ASSESSING THE RISK FACTORS ASSOCIATED WITH POTENTIAL FAILURE MODES IN THE SOFTWARE DEVELOPMENT PROCESS

by

Bashiru Lawal

Submitted in accordance with the requirements for
the degree of

DOCTOR OF PHILOSOPHY IN INFORMATION SYSTEMS

at the

UNIVERSITY OF SOUTH AFRICA

Supervisor: Professor Ernest Mnkandla
Co-supervisor: Professor Adéle Da Veiga

January 2023

Declaration of Authorship

I hereby declare that this thesis is my own independent work. It is submitted for the degree of Doctor of Philosophy in Information Systems at the University of South Africa. This thesis has not been submitted before for any degree or examination at any other university.

[illegible]

Bashiru Lawal
January 2023

Abstract

Poor management attitude has been identified as the major challenge to software project management. Lack of knowledge and understanding of other pertinent risk factor dimensions in software projects (such as potential failure modes in the software development process) by managers and the current scarcity of research on the applicability of safety and reliability engineering tools (e.g., Failure Mode and Effect Analysis (FMEA)) for software project risk management to support managers' intuition accounted for the poor managerial attitude to persist. This study was conducted to address the identified problems and provide answers to the main research question: 'How can the risk factors associated with potential failure modes in the software development process be assessed using the FMEA method?'

The Diffusion of Innovation Theory (DIT) and the Technology Acceptance Theory/Model (TAT/TAM) were combined to provide the theoretical foundation for the study. The study adopted a combination of case study and survey design methods employing a mixed-method approach: qualitative and quantitative methods for data collection. The Miles and Huberman (1994) method was used to analyse the qualitative data that emerged from the study while descriptive and inferential statistics were used to analyse the quantitative data. The findings revealed an authoritative list of potential failure modes in the software development process. The findings also revealed some likely causative factors of the identified failure and their associative risk effects on the software project outcome. The study also found that students that used the FMEA for their project risk management in a practical software engineering (SE) course acknowledged the method's ease of use and its applicability for the software project risk management practice.

Some contributions of the study include the provision of insight into another dimension of risk factors in software development projects and a modified risk theory adopted from FMEA method for risk assessment, thus, presenting a novel methodological contribution in the field of SE research.

KEY TERMS: Risk factors, risk assessment, potential failure modes, software development process, project risk management, FMEA method, project outcome, DIT and TAM.

Dedication

This thesis is dedicated to my late father Alhaji Hamza Oriolowo Lawal and my late cousin brother Pharmacist Nurudeen Olayinka Ayinde Lawal whom God used to be my role model that has brought me this far. May Aljannah Firdaus be their final abode. It is also dedicated to my family and friends.

Acknowledgements

My deepest gratitude and sincerest appreciation go to:

- ❖ My first supervisor, Professor Ernest Mnkandla, for his tremendous and valuable insight, detailed feedback, intellectual support, constructive criticism, suggestions, encouragement enjoyed in the research process enabled me complete this study;
- ❖ My second supervisor, Professor Adéle Da Veiga, for her intellectual guidance, insight and rigorous comments that provided a good shape to my work and make it far better;
- ❖ The following men and women for their support and contribution in one form or the other: Professor Conrad Mueller, Dr. B. Chimbo; Professor B. Mamba; Professor I. Osunmakinde; Ms. De Kock Estella and Mr. Thulare Obakeng;
- ❖ My colleagues and students from the computer science department Federal College of Education (Technical) Gusau – Nigeria for their support during the research process;
- ❖ The software development companies in Nigeria and their personnel that participated in this research and provided valuable support during the research field works. I say thank you all for your time;
- ❖ Special thanks to the University of South Africa Master's and Doctoral Bursary Unit for the 2022 bursary award given to me. The payment of my final year tuition fees and grants for research essentials kept me determined and resolute to complete this study;
- ❖ My late father, Alhaji Hamza Oriolowo Lawal, and my mother, Alhaja Wulematu Arinpe Lawal for their support and motivation;
- ❖ My family – my wife, Rukayya, my daughter, Fatimah, my sons, Mubarak, Mohammad and Abdullah, they stood by me throughout the research work when I was too busy to be with them or too preoccupied;
- ❖ Finally, and most importantly, my creator – the God Almighty for His wonderful protection, mercy and guidance on me throughout the duration of this programme.

Table of contents

Contents

List of Figures

List of Tables

List of Appendices

CHAPTER ONE

Introduction and Background

The aim of this opening chapter is to present a general introduction of the thesis by giving the background of the study that justifies the need to assess the risky potential failure modes in the process of software development. Section 1.2 discusses the problem to be addressed in the study and section 1.3 presents the research questions. Section 1.4 presents the hypothesis of this research. The research objectives are presented in section 1.5 and section 1.6 discusses the research scope. Section 1.7 is the summary of the research framework and methodology employed for the conduct of the research while the significance of the study is presented in section 1.8. Section 1.9 contains the structure of the thesis in an outline form and presented while section 1.10 is the summary of the chapter.

1.0 Introduction

Software development projects are known to be high-risk ventures that are incredibly difficult to manage. As at 2020, over 50% of world software projects are challenged, 31% are successful and 19% are completely failed projects (CHAOS Report 2020). Even though, the alarming statistics for software project failures at various levels as reported in the series of CHAOS Reports (e.g., CHAOS Report 1994, 1996) has been heavily criticized as ‘unjustified claims’ (Jørgensen & Moløkken-Østvold 2006) and the term ‘software crisis’ used in the report to describe the global project situation was not appropriate or fully justified (Glass 2006), statistics from the project management performance research conducted by research agencies in project development reveal that the failure rate of software development projects worldwide is still high (GitLab 2019; KPMG 2019; Wellington 2018; Giuseppe 2017; Lehtinen, Mäntylä, Vanhanen, Itkonen & Lassenius 2014; de Wet & Visser 2013). Unfortunately, the incidence of software project failure is becoming worse even as the industry is rapidly growing in complexity and criticality (Lehtinen et al. 2014).

The incidence of software project failures led to the establishment of software engineering discipline and software process improvement concepts about four decades ago (Lehtinen et al. 2014). These initiatives were intended to set a standard in the software developmental processes and improvement in the quality of software projects. Despite these initiatives, software developers have struggled to develop "consistently" successful software projects over the past three decades (AIPM & KPMG 2021; KPMG 2011). Most project administrators and managers have witnessed one of the many software project failures during their careers and continuous defiance of this manner is a growing concern (GitLab 2019; KPMG 2019; CHAOS Report 2015; Castsoftware 2015; KPMG 2011; Geneca 2011; Cerpa & Verner 2009). Similarly, a cursory study of some important quotes from respondents of the 2021 Australian Institute of Project Management (AIPM) and KPMG project management survey reveal that organisation inflexibility is still creating project obstacles, time delays and barriers to effective project delivery (AIPM & KPMG 2021).

Authors ascribe this unfortunate poor statistic of failure rates in software development projects to unprofessional attitude of managers towards managing the risks factors in software projects (Wellington 2018; Giuseppe 2017; Castsoftware 2015). Other important reasons from literatures are lack of interest in keeping up with modern technology and how best the delivery of software projects can be implemented (AIPM & KPMG 2021). Research on the global state of project management (e.g., Wellington 2018) found the largest contributing factor for project failure to be inadequate managerial training. Castsoftware (2015) shows that managers are not taking the right measures certified to understanding, estimating or evaluating and managing the risk factors that are associated with these projects. This factor highlights the fundamental problem associated with project management.

In addition to this, Wellington's research discovered risk management to be amongst the critical success factors for projects. However, most managers see risk management as optional or another discipline, which should take a parallel stance with developmental process (Wellington 2018; Mnkandla 2012). Instead of imbibing a prudent approach to managing these risks, they rely on their intuitive and other traditional testing techniques for their risk management (Castsoftware 2015).

Similarly, researchers and advocates of project risk management (such as Trzeciak 2021; Hopkin 2018; Giuseppe 2017; Aven 2016; Vahidnia, Özgür& Askerzadeet 2016; Castsoftware 2015; Lehtinen et al. 2014; Sarigiannidis & Chatzoglou 2011) claim that effective and prudent management of major risk factors in software development projects can minimize the incidence of project failure; hence, significant to project success. Over the past three decades, research on software risk management has primarily focused on developing guidelines for risk management in software development projects (PMI 2008; SEI 2006; Charette 1996; Boehm 1989). In 1989, Barry Boehm laid a foundation for risk management as a field of research in software development projects through seminar presentations and a pioneering book launch in risk management titled: “Software Risk Management”. Since then, researchers have developed numerous methods to guide managers on how to effectively analyse and manage the risk factors within any given project.

Castsoftware (2015) claims that proactive approach to managing the risk factors in software projects will move the traditional testing technique to the next level by identifying all potential problems that can emanate from high severity engineering flaws in all the stages of the software development life cycle (SDLC). The SDLC also known as the software development process is the series of identifiable stages that a software product undergoes during its lifetime (Pressman 1997). One other important benefit of proactive risk management is that it accounts for apparent issues that can influence response times to control the failure effect by preventing both project duration and budget overruns (Castsoftware 2015). These and other factors are important reasons why risk management is a dominant area of research in software development projects.

The PMBOK Guide (PMI 2017), specifically in the 11th chapter, is a project management guide that addresses project risk management activities. By this guide, risk management is expected to follow a set of risk management processes, which include planning, risk identification, qualitative and quantitative analysis, risk response strategies and the risk control measures. Yusuf (2018) summarizes the PMBOK guide, and recommends the following steps for project team members or project managers, in whichever capacity, to develop effective project risk management strategies. Firstly, team members or managers need to plan and expand on their knowledge of understanding of important risk factors, which are capable of influencing the success of software development projects. Secondly, qualitative steps must be taken to authoritatively identify these important risks. Thirdly, the relative criticality of these risks must be analysed and classified along with their relative effect on the project outcome (i.e., cost, time and quality). Lastly, managers should be adequately informed with alternative action plan to control or counter the effect of these important risk factors.

A critical analysis of Wellington's findings, which is in agreement, or corroboration with the PMBOK Guide to project risk management and the recommendations of Yusuf (2018) reveal that project managers' ignorance of other important risk factors in software development projects is a major contending challenge to software project success. A literature review conducted on project risk factors by the researcher shows that research on risk factors in software development projects is not detailed enough to inform software development practitioners on important dimension that needs more attention. In order words, research effort on risk factors in software development projects is expected to adequately prepare project managers and risk management team on how best to identify, analyse and control other important risk dimensions relevant to the software development process with a view to improving the chance of developing successful software projects

1.1 Background

A software development project is a type of project that is executed with the sole aim of evolving and delivering a software product (Pressman 1997). The activities may include re-use, maintenance, re-engineering of the existing projects and or development of new projects or any other engineering processes that will lead to the development of software product(s). Software development projects in the contemporary world are high-risk ventures that are prone to failure (Giuseppe 2017; Castsoftware 2015).

Project failure arises from projects that are not delivered on time and not in line with expectation (Giuseppe 2017; Castsoftware 2015). Failure can also occur if the project cost overruns its budget forecast in its business plan (McKinsey 2012). In other words, a project succeeds if it is delivered on time, within its financial forecast and with quality. However, recent studies from AIPM and KPMG (2021) reveal that success goes beyond cost, time and scope. Intended benefits must be delivered and achieved for projects to be considered a success. The ability to deliver successful and valuable project is not limited to users' expectation or financial involvement and timeline but extends to the acquisition of knowledge from the project best practices and project execution, including knowledge stemming from failures. Importantly, knowledge accounts for the key driver for resolving complex situations and avoiding bad decisions and their consequences.

In software development, knowledge on 'design patterns' are given due relevance (Stamelos 2010), and are reported in the literature as considerable best practices as relayed by experienced object-oriented software developers (Larma 2002). If patterns are best practices that can be re-used to resolve occurring problems, "Anti-Patterns" are project situations that involve risks, bad practices or frequently occurring malpractices that can lead to project failures. Some common software project management anti-patterns identified by Brown, Malveau and Mowbrav (1998) include the "Death by planning" (excessive preplanning software projects causing postponements), the "Poltergeist" (where do-nothing classes add unnecessary overhead), the "Cut-and-paste-programming" (directly copied programme segments causing significant maintenance problem), the "Boat Anchor" (a white elephant piece of hardware or software bought at great cost), the "Analysis paralysis" (striving completeness or perfectness in analysis phase causing project gridlock), and the "Golden Hammer" (a single technology that is used for every conceivable programming problem). Thus, understanding the dynamics and the causative factors of these anti-patterns and how to fix their consequences will make this type of software project management knowledge a success factor for building quality software system.

However, knowledge regarding the notion of “anti-patterns” in software project management is not given relevant attention in the business of software development. Knowing that so many software projects are high-risk ventures that are difficult to manage, it is logical to presume that software failures are caused by these error-prone bad practices. In other words, the lack of knowledge on the dynamics and consequences of one or more of these software project management anti-patterns on software projects is a challenge to successful software development projects.

In addition, statistics from the research in software development projects conducted by some globally recognized research agencies (AIPM & KPMG 2021; Standish Group 2020; GitLab 2019; KPMG 2019; Giuseppe 2017; Castsoftware 2015) in software development projects show that the rate of software projects failure at global context is still high. The researchers observe that situation is becoming more so as virtually all other spheres of disciplines are reliant on software product. Software project failure has a multiplier effect that is not limited to monetary aspects; it also threatens the survival of software development organizations (Yusuf 2018). Thus, research in the field revolves around how software project failures can be minimized.

The PMBOK (2017) Guide to project management provides *series* of approaches for scope management, budget and schedule control and risk management. Other chapters from the guide dwell on human resources project management with some points to select, develop, and manage a team. There is also a section that specifies indicators for performance evaluations and recommends how to use interpersonal skills to resolve conflicts. It is a comprehensive document often supported with graphs and charts. Wellington (2018) claims that scope management, budget and schedule control and risk management are the fundamental aspects of project management. However, out of all these techniques, effective risk management processes have been singled out as key to project success (Wellington 2018).

In spite of the PMBOK Guide that comprehensively spelt out the fundamental techniques for effective project management, research has shown that the level at which software project failures are occurring calls for concern. It is reasonable to wonder why not all projects are still completed on time, within scope, and slipped outside the stipulated budget. Large projects, especially those in software development projects, have poor statistics. Multiple studies (e.g., GitLab 2019; KPMG 2019; Giuseppe 2017; Castsoftware 2015; Standish Group 2015) show a significant share of software development projects overrun their original timelines or are never completed.

Most experts in software development agree that failure occurrence in software projects is far beyond expectation despite the fact that agents of these failures are predictable and preventable. Researchers and experts in the field also agree that there is more than one reason for a software project to fail. However, research conducted on 70 failed projects by Verner, Sampson and Cerpa (2008) to investigate the exact causal factors for their failure shows that the entire failed project investigated suffered from poor project management. Furtherance to this finding, Castsoftware (2015) and Wellington (2018) identify poor managerial skills as the main project management challenge. Several factors are responsible for this problem. According to Wellington (2008), the problem arises from the fact that project managers are not adequately trained for the job and are unwilling to undertake more training. Castsoftware (2015) relates the problem to managers relying on their intuition in lieu of project risk management when detailed information about other important project risk factors is not available to support their intuitive ability.

Significant research has been conducted in this study area. For example, McFarlan (1981) makes a list of 20 risk factors in software development projects based on three major dimensions: project size, technology experience and project structure. Cule, Schmidt, Lyytinen, and Keil (2000) identify 55 risk factors in software projects that are associated with four dimensions: task, client, environment, self and suggest corresponding risk management control for each of the risk item. Houston (2000) in his research work conducted a literature review to develop a list of 29 software risk items, which the researcher considers as the major risks that are highlighted in existing literature.

The works of Schmidt, Lyytinen, Keil and Cule (2001) and Wallace, Keil and Rai (2004) have contributed significantly to the project risk factors' research because of their distinctive and wide dimensional coverage, theoretical substantiation and the susceptive approach used to identify and classify risk factors in the research. Today, Wallace et al. (2004) and Schmidt et al. (2001) possess a domineering referral archetype in software development project risk management literature. Schmidt et al. (2001) developed a list of 53 risk factors that are associated with 14 different dimensions of which 11 have cross-cultural application. Similarly, Wallace et al. (2004) identify 27 risk factors that are associated with six dimensions.

There are other related studies in the field that focus on risk factors in software development projects. For example, Bilal, Gani, Liaqat, Bashir and Malik (2020) applied a structured questionnaire together with a survey method to discover 83 risks factors across the SDLC for a size-scaled enterprise that included small and medium enterprise software projects in Pakistan software industry. The study found that a number of the risks associated with the studied software projects in Pakistan software industry seemed to be significant and high.

In addition, Menezes, Gusmão, and Moura (2018) found 148 risk factors in software development projects with the requirement specification stage of the SDLC having the highest susceptibility to risk.   Hijazi, Alqrainy, Muaidi and Khdour (2014a) discovered 100 risk factors in the SDLC and described their nature in such a way that will support software practitioners' risk management experiences. However, findings from previous research (e.g., Menezes et al. 2018; Vahidnia et al. 2016; Hijazi et al. 2014; Wallace et al. 2004; Boban et al. 2003, Schmidt et al. 2001; Houston 2000; Cule et al. 1998) that identified and assessed risk factors in software development projects overlooked the causal factor and the effects of the analysed risks on project outcome. In addition, previous studies that identified and assessed software project risks at various dimensions, provided useful insight that empower managers on how to probe risk levels. However, previous research (e.g., Menezes et al. 2018; Wallace et al. 2004; Schmidt et al. 2001; Houston 2000; Cule et al. 1998) overlooked the dimension of failure modes in the SDLS, thus, do not provide managers information about risk factors associated with the potential failure modes in the software development process.

The implication of the identified gap is that the lack of information about the dynamisms of risk factors associated with the potential failure modes in the process of software development and their multiplier effects on the project outcome contribute to managers' ignorance of the impending threat that are attached to these risk. This in turn, contributes to some negative managerial decisions that have negative impacts on software project outcomes.

Another area of research in software engineering intended to improve the practice of risk management is how to integrate the safety and reliability engineering techniques for the conduct of risk management of software projects. This is supported by the recommendation of AIPM and KPMG (2021) on the need by risk managers to keeping up with modern techniques and gets trained on how best to use the techniques in the delivery of project. Safety and reliability engineering techniques such as the failure mode and effect analysis (FMEA) are well-known risk analysis tools that are widely used for quality improvement and risk assessment tools in manufacturing, aeronautics, defence industries, etc. (Grunske & Han 2008; Hansen 1989).

The application of the FMEA technique has long been explored in software engineering to identifying potential design flaws and development process failures in software-based system before they occur, with the sole aim of eliminating them or to reduce the risk associated with them (Mitrabinda & Durga 2011; Appukkutty, Ammar & Goseva-Popstojanova 2005). One of the procedural processes of the FMEA that makes it widely accepted is techniques in which team members are chosen and gathered from different functional departments to identify and assess the risk in a system using different perspectives and standpoints.

Over the years, there has been growing interest in the application of reliability engineering tools such as the FMEA for the conduct of risk analysis and management in the software engineering discipline. For example, Hassan, Goseva-Popstojanova and Ammar (2005) applied the FMEA methodology for analysing hazards in software architecture level.

Furthermore, Chin, Wang, Poon and Yang (2009) applied the FMEA for risk analysis in software-based system using a group-based evidential reasoning approach.

Mitrabinda and Durga (2011) deployed the FMEA methodology for software risk assessment at architectural level using UML diagrams while Gupta, Gogate, and Gupta (2012) used same methodology for software failure analysis at the requirement phase. However, the application of the FMEA technique as used in the previous research focused on how to identify and manage risks at early stages of the SDLC (i.e., requirement and design stages) but not across the entire stages of the SDLC. Other research are aimed at improving the reliability of software products through advanced management, which only address complexity but cannot effectively prevent failure incidences.

It is important to note that most of what was reported from the available literatures about the integration of the reliability engineering tools such as FMEA for the conduct of risk management in software engineering is productive and resourceful. However, the problems associated with the previous research on the application of reliability engineering techniques in software engineering is that those that integrate the engineering tools for their risk assessment in software engineering have not empirically justify whether the procedural requirements of the tools conform to the requirements of other prominent project management frameworks such as PMBOK, SEI and Boehm (Gupta et al., 2012; Mitrabinda & Durga 2011; Appukkutty et al. 2005). In addition to this, most of these tools have not been industrially validated to evaluate their effectiveness for the conduct of risk management in software engineering. This in turn, de-motivates project managers' interest and confidence in the use of reliability tools.

To sum it up, evidences from the literature show that the rate of failure in software development projects worldwide is still high (CHAOS Report, 2020; GitLab 2019; KPMG 2019; Wellington 2018; Giuseppe 2017). Despite a wider range of risk management techniques and huge resources invested by practitioners and researchers in software development to control the failure rate, software project failure does not seem to be decreasing. However, poor management attitude has been identified as the major contributor to the software failures (Wellington 2018). It is therefore desirable to explore ways of improving management practice in software development projects.

1.2 The problem statement

Attitude of managers towards risk factors management is a major challenge to successful software development. Many authors ascribe this challenge to three main factors. Firstly, the narrow knowledge and understanding of other pertinent risk factor dimensions in software projects by managers to support their intuitive capacity accounted for their persistent poor managerial attitude. This in turn contributes to managers' ignorance of risks multiplier effects on the project outcome (GitLab 2019; KPMG 2019; Menezes et al. 2018; Yusuf 2018). Secondly, there is a growing interest in literature by scholars and researchers in SE on the application of safety and reliability engineering tools such as FMEA for project risk analysis. However, the lack of evidence from literature that empirically confirms efficacy and effectiveness of the method as a competent software project risk management tool seems to pose problem to developers and managers who are searching which appropriate, reliable and cost-effective tool to use for the conduct of project risk management. Lastly, managers are still not getting it right in applying appropriate response strategies to address the threats posed by occurring risks in their projects.

There is, therefore, the need to extend the existing research on risk factor dimension to include other important risk dimension such as potential failure modes in the process of software development that has been neglected in previous research to support managers' intuitive ability to address occurring risks in their projects. There is also the need to explore more on the integration, efficacy and how best to use the safety and reliability engineering tools (such as FMEA) to establish best practices for the conduct of risk management in software development projects.

To address the identified problems in line with the objective of this research, the researcher will engage a team of software development experts and selected student developers in an exploratory case study research, focus group discussions, survey research and an interactive study to resolve the current challenges facing software `development projects. .

1.3 The research questions

The main research question raised to guide the conduct of this research is:

How can the risk factors associated with potential failure modes in the software development process be assessed using the FMEA method?

The followings are the explanation of the key variables of the main research question:

- *Risk factors* are agents that are technical, economic and behavioural in nature that can influence the successful completion of a project resulting into inadequate project operations and overruns (Thomas 2008). Examples are inaccurate requirement, bad defect tracking, inappropriate user involvement technology change, and so on.
- *Failure mode* is a characterization of the manner by which a product or process fails (Wikipedia 2015). It relays a detailed description of occurrence and severity (that is, effects) in relation to the abstract model of the causal factor of the occurring failure. For example, 'expected functionality is omitted' is a probable failure mode. Now, if an expected functionality of a system is omitted as a failure mode and the failure effect turns out to pose significant threat to the successful completion of a project leading to inadequate project operations and overruns during the development process, then it becomes a risky potential failure mode.
- *Software development process* is the series of identifiable stages that a software product undergoes during its lifetime (Georgieva 2010). The five (5) studied stages of the software development process adopted in this research are planning, Analysis and specification, Design, Coding, and Installation and maintenance.
- The *FMEA* stands for Failure Mode and Effects Analysis. It is a reliability-engineering tool used to identify potential failure modes in a system or process and their causes and effects (Tay & Lim 2016).

The research question is further divided into 6 sub-questions, which are answered through qualitative research involving review of theoretical concepts of prominent project management frameworks in literatures and certain exploratory case studies research involving focus group discussions to reinforce and validate the findings from available literatures. Other sub-questions relate to the actual experiences when the researcher with a team of software development experts and selected student developers, applying the FMEA method to assess the risk factors associated with potential failure modes in the software development process on the project outcome.

The research sub-questions are listed as follows:

1. What potential failure modes in the software development process do software development experts consider to be risky to the software project outcome?
2. What factors are considered by software development experts to be the likely causes of the identified potential failure modes?
3. What failure symptom(s) can be used to detect likely occurrence of the potential failure mode in the software development process?
4. What multiplier effects do risks associated with potential failure modes in the process of software development pose on the project outcome (cost, time & scope)?
5. To what extent is the FMEA feasible and practicable for the conduct of software project risk management?
6. What risk response strategies are considered appropriate actions for managers to address the threats posed by the identified potential failure modes in the software development process?

1.4 Research hypotheses

Relationships and tentative influence among risk factors, risk management methods and project outcome in software development projects have been explored and established by researchers (e.g., Thomas 2008; Deephouse, Mukhopadhyay, Goldenson, Kellner & Marc 2005; Barki, Rivard & Talbot 2001; Jiang, James & Klein 2000; Wallace 1999; Nidumolu 1995). However, these studies used restricted risk factors dimensions, which ignore other important dimensions such as risk factors associated with potential failure modes in the software development process. These studies are also characterized to have limited research that investigates the relationship between the reliability engineering tools (such as the FMEA) and the software project risk management process. Thus, linkages between the risk factors associated with potential failure modes in the software development process and the software project outcome, and linkages between the reliability risk management tools (FMEA) and software risk management process are generally overlooked in the literature.

In this study, hypothesis testing is conducted on variables relationship in the study research questions that are related to the positivist epistemological orientation based on empirical facts (i.e., Research questions 4 and 5). The aim is to obtain general, objective and scientific results that will represent the entire population and could be generalised over similar ones. On the other hand, variables linkages in the remaining research questions (Research questions 1, 2, 3 and 6) of the study are not investigated because the affected research questions are related to value judgments or subjective interpretation. The Research questions 1, 2, 3 and 6 adopt the interpretive epistemological orientation where the reality of a phenomenon is the result of social construction between the interacting social actors. Thus, the findings could not be generalised.

In addressing the established literature gap in line with the planned hypothesis testing, the researcher has formulated the following three hypotheses on the anticipated relationship among the variables in the study research questions 4 and 5.

- *Relationship between the risk factors associated with potential failure modes in the software development process and the project outcome (RQ 4)*

  H_{01}: There is no significant difference between the mean scores' multiplier effects of potential failure modes in the software development process on the project outcome (cost, time & scope/quality).

  H_1: There is significant difference between the mean scores' multiplier effects of potential failure modes in the software development process on the project outcome(cost, time & scope/quality).

- *Relationship between the FMEA and the Software project risk management process (RQ 5)*

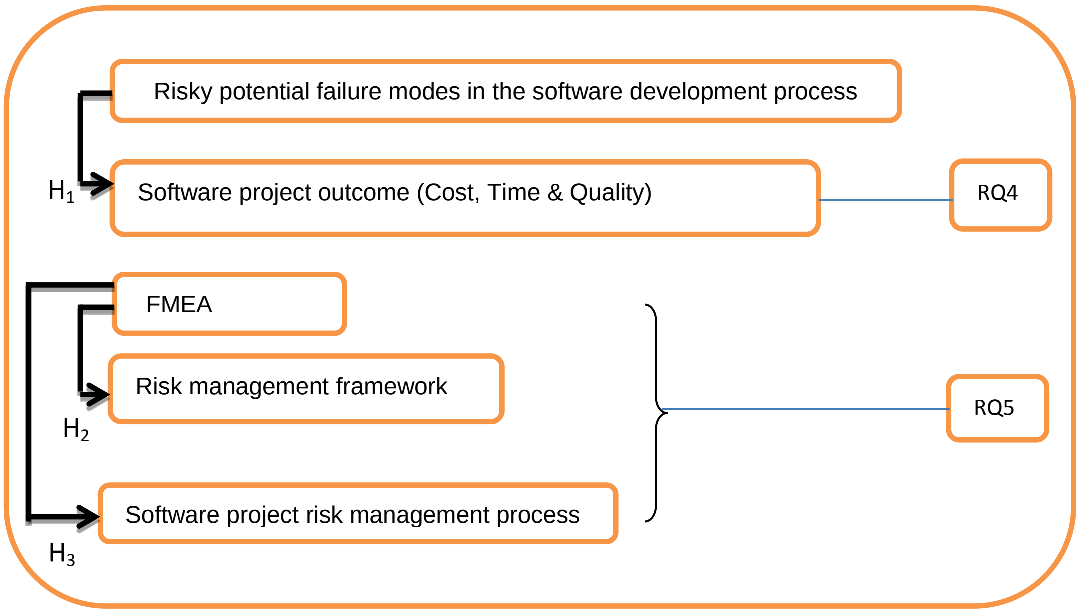
$H_{02:}$ There is no significant difference between the mean scores of FMEA adherence to other popular project risk management frameworks.

H_2: There is significant difference between the mean scores of FMEA adherence to other popular project risk management frameworks.

$H_{03:}$ FMEA method is not efficient for the conduct of software project risk ` management.

H_3 FMEA method is efficient for the conduct of software project risk management.

Figure 1.0 presents a mapping of the research hypotheses on the formal research questions of the study.



Risky potential failure modes in the software development process
H1
Software project outcome (Cost, Time & Quality)
RQ4
FMEA
Risk management framework
H2
RQ5
Software project risk management process
H3

Figure 1.0: Mapping the research hypothesis on the formal research questions of the study.

Figure 1.0 presents the mapping of the informal hypotheses against the formal research questions. Variables of the formal research questions are in the white boxes. Double-headed arrows indicate the informal hypothesis - relationship being investigated. The thin line connects the hypothesis testing with the formal research question of the study.

1.5 Research objectives

The objectives of this research are formulated in such a way that the objectives correspondingly provide answers to the research questions. The main objective of this research is to assess the effects of risky potential failure modes in the process of software development on the software project outcome using the FMEA method. Specifically, the following sub-objectives are stated to help in the attainment of the main objective:

1. *To identify the risky potential failure modes in the software development process that is considered by software development practitioners as threat to software project outcome*. This objective is implemented through an exploratory case study involving a focus group discussion with selected software development practitioners.

2. *To identify the likely causal factors of the identified failure modes.* This objective is attained through an exploratory case study involving a focus group discussion with selected software development practitioners.

3. *To identify the possible detection method that can be used to detect the likely occurrence of the potential failure mode in the software development process*. The same method used in sub-objective 2 and 3 is implemented here to attain the objective.

4. *To assess the likely multiplier effects, which the risky potential failure modes in the software development process pose on the project outcome. This objective is implemented by (i) conducting* a survey research using questionnaire *to analyse the perceptions of software development practitioners on failure mode multiplier effects on project outcome, and (ii)* by asking selected software development practitioners to apply the FMEA to analyse the likely multiplier effects on the projec*t outcome*
5. To investigate the feasibility and practicability of integrating the FMEA for the conduct of risk management in software development projects. This objective is implemented through survey study with experienced software

practitioners and through an FMEA interactive session with undergraduate software engineering students on the FMEA application for the software project risk management

6. *To suggest effective risk response strategies that managers will use to address the effect of the threat posed by each of the identified potential failure modes.* To accomplish this objective, results from objective 5 will be used with the criticality classification criteria of the USDOD (MIL-STD-1629 1980).

1.6 Research scope

Most of the existing projects on risk management models in software projects risk management such as Boehm and SEI admitted a two steps approach to project risk management, which are *risk assessment* and *risk control* and that risk assessment, must be the first step, which must be completed before risk control. This research aims at assessing the risk factors associated with a specific dimension in the software development process and determines its effect on software project outcome. The emphasis is on assessment of risk factors associated with potential failure modes in the software development process using the FMEA method. The FMEA process is conducted in all the studied stages of the software development process to identify, analyse, prioritise and classify the risk factors in the software development process.

In addition, the FMEA method used for the risk assessment in this work can be used beyond the boundaries of the software development industry. Obviously, that can be applied for risk assessment in any production-based project. However, the focus of this research is the use of FMEA method to assess risk factors emanating from software development projects.

1.7 Research methodology

In this study, a mixed method is used where both qualitative and quantitative methods are used to answer the research questions (Doyle, Brady & Byrne 2009). The quantitative and qualitative are concerned with positivists and interpretivism epistemology orientations respectively, and are combined not for competition but for complementary and parallel purpose (Weber 2004). The mixed-methods approach is considered appropriate for this research because quality data and deep perspectives from the software development experts on the possible failure occurrence and their likely causes in the SDLC are required. The risks associated with these failures are quantified and the different sources of evidence are crucial for validating the contribution of this research. Five fieldworks designed for this study are characterized as survey and case studies. The reason for this is that the challenges in risk management are more practically oriented. Thus, it is assumed that such practical oriented and industry-verified research is the appropriate design approach to validate the research model and findings.

The population of study is represented by three target groups, which are composed of:

(i) Selected software practitioners from software organizations in Nigeria who have established practical experiences in at least one of the following software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control.

(ii) Selected group of practitioners and researchers in software engineering around the world who have actual experiences in software development projects in at least one of the following software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control; and

(iii) Selected group of undergraduate Computer Science students who have registered for software engineering as a core course in their programme at a public university in Northern Nigeria.

Five field studies are conducted where multiple research instruments are applied to collect both qualitative and quantitative data to be used for this research. The research instruments used to gather the required data include: focus group discussions (used to collect qualitative data), hard copy and mailed questionnaires (used to gather quantitative data), audio/video tapes (for qualitative data) and PFMEA worksheet (used to collect qualitative and quantitative data). The multiple instruments are adopted to ensure validity and reliability of the captured data.

The method used to analyse the qualitative and quantitative data collected from the fieldworks is based on the Miles and Huberman (1994) method of analysing qualitative data and descriptive statistics respectively. The Miles and Huberman method is considered most appropriate method because of the small size of selected participants and data collected from the study. Thus, the application of specialized software for analysing the qualitative data was not required. Similarly, the descriptive and inferential statistics involving multiple statistical tools such as: Mean, Standard Deviation, Grand Mean, Percentage, and Frequency Distribution, ANOVA statistics and SPSS package are used to analyse data obtained from questionnaires data while FMEA process, standards and rating tables are used to assess, prioritize and classify risks across the studied stages of the software development process.

1.8 Significance and contribution of the study

This research adopts a proactive risk management strategy by applying validated method (FMEA) to identify, analyse, prioritise and classify risk factors associated with potential failure modes in the software development process. The study conducted software project risk assessment using the modified Probability Impact (PI) theory of FMEA process to prioritize and classify risk factors criticality in the software development process and their super set of risks and rating for reference and practical purposes.

One process through which this present research is conducted differently from the previous research is the adoption of FMEA tool for the risk assessment, which authoritatively identified failure possibilities i.e., the establishment of potential failure modes in the software development process, explored and described their likely causes and effects, assessed, analysed and classified the risk associated with the failure possibilities in the software development process and proposed to managers a control procedure to mitigate against the risk associated with the identified failure possibilities. Also, the group dynamic in which the FMEA team members are assembled from different background and functional departments for the conduct of the FMEA makes the FMEA method a widely acceptable method for risk assessment technique (Zhao, You & Liu 2017). The inclusion of detection method aimed at controlling failure occurrence factor adds more values to the risk analysis compared to other traditional approach (such as the risk matrix) that employ factors of Occurrence and Severity.

This study is expected to provide software development practitioners with useful information about series of potentially high threat failure modes that needs much attention during the software development process. The assessment and classification of their associated risk provides a unique strategy into how managers can focus more on other important risks in the software development process and determine what best risk control mechanism is to be deployed to mitigate against the risk. This aspect can be seen as a model that contributes to other project risk management frameworks.

1.9 Organization and structure of the thesis

All the studies conducted in this research are presented in the thesis in six chapters. Figure 1.0 illustrates an overview of the thesis structure.

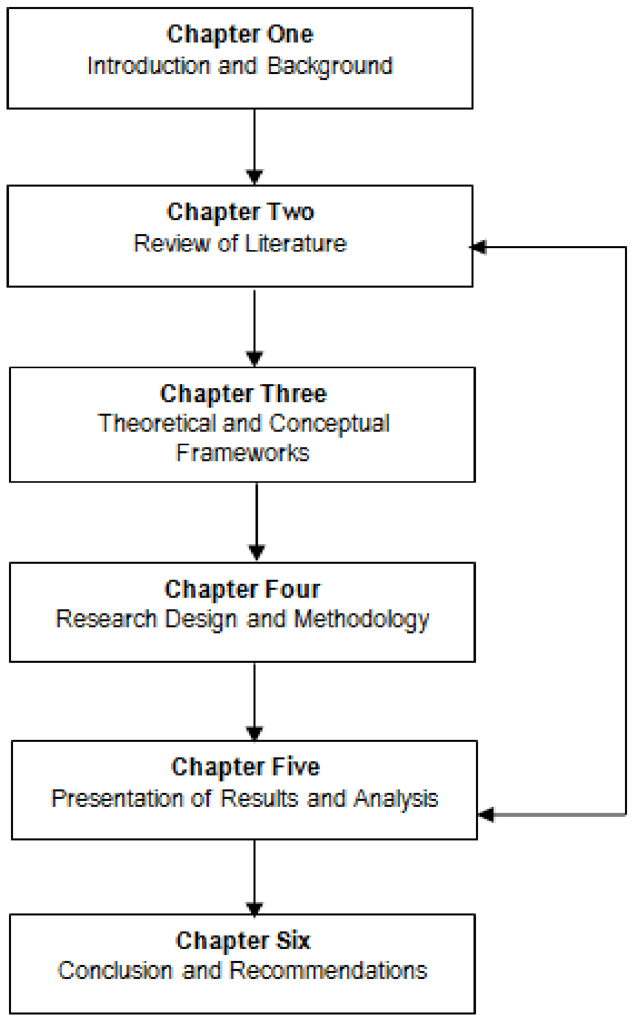

Chapter One
Introduction and Background
Chapter Two
Review of Literature
Chapter Three
Theoretical and Conceptual Frameworks
Chapter Four
Research Design and Methodology
Chapter Five
Presentation of Results and Analysis
Chapter Six
Conclusion and Recommendations

Figure 1 1: An overview of thesis structure

The overall objective of this thesis is to assess the risk factors associated with potential failure modes in the software development process. By using scientific procedures, the FMEA method is validated as suitable for the conduct of risk assessment in software project. Potential failure modes in the software development process are identified and the effects of their associated risk factors on project outcome are evaluated. Also, resulting risk criticality of individual failure mode is assessed, prioritized and classified and the FMEA method usability and practicability are statistically tested.

Chapter One presents a clear focus of what the research aims to achieve. Hence, issues addressed in the chapter include research background, problems statements, research questions, and aim and objectives of the study. Other topics include the research hypotheses, research scope, research methodology, research significance, structure of the thesis and chapter summary.

Chapter Two contains a review of academic literature concerning the global status of software project failures. The chapter also presents the review of previous research conducted on the area of risk factors and their dimensions and management in software projects, software project risk management models, failure modes in the stages of the software development process, safety and reliability engineering tools and their applications in software project risk management and the summary of key findings from the reviews are presented, limitations and drawbacks of previous studies and motivation for current studies.

Chapter Three presents the relevant theories that serve as the theoretical foundations of this study. The main objective of chapter three is to discuss the theoretical frameworks used in present a clearer structure and vision of the study. The chapter also presents the conceptual model used to illustrate the connectedness of the various constructs that informed this study without ambiguity

Chapter Four provides a clear focus of the research methodology used with close attention to the design and philosophy applied to conduct the research and in answering the research questions. The chapter also discusses the methodology used in identifying the potential failure modes in the software development process and method used to assess their associated risks in the software development process. The chapter also discusses data to be collected, the field studies conducted, and the research validation case studies.

Chapter Five is a presentation of the results analysed from both qualitative and quantitative data collected from the research. The analysis is organized by first presenting the qualitative data before analysing the quantitative data. The qualitative data are collected from the focus group discussions conducted in Field study 1 and Field study 2 while quantitative data are data collected from the four questionnaires used (Failure Mode and Effect Analysis Adherence Questionnaire (FMEAAQ), Failure Modes' Effects on Project Outcomes Questionnaire (FMEOPOQ), Potential Failure Mode Effects on Project Outcomes (PFMEOPOQ) and the Failure Mode and Effect

Analysis Evaluation Questionnaire (FMEAEQ)) and the FMEA worksheet across the five conducted field studies.

Chapter Six discusses the key findings of this research and conclusions drawn from this research. Chapter six also presents the research expected theoretical, knowledge, policy and practical contributions. In addition to this, the chapter highlights what the limitations of the study are and reflects on the study. The chapter concludes by recommending future academic research questions and makes concluding remarks about the study.

1.10 Chapter Summary

This chapter provides a clear focus of what the research aims to achieve with a detailed background of the study. The background explicitly discusses the cause and effect analysis of software project failure and identifies poor project management as the major challenge to successful project development. The chapter also justifies the need to explore another dimension of risk factor in software project development and the possibility of integrating the FMEA for the conduct of risk assessment in SE as a means of addressing the identified problem. The research problem statement, research questions, research hypotheses, aims and objectives and research scope are clearly stated in guiding the study. The chapter ends with a presentation of structural layout of the entire thesis.

The next chapter of this study provides the reader with the literature review conducted to establish the basis for this research.

CHAPTER TWO

Review of Literature

This chapter discusses some relevant topics related to the issues under investigation with the aim of identifying any research gaps and justifying reasons for the assessment of risk factors associated with potential failure modes in the software development process. The global trend of software project success or failure is first reviewed and presented in section 2.1. In section 2.2, the general concept of risk and risk management is briefly discussed. The review on research studies on software risk management and their limitations as observed from the literature is done in section 2.3.The review on the risk factors management in software development process and discussions of the task specifications in all the stages of the SDLC is presented in section 2.4. Review of reliability and safety engineering techniques as used for failure analysis and project risk management and their application in SE follows in section 2.5. Observations/deductions, summary, and motivation for the current research are presented in section 2.6 while section 2.7 is summary of the chapter.

2.0 Introduction

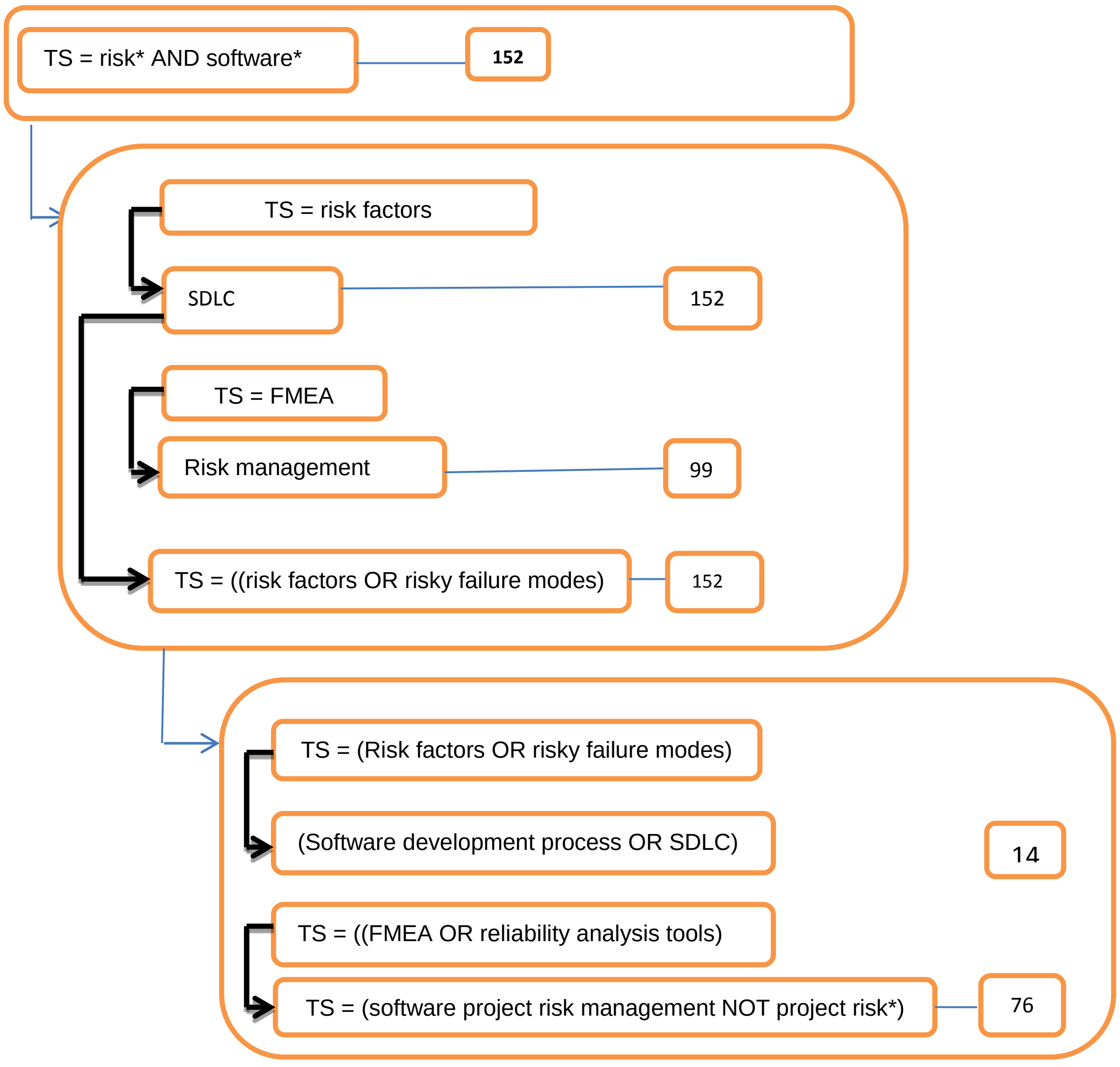
The quantum of resources (specifically time and effort) invested in creating research ideas and concept determines the degree of quality and the level of success of research. The specific objective of conducting a literature review according to Pinder, Wilkinson and Demack (2003) is to establish a convincing basis that will determine whether the developed ideas and concepts are worth pursuing or not. Hence, the assessment of risk factors associated with potential failure modes in the software development process begins with an extensive review of related literature. To establish a realistic trend of failure rates in software development projects, a comprehensive literature study was conducted on published reports of the past ten years of global researchers with relevance bordering on project development. This exercise was conducted mainly to ascertain whether the statistics on project failure rate is a mere exaggeration or a reality.

The fact of knowing that this research is intended to contribute to the improvement on the risk management practice of software development organization through the use of FMEA method for project risk management, it is essential to review existing academic research on software risk factors, risk management and the application of FMEA in software development in order to open potential research avenues. Thus, the literature review is aimed at: (i) assessing the rate of global software failure, its causative factors and remedy; (ii) assessing the meaning of risk as conceptualised in SE; (iii) reviewing software risk management and evaluating the current risk management strategies in SE; (iv) examining the risk factors management in software development projects and (v) discussing the possibility of applying the FMEA method for the conduct of risk management in software projects.

To achieve the stated objectives, a systematic review of literatures was conducted on published articles on risk, risk factors, risk and risk management methodologies in software projects of last three decades. The review focused on peer-reviewed articles, published academic journals specialising in project management, software project risks, software risk factor management, software project management models, stages of software development process and FMEA application in SE. The review utilised the key SE project management journal databases available through the UNISA library like: Science Direct, Springer, IEEE, Emerald, and Informs. The Google scholar search engine was also consulted.

The review used a predefined general search term: TS = risk* AND software* (yields 198 hits). At this point, a systematic refine search was conducted using the following search term: TS = risk factors AND SDLC (yields 152 hits), TS = FMEA AND risk. management (yields 99 hits), and TS = ((risk factors OR risky failure modes) AND SDLC)) (yields 152 hits), and involved checking the popular journals on software project development, project management, life cycle and application of FMEA for project risk analysis. Accessibility challenges of some journals were addressed by ordering articles through the UNISA library. The review period covered the search date of each journal up Until January 11, 2017. However, an updated search was conducted more recently to update the review. More than 200 articles were reviewed.

The evaluation of the 132 articles returned under a refined search term: TS = ((risk factors OR risky failure modes) AND (software development process OR SDLC)) found only 16 articles with relevance to the subject of this research. Using a more specific search term: TS = ((FMEA OR reliability analysis tools) AND (software project risk management NOT project risk*)) returned 76, out of which 12 were found to be useful for this research. However, using a refined specific search term: TS = ((software risk* OR failure mode risk*) AND (software development process OR SDLC)) 14 results were obtained, and after evaluating these results critically, no paper was found to be related with the objective of this current research. Figure 2.0 presents mapping the search, which indicated how the search terms were mapped with the number of articles that were returned from the search.



TS = risk* AND software*
152
TS = risk factors
SDLC
152
TS = FMEA
Risk management
99
TS = ((risk factors OR risky failure modes)
152
TS = (Risk factors OR risky failure modes)
(Software development process OR SDLC)
14
TS = ((FMEA OR reliability analysis tools)
TS = (software project risk management NOT project risk*)
76

Figure 2 0: Mapping the search

In all, 42 articles were selected to be useful for the purpose of this research. Each of the considered articles either has provided list of risk factor dimension in software projects or methodology for assessing risk or suggested a contribution towards improving risk management practice in software projects. Large numbers of those journals where the selected items were found are *IEEE Transactions on Software Engineering, IEEE Transactions on Engineering Management, International Journal of Project Management, Information Management and Computer Security and Journal of Management Information Systems*. The listed journals capture the current trends in research and practice of software project risk factors assessment and techniques used for project risk management. The remaining sections of the chapter discuss the analysis of these publications leading to the elicitation of the themes and developmental trends.

2.1 Status of software development projects: Failure or success?

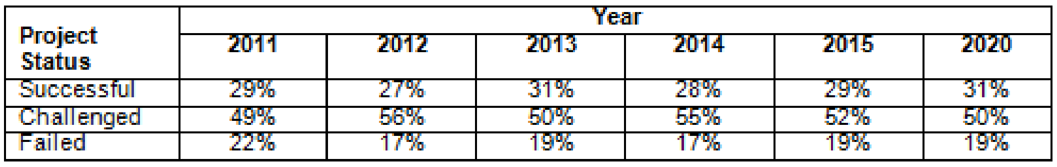
Software development project according to Pressman (1997) is a type of project executed with the sole aim of evolving and delivering software product. The activities may include re-use, maintenance, re-engineering of the existing projects and or development of new projects or any other engineering process that will lead to the development of software product.

The issue of whether the global status of software development projects can be rated as success or failure has generated hot debate among the software practitioners and researchers for many years. The most commonly cited definition of project failure/success is the one from the Standish Group (Standish Group 1999, 1994), which defines software project failure as projects with budget overruns, schedule slippage and one with unmet customer satisfaction. Kerzner (1995) attributed project success to the one that is completed within the period, within budget, and one meeting all business objectives and obtaining user acceptance.

Recent statistics by the 2021 AIPM and KPMG project management survey on project performance reveal that about 50% of world software projects encounter different obstacles to effective project delivery. Also, the KPMG (2019) survey on the future of project management: global outlook 2019 revealed that organizations around the world including software companies continue to face difficult challenges in delivering projects that meet all objectives around the iron triangle of time, cost and scope, along with achieving stakeholder satisfaction. The KPMG (2019) key findings revealed that 19 % of organizations surveyed deliver successful projects and less than 37 % of the organizations are likely to deliver projects on time and on budget at least most of the time.

Also, a comparative study of surveys conducted by Standish Group Research of 2020 and 2015 (i.e., CHAOS Report 2020, 2015) reveals that, the failure occurrence of software projects is still not at a decline as shown in Table 2.1. The table shows that as at 2020, 50% of world software projects are challenged, 31% are successful and 19% are completely failed projects.

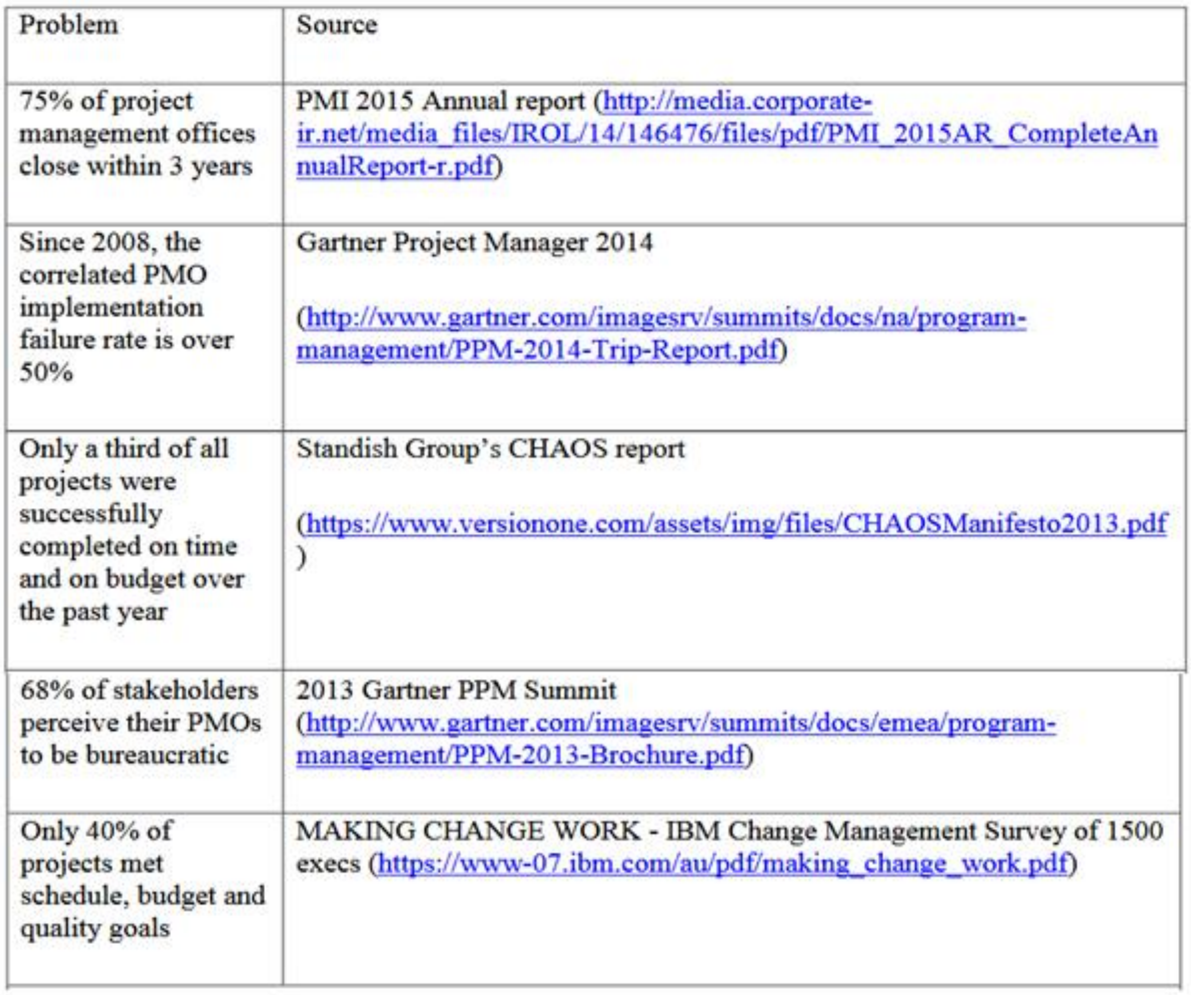
Table 2 1: Current status of software projects (Source: Chaos Report 2020. 2015)



Project Status	Year					
	2011	2012	2013	2014	2015	2020
Successful	29%	27%	31%	28%	29%	31%
Challenged	49%	56%	50%	55%	52%	50%
Failed	22%	17%	19%	17%	19%	19%

In addition to this, Giuseppe (2017), a publication of *Project Management (PM) World Journal* of 2017, empirically reveals that failure rate in software development projects is at the increase despite the high-tech inducement in the development process and that poor project management practice exhibited by managers is the major challenge to project success. Table 2.2 highlights the staggering findings from the report.

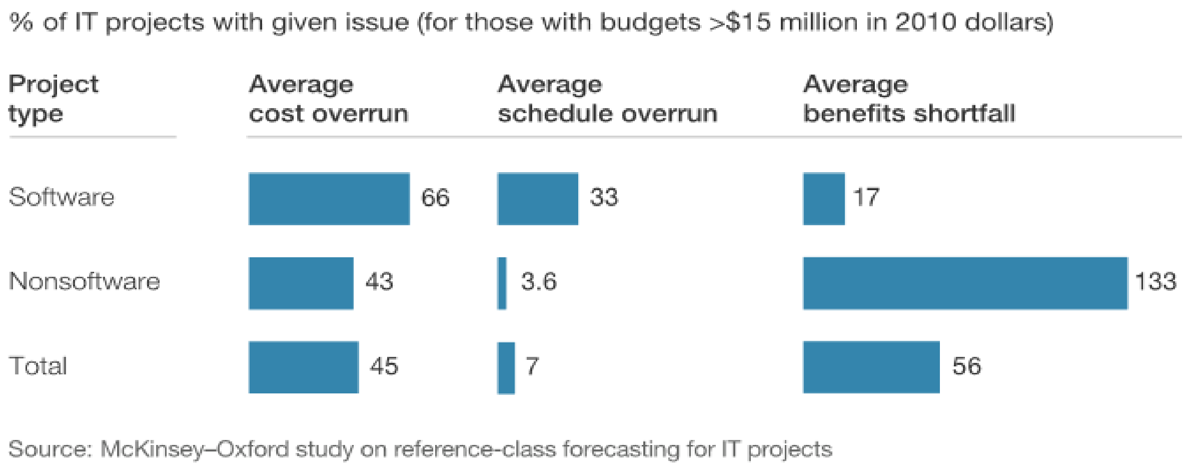
Table 2 2:Fact and figures of global status of software development projects (Source: Giuseppe 2017)



Problem	Source
75% of project management offices close within 3 years	PMI 2015 Annual report (http://media.corporate-ir.net/media_files/IROL/14/146476/files/pdf/PMI_2015AR_CompleteAnnualReport-r.pdf)
Since 2008, the correlated PMO implementation failure rate is over 50%	Gartner Project Manager 2014 (http://www.gartner.com/imagesrv/summits/docs/na/program-management/PPM-2014-Trip-Report.pdf)
Only a third of all projects were successfully completed on time and on budget over the past year	Standish Group's CHAOS report (https://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf)
68% of stakeholders perceive their PMOs to be bureaucratic	2013 Gartner PPM Summit (http://www.gartner.com/imagesrv/summits/docs/emea/program-management/PPM-2013-Brochure.pdf)
Only 40% of projects met schedule, budget and quality goals	MAKING CHANGE WORK - IBM Change Management Survey of 1500 execs (https://www-07.ibm.com/au/pdf/making_change_work.pdf)

Similarly, in the survey conducted by McKinsey and Company (2012) and the BT Centre for Major Programme Management at the University of Oxford on a study of 5,400 large scale IT Projects (projects with initial budgets greater than $15M) in 2012, it was discovered that 17% of the surveyed projects are challenged; 66% have budget overruns and 33% exceeded their duration. Table 2.2 presents the percentage of IT projects with given issues.

Table 2 3:Percentage of IT projects with given issues (McKinsey & Company 2012)



In conclusion, the above statistical reports are a clear indication that the failure rate of software development projects are rising with time and that the manner in which these projects are managed in most software development organizations are the root cause. These findings reinforce the findings of Wellington (2018) and Castsoftware (2015). One can also conclude that the extent of failure rate is threatening the survival of software development organizations and that as software development organizations continue to invest huge resources (money, time and technology) in software development projects, a major area of concern revolves around how to reduce the failure rate.

2.1.1 The way forward: Technology or project management?

A lot of techniques and approaches have been proposed to address this situation that is threatening the existence of software companies. Most of the proposed tools focused on how to improve software products through technology inducement or project management. Although an improved software product is desired, an in-depth analysis of the situation shows that technology inducement efforts address only 50% of the problems threatening the existence of software companies (Nogueira 2000). This is because technology inducements only address complexity but cannot prevent failure incidences. A project experiences budget overrun if the achieved performance does not equal the planned estimates (Nogueira 2000; Nogueira, Luqi & Berzins 2000).

The planned estimates that can help to address any impending overrun of time or budget can best be achieved through adequate project management (Wellington 2018; Castsoftware 2015; Keil et al. 2008; Schmidt et al. 2001; Ropponen & Lyytinen 2000). Also, researchers in software project management have proven that realistic and more accurate early risk estimates and fully integrated risk management into the process of software development could help software managers to reduce wasted resources that are associated with project cost and schedule overruns (Castsoftware 2015; Wallace et al. 2008; Keil et al. 2008; Schmidt et al. 2001; Ropponen & Lyytinen 2000). However, managers have divergent opinions on this. Some managers see risk management as additional responsibilities, which should go parallel with developmental activities (Mnkandla 2012; Bannerman 2008).

The researcher found that the effort of inducing high-tech in the software development projects as a measure to developing ‘*consistently*’ successful software projects cannot reliably address the impending overrun of time or budget. Also, research studies in project management have identified risks management in the early stages of project development as potential tool that can assist in minimizing project failure. However, in order to avert software project failure and minimize wastage of development resources associated with overruns, postponements, delayed and challenged projects, it is worthy of note that risks exist in all the stages (early, middle and later) of the software development process. Thus, a realistic and accurate risk management should be fully incorporated in all stages of the software development process from project planning through its design, execution and up to its deployment.

2.2 Concept of risk and risk management

Conceptually, at the project level, software projects in the today’s contemporary world are known to be high-risk ventures that have high failure rate. The high rate of failure in software project is a major concern to practitioners and other stakeholders in software projects. Project risk, according to PMBOK (2008) “is an uncertain event or condition which, if it occurs, has a positive or negative effect on one or more project objectives, such as scope, schedule, cost, or quality” (PMI 2008, p.194). Risk in software projects can be classified as generic and project-specific risk (Boehm & Ross

1989). The generic risks are risks common to all projects while the project-specific risks are risks that are associated with particular project. Large sections of these risks are not difficult to detect and manage. Some are more difficult to identify; thus, their occurrence and/or impact would be difficult to predict. The risk management efforts are also complicated by project multi-dimensional variables such as size, context composition long planning, structure, novelty, complexity and execution horizons.

2.2.1 Risk in software engineering (SE)

In SE, the term 'risk' is any form of situation or event, which can influence the outcome of a software development project (CAST 2015). Risk is the possibility of suffering loss or injury in some situations, and total risk exposure to a specific project is a function of both the *probability* and the *impact* in terms of loss.

The most common definition of risk in SE is in the form of risk exposure that is related to a specific factors that lead to undesired outcomes of a project. In this respect, risk in SE is usually defined as the likelihood of its occurrence and the effect on the project, if it occurs (Boehm1989; Charette 1996, 1989).

The mathematical definition of risk exposure is:

$$\mathbf{RE= Prbl} \; x \; \mathbf{Imp}$$

Where **RE** is the risk exposure related to a specific risk factor
Prbl is the likelihood that an unwanted event will occur, and
Imp is the severity of the effect of the undesirable event, if it occurs.

A close examination of the above risk concept shows that the conceptualization of 'risk' in SE may be narrower than the nature of the inherent problem in practice that is tenable in SE. For example, the definition in terms of probability of impact cannot adequately examine or assess what can go wrong (e.g. failure modes) in the software development process, much less being able to examine combination of conditions to fix the engineering flaws in the software development process.

2.2.2 Failure modes in software development process

A failure mode as described by the Wikipedia, the free encyclopaedia, is ‘the way in which a component failed "functionally" on the component level’ (“Failure cause” 2015). Relating this to software development process, a failure mode is a characterization of the manner in which a typical activity in the process fails
. This is because activities associated with high severity during the development process, and are having low possibility of occurrence, can be computed to yield low risk value factors and as so, they can be wrongly considered as factors with insignificant risk.

The problem of risk misconception makes it practically difficult for current risk assessment techniques to adequately identify the potential failure modes that exist in the software development process. Consequently, the effects of the failures associated with risk on project outcome remain unknown. It is therefore the aim of this study to modify the risk concept in SE with validated theory that will assist in assessing the effects of risk factors associated with potential failure modes in the software development process on the project outcome.

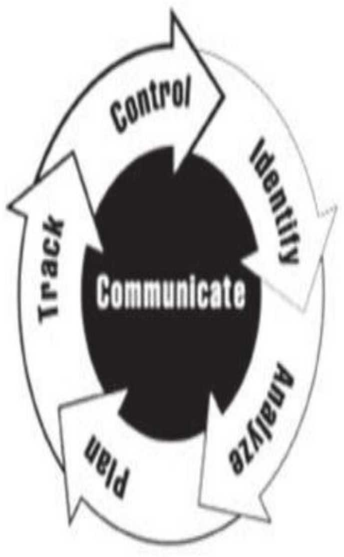
2.3 Concept of software risk management

Software risk management is a process, which provides an emerging disciplined environment for proactive decision-making to assess continuously areas with high threats, determine the relative importance of risks and analyse those risks that are important to deal with; and recommend plan actions to deal with those risks (SEI 2012). In contemporary world today, a leading success indicator for effective project management is the ability to manage risks more prudently and effectively. Risk management advocates and researchers (e.g., Wallace et al. 2004 & Schmidt et al. 2001) claim that proper risk management can assist the project manager to reduce wastages of development resources against both known and unknown risk factors on projects of all kinds (Wallace et al. 2004).

Wallace, Keil, and Rai (2004a) posit that risk in software projects is an aggregation of a number of factors or conditions, whose occurrences in projects may pose a serious threat to the project duration. Managing risk according to Huang and Han (2008) requires estimating its benefit, assessing its probability of occurrence and its likely effect on project quality, as well as the devising control strategies. The risk management process is generally accepted to be divided into two interwoven stages, i.e.: risk assessment and risk control. This gives a formalized approach to track, monitor and progressively control risks in order to achieve project success. The common perception in the above definitions is that risk management is a tactical, formalized, structured mechanism and continuous process, which is team-oriented, that is capable of napping project risks in order to ensure that projects failures are prevented.

2.3.1 Concept of continuous process of risk management

The continuous risk management process is a software engineering activity initiated by the SEI comprising the process and tools for risk management in software development project. The continuous risk management paradigm was proposed by SEI in 1996 as an improvement of the Barry Boehm's six steps to managing risk proposed in 1991. Figure 2.1 shows the paradigm as proposed by SEI.



Function	Description
Identify	Search for and locate risk before they become problems
Analyse	Transform risk data into a decision-making information. Evaluate impact, probability and timeframe, classify risks, and prioritize risks
Plan	Translate risks information into decision, and mitigating actions (both future) and implement those action.
Track	Monitor risk indicators from the risk mitigation plans.
Control	Correct for deviations from the risk mitigations plans.
Communicate	Provide information and feedback internal and external to the project on the risk activities, current risks and emerging risks. Note: communicate happens throughout all the functions of risk management.

Figure 2 1: Continuous risk management paradigm Curled from SEI (1996)

The paradigm depicts a formalized set of processes that are identified as necessary, which are continuous activities that must be practiced throughout the life cycle of a project to ensure project success (Joan-Pasto & Ramon-Roy 2005). Joan-Pastor and Ramon-Roy reaffirm that organization stands to gain huge benefits when adopting the continuous risk management paradigm in their risk management practice. Some of these include:

- Potential problems are identified, analysed and prevented before they occur
- Ensuring an uncompromised product quality
- Maximizing the use of developmental resources, and
- Promoting teamwork.

Owing to the benefits attributed with this paradigm, this proposed study will adopt the continuous risk management approach for the current research.

2.3.2 Risk management strategies

Two prominent risk management strategies are noticed from literatures. These are reactive and *Proactive* risk management strategies (Asadi 2015; Zardari 2009; Bannerman 2008).

The reactive strategist kicks into action once a problem occurs, or problems are identified after investigation. It is often compared to a crisis management or fire-fighting scenario. Here, investigations are conducted after the damage, and measures are taken to avoid similar problems from occurring in the future. Plans and further actions are taken to minimize the negative effects the problem could cause the project objectives and sustainability (Zardari 2009 & Bannerman 2008).

Contrary to the philosophy of reactive management strategist, proactive risk strategist seeks to identify all-important risks earlier, before a problem occurs. The objective is to minimize the probability of any form of problem from happening in future by identifying the boundaries of processes, where a breach of activity can lead to a problem (Asadi 2015; Zardari 2009; Bannerman 2008).

CAST (2015) reaffirms that reactive approach in form of guesswork; fire-fighting approaches are not effective means of managing risks. Rather, identifying, assessing, classifying and controlling risk are the effective predictive strategies for determining the probability that a software development project will experience an undesirable event. These include cancellations, abandonments, time and budget overrun, and overrun of project scope.

The above contributions reaffirm the need for software development teams and their managers to adopt a risk management method that will reliably identify, analyse and handle risk factors in all the stages of the software development process so as to the chances of project failures. Thus, the point of concern here is what risk management methodology can be adopted in the software development projects with scientific justification to reliably assess the effect of risks in the software development process on the project outcome?

2.3.3 Risk management in modern software development methodologies

Modern software development has evolved in a new dimension leading to many schools of thought to design best practices to execute a software development process. Several software development models are heard of: the Waterfall, Spiral, V-Model, Iterative, and some others as software development methodologies including the modern ones, Agile and DevOps, which are available in SE to help software developers to execute software development processes to develop software products (Svitla 2022). A software development methodology is a form of development paradigm consisting of log of tasks that assist developers in the development of software products. These logs of tasks result in a production of software products that explicitly explains the way in which the development process was carried out (Almeida, Simões & Lopes 2002). However, selecting the most appropriate software development methodology for the delivery of a particular software product requires mapping a unique approach to unique business needs, and strategically structures the future to develop a product roadmap that correlate with the needs of the client (Svitla 2022). This sub section introduces the most modern software development methodologies (Agile and DevOps) and examines their risk management competence.

Agile methodology

Agile development is an epitome of flexibility (Shore & Warden 2021). It is a development paradigm that requires much more communication and flexibility of development processes. It is an iterative development model that focused around swift delivery using adaptive planning, and always aspiring for continuous improvements thanks to staple resources such as the widely adopted Scrum (Svitla 2022). Agile methodologies arise from the need to overcome the uncertainty situations that are characterized by the application and implementations of traditional methodologies in project management. The 'constant feedback' to all members of Agile group, and the aspiration to work as a group as opposed to individualism and lack of communication exhibited by other paradigms make it a quality model (Tolfo, Wazlawick, Ferreira & Forcellini 2021). The Agile methodology presumes a minor gap between each delivery in order to make early and continuous delivery of software susceptible to evaluation a possibility (Shore & Warden 2021).

Going by project development history, the structured stages of Waterfall methodology works well, especially for assembly projects where planning, designing, building, testing, and delivering are effective. In modern software development, the reality is more straightforward and strategic. The current software industry need for a more flexible model that can quickly accommodate requirement change is what prompted the Agile paradigm to software development emerge (Svitla 2022). When comparing issues and advantages in agile and incremental development between state of the art and an industrial case, Petersen & Wohlin (2009) observed that software implementation in line with the Agile paradigm is interactive and incremental through small and more manageable procedures, ensuring all project artefact visible and well tracked, and enabling early confirmation of whether or not the delivered artefact meets the needs and making the respective corrections with low risk and cost. Thus, risk management is competently taken care of by Agile principles and procedures. This conclusion is also supported by the findings of Mnkandla (2012). In Mnkandla (2012), both XP and Scrum methodologies were proven to cater for project risk management.

DevOps Methodology

DevOps portends a change in mind-sets for IT culture by establishing trust and cohesion between developers and systems administrators. It is a combination of software development (dev) and operations (ops) (Luz, Pinto & Bonifácio 2018). In building on top of Agile paradigm, DevOps emergence became a crucial factor that transform this mind-set to a higher level. The intention is to ensure a culture of collaboration between the development team and the operations teams with the aim of enhancing the flow of completed work. The DevOps paradigm focuses on incremental development and swift delivery of software products. As defined by Luz, Pinto and Bonifácio (2018), DevOps software development paradigm merges the work of development teams and operations teams using a means of collaboration and shared responsibility. Others view it as a combination of principles, values, practices, methods and tools (Stahl, Mårtensson & Bosch 2017). The DevOps movement comes to break the traditional paradigm, whereof, no interaction between development teams and system operators. The overall motive behind its emergence is to improve the rate of deployments while increasing the stability and robustness of the production environment (Leite, Rocha, Kon, Milojicic & Meirelles 2019).

Beyond a cultural change, the DevOps methodology comprises of four major principles that guide its deployment and implementations. These are: Automation of the software development lifecycle (here, all the stages of development such as testing, development environment, releases, configuring infrastructure, among others are automated and other manual processes). DevOps has transformed the way software development and operations are developed in the modern world. Its emphasis on collaboration, automation, and continuous improvement has resulted in faster time to market, improved quality, enhanced efficiency, and increased innovation. (Rajapakse, Zahedi, Babar & Shen 2022; Leite, Rocha, Kon, Milojicic & Meirelles 2019). The main objective of DevOps methodology is to imbibe the culture of collaboration, automation and continuous process improvement between the development team and operation team to eliminate any form of disconnect between the two teams while ensuring swift delivery of software product in an agile, safe, and stable way (Rajapakse, Zahedi, Babar & Shen 2022).

DevOps has transformed the way software development and operations are developed in the modern world. Its emphasis on collaboration, automation, and continuous improvement has resulted in faster time to market, improved quality, enhanced efficiency, and increased innovation. By embracing the principles of DevOps, organizations can unlock the full potential of this transformative methodology and thrive in the dynamic landscape of software development and operations.

In summary, projects execution in traditional approach based on the waterfall model are undertaken in a linear way with several events in which their principles and practices are often silent about risk management. However, project development with modern engineering approach such as Agile and DevOps, are undertaken through interactive, incremental and successive progress (Larman & Basili 2003), some of which principles and practices provide guidance on how project risk management can be implemented in a given project (Mnkandla 2012).

2.4 Risk factor management in software development projects

This section presents the various steps and principles guiding an effective risk factor management in software development projects.

2.4.1 Risk Factors: Identification

In 1989, Barry W. Boehm launched a pioneering book in risk management titled: “Software Risk Management”, which laid a foundation for risk management as a field of research in software development projects. Since then, risk management in software development projects has become a priority of many researchers in software engineering (Sarigiannidis & Chatzoglou 2011). This breakthrough has spurned a successive network of activities, studies and thorough research from software development scholars and researchers, which are geared towards improving the quality of software development projects. A practical example is the implementations made by the Software Engineering Institute (SEI) in the early 1990s from the work of Boehm (1989, 1991) and Charette (1989, 1990), which have brought numerous contributions that even today acts as a referral in several risk management literatures (Sarigiannidis & Chatzoglou 2011). Previous studies on identification of risk factors in software projects are summarized in Table 2.4.

Table 2 4 :Research on risk factors in software development projects

Checklist	Number of software risk	Dimension of risks	Sources
Boehm (1991)	10	0	A Survey research of experienced project managers
Barki, Rivard et al., (1993)	23	5	A Systemic literature survey
Heemstra & Kusters (1996)	36	9	A combination of Literature survey with professional experiences
Moynihan (1997)	21	4	Interviews of experienced software developers
Cule et al., (2000)	38	4	A Delphi study of experienced project managers
Schmidt, Lyytinen et al., (2001)	53	14	A Delphi study of experienced project managers
Wallace, Keil and Rai (2004)	101	6	comprehensive literature review and rigorous instrument development process with project managers
Hang and Huang (2007)	27	6	Analysis of 115 software projects
Pare et al., (2008)	19	4	A Delphi Study of clinical risk factors information system
Keshlaf & Reddle (2010)	52	2	Survey of web distributed software development project risk
Arnuphaptrairong (2011)	Top Ten	7	Literature survey
Hijazi, et al., (2014a)	110	1	Survey of Literature
Menezes, et al., (2018)	148	1	A systematic literature review
Bilal et al. (2020)	83	1	Survey based approach and structured questionnaire

The general approach to curtail and to minimize the likelihood of an undesirable project outcome in software projects is that all potential risk factors should be identified within a specified dimension (Barki, Rivard & Talbot 1993) analysed using a grounded method at the start of the project (Schmidt et al. 2001). The individual risk factor is then assessed using a suitable risk theory (Keil et al. 2003) and the estimated risks are then prioritized to identify the risks that pose the highest threat to the project outcomes (Iversen, Mathiassen & Nielsen 2004). Risk managers can then focus their attention more on the risk factors with the highest risk values to minimize the likelihood of their occurrence and/or the effect of the threat using a more effective control strategy (Wallace et al. 2004).

The contribution of Barki et al. (1993) towards an effective project risk management suggests a multidimensional approach to address the multidimensional nature of occurrence of risks factors in the SDLC. Barki et al. also, recommend an alternative approach, in which, “every risk should be separately defined and theoretically investigated as well as empirically analysed” (Barki et al. 1993). Keil et al. (2008) posit that applying multidimensional approach to address software project risk factors can bring a clear direction for research and practical purposes. However, despite the fact that previous studies (e.g., Barki et al. 1993) highlighted benefits that can result from dimensioning risk factors in software projects, research in this area this is not detail enough to explore the research opportunity in this direction.

There have been interesting studies in the domain of risk management in software development projects that are reported from literatures. Most of these researches focused mainly on risk factors associative dimensions in software development process as suggested by Barki et al. (1993), to help improve existing risk management practices. For example, McFarlan makes a list of 20 risk factors in software development projects based on three major dimensions (project size, technology experience and project structure). Cule et al. (1998) identified 55 risk factors in software projects that are associated with four dimensions (task, client, environment, self) and suggest corresponding risk management control for each of the risk item. Houston (2000) in his PhD work conducted a literature review to develop a list of 29 software risk items, which the researcher considers as the major risks that have been widely investigated in similar studies.

The work of Schmidt et al. (2001) and Wallace et al. (2004) have contributed significantly to research on risk factors because of their distinctive and wide coverage, theoretical substantiation and their susceptive approach used to identify and for classifying the risk factors in the research. Even today both kinds of research possess domineering referral archetype in software development project risk management literatures. Schmidt et al, (2001) developed a list of 53 risk factors that were associated with 14 different dimensions, of which 11 have cross-cultural application factors. Wallace et al. (2004) identified 27 risk factors that are associated with six dimensions.

There are other studies on risk factors (e.g., Bilal et al. 2020; Menezes et al. 2018; Pasha, Qaiser & Pasha 2018, Vahidnia et al. 2016) that focused on risk that are generally associated with software development projects. Some were conducted over the phases of the SDLC (e.g., Hijazi et al. 2014a & 2014b) to garner an elaborate list of software related risk to support software practitioners' risk management experiences. However, earlier efforts from these researches did not specifically describe what factor combination and ordered events would possibly lead to an effective risk management in the software development process. While the specifications and assessment of risk factors are in various dimensions, which most previous research bemoaned, they do not provide them with useful information that will assist them to hypothesize a tailored action plan for countering the threats posed by the potential failure modes across all stages of the software development process.

The study of Menezes et al. (2018) extracted and classified risk factors from literatures according to the taxonomy developed by SEI. Menezes et al. (2018) identified and classified 148 risk factors and they found evidence based on the study suggesting that risk factors relating to software requirements are the recurrent and cited. In addition, analysis of the study findings shows that the most mentioned risk factors were the lack of technical skills by the staff. Furthermore, Bilal et al. (2020) used a survey-based approach and a structured questionnaire to discover 83 risks factors across the SDLC for small and medium software projects in Pakistan software industry. The study found that majority of risks associated with small and medium software projects in Pakistan software industry appeared to be significant and high.

The contribution of these studies emphasises the need for more research studies on risk factors in software development projects as fundamental panacea to perennial failure rate of software development projects.

2.4.2 Risk factors in the software development process

Software development process is the series of identifiable stages that a software product undergoes during its lifetime (Guarav & Pradeep 2014). It is a process used by software development organizations to design, develop and maintain or upgrade specific software. The framework contains the detailed plan to be followed by software industries for software projects. The software development process is aimed to develop high quality software within the appropriated budget that is completed within the right period, which must meet customers' expectations.

Risk factors in the software development process are agents of uncertainties in the software development process (Hijazi et al, 2014a). The software development process is susceptible to different multidimensional risk factors, which are discussed extensively in section 2.4.3. The most referred dimension in literatures according to the Menezes et al. (2018) is planning and control. However, other associative dimensions of risks exist in the software development process, which are not reported in literatures (e.g., risk factors associated with potential failure modes).

The ISO/IEC 12207 is an international standard (ISO/IEC 1995) for software development process, which describes a common framework that all software development experts can follow to develop and maintain software. The standard describes the whole conception from the planning stage through development to testing, operation and maintenance processes. Figure 2.2 presents the graphical description of the ISO/IEC 12207 stages of the software development process.

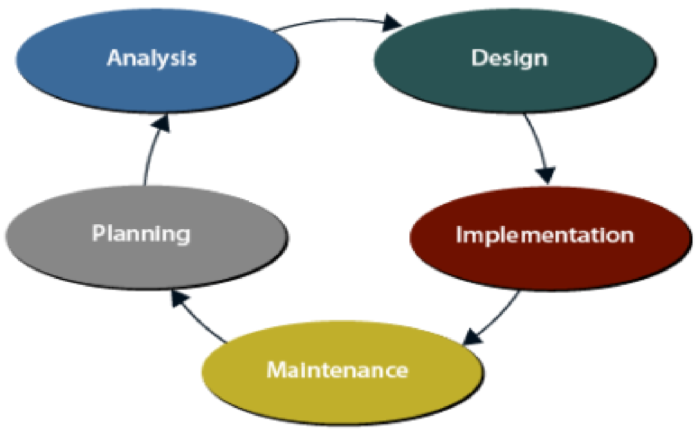


Figure 2 2: Stages of software development process/SDLC

Stages of software development process

A typical software development process consists of the following stages:

1. Planning stage:

This stage is the first step of any software development project (Georgieva 2010). Here, the project team examines all the required tasks (it is in the form of planning and scheduling) to plan and manage a software development process. The main concept of planning phase surrounds the feasibility studies on areas such as: project planning, timing, risk management, team composition, and responsibility designation amongst team members, establishing the technology priority scale preference and application selection and project costing (Hijazi et al. 2014a). Hijazi et al. also state that both the quality assurance specification and the identification of risks that are associated with the proposed project are done at the planning stage.

2. Requirement analysis stage:

In this stage, the project team engages in intensive discussions with the stakeholders from the project problem domain to come up with detail information that has the capacity of developing the required solutions to the problem. The main goal of this stage is for the developers to be able to clearly characterize problems, which the system intends to solve (Georgieva 2010). Other specifications are also described (such as safety and reliability behaviours).

The requirements are gathered and further analysed to be categorized as either user, functional or system requirements (Georgieva 2010).

The requirements are gathered using the following practices:

a. Conducting a review of the existing system and software
b. Creating analysis and needs of the system by interviewing the users and other stakeholders
c. Analysing answers collected from interviews or questionnaires

3. Design stage:

Activities of the design stage transform the system requirement specified into architecture. The focus of this stage is to ensure that all features and operations configurations that are attributed to the software are detailed and correctly assigned and documented to facilitate several design approaches. The development team and other important stakeholders using certain criteria such as risk factors management, cost, schedule constraints, modularity, etc. to select the best design approach for the product then review this design document.

Activities that are undertaken in this stage according to (Hijazi et al. 2014a) are:

- Scrutinize the operation relationship between objects and functions on the system.
- Data and database schema are analysed and created
- User interface is designed

4. Implementation and coding stage:

In this stage, the actual system development commences and the product is developed. This goal is actualized by implementing the design conception as fashioned out in the design phase.

To implement the design conception as listed in the design phase, coding guides and principles must be strictly adhered to by the developers using the appropriate programming tools. These tools may include but not restricted to debuggers, case tools, compilers and interpreters.

During the implementation stage, the following events followed:

- Development of the database in line with the design scheme
- Development of application systems in line with the system design
- Debugging process in form of testing and upgrading the application developed.

5. Installation and maintenance stage:

The goal of this stage is to ensure that the developed system is able to function adequately in the stipulated environments. The feedback received after the product deployment will suggest whether to display the product in the current form or to improve on its current functions. After the deployment, maintenance can then be introduced.

In conclusion, the software development process (also known as SDLC) is a series of related activities, which are bundled in phases that often leads to the development of the software. The five stages of the software development process recommended by ISO/IEC 12207 for developing software are: planning, requirement specification analysis, design, implementation and coding and installation and maintenance. The software development process is susceptible to different dimensions of risk factors, which are discussed in sections 2.4.2.1 and 2.4.2.2. However, this study will focus on risk factors associated with potential failure modes in the software development process.

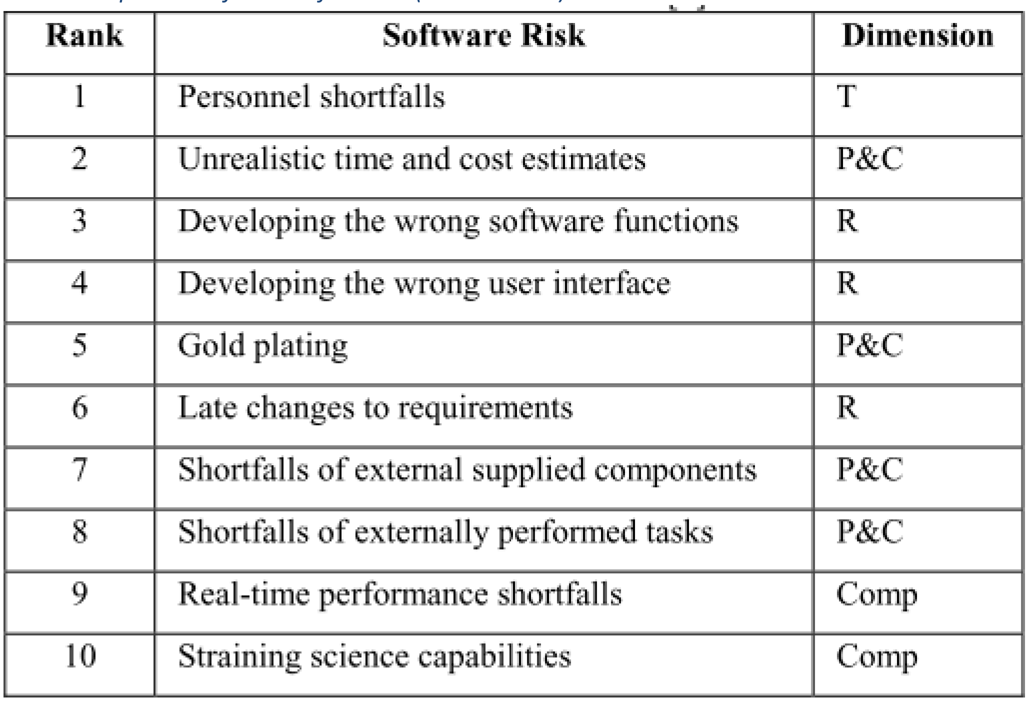
2.4.3 Risk factors: Management

There are four main methodologies found in literature and practice for the conduct of software risk factors management. According to Bannerman (2008), the four risk factors management methodologies found in literatures can be classified as: *checklists*, *analytical frameworks*, *process models*, and *risk response strategies*.

2.4.3.1 The check lists approach

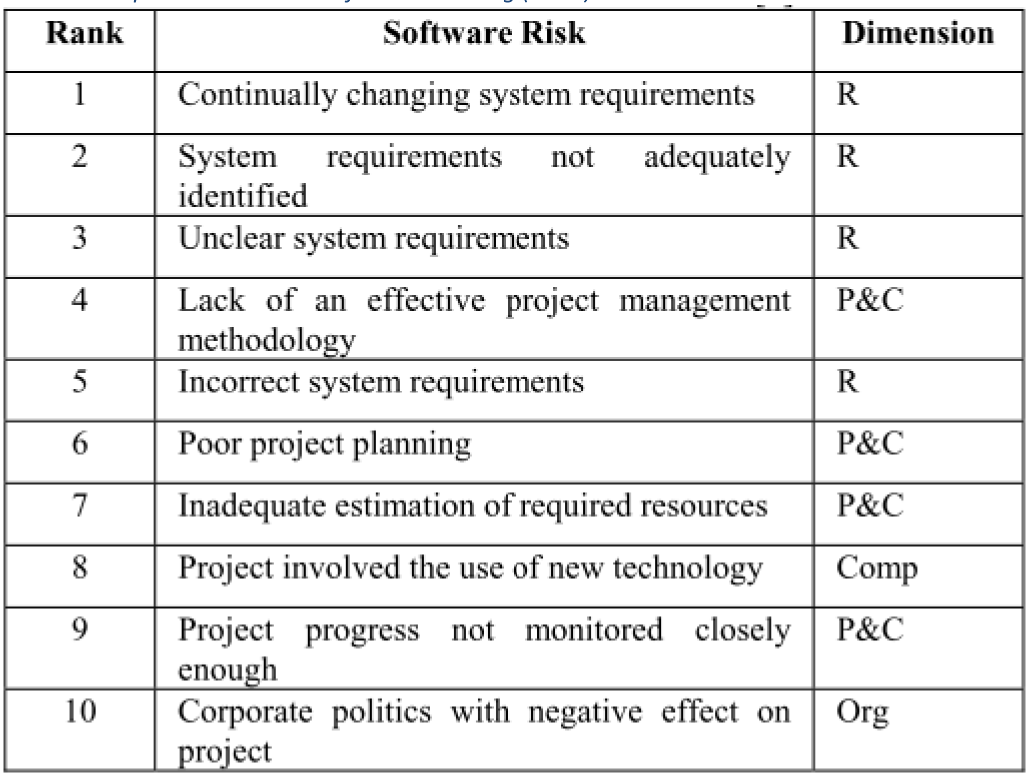
The checklists approach is concerned with the provision of a list of top ranked risk factors that can influence a project (Iversen et al. 2004). The developed list of risk items assists a project manager pay absolute attention to all potential risk sources. The rational of checklist researchers is to develop a list and use the developed list as a template to review other similar projects. This is to ensure that every identified risk in the project is adequately monitored (Bannerman 2008). Various research on risk factors in software projects have been reported in literatures which cut across different dimensions to facilitate risk identification as shown in tables 2.5 to 2.10. The associated dimensions are: User (U), Requirement (R), Complexity (C), Planning and Control (P and C), Team (T) and Organizational Environment (OE).

Table 2 5: Top ten risk factors of Boehm (Boehm 1991)



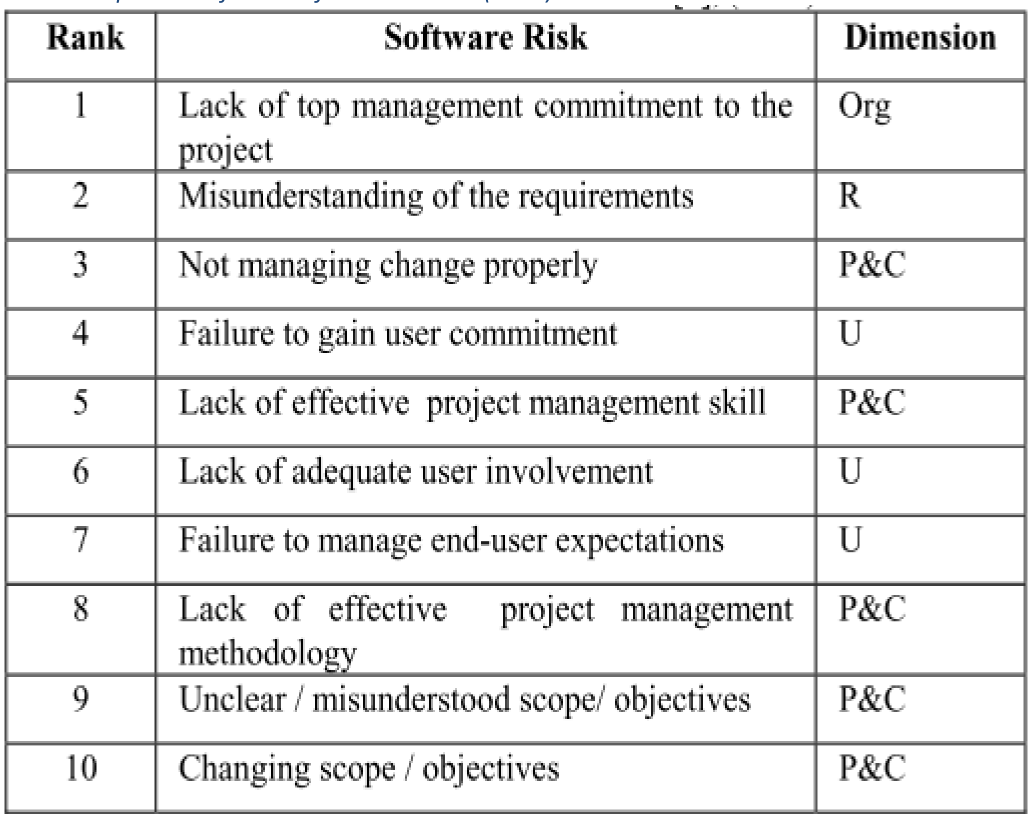
Rank	Software Risk	Dimension
1	Personnel shortfalls	T
2	Unrealistic time and cost estimates	P&C
3	Developing the wrong software functions	R
4	Developing the wrong user interface	R
5	Gold plating	P&C
6	Late changes to requirements	R
7	Shortfalls of external supplied components	P&C
8	Shortfalls of externally performed tasks	P&C
9	Real-time performance shortfalls	Comp
10	Straining science capabilities	Comp

Table 2 6: Top Ten Risk Factors of Han and Hung (2007)



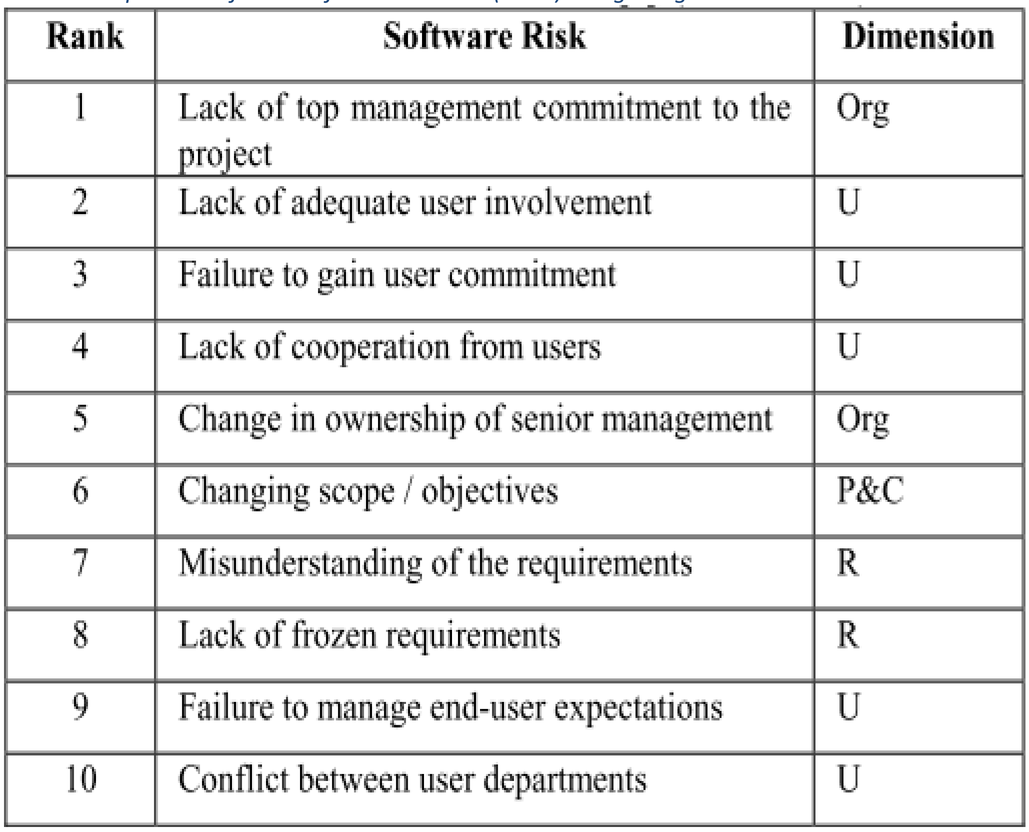
Rank	Software Risk	Dimension
1	Continually changing system requirements	R
2	System requirements not adequately identified	R
3	Unclear system requirements	R
4	Lack of an effective project management methodology	P&C
5	Incorrect system requirements	R
6	Poor project planning	P&C
7	Inadequate estimation of required resources	P&C
8	Project involved the use of new technology	Comp
9	Project progress not monitored closely enough	P&C
10	Corporate politics with negative effect on project	Org

Table 2 7:Top ten risk factors of Schmidt et al. (2001) USA



Rank	Software Risk	Dimension
1	Lack of top management commitment to the project	Org
2	Misunderstanding of the requirements	R
3	Not managing change properly	P&C
4	Failure to gain user commitment	U
5	Lack of effective project management skill	P&C
6	Lack of adequate user involvement	U
7	Failure to manage end-user expectations	U
8	Lack of effective project management methodology	P&C
9	Unclear / misunderstood scope/ objectives	P&C
10	Changing scope / objectives	P&C

Table 2 8:Top ten risk factors of Schmidt et al. (2001) Hong Kong



Rank	Software Risk	Dimension
1	Lack of top management commitment to the project	Org
2	Lack of adequate user involvement	U
3	Failure to gain user commitment	U
4	Lack of cooperation from users	U
5	Change in ownership of senior management	Org
6	Changing scope / objectives	P&C
7	Misunderstanding of the requirements	R
8	Lack of frozen requirements	R
9	Failure to manage end-user expectations	U
10	Conflict between user departments	U

Table 2 9:Top ten risk factors of Schmidt et al. (2001) Finland

Rank	Software Risk	Dimension
1	Lack of effective project management skill	P&C
2	Lack of top management commitment to the project	Org
3	Lack of required knowledge /skill in the project personnel	T
4	Not managing change properly	P&C
5	No planning or inadequate planning	P&C
6	Misunderstanding of the requirements	R
7	Artificial deadlines	P&C
8	Failure to gain user commitment	U
9	Lack of frozen requirements	R
10	Lack of "people skill" in project leadership	T

Table 2 10:Top ten risk factors of Pare et al. (2008)

Rank	Software Risk	Dimension
1	Lack of project champion	T
2	Lack of commitment form upper management	U
3	Poor perceived system usefulness	U
4	Project ambiguity	P&C
5	Misalignment of system with local practices and process	P&C
6	Political games or conflicts	Org
7	Lack of required knowledge or skill	T
8	Changes to membership on the project team	T
9	Organizational instability	Org
10	Insufficient resources	P&C

The above serial studies reeled from literature presented in tables (2.4 to 2.9) consider software risks along several dimensions (User, Requirement, Complexity, Planning and Control, Team and Organization) and have provided some empirically founded insights of typical software risks and risk management strategies to mitigate them.

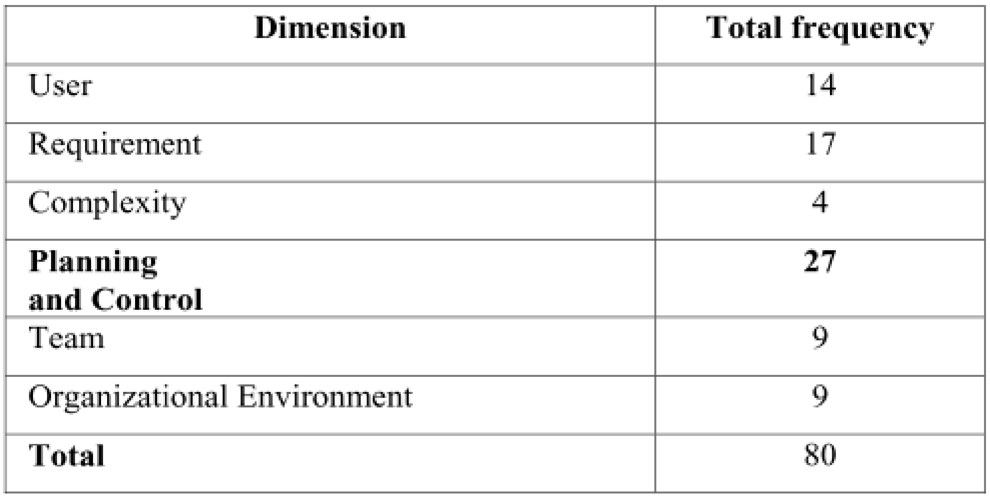
Above all, these studies provided detail features of risk management deliberations, but cannot adequately describe the direct relationship that exists between the risk factors and risk management processes on the project outcome (cost, time and quality). In addition, the problem associated with this approach is that most of the available list of risk items reported may not adequately cover the specific project of contention. Also, the lists only prescribe risk factors but are found short of an adequate action plan to counter the effect of the threat posed by the risk items. Above all, research in these areas is weak in exploring the relationship that exists amongst the project risk factors, the risk management and the project outcome.

2.4.3.2 **Analytical frameworks**

This approach is also referred to as non-process based analytical framework for managing risk. The approach is similar to a checklist. It is based on categorizing some risks found in the checklist to have same causality relationship into specific dimensions in order to devise an effective means of identifying and managing the risk (Cule et al.2000). This approach provides a better alternative to checklist management in the sense by applying an individual management measure to one or more categorized risks, it will rather be more efficient and cost effective than treating each individual risk factor (Addison & Vallabh 2002).

In a previous review of the top ten risk checklist which spanned across about 80 dimensions reported from software project literature, and which was conducted by Arnuphaptrairong (2011) shows that *planning and control, requirement and user* are the three most mentioned dimensions in the literature as shown in the Table 2.11.

Table 2 11: Risk factors' associated dimension appearance in literature survey 1991-2011



Dimension	Total frequency
User	14
Requirement	17
Complexity	4
Planning and Control	**27**
Team	9
Organizational Environment	9
Total	80

Similarly, risk factors associated with the technical complexity was found to be reported as the least important. The implication of this study is that across all research concerning risk factors in software development projects reviewed, project managers advocated that risk factors associated with the three earlier mentioned dimensions are the most important risks. In other words, managers should pay more attention to these risks factors when working on any software project.

Another very important observation of this research is that in all the eight top ten software project risk factors reviewed (Table 2.2 to Table 2.9), none of the studies took cognizance of risk associated with potential failure modes in the software development process. The researcher found most of the risk factors research and risk management to be mainly quantitative; which only probed the risk nature from the surface and did not investigate qualitatively the cause-effect factors of the identified risks as illustrated in Tables 2.2 to 2.10. It is clear from these findings that a more detailed study that will conduct the cause-effect analysis of these risks would go a long way in estimating the risk magnitudes and assessing their effects on project outcome.

2.4.3.3 Risk management frameworks and standards in software projects

The standards and frameworks that are internationally recognized for the conduct of risk management in software projects include: the ISO 3100 (ISO 2009); MPS.BR (SOTEX 2006); the Capability Maturity Model Integration (CMMI) developed by the

Software Engineering Institute (SEI 2006); the PMBOK 4th edition developed by Project Management Institute (PMI 2004); the AS/NZS 4360 standard (Standard Australia & Standards New Zealand 2004); the Rational Unified Process (RUP) developed by IBM (IBM2003); the ISO 10006 guidelines (ISO 2003); the Microsoft Solutions Framework (MFS) developed by Microsoft (Microsoft 2002); the ISO/IEC 15504-5 standard (ISO 1999) and the Boehm's list of software risk items (1991).

Figure 2.3 presents the chronology of the standards frameworks that are internationally recognized for the conduct of risk management in software projects, which was originally presented in Gusmao (2007) and updated by Sandra and Carlos (2016).

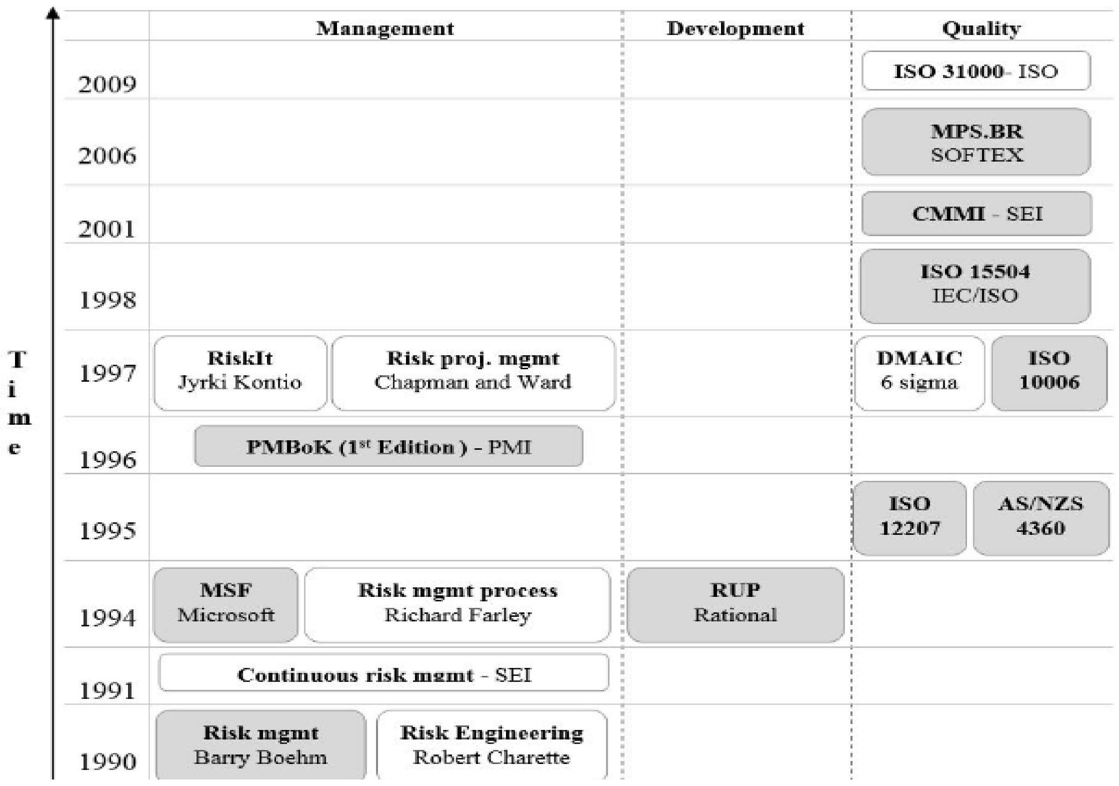
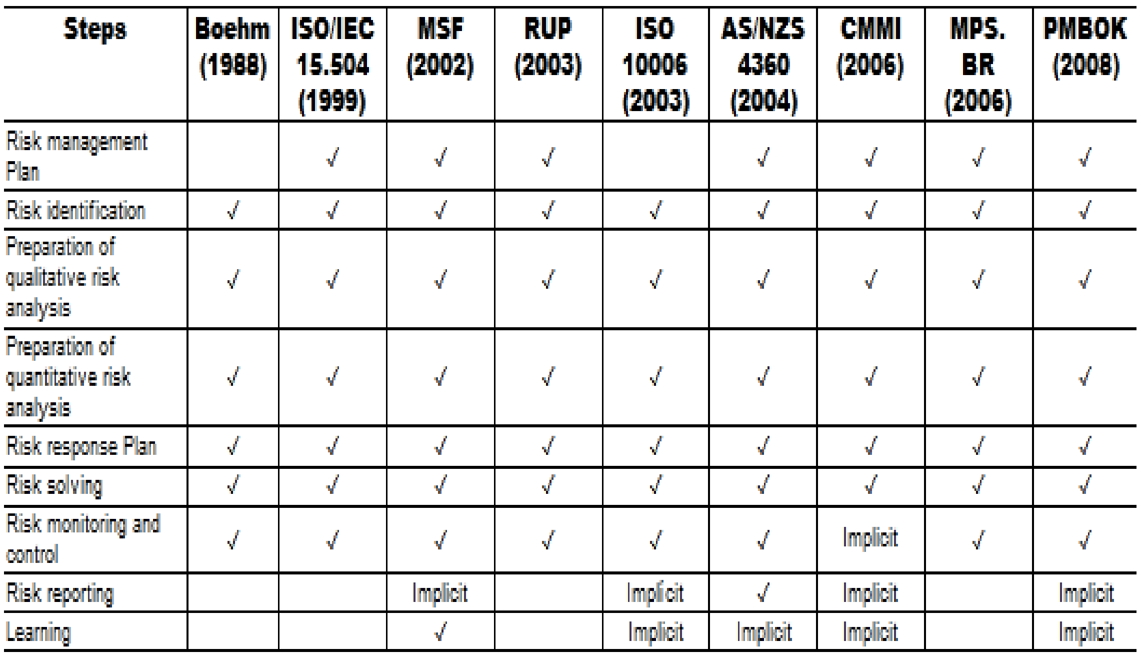


Figure 2 3: Chronology of frameworks for software project risk management (Sandra & Carlos Eduardo (2016))

A comparative analysis of the frameworks for software project risk management was carried out by Neves, Silva, Salomon, Silva and Sotomonte (2014) in order to facilitate the visualization of the steps required for risk management in software project. Neves, et al., combined the steps of the project risk management knowledge area, described by the PMBOK Guide (PMI 2008) and the steps of "solving risks", "communicating risks", and "learning", and selected from other frameworks of the AS/NZS 4360 standard (Standard Australia & Standards New Zealand 2004) and the Microsoft Solutions Framework (MFS) developed by Microsoft (Microsoft 2002) as the basis for the comparison.

The Neves et al.'s comparative analysis of software project risk management frameworks is presented in Table 2.12.

Table 2 12: Comparison of software project risk management frameworks (Source: Neves et al. 2014))



Steps	Boehm (1988)	ISO/IEC 15.504 (1999)	MSF (2002)	RUP (2003)	ISO 10006 (2003)	AS/NZS 4360 (2004)	CMMI (2006)	MPS. BR (2006)	PMBOK (2008)
Risk management Plan		√	√	√		√	√	√	√
Risk identification	√	√	√	√	√	√	√	√	√
Preparation of qualitative risk analysis	√	√	√	√	√	√	√	√	√
Preparation of quantitative risk analysis	√	√	√	√	√	√	√	√	√
Risk response Plan	√	√	√	√	√	√	√	√	√
Risk solving	√	√	√	√	√	√	√	√	√
Risk monitoring and control	√	√	√	√	√	√	Implicit	√	√
Risk reporting			Implicit		Implicit	√	Implicit		Implicit
Learning			√		Implicit	Implicit	Implicit		Implicit

Result analysis of comparison data presented in Table 2.12 shows that all the studied risk management frameworks have the same context. Neves et al. posit that some of these frameworks have higher level of adherence to the steps (e.g., PMBOK, (PMI 2008) and CMMI (SEI 2006). However, other frameworks are perceived by Sandra and Carlos (2016) to have similar compliance with AS/NZS 4360 (2004) standard but do so implicitly.

The above analysis shows that frameworks of PMBOK, SEI and Boehm are the prominent frameworks and this is unconnected to their explicit compliance with all requirement steps for conducting project risk management. This is also reinforced by the review of literature that returned Boehm and SEI having the highest referral in all the reviews conducted on risk management studies. Thus, these three prominent frameworks will be the basis of comparison for the method used for the risk management in this study.

The following sections review briefly the three prominent risk management frameworks that are in practice in software engineering (I.e., PMBOK, SEI & Boehm).

2.4.3.3.1 Boehm risk management model

In his contribution, Boehm (1991) developed a risk analysis model by applying a decision tree approach. This model involves a two-step approach: risk assessment and risk control as presented in Figure 2.4. The Boehm's model is presented in Figure 2.4.

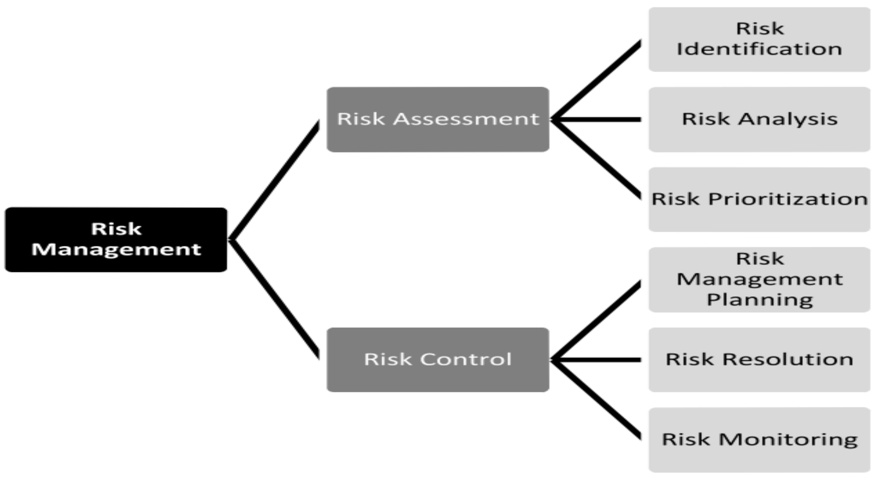


Figure 2 4: Risk management steps (Boehm 1991)

The Boehm's risk assessment comprises identification, analysis and prioritization while the risk control component is divided into planning, reduction and monitoring, (this is illustrated in Figure 2.3).

The explanation of each step according to Boehm (1991) is summarized as follows:

Risk Identification: Finding possible sources of risk that might influence a project and develop awareness of the specific risks associated with a project.

Risk Analysis: Encompasses the identification and characterization of potential risk mechanism in the system under development.
Risk Prioritization: Involves the process of consideration of the identified risk based on their relative importance.

Risk Management Planning: Involves the management of risk by developing a risk control strategy that will prioritize implement, and maintain the system.

Risk Resolution: Involves the control process that could minimize or eliminate the identified risks.

Risk Monitoring: Involves appropriate mechanism that is put in place for the risk-reduction process.

The benefits of Boehm's model over other risk management models are its simplicity and its scope in all the stages of the software development process. However, the limitation of Boehm's model is that it can only be implemented explicitly on project specific risk.

2.4.3.3.2 Project management body of knowledge (PMBOK)

PMBOK (PMBOK® Guide (PMI 2000)), by the Project Management Institute (PMI), is a project management guide that addresses project risk management activities in the 11th chapter of the guide. By this guide, risk management should follow a set of processes, which are:

• *Risk Management Planning*: The three risk management plans identified by the PMBOK are methodology, roles, responsibilities, and budget.

• *Risk Identification*: This involves the establishment of important risk that can determine the success of a project.

• *Qualitative Risk Analysis*: This is the subjective assessment of the impact and the probability of the identified risk.

• *Quantitative Risk Analysis*: This is the computation of the impact and the estimated probability for the identified risks.

• *Risk Response Planning*: This involves execution of recommended action to reduce the effects of the identified risks.

• *Risk Monitoring and Control*: These are a set of mechanisms that span across the entire project due to constant risk variation during the project life-cycle (PMBOK® Guide (PMI 2000)).

The feature of the PMBOK model that makes it unique among other risk management models is that its frameworks addresses the steps required for project risk management in more detail than others (Sandra & Carlos 2016). However, ratings in PMBOK model are assigned in terms of monetary values rather numerical scale. Although naturally, assigning rating by financial values can continue to drive the analysis, the use of scale makes risk analysis activities much easier (Flávio & Sandro 2008).

2.4.3.3.3 Software engineering institute (SEI)'s software risk management (SRM) methodologies

The SEI-SRM model is a very efficient risk management tool used by software development companies. The risk identification method of SEI-SRM model is more detailed than any other risk management models because it applies standard and well-tested approaches.

The SEI-SRM model for software risk management is supported by three groups of practices:

1. Software Risk Evaluation (SRE)
2. Continuous Risk Management (CRM)
3. Team Risk Management (TRM)

This framework enables software engineers, project managers and their team members to identify early enough, risk functions in software development projects; so that a well-coordinated risk management and contingency actions can be planned and executed within a specified period. The SEI Risk Management framework is depicted in Figure 2.5.

Control

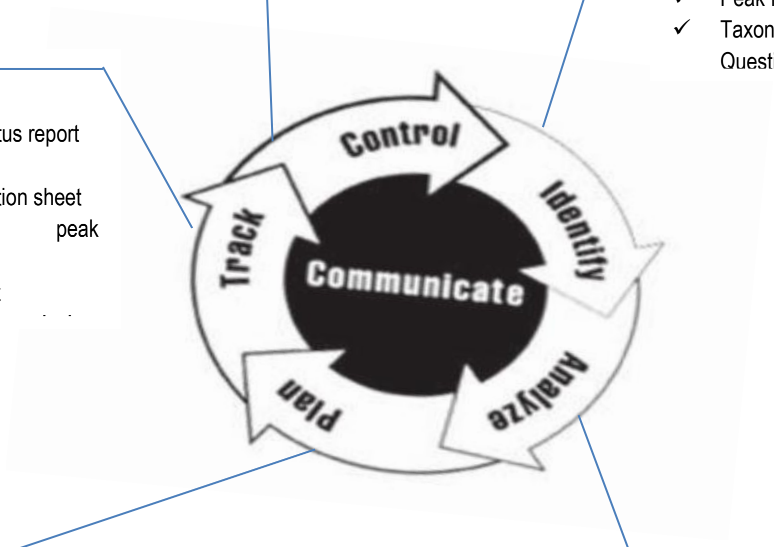
- ✓ Cause and Effect Analysis
- ✓ Change a peak
- ✓ PERT change
- ✓ Motivating
- ✓ Problem solving planning
- ✓ Peak information sheet
- ✓ Spreadsheet peak

Identify

- ✓ Brainstorming, Identification and analysis.
- ✓ Brainstorming
- ✓ Periodic peak
- ✓ Project profile questions
- ✓ Peak information sheet
- ✓ Peak form
- ✓ Taxonomy-Based Questionnaire(TBQ)

Track

- ✓ Mitigation status report
- ✓ Bar graph
- ✓ Peak information sheet
- ✓ Spreadsheet peak tracking
- ✓ Plight chart



Plan

- ✓ Action item list
- ✓ Planning flowchart
- ✓ Planning worksheet
- ✓ Affinity grouping
- ✓ Brainstorming
- ✓ Cause effect analysis
- ✓ Cost benefit analysis

Analyze

- ✓ Item grouping
- ✓ Bar cock
- ✓ Brainstorming, identification and analysis
- ✓ Binary evaluation
- ✓ Motivating
- ✓ Peak form
- ✓ Peak information sheet
- ✓ Peak top N

Figure 2. 5: SEI risk management model (source: SEI 1999)

The framework presents a set of activities that are represented as continuous process throughout the lifespan of a project. The main advantage of SEI-CMM model is that it is practically oriented and not purely theoretical. There is no reported case where the model was unsuccessful. In addition, the SEI-CRM model was developed by institute that is software oriented, and can therefore be used in any software projects. The disadvantages in SEI-CRM method are that the risk management template of SEI-SRM is meticulously implemented as designed by the institute. Thus, it will be difficult to implement the framework to cover projects that have specifications outside the SEI template. Also, the risk management outcome of SEI-CRM model may be inconclusive when assessed by the SRE team.

The Capability Maturity Model (CMM) of the SEI has been criticized for being too process-oriented and not goal-oriented enough, thus, underwent some improvements. Jones (1994) and Gerald (1993) are two noteworthy critics of the CMM. Software organizations have found it difficult to tailor the CMM to specific goals and needs. Thus, the emergence of a newer and improved version called the Capability Maturity Model Integration (CMMI). The SEI developed the CMMI to integrate and standardize the CMM, which has different models for each function it covers. These models make the software development process more efficient and flexible. Another improvement introduced on to the CMM is the integration of Agile culture into the development processes to improve the management configuration and the quality product. This is implemented in parts using a pattern of continuous improvements and feedback into the software development process.

Several versions of CMMI have been released by the SEI. The SEI released the first version of the CMMI in 2002. In 2013, the CMMI Institute was formed to take charge of the CMMI services and future model development. The latest version -- CMMI V2.0 – was launched in 2018. It aims at establishing business objectives and tracking those objectives at every level of business maturity.

2.4.3.4 Risk response strategies

Risk response strategies are the various responsive methods applied in creating strategic alternatives, and applying appropriate actions, to enhance opportunities and minimize threats to the project’s business objectives. For example, the risk: *unrealistic requirement* can be avoided by applying standard requirement engineering principles to capture vital system requirements (Abubakar & Lawal 2020; Hijazi 2014a). In addition, risk transfer strategy can be applied to mitigate the threat that can result from the risk “Ambiguous or unclear Design Document” by incorporating skilled developers to participate in part of system documentation activities (Hijazi et al 2014a). Others are best mitigated by accepting the risks when it is sure that none of the other risk strategies can work or when risk response is not warranted due to the severity of the risk (Abubakar & Lawal 2020). In all, five response strategies are common in all literature reviewed and are summarized by Bannerman (2008) as follows:

a. *Avoid:* Risk can be avoided by either countering the causative factor or by implementing the project in a different way and still ensure that the project achieve its business objectives. However, it is not all risks that can be avoided, removed or eliminated. Using this approach may be time demanding and may be too costly to apply. All the same, it is a highly recommended strategy to be considered before any other one.

b. *Transfer:* This strategy involves shifting of responsibility for the risk management to other stakeholder who will be responsible for the risks should it occur. This is to ensure that the risk is handled by the best handler. This strategy usually goes with payment of a premium, and necessary decision should be taken on its cost effectiveness before embarking on the risk transfer strategy.

c. *Mitigate:* This strategy reduces the chance and or the effect of risk scenario to an acceptable threshold. Taking a proactive measure to minimize the chance and or the impact of risk is usually more effective than to control it after occurrence. This strategy is time consuming and may require other resources. It is however better to mitigate so as to minimize the risk that emanates.

d. *Accept:* This is a strategy that is adopted when it is sure that none of the other risk strategies can work or when risk response is not warranted due to the severity of the risk. When a decision is made by the project manager and the project team to accept risk, such an agreement is without any change to the project. A contingency plan or work around plan may be put in place in case of any eventualities.

The above risk response strategies consider risk response alternatives based on attributes of risk principles and practice. The strategies lack the capability of planning control action for issues of risks described with their values. Thus, there is the need to develop and implement response strategies to risk using risk values in compliance with relevant safety standard.

2.4.4 Risk assessment techniques for software projects

Risk assessment in software projects involves the process of identifying, analysing, prioritizing and estimating the risk factors in software projects (Boehm 1991). Risk assessment process can be actualized using a variety of techniques. The following are the existing software project risk assessment techniques that are found in the literature as enumerated by Taroun (2012): Fuzzy Sets Approach (FSA); regression analysis; Decision Tree and Fault Tree Analysis; Monte Carlo Simulation; Cause-And-Effect Diagram; Delphi Technique; Multiple Criteria Decision Making (MCDM); Neural Networks; Influence Diagrams; PERT; expert systems; Sensitivity Analysis; FMEA, combination of FMEA and FTA.

Individual analyses of these techniques are not included in this research. However, review of previous studies conducted to analyse the strength and weaknesses these techniques (e.g., Taroun 2012; Abdelgawad 2011; Georgieva 2007) reveal that some of these techniques can only be used for qualitative assessments. That is, in most cases, these techniques can only conduct risk assessment by ranking, using rating scores or numerical values rather than quantifying the risk magnitude. Thus, the conduct of quantitative risk assessment practically becomes impossible for risk managers.

2.4.4.1 Security risk analysis for web applications (SRAWA)

Another important software risk assessment framework called SRAWA was developed to address the need for more effective risk management. SRAWA is a risk assessment framework developed by Dimitrakos and Ritchie in 2002. It was built on prominent reliability engineering techniques such as: (1) Hazard and Operability study (HAZOP) (2) Fault tree Analysis (FTA) (3) Failure Mode and Effect Critically Analysis (FMECA) and (4) Markov risk control strategies. The SRAWA Model was developed for Security Risk Analysis for Web Applications using CORAS 172 approach. It is important to note that the reliability and safety engineering techniques introduced in this approach enhances the confidence of this approach.

The overall concluding remark of the findings from these sections is that most of the available risk assessment frameworks do not cover in detail the required steps recognized for formal project risk assessment, hence may not be capable of assessing generic risks (Taroun 2012). Some of these techniques are not validated before use. The implication of this is that any alteration of the risk scenario will require recalibrating the risk assessment process from the start. In addition to this, some of the assessment techniques do not support the initial risk identification processes (Abdelgawad 2011) and some support early stage development assessment but cannot conduct risk assessments in all the stages of software development process (Georgieva 2007). Thus, there is a pressing need to develop a comprehensive and more effective risk assessment framework that will address the limitations of previous studies.

The next section discusses the reliability and safety engineering techniques as risk management tools and its applicability in SE.

2.5 Reliability and safety engineering techniques in SE

Reliability and Safety Engineering Techniques (e.g., FMEA, FTA and HAZOP) are well-known risk analysis tools with an established position in traditional reliability analysis. These are widely used quality improvement and risk assessment tools used in manufacturing, aeronautics, defence industries, etc. (Hansen 1989; Hassan et al. 2003; Grunske & Han 2008). Issues about the modelling of these engineering techniques to align with some notable risk management framework, such as PMBOK Guide (PMI 2000), for the purpose of developing better and reliable risk management methodology have been explored in project management (e.g., Souza & Carbal 2008; Carbonne & Tippet 2004).

Further applications of these techniques are currently being explored in software engineering to identify potential design flaws and development process failures in software-based systems before they occur, with the sole aim of eliminating them or to reduce the risk associated with them (see Mitrabinda & Durga 2011; Appukkutty et al. 2006). This is because different forms of risks exist in a software-based system. Software-based systems most of which are safety critical are prone to failure. These risks can be assessed at various developmental stages. In order to improve software safety and to achieve a better quality and reliability, it is essential to conduct a reliability-based risk assessment at the early developmental stages (Mitrabinda & Durga 2011).

Reducing the failure rate of a software development project is one of the core challenges in software industries. There are various system reliability-engineering approaches to identifying and recovering from system failures. FMEA and FTA have provided their clear merits for reliability analysis. These two safety and reliability analysis techniques, are widely used in many software industries. They are used to detect failures, and prioritize these failures based on user perception to perform sensitivity analysis of different components of the system architecture. These techniques are well-established reliability engineering tools widely used in various fields of industries. The aim is to investigate and identify likely failure modes in the system components, assess their impact on system behaviour and suggest adequate control action to address these effects. Other examples of these techniques are: HAZOP, Functional Failure Analysis (FFA), etc. (Grunske & Han 2008; Hansen 1989; Hassan et al. 2003). For the purpose of this study, the scope is limited to FMEA which is discussed below.

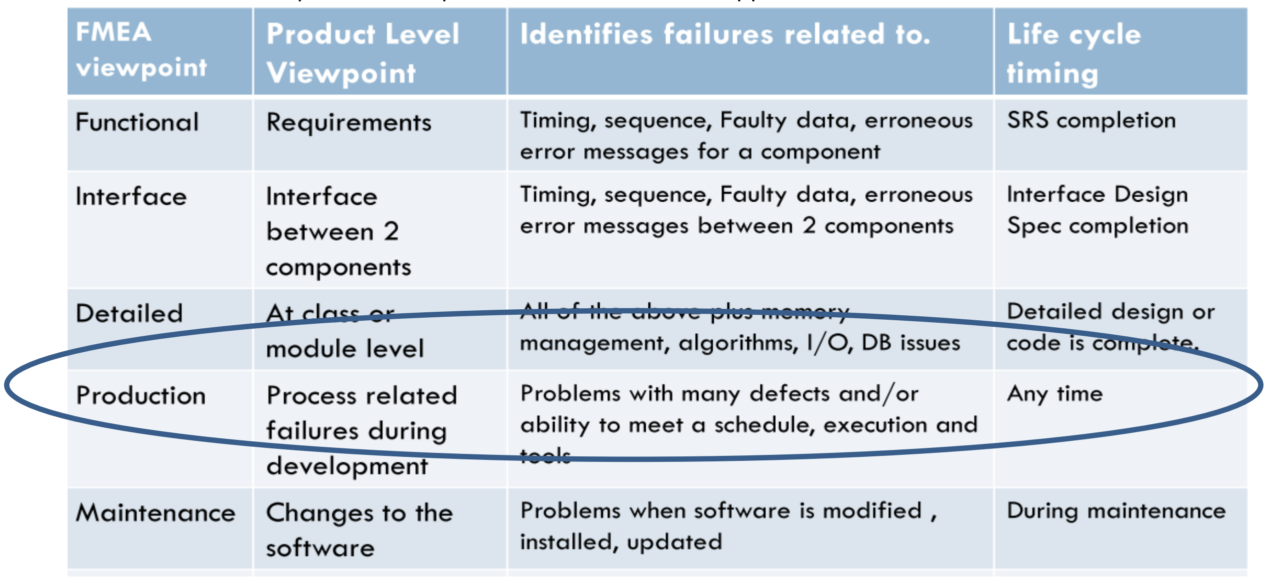
2.5.1 FMEA/FMECA

The notion of Software Failure Modes and Effects Analysis (SFMEA) has been proposed in SE as tools for software safety and reliability in the embedded systems community in the last two decades (Maier 1997). Maier used FMEA and FTA to support safe design of embedded software in safety-critical systems. The motivation has been both the success of FMEA as a standard (IEC 61508) technique to improve reliability and safety and the increasing contribution of software to many products. The inevitable consequence is the increasing proportion of safety and reliability challenges that are attributable to software failure. FMEA has multiple important view point in software development.

2.5.1.1 FMEA viewpoints in software development

The FMEA viewpoints describe the area of application of FMEA in software development. Table 2.13 describes the five important viewpoints from which Software FMEA can be conducted.

Table 2 13:: FMEA important view point in Feasible Areas of Application in Software



FMEA viewpoint	Product Level Viewpoint	Identifies failures related to.	Life cycle timing
Functional	Requirements	Timing, sequence, Faulty data, erroneous error messages for a component	SRS completion
Interface	Interface between 2 components	Timing, sequence, Faulty data, erroneous error messages between 2 components	Interface Design Spec completion
Detailed	At class or module level	All of the above plus memory management, algorithms, I/O, DB issues	Detailed design or code is complete.
Production	Process related failures during development	Problems with many defects and/or ability to meet a schedule, execution and tools	Any time
Maintenance	Changes to the software	Problems when software is modified , installed, updated	During maintenance

Table 2.13 describes the five important viewpoints of FMEA in software development. Analysis of viewpoints indicates that FMEA can be used to identify failures related functions at requirement level; at class or module level; and process related failures during software development and maintenance during changes to software systems. The appropriate timing for the application of the FMEA in software life cycle is as well presented in the Table 2.13.

The viewpoint that is circled in the Table 2.13 describes the scope of the proposed application of FMEA in the study. The FMEA will be used to identify processes related failures during development of software. The viewpoint is production, which indicates software development. It is worthy to note that the application of FMEA extends to analysis, estimation, prioritization and classification of risks in the process of software development.

2.5.1.2 Steps to conduct FMEA process

Naturally, FMEA method has its traditional steps to be followed for its implementation as tool for project risk analysis. The traditional steps for conducting FMEA process in project risk management is presented in Figure 2.6.

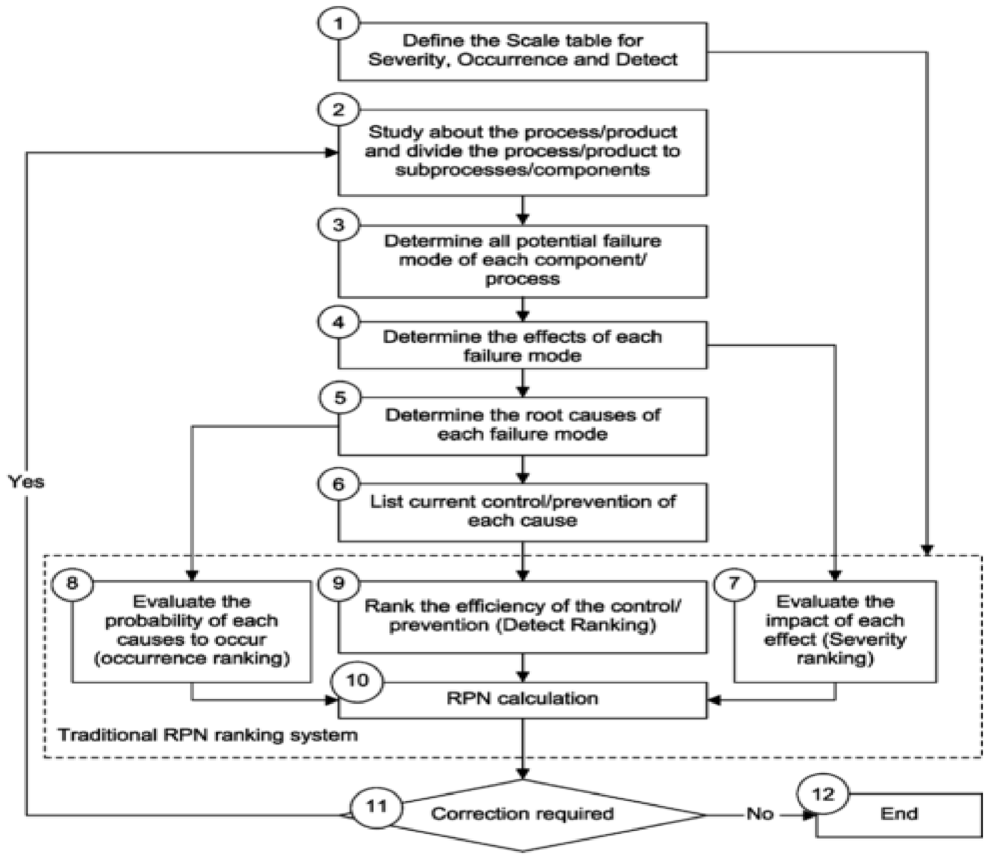

1
Define the Scale table for Severity, Occurrence and Detect
2
Study about the process/product and divide the process/product to subprocesses/components
3
Determine all potential failure mode of each component/ process
4
Determine the effects of each failure mode
5
Determine the root causes of each failure mode
6
List current control/prevention of each cause
Yes
8
Evaluate the probability of each causes to occur (occurrence ranking)
9
Rank the efficiency of the control/ prevention (Detect Ranking)
7
Evaluate the impact of each effect (Severity ranking)
10
RPN calculation
Traditional RPN ranking system
11
Correction required
No
12
End

Figure 2 6:Steps of conducting FMEA process (Source: Tay & Lim 2006)

The steps in Figure 2.5 describe the steps to conduct a traditional FMEA process. For the purpose of this study, the traditional FMEA process will be transformed into SFME production viewpoint, which is the focus of this research. The SFMEA process will require gathering a team of practitioners and researchers in software development to conduct the SFMEA using the steps 1 – 12 as indicated in Figure 2.5.

However, researchers that use FMEA for their risk analysis can modify the traditional steps or procedures indicated in Figure 2.5 to suit the purpose of their studies. For example, Georgieva (2010) conducted FMEA over the stages of the software development process using the steps presented in Figure 2.7.

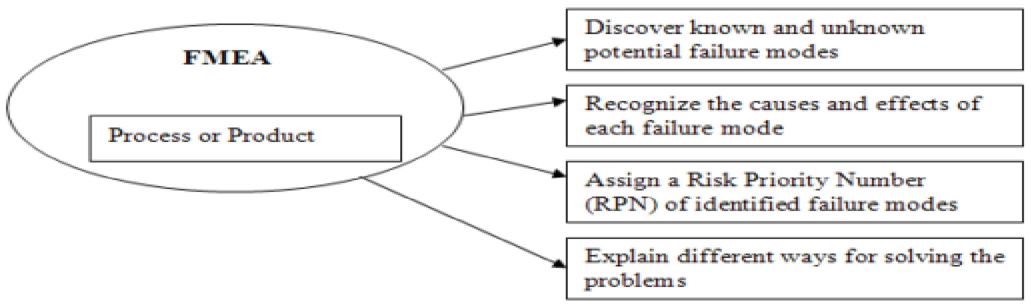

FMEA
Process or Product
Discover known and unknown potential failure modes
Recognize the causes and effects of each failure mode
Assign a Risk Priority Number (RPN) of identified failure modes
Explain different ways for solving the problems

Figure 2 7: Steps to conduct FMEA over software development process (Source: Georgieva 2010)

Georgieva in her study streamlined the traditional twelve (12) steps of FMEA process into four (4) steps to conduct FMEA process over the software development process. All the same, further analysis of her study indicates that most of the traditional steps were implemented implicitly in the study.

Aspects that should be noted that make SFMEA production viewpoint analysis to go faster and more effective are:

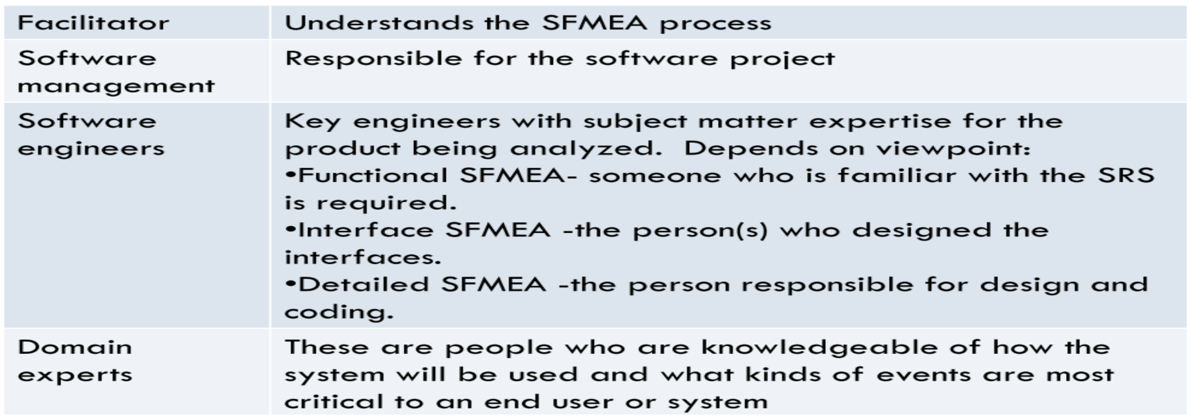
- More detailed process documentation such as Software Requirement Specification (SRS), Interface Design Specification (IDS), Design Documents, Test plans, etc.
- Software developers who are willing to appreciate the likelihood of software failure and not those who are always optimistic.

2.5.1.3 Personnel required for the conduct of software FMEA

The composition of personnel required to conduct FMEA in software development are: Facilitator who understands the SFMEA process; Software project managers; Software engineers and Doman experts (softrel.com).

Table 2.14 presents the categories of software development professionals that are required to participate in the software FMEA (softrel.com)

Table 2 14: Personnel required for Software FMEA production viewpoint (Source: softrel.com)



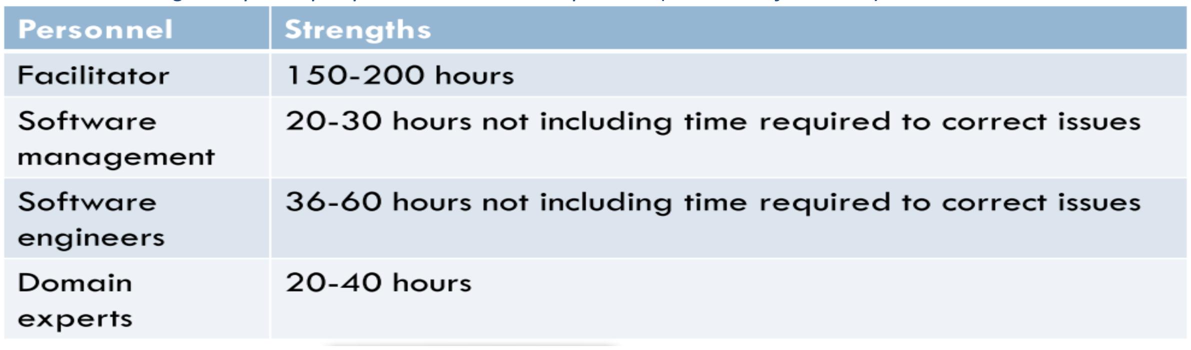
Personnel	**Strength**
Facilitator	Understands the SFMEA process
Software management	Responsible for the software project
Software engineers	Key engineers with subject matter expertise for the product being analyzed. Depends on viewpoint: •Functional SFMEA- someone who is familiar with the SRS is required. •Interface SFMEA -the person(s) who designed the interfaces. •Detailed SFMEA -the person responsible for design and coding.
Domain experts	These are people who are knowledgeable of how the system will be used and what kinds of events are most critical to an end user or system

As mentioned earlier, the viewpoint production of software is also referred to as software development. The compositions of personnel that will form a team to conduct FMEA process in software are presented and the role of individual professional are outlined in the Table 2.14. Analysis of the compositions of personnel from the table shows that the compositions of the team to conduct SFMEA include software developers; users of software to be developed and professionals from and outside SE.

2.5.1.4 Strengths (in hours) required per personnel for the conduct of software FMEA

This section describes the range of strength (period of engagement in hour) required per personnel for the conduct of SFMEA. Table 2.15 presents the duration of engagement (in hours) of individual personnel that will participate in SFMEA process.

Table 2 15: Strength required per personnel in SFMEA process (Source: softrel.com)

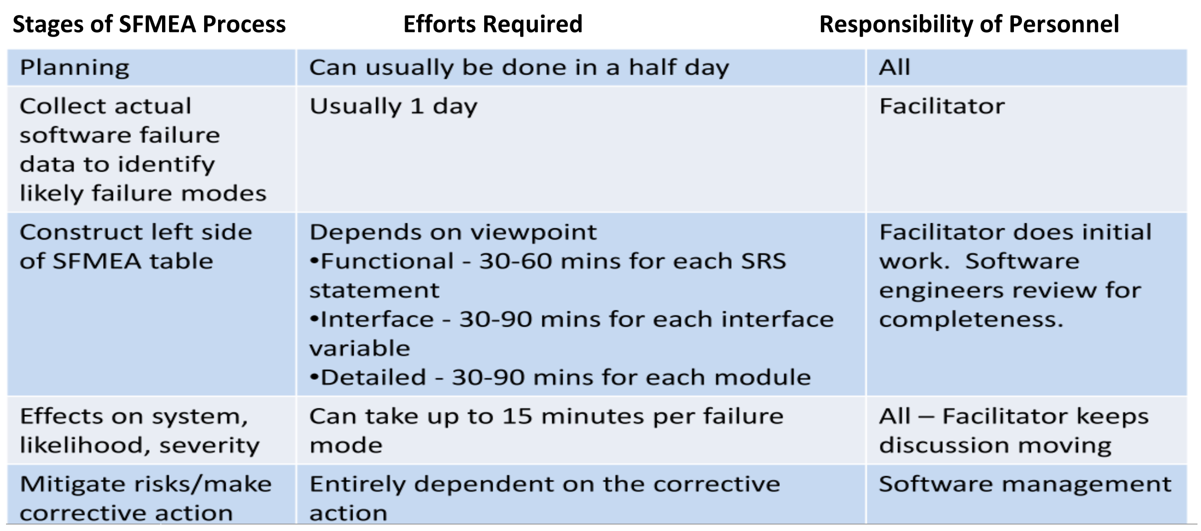


Personnel	Strengths
Facilitator	150-200 hours
Software management	20-30 hours not including time required to correct issues
Software engineers	36-60 hours not including time required to correct issues
Domain experts	20-40 hours

Analysis of the duration shows that the facilitator is expected to commit the highest engagement in the SFMEA process. This is expected because the facilitator will coordinate all the sessions of the SFMEA process, hence, present in all the sessions. Software engineers are expected to be more engaged than domain experts in software project management .

The engagement period and the responsibility of personnel required to conduct the stages of the SFMEA process are presented in Table 2.16.

Table 2 16: Personnel responsibilities and effort required to conduct the stages of SFMEA



Stages of SFMEA Process	Efforts Required	Responsibility of Personnel
Planning	Can usually be done in a half day	All
Collect actual software failure data to identify likely failure modes	Usually 1 day	Facilitator
Construct left side of SFMEA table	Depends on viewpoint •Functional - 30-60 mins for each SRS statement •Interface - 30-90 mins for each interface variable •Detailed - 30-90 mins for each module	Facilitator does initial work. Software engineers review for completeness.
Effects on system, likelihood, severity	Can take up to 15 minutes per failure mode	All – Facilitator keeps discussion moving
Mitigate risks/make corrective action	Entirely dependent on the corrective action	Software management

The contribution of these specifications of responsibilities and engagement duration allow risk analyst to plan and design an effective FMEA process before applying the FMEA process for the conduct of any form of risk analysis. It is beneficial for this study to adopt the specifications of responsibilities and engagement duration as specified in the Table 2.16.

2.5.2 FMEA application in SE

This section discusses the previous research conducted in software risk management using FMEA method. At the end of the presentation, research outcomes involving the application of the FMEA method will be analysed to evaluate the feasibility of integrating the method into risk management of software projects.

In recent years, software safety and reliability engineers and researchers have intensified their efforts towards applying powerful and effective reliability and safety analysis techniques to examine possible failure mechanisms, risk assessment and to eliminate potential failures and risks across system development lifecycle (e.g., Khaiyum & Kumaraswamy 2014; Gupta et al. 2012; Mitrabinda & Durga 2011; Hassan et al. 2005). Other researchers have applied the technique of FMEA to the software development phases to describe possible failure components in software-based systems (e.g., NASA 2004; Ozarin & Siracusa 2003; Lutz & Woodhouse 1997; Reifer 1979).

Some have adopted the application of Fault Tree Analysis (FTA) technique, another powerful static–analysis tool, which has been imported from mechanical engineering (Gupta et al. 2012), to investigate the potential software related causes of a fault or hazard (Dehlinger & Lutz 2004). Software FTA is an engineering activity, which is primarily used in software development to identify software defects. It is mostly efficiently applied when detailed requirements or design documentation exists (Lutz & Woodhouse 2004). Software FTA has also been used to verify software code (Leveson 1995).

Functional Failure Analysis (FFA) is another technique, which is used as a top down approach based on system scenarios to identify the system level failures (Hassan et al. 2005). It is a tool used to measure, estimate, and forecast the reliability of software system during various stages of development. The FFA uses system level scenario diagrams as an input to identify all system level failure modes. This high-level analysis provides a comprehensive view of the ways in which the systems fail (Hassan et al. 2005).

Some researchers have adopted integrated techniques, combining two or three of the FMEA, FTA and FFA tools to improve the failure assessment activities. The Bi-Directional Safety Analysis (BDSA) is an approach of integrating both the FMEA and FTA to assess software failure modes (see Gupta et al. 2012; Feng & Lutz 2005; Lutz & Woodhouse 1997). Others have conducted software risk assessment on software development artefacts using the combination of the trio: FMEA, FFA and FTA (see Hassan et al. 2005).

Furthermore, there has been a tremendous achievement in extension of these integrated techniques from software development activities to product–line applications such as flight – instrumentation displays and developed tool support for the reuse of failure-analysis artefacts within a product – line (see Dehlinger & Lutz 2006; Feng & Lutz 2005; Dehlinger & Lutz 2004). However, further research is highly required in this direction (Lutz 2007). Other contributors have refined these integrated approaches to support those projects having tight budget and or duration constraints for the failure analysis efforts (e.g., Lutz & Shaw 1999).

Feather, Cornford, Dunphy and Hicks (2002), Cornford and Feather (2001) conducted risk assessment at the requirement stage based on a multiple experts' knowledge. These two research approaches employed the same methodology by first identifying the possible mode of failures for a high-level requirement and then tried to estimate the impact of these failures on the requirement. These methods are purely subjective. The two approaches do not take any analytical approach, e.g., using architectural level information for their analysis, and therefore are more error-prone on account of being human intensive. The risk assessment proposed in this research will employ both analytical and subjective approaches.

Appukkutty et al. (2006) proposed a risk assessment method by identifying likely failure possibilities of a scenario and analysing the scenario complexity in each failure mode. Their proposal recommended risk assessment at the requirement level. However, in the proposal framework, low-level associative risk details such as a component or connector is not considered.

Mitrabinda and Durga (2011) applied the FMEA to assess the risk factors for a component-based scenario at the early stage of SDLC. Interestingly, the risk assessment method relied on the dynamic composition of the FMEA team by capturing both the analytical and subjective views of the team to analyse the technical and some non-technical risks of the software based system at early stage of development. The weakness of the assessment, however, was the narrow assessment proportion of the SDLC, which cannot effectively address other important risks factors in all the phases of the SDLC. In addition, the assessment was based on fictitious assumption of fixed requirements and did not include factors of conflicting views of the assessment team.

Georgieva (2010) advances the use of FMEA in SE by applying the FMEA method to identify the probable failure modes in the software development process. The work exhaustively describes steps to follow in order to conduct an effective application of FMEA over the SDLC. Georgieva (2010) identified fifty-two (52) probable failure modes across eight (8) studied stages of the software development process. Though, Georgieva work introduced a cost effective and reliable technique to identify likely failure points in the software development process, it failed to assess the risk associated with the identified failure mode. Also, the information about the failure effect on the software project outcome and control strategies to mitigate the effect were not included. This gap is the focus of the current study.

In addition, FMEA have been applied over concrete programs in the software development and not purely theoretical (Lauritsen & Stålhane 2005). Luke (1995) highlights the role of FMEA in developing quality software products through the identification of the probable failure modes early enough in the software development process. However, method implementations of the FMEA to analyse the risk factors that are associated with the potential failure modes in the SDLC have not been explicitly formulated. In addition, the criticality classification of the failure modes in the SDLC and their impact on project outcomes remain largely unexplored.

2.5.3 The technical benefit of FMEA application in SE

FMEA is a proactive approach to failure prevention and can be applied to software development process. Application of FMEA to software allows us to anticipate and plan for problems or failures before they occur, thus catalysing the building of quality into our software products. It involves structured brainstorming to analyse potential failure modes in software, rate, rank and prioritize the risk to the software and take appropriate actions to mitigate the risk. This process is used to improve software quality, reduce cost of quality in software projects and to improve project scheduling.

Some of the benefits according to the SW-quality (2016) are:

- It enhances the development of software that is more robust and reliable.

- FMEA is a cost-effective tool. It can significantly reduce the cost of testing and risk analysis in software projects.
- It increases productivity of the software development organization, in the aspect of developing software that are reliable and of high quality within a short period.
- It enhances improvement in software project scheduling.

In all, studies and findings reviewed in these sections reveal that there is a growing interest in the literature by practitioners, scholars and researchers in SE on the application of the FMEA for software project risk analysis. However, analysis of the reviewed literature showed that the adoption of the reliability engineering techniques (e.g., FMEA) in software engineering has been largely marred with unstructured principles and technical uncertainties (the techniques' procedural requirements used for risk analysis not tested against standard requirements of the formal project risk management framework). Thus, generating scepticism leading to slow adoption of the engineering techniques among risk managers. These waters down the confidence of accepting the FMEA as a competent risk management tool among developers and managers who are searching which appropriate, reliable and cost-effective tool to use for the conduct of project risk management.

Furthermore, analysis of the reviewed literature indicated that most of the reported risk assessments conducted with the FMEA in SE mainly focused on risk analysis and risk assessment (e.g., Khaiyum & Kumaraswamy 2014; Gupta et al. 2012; Mitrabinda & Durga 2011; Hassan et al. 2005). Hence, there is need for research that will improve the application of the FMEA to cover areas of risk control and management in such a way that will suggest efficient risk response strategies to mitigate the effect of the occurring risks on the project outcome (Georgieva 2010).

This research will rely on the findings from the literature to address the identified gaps beginning with mapping the procedural requirements of the FMEA for risk analysis against the standard requirements of the formal project risk management framework. The aim is to provide empirical evidence on the FMEA's efficacy and effectiveness for the conduct of software project risk management. Secondly, this research will explore means of improving the application of the FMEA in SE to cover risk control and management. Lastly, applying the FMEA method to assess the risk factors associated with the potential failure modes across the stages of the SDLC. The assessment will begin with identifying the potential failure modes, then conduct a cause-effect analysis on the failure, assess their associated risks, prioritize and classify their risk criticality and suggest appropriate risk response strategies to control the effect of the risks.

2.6 Deductions from reviewed literature

Observations/Deductions

Literatures studied revealed five contending issues that need further examination.

- Research agencies in project development that investigated the global status of software projects failures reveal that the failure rate of software development projects worldwide is still high and far beyond expectation (CHAOS Report 2020; GitLab 2019; KPMG 2019; Wellington 2018; Giuseppe 2017; Lehtinen et al. 2014; de Wet & Visser 2013). The study also found that a realistic and more accurate early risk estimates and fully integrated risk management into the process of software development could help software managers to reduce wasted resources that are associated with project cost and schedule overruns. These findings are a clear indication that the failure rates of software development projects are rising with time and that the manner in which risks in these projects are managed in most software development organizations deserves urgent attention.

- The concept of ‘risk’ as being conceptualized in SE is narrower than the nature of the inherent problems in practice that are tenable in SE. Thus, there is the need to modify the risk theory to adequately capture other flaws in the software development process (e.g., risk factors associated with failure modes), identifying their causative factors, assess the risk effects on the project outcome and predict suitable control measure to fix the risks.

- Researchers and scholars in SE view risk management as a tactical, formalized, structured mechanism and continuous process which is team-oriented and is capable of napping project risks in order to ensure that projects failures are prevented. The risk management process is generally accepted to be divided into two interwoven stages, i.e., Risk Assessment and Risk Control. The study also found two popular risk management strategies adopted in SE from literatures. These are: *Reactive* and *Proactive* risk management strategies (Asadi 2015; Zardari 2009; Bannerman 2008). In the reactive strategy, risks are identified after problems had occurred and problems are identified after investigation while the proactive risk strategy seeks to identify all-important risks earlier, before a problem occurs. The implication of these findings suggests the need for software development teams and their managers to adopt a risk management strategy that will reliably identify and handle risk factors in all the stages of the software development process to prevent the incidence of project failures. Thus, the point of concern here is: what risk management methodology can be adopted in the software development projects with scientific justification to reliably identify, assess and control risks factors in the software development process?

- The literatures reviewed reveal that managing software project risks has been a dominant focus of current research in software engineering. Several software risk management models and frameworks have been developed and have been proven very effective in software engineering (Neves et al. 2014). However, three out of all reviewed standard frameworks (SEI/SRE risk management (1996), Boehm model (1991), PMBOK/PMI (2000)) stand out, for they are found to have detailed coverage of the required steps for project risk management. It is worthy of note that all the reviewed risk management frameworks aimed at minimizing the incidence of project failures are too difficult to manipulate to admit project specific variables and so do not give room for human contribution.

  The study also found that research on risk factors in software development projects are not detailed enough to inform software practitioners on other important dimensions that deserve more attention. Researchers used many approaches to identify and manage these risk factors. Some based their approaches on their personal experience with software projects, some on extensive literatures reviewed, and some combine literature review with intuitivism from experienced project managers (details shown in Table 2.2). The reality is that a good number of them are not validated before use and they lack theoretical justification. However, the work of Wallace et al. (2004) stands out as shown in the Table 2.2, just as the work was based on comprehensive literature review and the instrument development process was later validated rigorously with project managers. However, since her work covers United States of America (US) companies only, the outcome of her work may not have captured risk factors that are specific to software development organizations outside the US companies.

Further to this, most of the published lists of risk items may not adequately cover the specific project of contention as shown in the Tables 2.2 – 2.10. Also, the lists only prescribe risk factors and their associated dimensions but are found short of evaluating the risk effect on project outcomes. Others cannot provide effective means to counter the effect of the threat posed by the risk factors. Above all, research in these areas is weak in exploring the links that exist between the potential risk factors in the software development process and project outcomes.

- Lastly, the review on the possibility of adoption of safety and reliability engineering techniques (e.g., FMEA) research in software engineering has shown that very impressive attentions of software engineers and researchers are currently focused on the application of reliability and safety engineering technique as shown in section 2.5.2. Other studies focus on risk assessment of hardware component of software-based system (see section 2.5.2). However, little concern is expended on the use of safety engineering techniques to identify and assess risk factors in the software development process. It has been found out that no existing software risk methodologies have been proposed or reported from the literature, which estimate detailed technical uncertainties (e.g., risk associated with the probable failure mode often ignored) in all the phases of the software development process. Hence, there is the need for further research in this area.

  Moreover, reviewed research concerning the application of reliability engineering techniques in literature showed that application of the safety engineering techniques in software engineering have been largely marred with unstructured principles (procedures used for risk analysis not tested against standard requirement of any of the dominant risk management framework e.g., PMBOK/PMI or SEI/SRE) (Georgieva 2010). Some are under-explored (used individually), and efforts have not been dedicated to analyse and classify the risk associated with the failure modes across all phases of SDLC in a way that will suggest efficient control strategies (Georgieva 2010).

Finally, the procedures for the conduct and application of some of these safety engineering tools for software risk assessment have also been largely missing from literature (as shown in section 2.5.2). The reason according to Georgieva (2010) is due to a narrow scope of research on reliability engineering tools in the software development process. Also, no paper has been identified or reported on a particular project where these safety-engineering tools were used unsuccessfully in the literature. In addition to this, studies over the use of FMEA and other methods that could help project managers to identify and reliably determine the potential threat posed by these failure modes in order to develop effective strategies to deal with these challenges have not been given the desired concern by researchers (Georgieva 2010; Luke 1995; Banerjee 1995). Hence, there is the need to conduct further research in this regard.

The work of Georgieva (2010) puts a new milestone on this quest. It formulates a practical exhibition by mapping out strategies and guidelines for the application of FMEA method over the software development process. However, criteria for the identification of the listed failure modes were not detailed in the work and neither the investigation concerning the relative importance of the listed failure mode nor an attempt to classify and suggest measures to counter the effect of the identified failures were conducted in the research. Therefore, the focus of this study is to address these identified gaps and improve on the previous research that applied the FMEA for the conduct of risk assessment in SE.

2.7 Chapter summary

This chapter presents reviewed literatures that are relevant to the background on the use of the FMEA method for assessing the risk factors associated with potential failure modes in the software development process. In the chapter, a detailed discussion of the relevant issues pertains to the phenomenon investigated. The global status of software project failures is enumerated from different perspectives of research agencies in project development to ascertain the degree of crisis in software development projects. The chapter also provides information about the concept of 'risk' as being conceptualized in SE and suggests the adjustments required of the risk theory to accommodate other dimensions of risk factors such as failure modes in the software development process. Other contending issues such as risk factors in software projects, software risk management, software development process, failure modes, risk management frameworks and the possibility of integrating the safety and reliability engineering tools for the conduct of risk assessment in SE are discussed in details to set the basis for the study.

The FMEA method is widely used for quality improvements and risk assessment tools in the manufacturing, aeronautics, defence industries, etc. This engineering paradigm has long gained a wider acceptance in software engineering. The notion of software FMEA has been tried in the embedded systems community over the past 20 years. The motivation has been both the success of FMEA as a standard (IEC 61508) technique to improve reliability and safety software products. The FMEA was therefore considered appropriate for this study due to its emphasis on continuous and proactive risk management strategy, team-oriented process and most importantly modelling efficiency to capture risk factors associated with potential failure modes in the software development process.

The chapter concludes with deductions from the reviewed literature, which identified gaps that require filling. The following is the summary of contending issues deducted from the reviewed literature that are critical to guide the conduct of the research.

- The failure rate of software development projects worldwide is still high.
- The concept of 'risk' as being conceptualized in SE is narrower than the nature of the inherent problems in practice that are tenable in SE
- Researchers and scholars in SE agree that risk management is team-oriented and is capable of napping project risks in order to ensure that projects failures are prevented.
- The study also found that research on risk factors in software development projects are not detailed enough to inform software practitioners on other important dimensions that deserve more attention.
- Most of the published lists of risk items may not adequately cover the specific project of contention as shown in the Tables 2.2 – 2.10.

- Application of the reliability engineering techniques (e.g., FMEA) in software engineering has been largely marred with unstructured principles and trust deficit among developers and managers who are searching which appropriate, reliable and cost-effective tool to use for the conduct of project risk management.
- Application of the FMEA in SE is restricted to risk analysis and assessment
- There is need for research that will improve the application of the FMEA to cover areas of risk control and management in such a way that will suggest efficient risk response strategies to mitigate the effect of the occurring risks on the project outcome

The next chapter presents the theoretical and conceptual frameworks of the study.

CHAPTER THREE

Theoretical and Conceptual Frameworks

This chapter presents the relevant theories that serve as the theoretical foundations for this study. The main objective of the chapter is to discuss the theoretical frameworks used to present a clearer structure and vision of the study. Section 3.1 is a presentation of the two theories guiding the conduct of this study. Section 3.2 presents the conceptual model used to illustrate the relationships among the various constructs that informed the study while the chapter summary is presented in section 3.3.

3.1 Theoretical framework

3.1.1 Introduction

Adopting the most appropriate theoretical framework in a research process is one of the most important components of research. The benefit to be derived from using a theoretical framework in a study cannot be overemphasized. The theoretical framework is the foundation from which all knowledge is constructed (metaphorically and literally) for a research study. The framework presents the foundation and basis for the study (Grant & Osanloo 2013). Grant and Osanloo provide Eisenhart's definition of a theoretical framework as "a structure that guides research by relying on a formal theory constructed by using an established, coherent explanation of certain phenomena and relationships" (Eisenhart 1991, p. 205). Thus, the theoretical framework is a composition of relevant theories that underpin the opinion of a researcher in relation to how he reasons and plans to research his topic, as well as the constructs and definitions of variables from the theory (theories) that are relevant to his research title. Na-Allah (2019) studied the importance of theoretical and conceptual frameworks to qualitative research and concluded that, "they help greatly in making sense of data requiring clear thinking" (p.74). Thus, theoretical and conceptual frameworks are tools used by researcher to organize and make clarity of data used in his research to the readers.

This chapter discusses the theoretical frameworks used as a lens for the introduction and use of FMEA to assess the risk factors associated with potential failure modes in the software development process. Thus, two theories are combined to provide the theoretical foundation for the conduct of this study. These are the Diffusion of Innovation Theory (DIT) and the Technology Acceptance Theory/Model (TAT/TAM). Section 3.1.1 presents the Diffusion of Innovation Theory while the Technology Acceptance Theory is discussed in section 3.1.2.

The DIT theory was considered suitable for adoption in this study to: (i) introduce and spread the idea of using FMEA for software project risk management; (ii) to brainstorm on the relative advantages and compatibility of the FMEA with other existing project risk management tools; and (iii) to implement the FMEA to assess the risk factors associated with the potential failure modes in the software development process based on the software development practitioners' perception, while the TAM is considered for adoption in this study to provide an understanding of the determinants of the actual user behavioural use of the FMEA for the conduct of software project risk management activities.

Many studies (e.g., Kanchanatanee, Suwanno & Jarernvongrayab 2014; El-Gohary 2012) have integrated DIT and TAM in similar context with this research to examine the degree of adoption and behavioural usage of a new product, method or technology. For example, El-Gohary (2012) combined the TAM and DIT to validate a conceptual framework by extending the combined models to illustrate E-Marketing adoption by Egyptian tourism organizations. Similarly, Kanchanatanee, Suwanno and Jarernvongrayab (2014) used the same model (combined DIT and TAM) to establish the effect of attitude toward using, perceived usefulness, perceived ease of use and perceived compatibility on intention to use E-Marketing of small and medium sized business owners in the three southern border provinces of Thailand. Both studies explained the E-marketing adoption utilising a quantitative approach, in which data is collected based on survey strategy through questionnaires to address different levels of the study. Findings from the studies confirmed that IT theories (namely TAM and DIT) are valid models to illustrate E-Marketing adoption. Both studies also added to the extremely limited number of empirical studies that has been conducted to investigate adoption of new technology (such as E-Marketing) in developing countries.

Even though there are other related theories and models that may be applicable to this research, the DIT and TAM are chosen because they adequately capture the divergent opinions and experiences that software development practitioners will hold when applying a new technique and the actual behavioural use of the new technique for the conduct of risk assessment of software development projects. The resolve for the adoption of DIT and the TAT/TAM are based on the notion that models are the main route for researchers to develop conceptual frameworks, while theories lead to theoretical frameworks (Ngulube, Mathipa & Gumbo 2015). However, models simply describe a phenomenon, unlike theories that have the capacity to explain and predict, but the focus of the research is to expose user to a new technique and provide a critical assessment of it using critical qualitative and quantitative approaches and reliable methods discussed in Chapter 4 of this research.

After a detailed review of the literature on the DIT as propounded and used by Rogers (1998), five constructs were identified as significant determinants of communication for diffusion of innovation through certain transit within a defined period and across a designed social system. These are knowledge, persuasion, decision, implementation and confirmation. This was the basis upon which Rogers (2003, 1998) describes the elements of the DIT, which he found to play a significant role, as communication for diffusion of innovation.

Similarly, the literature studies conducted on TAT/TAM as propounded and adopted by Davies (1998) revealed three constructs (*the actual system use, behavioural intention and attitude toward use*) as significant determinants on the manner in which target population accepts and uses an information system or technology. This was the basis upon which Davis (1998) analysed and theorised the two fundamental constructs, which he found to crucial determinant of technology adoption. The two fundamental constructs are: *perceived usefulness* and *perceived ease of use*.

The resulting seven fundamental constructs extracted from the DIT and TAT/TAM are the main focus of this study, because they are informed by the perceptions, beliefs and experiences that users hold about innovation and its usage. Details about the selected theories are presented in the next sections.

3.1.2 Diffusion of innovation theory (DIT)

The DIT was propounded by Rogers in 1962 to describe the manner in which an idea diffuses across a population. The aftermath of this diffusion is that elements of the target population embrace a new idea or product. Adoption simply refers to when people or communities apply a new idea or new one(s) with a reason that the new product is better than the previous or existing product. It may as well result in acquiring and performing a new behaviour. In addition, adoption is said to have taken place when adopters perceive the product, idea or behaviour as new, beneficial and innovative (Rogers 2003, 1995)

The DIT was developed to explain how innovation spreads. The spreading of innovation is a process by which new idea, product or behaviour is constructed across elements of target population (Rogers 2003, 1995). Consequently, the spreading process informs members of the target population on the benefits of the idea or discovery sources. Rogers (2003) propounded this model to explain the spread of innovation process. According to Rogers, the diffusion of innovation is a process of communicating new ideas, method or product through designed channels among the members of a community over a period of time (Rogers 1995). The DIT explains how innovation spreads out from its source to the prospective user or its adapter; it emerges in the society as a group process (Rogers 2003). In Roger's views, successful innovation spread process is composed of four elements, which include: (a) The idea or practise to be spread must be categorized as innovation; (b) The innovation spread must be transmitted through certain channels; (c) The innovation must be used among members a target community; (d) The innovation process must take into consideration adoption time for the practise or idea. The first two listed elements: *innovation* and *communication channels* are the focus this study.

At this early stage, it is important to make it clear that the first and second elements of the DIT are relevant to this study (i.e., Innovation and Communication Channels), since the research scope covers the adoption of a new approach to assess the risk factors associated with potential failure modes in the software development process. Though, certain scientific means of validating the research findings are included, the conclusion as to either adopt or reject the innovation and the time frame for the adoption are areas of consideration in future research. However, the researcher discusses the concepts of the two relevant elements of the DIT (Innovation and the Communication channels) before presenting how they relate to this research.

Innovation

The diffusion process starts with innovation of idea, method or product that is perceived as novel or new by users of the product are considered as appropriate to use or adopt. The adoption levels are clarified by the characteristics of innovation which Rogers (1995, p.15) describes as follows:

• ***Relative advantage****.* According to Rogers, the relative advantage is the extent to which the innovation on the new idea or product is adjudged to be better than the existing idea. Hence, adopters will consider the new idea if they perceive it to be having some relative advantage over the existing product. The level of comparative advantage between the new idea and the existing one determines the rate of adoption of the new idea or method (Robinson 2009, p.2).

• ***Compatibility***. In Rogers's view, compatibility with the existing values and practices is the extent at which the new idea or the innovation is adjudged to be consistent with past experiences, and or the existing values, and the needs of the potential adapters. An idea that is compatible with the norms and values is more liable to be adopted faster than the incompatible idea (Rogers 1995, 2003). This means that people tend to adopt an innovation that is capable with their cultural norms and practices faster.

• ***Complexity****.* Rogers describes complexity as the degree to which an innovation is perceived as difficult to understand and use. Thus, people are more likely to adopt innovations with ease of use and which are easy for them to understand.

• ***Trialabillity*** *(same as ability to be tested)*. An innovation is perceived as difficult to comprehend or use (Rogers 1995, 2003) to the extent. If innovation is not tested, it cannot be expected to be successful. Verifiable innovation represents less uncertainty to the individual who will consider it for adoption. After its adoption, the individuals learn by doing practice or trial (Rogers 1995, 2003). Thus, individuals tend to adopt an innovation if they can test it before considering to its adoption

• ***Observability*** *(same as distinctiveness of the results).* The outcome of innovation is distinct from the existing system to the extent. The more visible the outcome of the innovation is noticed by the people, the more likely the innovation is adopted by them. Such distinctive result stimulates interest with further discussion with other people of different interest, which requires assessment information on innovation.

3.1.3 Communication channels

A new idea or product that possesses the five characteristics suggested by Rogers has to be communicated to prospective adopters in order to be accepted for adoption. Thus, Rogers proposed a channel of important factors for a successful diffusion process. In Rogers's views, one-on-one communication between people is more important than mass media communication. Rogers also believes that it is the members of the communities that decide about whether to adopt or reject the innovation. In identifying the channels for communicating the innovation, Rogers (1995, 2003) suggests a five-stage communication channel of innovation that is made up of a linear mental process. Rogers views this linear stage as a mental process from which an innovation can be communicated within a specific period and within a social system. In Rogers's view, "the process starts from the first innovation recognition to the formation of an attitude towards it, then to the decision to be adopted or rejected, later on to the implementation and use of the idea or the new practice, and finally to decision confirmation" (Rogers 1995, 2003). Figure 3.1 presents the communication channels for diffusion of innovation as identified by Rogers (2003).

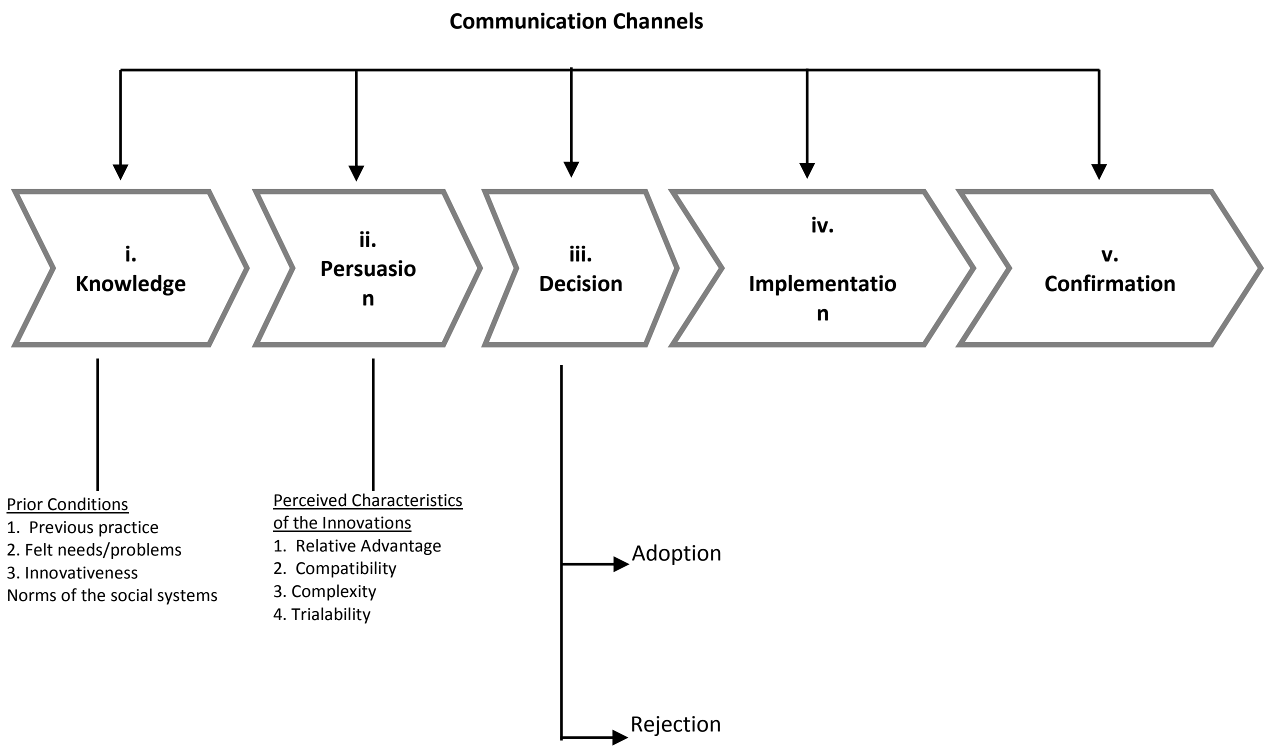


Figure 3 4: Communication channels for diffusion of innovation as identified by Rogers (2003)

Rogers (2003) identifies knowledge, persuasion, decision, implementation and confirmation as the fundamental channels through which an innovation, an idea or product can be communicated disseminated to gain momentum over time through a specific population or communities.

1. **Knowledge:** This is the first stage of diffusion of innovation channel. This stage first exposes the adopter to the real innovation. The prospective adopters may be ignorant of the implication of the adopters at the early stage and they have to be implemented adequately enough to know more about the innovation. In this stage, important information concerning the existence of the innovation are learnt by individuals and other useful information are gathered.

2. **Persuasion:** In this stage, aspiring individuals who are also active seek knowledge that will assist in the decision process. Persuasion is the point at which the prospective adopter is open to the idea of the new approach. They also seek for useful information that justifies their conclusive decision.

3. **Decision:** This is the stage where the individual adopts or rejects the innovation. Here, eventually the prospective adopter must make a decision. They will have to compare and contrast several factors concerning the innovation before finally making decision about the adoption to either accept or reject the innovation.

   Rogers considers this decision stage as the most difficult stage to acquire intelligence because individuals make wrong decisions in many instances. At times, wrong decisions are made following their underlying views and feelings and following the decision. Thus, the decision making process is challenging – the reason given is that following a decision is not likely to be representative of the actual reason that a decision was made.

4. **Implementation:** This is the stage in which prospective adopters implement the innovation and assesses its values and benefits. Once a decision to adopt an idea or innovation is positive, the innovation will then be accepted by the adopter. During the implementation, the adopter is expected to advance in knowledge about the benefits and also seek further information that will either improve the implementation of the innovation or to acquire better understanding about the product in context.

5. **Confirmation:** In this stage, the adopter continues to seek out more information that will justify that the adoption of the innovation has relative advantage and was beneficial. This stage stimulates a decision by the adopters to either continue using the innovation or discontinue its use.

In this study, an approach of integrating the FMEA methodology is considered as a reliable and effective means for software project risk management in addressing one of the problems associated with poor management practice in software development projects. Thus, the innovation is the use of FMEA to assess the risk factors associated with the potential failure modes in the software development process. The researcher also considered the characteristics of the FMEA method reviewed in the literatures for this thesis (section 2.5.1 to section 2.5.1.4) for the conduct of software project risk management in relation to the characteristics of innovation theorized by Rogers (1995) (i.e., relative advantage, complexity, trialability, and observability).

The procedural characteristics of the FMEA method for the conduct of software project risk management is diffused or spread to two different social systems. The social system is viewed as a confinement consisting of interrelated units that proffers solution to a common problem in order to achieve a common goal (Rogers, 1995, 2003). The social system forms the area where innovation spreads. Constituents of a unit of a social system can be individuals, informal groups, organizations and so on. The social systems for this study are: *software development practitioners* and *undergraduate software engineering students*. The channels of communicating the innovation are through exploratory case study, focus group discussions, workshops and interactive sessions. The communication channels follow the five stages of communicating innovations identified by Rogers (1995) (i.e., Knowledge, Persuasion, Decision, Implementation and Confirmation).

The diffusion of the innovation to the members of the social system follows series of processes as laid down by Rogers (1995). The procedural characteristics of the FMEA are first presented in a workshop to the participating software practitioners; an exploratory case study was then conducted to qualitatively examine the FMEA procedural characteristics and then followed by a focus group discussion with the same practitioners. The workshop was intended to inform the participants (knowledge) about the procedural characteristics of the FMEA for the conduct of the software project risk management. Also, the focus group discussion was conducted to collectively brainstorm amongst practitioners on the perceived FMEA procedural characteristics and the FMEA capabilities for the conduct of software project risk management. The exploratory case study was conducted to examine the procedural characteristics of FMEA method in software project risk management and compared same with the procedural frameworks of other popular project risk management frameworks (PMBOK, SEI & Boehm). It also examines and gathers input on various artefacts associated with the application of the FMEA method.

At the end of the exploratory study, a questionnaire survey was conducted with members of the social system (participating software development practitioners) to assess the extent to which the FMEA framework adheres to other popular project risk management frameworks. During the survey, practitioners are requested to complete questionnaire asking them to assess the extent to which the FMEA procedural characteristics adhere to other popular project risk management frameworks. Thus, the hypotheses for the evaluation are defined as:

H_{02}: There is no significant difference between the mean scores of FMEA adherence and the other popular project risk management frameworks.

H_2: There is significant difference between the mean scores of FMEA adherence and the other popular project risk management frameworks.

In addition to this, the perceived characteristics serve as influencing factors (persuasion) for trialability by the participants. In testing the innovation (trialability), the participating practitioners are engaged in a focus group discussion to brainstorm on the probable risk associated with failure modes that can occur in all stages of the software development. The trialability also extends to other stages of risk management activities (i.e., risk identification, analysis, and assessment). The study then examines and assesses the risk effects on project outcomes. Thus, the hypotheses for the evaluation are expressed thus:

H_{01}: There is no significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality).

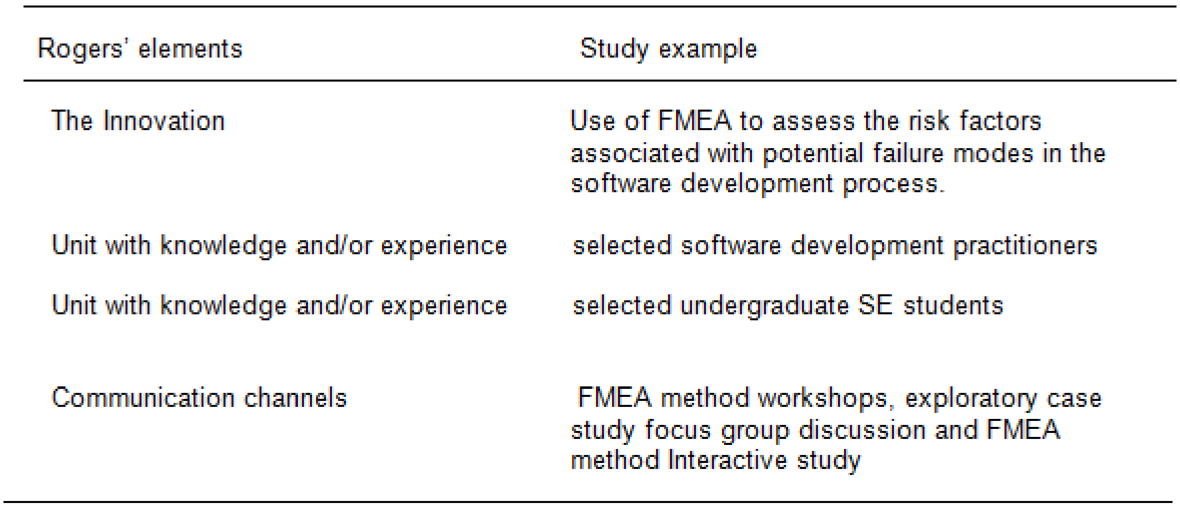
H_1: There is significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality)

The decision stage of the communication channel (as whether to adopt or reject the innovation) is considered an area for future research as earlier stated. Thus, the outcome of the trialability stage, which conducted the risk management activities, provided answers to the questions of this research.

Furthermore, to advance the spread of the innovation, the stages of communication channels identified by Rogers (1995) are adopted for an FMEA method interactive study with undergraduate software engineering students. Here, students are introduced to the procedural characteristics and unique capabilities of the FMEA method as a software project risk management tool (knowledge stage). They are also trained on how to use the FMEA method to conduct risk management process in software project development (persuasion and implementation stages). The diffusion of the innovation with students captures the process of introducing and engaging students in a practical training on how to use the FMEA method to manage the risks in software development projects in one of the bachelor-degree programmes. In spreading the innovation to the second social system (undergraduate SE students), the Rogers' diffusion theory was used to introduce and train the target community (undergraduate SE students) on FMEA strengths and weaknesses.

To summarize the implementations of the DIT as applied in this study, the innovation for this study was the use of FMEA to assess the risk factors associated with potential failure modes in the software development process. The unit with experience is software development practitioners. The unit with little or no experience with the innovation is the undergraduate software engineering students. The communication channels are FMEA method workshops, exploratory case study, focus group discussion, and FMEA method interactive study. Table 3.1 reflects how this study relates to Rogers' diffusion theory.

Table 3 1: Roger (1995) Diffusion Theory elements and study example



Rogers' elements	Study example
The Innovation	Use of FMEA to assess the risk factors associated with potential failure modes in the software development process.
Unit with knowledge and/or experience	selected software development practitioners
Unit with knowledge and/or experience	selected undergraduate SE students
Communication channels	FMEA method workshops, exploratory case study focus group discussion and FMEA method Interactive study

Even though, the DIT framework was found most appropriate for spreading the innovation, the theory does not recognize the perceived ease of use and usefulness, which act as direct determinants of the actual user adaptation of the innovation. Thus, a combination of models was necessary to achieve the aim of assessing the perceived ease of use and the perceived usefulness of the FMEA. The adoption of *Technology Acceptance Theory/Model (TAT/TAM)* was considered most appropriate theoretical framework because users' perceptions of usefulness and ease of use are significant determinants of user behavioural usage as clarified by TAT/TAM.

The next section discusses the TAT/TAM and its implication on using the FMEA as instrument to assess the risk factors associated with potential failure modes in the software development process.

3.1.4 Technology acceptance theory/model (TAT/TAM)

The TAT/TAM is an IS theory that frames the manner in which users accept and use an information system or technology. It is a model extended from the Theory of Reason Action (TRA) done by Davis (1989). TAM came into existence after the introduction of information systems into organizations (Momani & Jamous 2017). The TAM states that the perceptions of users about the usefulness and ease of use of a technology are significant determinants of technology acceptance or adoption (Davis 1989). The TAM is applied to investigate and determine the role of the end-user when new technology is initiated. It also analyses the impact of external variables on the acceptance and usage of technology (Momani & Jamous 2017).

According to Momani and Jamous (2017), the three stages that are involved in the development of TAM are: adoption, validation, and extension. TAM was tested and adopted in the adoption phase using multiple information system applications. In the validation phase, the TAM was observed to implement correct measurement of users' acceptance behaviour in different technologies. In the extension phase, many introduce some new variables and relationships between the TAM's constructs (Momani & Jamous 2017).

Over the years, TAM has come of age and has emerged as a strong and parsimonious means of presenting the antecedents usage of system through two factors: perceived ease of use and perceived usefulness of information system (Davis, 1989, 1993) Figure 3.2 presents the conceptual model of TAT/TAM.

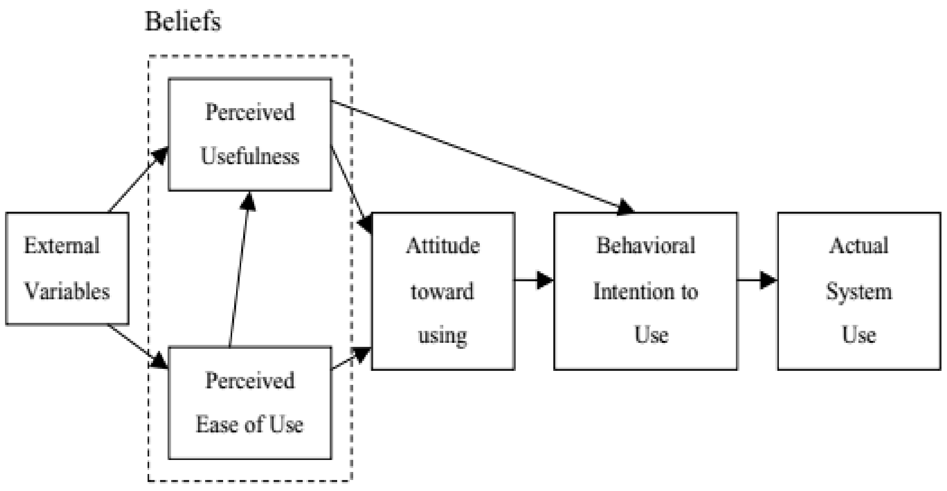


Figure 3 5: Conceptual model of TAM (source: Davis 1989)

The various constructs that formed the TAM are described according to Davis (1989) as:

Belief: This is the subjective probability the individual possesses about the outcome of performing the target behaviour will have.

Perceived ease of use: "The degree to which a person believes that using a particular system would be free of effort" (Davis 1998, p 320).

Perceived usefulness: "The degree to which a person believes that using a particular system would enhance his or her job performance" (Davis 1998, p 320).

External variables are important factor that are used to determine the attitude. The availability of TAT/TAM influences people's attitude and intention to use the technology. However, their views may change depending on other factors such as age and gender due to individual differences (Davis 1998).

The *actual system use* is the terminal where people engage with the technology. *Behavioural intention* is a factor that influences people to adopt the technology. The *attitude* (A) refers to the general impression people have about the technology and this influences the behavioural intention (Davis 1989).

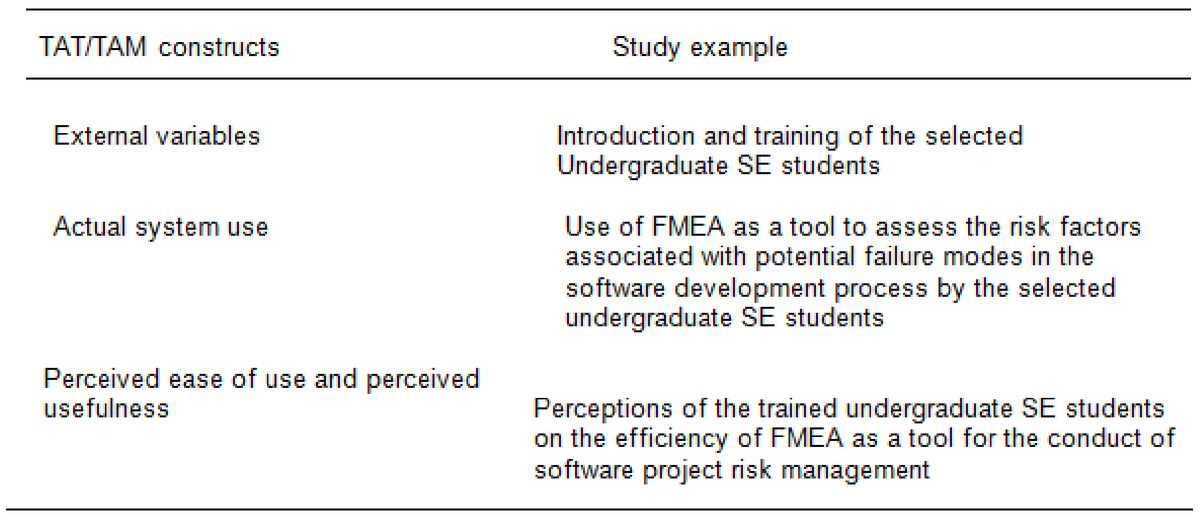
The application of TAM was adopted in this study through an extension of DIT by replacing the *complexity* construct from the communication channels of innovation with *perceived ease-of-use* and *perceived usefulness*. These two constructs measure the way in which the trained undergraduate SE students perceive the practicability and usability of the FMEA as a tool for the conduct of software project risk management activities as designed in the interactive study of field study 5 (discussed in chapter 4). Students' notable perception about the *usefulness* and *ease-of-use* are analysed to evaluate the efficiency of the FMEA application for the conduct of risk assessment in software development projects. Thus, the last hypothesis for the study can be expressed as:

Ho$_{3}$: FMEA method is not efficient for the conduct of software project risk management.

H$_{3}$: FMEA method is efficient for the conduct of software project risk management.

The implementation of the modified the TAM as applied in this study is presented in the Table 3.2.

Table 3 2: Davies (1998) TAT/TAM constructs relations with the study example



TAT/TAM constructs	Study example
External variables	Introduction and training of the selected Undergraduate SE students
Actual system use	Use of FMEA as a tool to assess the risk factors associated with potential failure modes in the software development process by the selected undergraduate SE students
Perceived ease of use and perceived usefulness	Perceptions of the trained undergraduate SE students on the efficiency of FMEA as a tool for the conduct of software project risk management

3.1.5 Strength and limitation of the chosen theoretical frameworks

The strength of the DIT model as adopted in this study is recognized based on its structure and applicability. The DIT is a well-developed theory, globally accepted and widely applied. In spite of its acceptability, early studies that applied the theory of spread of innovation and subsequent criticism discovered that innovative behaviour studies in the organization remain relatively underdeveloped due to their unconvincing, contradictory nature and are also characterized by a low level of explanation (Downs & Mohr 1976; Damanpouer 1991; Wolfe 1994). Since the DIT's publication in 1962, Rogers and his team have continuously improved the DIT through empirical research in 1971, 1983, 1995, and 2003. The consistent improvement of the theory by Rogers led to the application of the theory of Innovation spread in more than 5000 studies in various disciplines that cut across scientific, business, education and social studies (Rogers 2003). In addition to this, Robinson (2009) concluded that "the theory of Rogers has been tested through more than 6,000 research studies and this has proven its reliability" (Robinson 2009, p.1).

Furthermore, the DIT has been applied in different research studies related to software engineering in the past. For example, Livari (1996) employed the DIT to investigate the adaptation categories of the Computer-Aided Software Engineering (CASE) tools in organizations. In addition, Balzer, Litoiu, Müller, Smith, Storey, Tilley and Wong (2004) used Diffusion of Innovation Theory to conduct a long-term workshop series on Adoption-Centric Software Engineering (ACSE) to highlight the importance of addressing adoption problems in Software Engineering research.

As pointed earlier, the application of the DIT was adopted by various disciplines and its application cuts across countries and cultures (Anis 2009; Robinson 2009; Rogers 1995, 2003). For as long as the DIT is capable of explaining the process of innovation spread irrespective of discipline, culture and national border, the model will be useful in this current research.

In the case of TAM, the strength of the TAM as adopted in this study is recognized based on its simplicity and specification. It suggests a small number of variables, which jointly account for its use. The concepts are specific and easy to understand and can be modified through multiple system design and application. In addition, they can be generalized across different settings. Researchers (e.g., Davies et al. 1989) that applied AT/TAM gave the model empirical support in IT research. Davies, et al., discovered that TAM /TAT suggested software intention is better than the Theory of Reasoned Action.

However, empirical research of TAM/TAT showed that TAT only explains much of the variance in usage intention and self-reported usage (Davis 1989, 2003; Davies et al. 1989, 1992). This drawback was because TAM has not been tested with actual measures of usage behaviour. Reason for this could also be that TAM has not been tested as a complete model simultaneously; rather, various components of the model have been independently examined. This study incorporated the actual measure of usage to fully examine the extent to which TAM can help understand usage behaviour of trained undergraduate software engineering students that applied the FMEA as a tool to analyse and manage risks in software development projects.

The following section presents the conceptual framework for the conduct of the study.

3.2 Conceptual model for assessing risk factors in software development projects

The aim of this study is to assess the risk factors associated with potential failure modes in the software development process using the FMEA as risk assessment instrument. As a mixed-method study, the study intended to describe the research constructs and to establishing how the research constructs relate to the study. Despite the adoption of a combined model (DIT and TAM) that serves as the theoretical framework for this study, it is imperative to develop a conceptual model to illustrate the relationships among the various concepts that informed the study without ambiguity.

Huberman and Huberman (1994, p.18) describe a conceptual framework as:

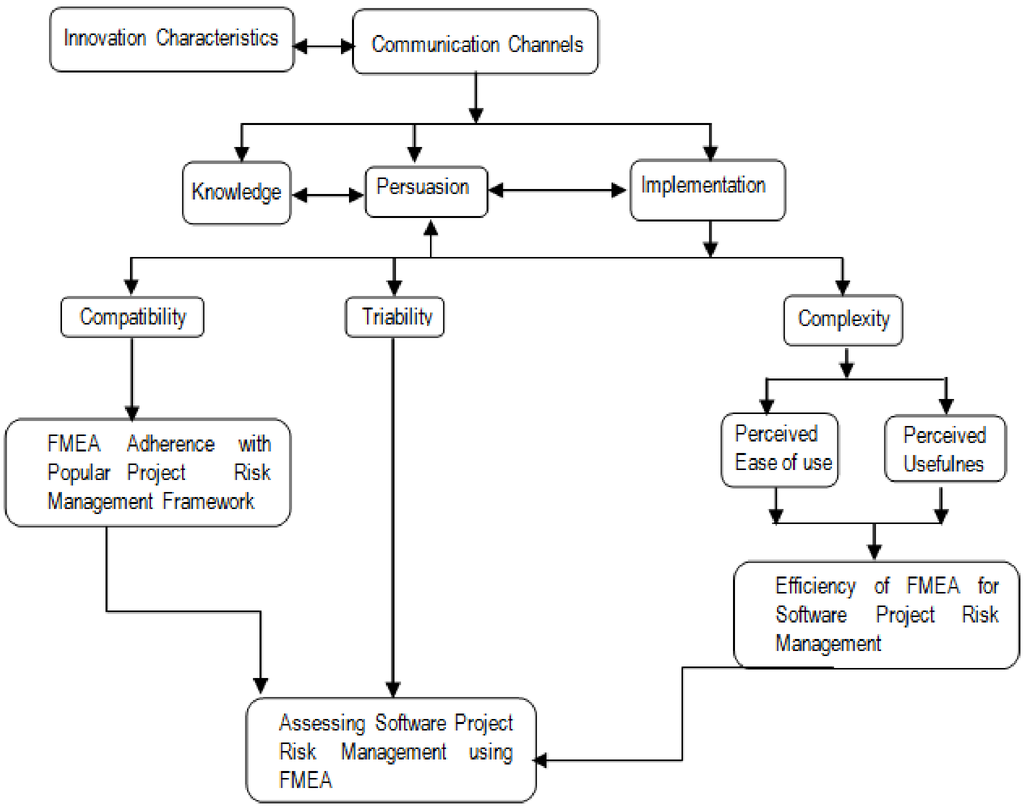
> “A conceptual framework explains either graphically or in a narrative form, the main dimensions to be studied, that is, the key factors or variables and the presumed relationships. A framework can be rudimentary or elaborate, theory driven or commonsensical, descriptive or causal” .

In this study, a conceptual framework was formulated using the seven constructs extracted from the combined models DIT and TAT/TAM (five from DIT and two from TAM/TAT) that relate to this study. These constructs are relevant to the study because they are direct determinants of users' perception of new idea, approach or innovation (Rogers 2003) and actual behavioural usage of a system (Davies 1989). In the case of this study, the five fundamental constructs extracted from DIT are considered as determinants to assess the perceptions which software development practitioners hold towards the use of FMEA to assess software project risk factors and the effects they considered the risk factors will have on project outcome, while the other two constructs adopted from TAM are considered direct determinants of behavioural usage of the FMEA for the conduct of software project risk management.

Each of the seven constructs presented in the conceptual framework addresses the research questions and is also applied in the hypothesis testing. For example, compatibility is concerned with software practitioners' perceptions of FMEA adherence to other project risk management models, while trialability identifies and assesses the software project risk factors' effect on project outcome as narrated and perceived by the software participants. The perceived ease of use and usefulness in the conceptual model is concerned with the efficiency of the FMEA for the conduct of software project risk management as perceived by the trained undergraduate SE students. Interestingly, all the constructs function harmoniously in an interrelated and interdependent way in using the FMEA to assessing the risk factors associated with potential failure modes in the software development process

.

In this section, the researcher engages with the constructs presented in the conceptual model (Figure 3.2) individually and then provides detailed explanations and relationships in relation to the implications to using the FMEA in assessing the risk factors associated with potential failure modes in the software development process. Figure 3.3 shows the conceptual model for assessing the risk factors associated with potential failure modes in the software development process.



Innovation Characteristics
Communication Channels
Knowledge
Persuasion
Implementation
Compatibility
Triability
Complexity
Perceived Ease of use
Perceived Usefulnes
FMEA Adherence with Popular Project Risk Management Framework
Efficiency of FMEA for Software Project Risk Management
Assessing Software Project Risk Management using FMEA

Figure 3 6: Conceptual model for assessing the risk factors associated with potential failure

3.2.1 Conceptual constructs' relationship with the study

This study is specifically concerned with the seven core constructs extracted from both the DIT and TAT/TAM. Five constructs extracted from the DIT are: knowledge, persuasion, trialability and implementation. The two constructs extracted from TAM are: perceived ease of use and perceived usefulness. The five constructs from DIT are direct determinants of users' rate of adoption of innovation (Rogers 2003). Similarly, the constructs: perceived ease of use and perceived usefulness are direct determinants of users' behavioural usage of Information system or technology.

However, in the case of this study, the five DIT constructs are regarded as the influencing factors, which gear software practitioners to learn and apply the experiences gained from the exposure and to use the FMEA for the practice of software project risk assessment. The TAM constructs are regarded as opinion held by the software engineering students on behaviour usage when they use the FMEA for the conduct of their software project risk management in one of their core undergraduate SE courses.

The seven core constructs that form the conceptual model appear to be independently presented, but the researcher believes they are linked together as the perceived characteristics of the FMEA and the channel of communication employed to spread the innovation to practitioners and students, influence the use of the FMEA to assess the software project risk factors and subsequent measure of actual behavioural usage. As a matter of fact, the constructs are interrelated and interdependent towards procedural fit assessment, the risk assessment and the actual user behaviour (efficiency). The only difference is that while some are explicit, that is external to the user (trialability, implementation, perceived ease of use and perceived usefulness), and others are implicit. That is to say they use internal to the user (knowledge and persuasion). Therefore, the constructs will be discussed in details as they relate to the study.

Characteristics of innovation: The characteristics of innovation are defined by Rogers (1998) as reasons for adoption of the innovation at all stages. Rogers identifies the following characteristics set for any innovation to be adopted: Relative advantage, Compatibility, Trialability, Complexity and Observability. These characteristics set have already been discussed in detail in section 3.1.1. In this study, the characteristics of FMEA reviewed from literature (section 2.5.1 to section 2.5.1.4) are regarded as the perceived characteristics of FMEA.

Communication Channel: This is regarded as a means through which the innovation gets communicated to people in order to be adopted (Rogers 1998, 2003). Rogers also identifies communication channels as an important element of the diffusion process. Rogers (1998) identified a linear five-stage mental process through which the perceived innovation characteristics can be spread among members of a social system.

In this study, the perceived FMEA characteristics (reviewed in section 2.5.1 to section 2.5.1.4) are communicated to the social systems by means of workshop (field study 1 in section 4.8), exploratory case study (Field study 1 in section 4.8), focus group discussion (Field study 1 in section 4.8; Field study 2 in section 4.9) and FMEA interactive study with students (Field study 5 in section 4.12). Software development practitioners and undergraduate SE students represent the social systems.

Knowledge: This is the first stage of spreading the innovation. Here, members of the social systems are exposed to the new ideas/approaches that are considered as innovation. At initial stage, it is assumed that they have very limited information to decide to adopt an innovation and also not sufficiently inspired to find out more. In this stage, members of the community are exposed to the new ideas or products and gather more information about them (Rogers 1998).

Persuasion: Persuasion is the location at which the members of the community are open to the new approach or idea (Rogers 1998). This stage involves formation of an attitude towards innovation itself. It is the stage where the users actively find out more knowledge that influences their decision on the innovation.

In this study, lecture presentations and training that informed student developers on the approach and usage of FMEA for the conduct of software project risk management were delivered through the various channels of diffusion stated above. These activities are considered effective enough to expose and train the participants on how to use the FMEA for the conduct of software project risk management.

Implementation: In this stage, an individual uses the innovation to evaluate its benefits (Rogers 1998). Prospective adopters may also find out more information to either justify the usability of the innovation or reject it. In the context of this study, three characteristics of innovation are implemented with respect to the FMEA for the conduct of software project risk management. First: *Compatibility*, software development practitioners were involved in a questionnaire survey to assess the level of adherence of the FMEA framework with the popular project risk management frameworks. Second: *Trialability:* here, software practitioners are engaged in an evidence-based research using the FMEA to assess software project risk factors. A questionnaire survey was then conducted to assess the practitioners' perceptions on the effect of software project risk factors on the project outcome. Lastly, after their training in an FMEA interactive study, undergraduate SE students were guided to use FMEA techniques for their practical SE course. They were required to use FMEA for software project risk management. At the end of the practical engagement, participating students were requested to complete questionnaire to assess their opinion on their *perceived ease of use* and *perceived usefulness* of the FMEA tool.

Perceived ease of use: “The degree to which a person believes that using a particular system would be free of effort” (Davis 1998, p 320).

Perceived usefulness*:* “The degree to which a person believes that using a particular system would enhance his or her job performance” (Davis 1998, p. 320).

Their perceptions on the *perceived ease of use* and *perceived usefulness* are determinants of efficiency of the FMEA in the conduct of software project risk management.

The next section presents the summary of the chapter.

3.3 Chapter summary

This chapter presents the relevant theories that served as the theoretical frameworks and support to the entire research project. A combined theory of DIT and TAM was the theoretical model that served as the framework of this study. The chapter provides a comprehensive description of the genesis and elements of the DIT and TAM models. A justification for adopting the models is also provided in this chapter. The chapter further describes how the seven core constructs extracted from the DIT and TAM were modified to generate a conceptual framework that explains the findings of the study. The conceptual model contains the seven fundamental constructs extracted from the DIT and TAM in relation to this study and how each of the constructs concerns the use of the FMEA for risk factor assessment in software development projects.

The next chapter presents the qualitative and the quantitative approaches and reliable strategies deployed to arrive at the findings of the study.

CHAPTER FOUR

Research Design and Methodology

The chapter provides a clear focus on the research design and the data collection techniques used in gathering relevant data for this research and then a justification for the choice made. The chapter begins with explaining a suitable epistemology paradigm for which the research is located. The chapter then describes the design of the project and then explains the strategy that is used in conducting the study. An exhaustive explanation of the research population and sampling techniques adopted are then presented. In addition to this, the chapter describes how various field studies in this research were conducted to assess the risk factors associated with potential failure modes in the software development process. Five field studies were carried out to be able to generate both qualitative and quantitative data that will be analysed and interpreted to provide answers to the research questions and hypothesis. The field studies are sequentially reported as they were carried out.

4.0 Introduction

In every research, there must be a clearly defined methodology, philosophical orientation, tools and techniques for data collection and analysis. Research methodology refers to the combination of processes, methods and tools as well as the underlying theoretical and philosophical assumptions and their implications for the methods adopted (Creswell & Plano 2011; Saunders, Lewis & Thornhill 2009). In section 1.7, a brief discussion of the research methodology was presented. In this chapter, a detailed account of the research methodology and justification for the techniques chosen for the data collection methods is presented. The chapter begins by justifying the selected epistemological paradigm, known as the philosophical foundations, on which the study is predicated. The critical realism is considered most appropriate to foreground the study. This discussion is followed by descriptions of research design and strategies adopted for the conduct of the research and the choice of data collection and analysis approaches.

The next section presents the epistemology paradigm of the research.

4.1 Research philosophy

This research is based on Critical Realism (CR) epistemology orientation. It requires a theory that stresses how people perceive the universe as theory-laden, but not theory-determined (Danermark, Ekström, Jakobsen & Karlsson 2002). It aligns with the fact that there is a real social world we can strive to know or perceive through philosophy and social science; and also, certain knowledge can be more realistic than other form of knowledge.

Over the years, several research in the field of social science have been conducted using the (CR) as a philosophical framework. The CR was carved out of the positivist/constructivist 'paradigm wars' of the 1980s (Denzin & Lincoln 2011). It was propounded in the 1970s and 1980s from the work of Bhaskar. It combines the features of both positivism and constructivism to provide a clearer ontology and epistemology, but draws elements from both methodological strains in its account of ontology and epistemology, this making CR a comprehensive philosophy of science (Brown, Fleetwood & Roberts 2002). Critical realists such as Andrew Collier (1994) and Lawson (1997) further manipulated the CR. It later emerged as a scientific alternative to both positivism and constructivism (Denzin & Lincoln 2011). As a philosophy of science, CR functions as a general methodological framework for research but is not associated with any particular set of methods (Brown et al. 2002; Nielsen 2002).

As a realist, the researcher believes that knowledge is dynamic – people's perception of a situation can change. As objects exist regardless of how people perceive them, human do construct knowledge about them. While human strives to construct new knowledge, sometimes misconceptions about theories can set in, thus people's perceptions about knowledge of the world is dynamic. This CR epistemological perspective means that the researcher recognizes that using the FMEA method to assess the risk associated with potential failure modes in the software development process over time can be more beneficial, thereby accepted or rejected, notwithstanding the researcher's attempt to ensure its trustworthiness and practical adequacy. A theory is not intransitive, as reality is.

This research follows the typical stages of mixed-method research while applying important considerations of CR ontology and epistemology by using the FMEA method to assess the likely effect of potential failure modes in software development projects and it further engages in a critical dialogue with software development practitioners to develop a rational judgement about failure modes' effect on project outcome. It also seeks to establish and interpret the behavioural use of the FMEA methodology for the conduct of software project risk management based on the practitioners' perceptions.

4.1.1 Preliminary study

This section presents a description of the preliminary study that was conducted prior to the commencement of the fieldwork. The preliminary study was conducted on the application of the FMEA on only one component of project risk management process in SE to assess the effectiveness of the method in software project risk assessment. To address the issue concerning bias factor in the selection of the FMEA method for the conduct of risk management activities, the method implementation and findings of the preliminary work were presented in academic forum for deliberations.

As a preliminary study to the PhD research, identification of risky potential failure modes in the process of software development was conducted using the FMEA method. At the risk identification stage, the FMEA procedural requirement stipulates three procedural steps:

1. Selecting experience team of between (9 to 15 members)
2. Reviewing the process or product
3. Brainstorming on unknown risks in the process

The risk identification process was conducted using the three procedural steps of the FMEA in an exploratory research involving 15 randomly selected software practitioners in Zamfara State in Nigeria. During the study, the selected practitioners engaged in a focus group discussion to brainstorm on possible failure actions, events, occurrences or situations in the process of software development, whose uncertainty can pose severe threat to software project outcome. Five stages of development process were studied (Planning, requirement specification, design, coding and Installation). The discussion also explored the causative factors of the identified failure modes, their likely effects on the project outcome and their symptomatic mechanism for detecting the identified failure modes.

Miles and Huberman (1994) method was used to analyse the qualitative data that emerged from the preliminary study. It is important to report here that the research findings generated positive interest from panellist and participants at different academic gatherings where the study was presented. Also, motivation received from one of the conference organizers resulted into publication of the preliminary findings (Abubakar & Lawal 2020).

Extensive reviews of the findings of the preliminary study (Abubakar & Lawal 2020) together with other related literatures in the field (such as: Khaiyum & Kumaraswamy 2014; Gupta et al. 2012; Mitrabinda & Durga 2011; Georgieva 2010; Hassan et al. 2005) informed the adoption of the FMEA and provided a preliminary justification for the selection of the proposed FMEA method for the conduct of risk management activities and were helpful in formulating the research questions. The reviews were also helpful for developing the questionnaires, the focus group protocol adopted for the PhD research and more importantly in shaping the design of the PhD research.

4.2 The research design

This research adopted a combination of case study and survey design. The research is designed based on the seven fundamental constructs extracted from the DIT and TAM in relation to the study. Five constructs were extracted from the DIT and two constructs extracted from TAM. The five constructs extracted from the DIT are: knowledge, persuasion, compatibility, trialability and implementation. The two constructs extracted from TAM are perceived ease of use and perceived usefulness. These seven constructs were discussed in detail as related to this study in section 3.2.1. Five field studies were designed to validate these contributions practically. The field studies include two exploratory case studies, one validation - survey and two interactive case studies. The field studies required spreading an innovation using several channels that include workshops, focus group discussions, lecture presentations and training that would inform both the software development practitioners and SE students on the use of FMEA to assess the risk factors that may lead to failure modes in software development process.

The field studies started with an exploratory case study and focus group discussion with selected 15 software development practitioners in Nigeria (Field studies 1 & 2). The studies engaged the practitioners in a workshop and in a focus group discussion to discuss the possibility of using the FMEA for the conduct of project risk management. The study also identified the probable failure modes in the software development process and analysed their cause-effect possibilities. The exploratory case study and the focus group discussion are purely qualitative, thus, adopted inductive approach to develop a rational judgment on the issues discussed.

Part of the studies, specifically, Field studies 4 and 5, were designed to assess the risk factors associated with potential failure modes in the software development process using the FMEA theory and determine the developers' perception of the effectiveness of using the FMEA for the conduct of the risk assessment. In this case, the Field studies 4 and 5 are theory-laden studies where adopted theories require testing their practice. Thus, a deductive research approach was adopted.

The next section presents the strategy adopted for the conduct of the research

4.3 The research strategy

In this study, a mixed – method research strategy was applied where both quantitative and largely, qualitative, methods were used to provide answer to the research questions as raised in section 1.4. The quantitative research is concerned with studying social phenomenon at the event level and it employs a very objective procedure to handle the collected data (Doyle, Brady & Byrne 2009).   However, the qualitative research focuses on the social events and their causal factors so as to determine their real causes (Sayer 1992). The research strategy investigates how generative procedure works and explains the interaction between the powers that develop a social phenomenon (Danermark 2002).

The two strategies (both the quantitative and qualitative) are concerned with positivists and interpretivism epistemology orientations respectively, and can be applied to understand the world (Weber 2004). In addition, both approaches are not mutually exclusive and can also be used to analyse data (de Villers 2005). The quantitative approach represents a positivist epistemological orientation with an aim of describing and predicting, based on empirical facts. It avoids any value judgments or subjective interpretation of the researcher (Scapens 1990). The main goal of this approach is to obtain a clearer objective and reliable scientific results that is able to represent the entire population and could be generalised over similar ones.

On the other hand, qualitative research adopts interpretive strategy to determine how to acquire and use knowledge (Scotland 2012). The reason for such orientation is that the use of scientific research to investigate social phenomenon is not appropriate (Riegler 2012). In other words, researchers have to source for detail data and information by engaging physically with the social constituents to fully investigate the social system (Ulum 2016). In addition, this research approach adopts the realism ontological orientation from which the reality of a situation is obtained from the physical engagement between the researcher and members of the target population. Thus, real picture and accurate explanation of a phenomenon is obtained when researcher fully participate in the investigation process of such phenomenon (Ulum 2016).

The next section presents the study population and the sampling techniques adopted for the conduct of this study.

4.4 Population and sampling

In order to find an appropriate representation of respondents in this research, the research was conducted following the use of convenience sampling technique. The selection and choice criteria adopted was based on easy availability and accessibility as suggested by Robson (2002). Three target groups represent the population of the study. These groups are presented as follows:

(i) In the first group, 5 software organizations from Nigeria and which are purposefully chosen using the designed criteria that will categorize selected organizations as: business type; size and software specific development product. The participants from the selected organizations are purposefully selected based on certain required experiences that cover the interest of the proposed research. The minimum requirements are that developers or participants have experience in at least one successful software project using any development model such as waterfall, agile (such as SCRUM and X-programming) and or iterative and RUP and have been involved in risk management processes in their organizations. In addition to this, participants must have expertise in any of the listed software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control. Fifteen (15) out of the twenty-one (21) software development experts that were invited turned up and participated in the group discussion. The selection size was considered sufficient based on the required team size (6 - 15) to conduct FMEA process (Tay & Lim 2016). These sets of participants are engaged in the field studies 1, 2 and 4, as presented in the participant column of Table 4.1.

(ii) The second group is composed of software development practitioners and researchers in software development who have actual experiences in software development projects using varieties of software models and have expertise in at least one of the software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control.. To enable representatives from all over the world, request for data was sent to the participants and the following source was chosen using a purposeful random selection method (and respondents were addressed via email): The authors of articles and participants in panels at all the proceedings and workshops of the International Conferences on Software Engineering sponsored by ACM and IEEE CS between the periods of 2007 to 2017 were consulted. Two-hundred and fifty (250) practitioners were contacted via email. One-hundred and twenty three (123) out of the invited two-hundred and fifty

(250) software practitioners around the world signified intention to participate in the survey study via email.

The decision to cancel the online questionnaires made it impractical for some software experts to respond to the e-mailed questionnaires even after their initial agreement to participate in the research. After several communications via email inform of reminders, only seventy-seven (77) out of the 123 practitioners around the world that initially signified interest to participate completed and returned the survey questionnaire via email (Field study 3).

(iii) The third group is composed of selected group of undergraduate computer science (CS) students that registered for SE as a core course in their programme of study at a public university in Nigeria. Interested students were given the consent form to complete (see Appendix 2A). Fifty-three students signified interest, however, 50 out of the 53 that completed and returned the consent forms turned out to participate in the survey (Field study 5).

The following section presents the field studies and the instrumentation design for the study.

4.5 Field studies and instrumentation design

This section describes the design of the various field studies carried out for both the case study and survey research with reference to the research instrument used for the study.

4.5.1 Field studies design

As earlier stated, five field studies were conducted in this research to generate data that were analysed to provide answers to the research questions. However, most of the field studies conducted are characterized as survey and case studies (such as: exploratory case study with software development practitioners in Nigeria, survey study with researchers in SE around the world, FMEA process to assess risks in software projects and FMEA interactive study with undergraduate SE students). The reason for conducting the field studies is that the challenges in risk management are more practical oriented; hence, such practicality oriented and industry-verified research are the appropriate approaches to validate the FMEA method and the research findings. Detail structure of the field study design architecture is presented in Table 4.1.

Table 4 1:: Field study design, instruments used for data collection and the study objectives

		Instruments for the Data Collection				
Field Study	**Participants**	**Focus Group Discussion**	**Questionnaires**	**Post-mortem Study**	**FMEA Worksheet**	**Objectives of the study**
Study 1: Exploratory case study with software experts in Nigeria	*15 Software experts in Nigeria*	✓	✓ FMEA Evaluation Questionnaire (FMEAEQ) (see Appendix B1)	✓		1. To examine the procedural requirements of the FMEA and that of popular project risk management frameworks 2. To assess the level of adherence of the FMEA standard procedures to the formal project risk management frameworks 3.Validate the various artefacts and data to be used for the research
Study 2: Focus group study with software experts in Nigeria	*15 Software experts in Nigeria*	✓	✓ Failure Mode Effect on Project Outcome Questionnaire (FMEOPOQ) (see Appendix B2)	✓	✓	1.To identify the risky failure modes in the studied stages of the SDLC 2. To identify the failure likely causes and their detection methods 3. To assess the effect of potential failure modes on project outcome (cost, time and scope)
Study 3: :Survey Study with experts and researchers in SE around the world	77 Software experts and researchers in SE around the world		✓ Potential Failure Modes Effect on Project Outcome Questionnaire (PFMEOPOQ) (see Appendix B3)			To validate the findings of field study 2
Study 4: FMEA Process with software experts in Nigeria to assess risk factors associated with potential FMs in the SDLC	15 *Software experts in Nigeria*				✓	To use FMEA method to assess, prioritize and classify risks factors associated with potential FMs in the studied stages of the software development process
Study 5: FMEA interactive study with SE undergraduate students	50 SE undergraduate students		✓ FMEA Evaluation Questionnaire (FMEAEQ) (see Appendix B4)	✓	✓	To evaluate the ease of use and the practicability of adopting FMEA in software project risk management

Before the commencement of the fieldwork, series of consultations were made through the National Information Technology Development Agency (NITDA) in Nigeria with respect to the research intentions and to interface with the target population in Nigeria (i.e., the software development practitioners across the five (5) targeted software industries in Nigeria). The agency also interfaced with a software industry that was proposed to be used as the research centre. Formal approvals granting the permission to participate in the research were obtained from the management of all the contacted software industries. Also, an ethical approval that duly granted the use of the company as the base for conducting the research activities was obtained from a software industry in Abuja – Nigeria (see Appendix 1C).

Similarly, attention was given to ethical issues (see Appendix 1B and 1C) and after due application for the conduct of the field work, research Ethics Clearance was granted by the UNISA College of Science, Engineering and Technology's (CSET) Research and Ethics Committee for the proposed research. The ethics approval was granted for a period of five years, from 03 December 2018 to 03 December 2023 (see Appendix 1A).

4.6 Field study 1: Exploratory case study in Abuja - Nigeria

Background

The review of the Abubakar and Lawal (2020) paper informed the selection and adoption of the FMEA method as a risk assessment tool in this research. The research field work began after due consultations with participants and agencies whose facilities are required for the conduct of the study. The Field study 1 started with an exploratory case study to: (a) examine the procedural requirements of the FMEA and popular project risk management frameworks, and (b) assess the level of adherence of the FMEA standard procedures with the formal project risk management frameworks using the PMBOK, SEI and Boehm as multiple case studies. An exploratory case study research is described as a qualitative approach in which the investigator explores a bounded system (a case) or multiple bounded systems (cases) over time, through detailed, in-depth data collection involving multiple sources of information (such as: observations, interviews, audio-visual material, documents and reports), and reports a case description and case-based themes (Creswell 2017).

Researchers in case studies use data triangulation as part of their data collection strategy, resulting in a detailed case description (Ridder 2016). Potential benefits of case study research reflect an in-depth description and analysis to garner a detailed understanding of “how” and “why” things happen (Yin 2015).

The Field study 1 began with a workshop on a procedural requirement on the case study (PMBOK, SEI-CRM and the Boehm). It was followed by a focus group, then survey with the representative experts and researchers in Software Engineering. The activities were conducted at the conference centre of a software industry in Abuja – Nigeria.

4.6.1 The exploratory study design

The study was conducted with a selected group of experienced software development practitioners with expertise in at least one of the software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control, from five different software development industries in Nigeria. The research was conducted in collaboration with a software development company based in Abuja Nigeria (where the workshop took place). Fifteen out of the twenty-one software development experts that were invited turned up and participated in the group discussion. The selection size was based on the required team size (6 - 15) to conduct FMEA process (Tay & Lim 2016). Throughout the field studies that engaged the selected software experts in Nigeria (N = 15) in this research, special attention was paid to the ethical aspects of consent form and approval to use facilities for research (see Appendix 1B &1C respectively). This was ensured to strengthening the data integrity and to protect the confidentiality of the participants.

Four working days of two weeks were used to conduct the exploratory research.

- Day 1: Three hours were spent on workshop and method introduction on FMEA model.
- Day 2: Three hours were spent on workshop and method introduction on PMBOK model.
- Day 3: Three hours were spent on workshop and method introduction on SEI-CRM model.

- Day 4: Three hours were spent on workshop and method introduction on Boehm's model.

During the exploratory study, the researcher engaged a team of selected software development experts in a focus group discussion to brainstorm on the procedural similarities that exist between the FMEA model and that of the formal project risk management model. The study focused on a risk management methodology and used four well-known sets of concepts: PMBOK, the Software Engineering Institute (SEI)'s software project risk management model, the Boehm's risk management model and the FMEA as a multiple case study.

Prior to the focus group discussions, the researcher anchored a three-hour (3 hrs) mini workshop each day of three working days with the aim of diffusing the relative advantage of the proposed method (FMEA) with reference to the concepts and procedural requirements of the case settings (i.e., FMEA, PMBOK, SEI-CRM and Boehm) models. The three-hour workshop was organized by the researcher with the selected team on the topic: Concept and procedure of FMEA, the content of chapter 11 of the PMBOK, the concept and the six step procedures of the SEI software risk management model and the steps of Boehm's software project risk management model. The workshop was conducted to introduce the concept of each of the case models, motivating participants to investigate it further on their own, or to demonstrate and encourage how the practices of actual methods are conducted. The researcher also presented a brief review of the existing tables of Occurrence, Severity and Detection during the workshop.

After the workshop, an exploratory study on the case settings was conducted. During the study, the researcher engaged the participants in a focus group study on an adherence analysis between the procedure for conducting FMEA (Georgieva 2010) and PMBOK (PMI 2004, CH 11), the procedural steps of SEI model and the steps of Boehm' risk management model to arrive at a conclusion. Similarly, the review of the existing tables culminated in the development of final operational tables of Occurrence, Severity and Detection to be used for the assessment of risk factors in the study (see Tables 4.2 to 4.4).

After the exploratory study, a survey study using questionnaire (FMEAAQ) was conducted with same members of the team that participated (15 software development experts) to assess the level of adherence of the procedural requirements of the FMEA model and other prominent project risk management frameworks. Fifteen participants completed the questionnaire. The questionnaire was used to gather and analyse participants' perceptions on how well the standard procedures of FMEA satisfy the procedural requirements of prominent standard project risk management models (PMBOK, SEI-CRM & Boehm).

The survey was conducted after the participants filled-in the consent form (see Appendix 1B) signifying their intention to participate in the survey. Throughout the survey process to the process of data analysis, due attention was paid to the ethical issues as specified in the consent form.

4.6.2 The Exploratory method implementation process

4.6.2.1 Focus group meetings

A focus group meeting was conducted to identify the common procedural requirements between the FMEA method and that of the prominent standard project risk management models (PMBOK, SEI-CRM & Boehm). A three-day focus group session was organised with each meeting lasting for a period of three hours. The sessions were coordinated and certain predefined guidelines were followed to ensure that the sessions remain focused on the earlier agreed theme. The agenda of each of the meetings were FMEA standard procedures, FMEA adherence analysis to other standard project risk management frameworks and reviews of the existing tables of Occurrence, Severity and Detection respectively. Detailed outline of the discussions is presented in Table 3.2.

4.6.2.2 Structure of the FMEAAQ

The FMEAAQ has three sections with a total of thirty-two questions and consists of five printed pages. The questions are self-designed by the researcher using the outcome of the focus group discussion as background. Most of the questions are close-ended. The purpose of this questionnaire is to establish a definitive conclusion on whether FMEA model can be considered a proficient and reliable tool for use in project risk management, more specifically, in software project risk management. The survey was also intended to identify the strengths and weaknesses of the FMEA as a software risk management tool. The FMEAAQ is provided in Appendix B1.

Section 1 of the questionnaire elicited about participants' personal information. In Section 2, participants are asked to rank the listed FMEA standard procedures they consider (as: "Highly Important" (HI), "Important" (I), "Less Important" (LI) and "NOT Important" (NI)) for risk management process and to state their application. In section 3, questions related to FMEA adherence with popular project risk management standard (PMBOK, SEI & Boehm) were asked. The entire fifteen (15) questionnaire forms administered by the researcher were correctly filled and returned making the returned questionnaire rate 100%.

The decision rule consideration for the study are: any questionnaire item with a mean value of 2.50 or above are interpreted to mean a high level of importance or adherence of such item while a mean score of below 2.50 indicates a low-level importance or adherence response for the item. Mean statistic was used to analyse the result while hypotheses were tested at 0.05 level of significance using the ANOVA. The ANOVA was chosen because of its capability to analyse and interpret both large and small number samples.

4.6.2.3 Control of validity of the exploratory case study

In order to ensure that the research constructs in the exploratory case study was valid and tailored towards the goals of the study, three techniques were used. These are:

- ❖ First, the researcher ensured that the timing, focal content and arrangement of each of the sessions of the study and the presentations therein are kept constraint.
- ❖ Second, all discussions and activities that took place in the exploratory study were tape-recorded using audio and video recording devices to ensure that instrumentation errors were reduced to the barest minimum and to ensure that none of the issues raised was missing.

- ❖ Third, all the analysis and interpretations made from the results obtained in the study were reviewed by three research experts from the Computer Science (CS) Education and Measurements and Evaluation Departments for validation from a recognised public University in Nigeria to reduce potential bias.

4.6.2.4 The Field study 1 conclusion

The exploratory case study explored and compared the procedural requirements of the FMEA with the prominent project risk management frameworks. Prior to the exploratory study, a workshop to spread the idea of using the FMEA for project risk management was organized and facilitated by the researcher. The researcher plays the role of the innovator that compares the innovation with the existing system. The method introduction of FMEA and the prominent project risk management frameworks (PMBOK, SEI & Boehm) formed the theme of discussion at the workshop.

The method requirements of risk identification, risk analysis, risk prioritization, risk resolution and risk control were the main tasks in the focus group discussion part of this study. The study method, though largely qualitative (focus group discussion) established knowledge about the FMEA procedural requirements that are important for project risk management, while the quantitative approach (the use of FMEA Adherence Questionnaire) assessed the level of adherence of the FMEA standard procedures to frameworks of the PMBOK, SEI and Boehm. In the study, practitioners used the knowledge gained about the FMEA standard procedures to discuss the relative advantage and compare same with the method requirements of prominent standard project risk management frameworks (PMBOK, SEI & Boehm). Results of the exploratory case study are presented and analysed in chapter five.

4.7 Field Study 2: Focus group study at Abuja - Nigeria

Background

The literature review conducted for the purpose of this research revealed that research on risk factors in software development projects are not detailed enough to inform software practitioners on other important dimensions that deserve more attention. Most of the reported risk factors do not capture in detail, the technical uncertainties (e.g., potential failure modes) that can occur during the development process, which often pose greater threat to the success of software development projects. Also, while the classification and cause-effective analysis of the risk factors provide useful information that empowers managers to probe risk levels, previous researches on software risk factors do not provide project managers with useful information that will assist them to hypothesize a tailored action plan for countering the threat posed by the identified risk.

At the end of the focus group session, a survey study was conducted using the FMEOPOQ to assess the effect of the identified probable failure modes on the software project outcomes (cost, time & scope). The identified failure modes from the focus group discussions were used as input for the design of the FMEOPOQ. All questions from the FMEOPOQ are close-ended. The questionnaire enquired from the software development experts in Nigeria evaluation of what they perceived as the effect of the listed probable failure modes on the project outcome (cost, time & scope). Participants are requested to select from the Likert scale of VH = Very High (5), H = HIGH (4), M = Moderate (3), L = Low (2) and VL = Very Low (1) to evaluate the effect of the listed failure modes in the SDLC on the project outcomes (cost, time & scope).

4.7.1 The focus group study design

The focus group discussion adopted similar research design techniques that were used in the Field study 1: same participants, same research techniques (secondary research and formal qualitative research technique) to attain the stated objectives of this study.

The secondary research technique used for the Field study 2 reviewed the following literature:

- The report of the preliminary study (Abubakar & Lawal 2020) published in the *International Journal of Science for Global Sustainability* 'Exploring the Probable Failure Modes in the Software Development Process'
- The work of Hijazi et al. (2014a) 'Risk Factors in Software Development Phases'.

- The research of Hijazi et al. (2014b) ‘Identifying causality relationship between Software Projects Risk Factors’.
- The research of Georgieva (2010) ‘Conducting FMEA over the software development process’.
- Review of the five studied stages of the software development process to be used for the research with reference to the ISO (1995). ISO/IEC 12207.

The formal qualitative research technique used questioning techniques that further triggered an in-depth discussion on the subject matter following the guiding questions and protocols stated on pages 3 and 4 of the focus group discussion guides (see Appendix 2A), as presented in the next section.

4.7.2 The focus group method implementation process

In this section, discussion is focused on identification and analysis of risky probable failure modes that exist in all the studied stages of the software development process (i.e., Planning, Requirement Analysis, Design, Implementation and Coding, Installation and Maintenance).

After observing all the necessary protocols, the meeting session began with the distribution of artefacts to all discussants in the meeting. The artefact comprises all the 4 earlier listed secondary research documents (see section 4.7.1) and a brief description of the 5 studied stages of the software development process. After the distribution of the documents, the facilitator read the introductory question as:

“Dear participants, you are given five minutes to study the distributed artefacts and think about your experiences on the activities at each stage of the software development process in any of the software project development you are involved and tell this gathering the manner in which you think a failure can occur any of these stages. Let’s start from the top to down (i.e., planning down to maintenance stage).

“Question: *Is anyone happy to share his or her experience of the Planning stage and identify those activities or events that, if not effectively or correctly handled can impact the outcome of the project?”*

Other guiding questions that were used to elicit responses from discussants are presented in focus group guide/protocol in Appendix 1C. All responses generated from the focus group meetings were analysed using the Miles and Huberman (1994) method of analysing qualitative data. The original responses were recorded and transcribed into document containing list of all probable failure modes in the software development process, their likely causes and effect and the corresponding failure detection method. These set of questions were repeated for other four stages of the software development process (i.e., Requirement Analysis, Design; Implementation and Coding, Installation and Maintenance).

4.7.3 The Field study 2 conclusions

The theme of the Field study 2 is *the identification of probable failure modes in the software development process: causes-effect analysis and possible failure detection methods.* Focus group discussion method was used mainly to extract qualitative information about the theme of the study. The study was qualitative which intended to explore and identify the risky probable failure modes that can exist in all the studied stages of the software development process (i.e., Planning, Requirement Analysis, Design, Implementation and Coding, Installation and Maintenance). The study used secondary research and formal qualitative research techniques to accomplish the stated objectives of the study.

One important aspect of this study that makes it systematic and comprehensive for failure identification and analysis method is the approach of combining brainstorming and checklist of possible failures from previous research. This was further supported by the techniques adopted for the selection of participants (software practitioners) with different backgrounds and expertise in software development projects that participated in the study. However, the focus group study conducted in this work did not consider other dimension of risk factors (such as task, client, environment, self, etc.) in software development projects.

4.8 Field study 3: survey study with experts and researchers in SE

Background

This study session was conducted to generate a list of potential failure modes that can exist in the software development process and to assess their likely effect on the software project outcomes. The study was also aimed at validating the findings of the previous study session (Field study 2). Here, survey research was conducted using a questionnaire, titled: Potential Failure Modes Effect on Project Outcome Questionnaire (PFMEOPOQ) as research instrument (see Appendix B3). The PFMEOPOQ containing a checklist of all the suggested probable failure modes in Field study 2 were organised according to the studied stages of the software development process and distributed to all interested experts and researchers in software engineering across the globe via e-mail.

Before the distribution of the questionnaire to the interested participants, an introductory letter notifying prospective participants about the aim of the research and seeking their interest and consent to participate in the research was sent to about 250 prospective participants across the world via their e-mails. This is presented as the anonymous page (see front page of Appendix B3). This was intended to ensure larger opinion concerning the theme of the survey and to validate the findings of the Field study 2. Participants that have confirmed their interest to participate in the research were asked to fill the consent form (Appendix 1B). Questions from the PFMEOPOQ requested the participants to express their opinions on whether the identified probable FMs listed in the questionnaires are potential threat to the software project outcome or not, and also to assess their likely effect on software project outcomes.

The questionnaire was chosen as instrument for data collection because of the need to reach out to a large number of researchers and experts in SE globally in a relatively easy and economically efficient manner in order to establish a generalised outcome from the research. In addition, there was previous information, which formed the basis for adequately prepared questionnaire that will provide quantifiable answers to the theme of the study in a way that will be relatively easy to analyse by the researcher.

In addition, questions that were asked in the questionnaires are appropriate. It was ensured that they provided detailed insights to the research topic as there are previous information on the subject, which were gathered from previous session (Field study 2). On the one hand, the questionnaire provided adequate options for respondents to select from.

4.8.1 The Survey study design

In this research, descriptive survey design was employed as the study design while questionnaire (PFMEOPOQ) was used as research instrument (see Appendix B3). The researcher adopted the list of probable failure modes identified in Field study 2 for the questionnaire design. This method is deemed appropriate because of the need to gather the opinion of a larger size of professionals concerning reliability and the consequential effect of the established failure modes on software project outcomes.

The population for the study is composed of all researchers and experts in SE across the world. Convenience and purposeful sample techniques were used to find an appropriate representation of respondents for the study. By these techniques, participants selected composed of software experts and researchers in SE who have actual experiences in software development projects using varieties of models in at least one of the software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control., from all over the world.

4.8.1.1 Structure of the PFMEOPOQ

The survey questionnaire (PFMEOPOQ) has two sections with a total of fifty-seven (57) questions. The questions were self-designed by the researcher the outcome of the focus group discussions in Field study 2. Most of the questions are fairly close-ended.

Section 1 of the PFMEOPOQ asked about participants' personal information. In Section 2, for each of the listed probable failure modes in their corresponding stages, participants were asked to circle the response in the left column that best describes their view on the *level of Acceptance/Agreement* (as: 5 - "Strongly Agreed" (SA), 4 - "Agreed" (A), 3 - "Medium" (M), 4 - "Disagreed" (D), and 5 - "Strongly Disagreed" (SD)of the listed probable failure modes as a potential risk factor in the studied stage. On the right-hand column, participants were asked to circle the responses that best describes the *Likely Effect* of the listed probable failure modes on software project outcome (cost, time and quality/scope). Sample question and answers were provided in the questionnaire to guide all respondents on how to complete the questionnaires. All the items in the PFMEOPOQ were reviewed by three research experts from CS education department and two from Measurements & Evaluation unit of a public University in Nigeria for validation.

4.8.1.2 The study validity and reliability

The reliability of the instrument was established through the 'Test-re-test method. The first test conducted a pilot survey with five software experts in Nigeria that would not participate in the research. At the end of the pilot study, an important adjustment in the PFMEOPOQ was made, which added the option to first verify whether the identified probable failure modes can be regarded as potential threat to project outcomes or not. The second test then conducted validation test where all the items in the PFMEOPOQ were reviewed by three research experts from the Computer Science (CS) education department and two from Measurements & Evaluation from a public university in Nigeria.

4.8.2 The survey study method implementation

The survey implementation begins with sending a request for participation in the proposed research in form of an introductory letter to the 250 sampled prospective participants and the following sources were chosen using a purposeful random selection method: The authors of articles and participants in panels at all the proceedings and workshops of the International Conferences on Software Engineering sponsored by ACM and IEEE CS between the periods of 2006 to 2017. At the initial stage, a total of 130 signified interest to participate in the survey. However, this number declined to about 77, which shows that only 77 software practitioners and researchers in SE around the world participated in the survey study (Field study 3). All the fifteen (15) questionnaires administered by the researcher were correctly filled and returned making the return rate of 100%. Mean statistics was used to analyse the data obtained from the PFMEOPOQ. The data collected were interpreted using descriptive statistics and detail of the analysis is provided in chapter five.

4.8.3 The Field study 3 conclusions

Even though this study used a descriptive survey design to clarify issues concerning the probable failure modes identified in the Field study 2 and determine their possible influence on software project outcomes, the main purpose of this study was to generate an authoritative list of potential failure modes that can exist in the studied stages of software development process, which can influence the outcome of a project from software practitioners' perceptions.

4.9 Field study 4: FMEA Process to assess risk factors in the software development process at Abuja - Nigeria

Background

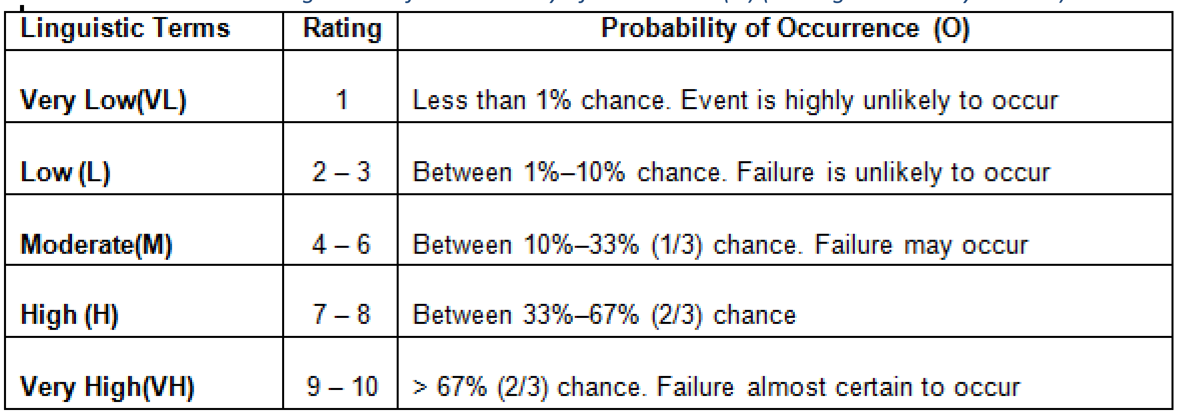
This study applied FMEA process to assess the risk factors associated with potential failure modes in the software development process. It builds upon the outcomes of Field studies 1 and 2. The study commenced with composition of perfect project team with a representation of 10 most experienced practitioners in software development projects that participated in Field studies 1 and 2. The 5 stages of software development process (Planning, Requirement Analysis, Design, Coding, Installation and maintenance) are the elements of processes to be examined. The study was intended to identify the stages of software development process that contains the potential failure mode that pose a high threat on the software project outcomes (cost, time and scope/quality). The study was conducted in same location as Field studies 1 and 2.

4.9.1 The FMEA process design

The list of potential failure modes established (from studies 1 and 2) for each of the software development process stages were adopted for this study. The project team was responsible for assessing the level of risk severity (S), occurrence (O), and detection (D) using the tables of severity, occurrence and detection. Companies and organizations usually establish their own versions of the O, S and D rating scales that fit their rating requirements for their FMEA process. However, for the purpose of this study, the tables of severity, occurrence and detection adapted from Abdelgawad and Fayek (2010), which were reviewed and validated in Field study 1 (see Table 4.2 to Table 4.5) were adopted for the risk factors assessment.

Moreover, for every potential failure mode in all the studied stages of the software development process, the team assessed the probability of occurrence for each possible failure (known as “occurrence”), using a linguistic model (Low, Very Low, moderate, High and Very High) on a rating scale of 1-to-10 for the adapted *Table of Occurrence* as proposed by Abdelgawad and Fayek (2010) (see Table 4.2).

Table 4 2:: Assessment Rating Criteria for Probability of Occurrence (O) (Abdelgawad & Fayek 2010)



Linguistic Terms	Rating	Probability of Occurrence (O)
Very Low(VL)	1	Less than 1% chance. Event is highly unlikely to occur
Low (L)	2 – 3	Between 1%–10% chance. Failure is unlikely to occur
Moderate(M)	4 – 6	Between 10%–33% (1/3) chance. Failure may occur
High (H)	7 – 8	Between 33%–67% (2/3) chance
Very High(VH)	9 – 10	> 67% (2/3) chance. Failure almost certain to occur

The team continued by assessing the damage each failure will inflict should the failure actually occur (termed as “severity”) using the same linguistic model for the adapted *Table of Severity* (see Table 4.3).

Table 4 3: Assessment Rating Criteria for Severity (S) (Abdelgawad & Fayek 2010)

Linguistic Terms	Rating	Severity (S)
Very Low (VL)	1	No effect
Low (L)	2 – 3	Effect slightly noticeable
Moderate (M)	4 – 6	Noticeable effect on sub-process or subsystem
High (H)	7 – 8	Effect on major project or system but not on safety or government regulated compliance item
Very High (VH)	9 – 10	Absolute effect on the project or system and safety or involving noncompliance with government regulations

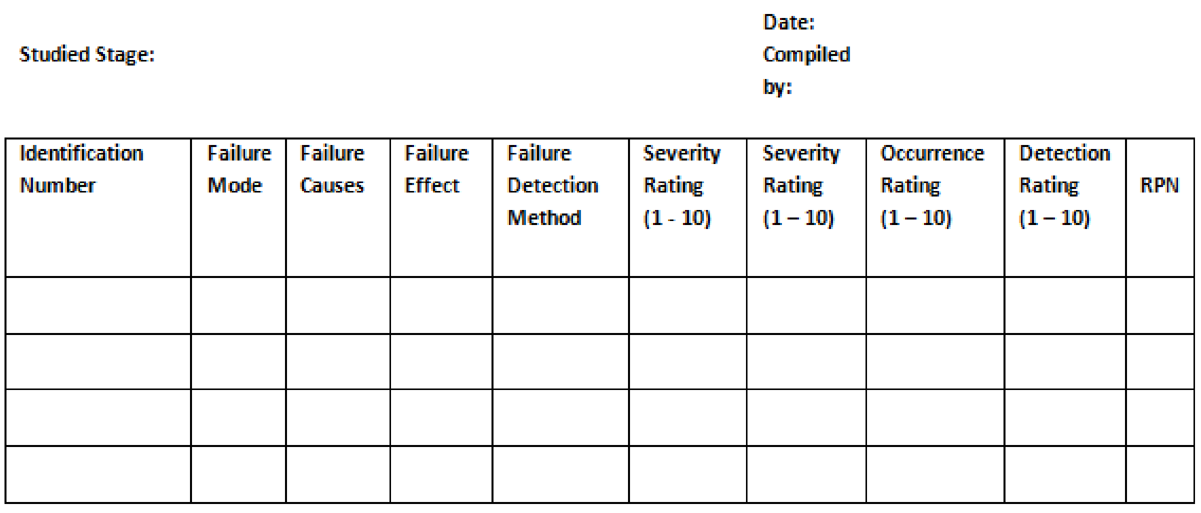
Progressively, the team then rated the likelihood of detecting such failures using the failure detection method specified against each failure mode before final delivery (called "detection") with the linguistic model 1-to-10 scale of the adapted *Table of Detection* (see Table 4.4).

Table 4 4: Assessment Rating Criteria for Detection (D) (Abdelgawad & Fayek 2010)

Linguistic Terms	Rating	Detection/control
Very Low (VL)	9 – 10	The project team was unable to identify a risk response strategy capable of detecting the risk event, controlling root causes, and controlling the consequence of the risk event.
Low (L)	7 – 8	The project team has identified a risk response strategy with a low chance of detecting the risk event, controlling the root causes, and controlling the consequence of the risk event.
Moderate (M)	4 – 6	The project team has identified a risk response strategy with a moderate chance of detecting the risk event, controlling the root causes, and controlling the consequence of the risk event.
High (H)	2 – 3	The project team has identified a risk response strategy with a high chance of detecting the risk event, controlling the root causes, and controlling the consequence of the risk event.
Very High(VH)	1	The project team has identified a risk response strategy that has been proven in the past to have high effectiveness in detecting the risk event, controlling the root causes, and controlling the consequence of the risk event.

All the previously assigned ratings for the three risk factors (Occurrence, Severity and Detection) for each failure mode in all the studied stages of the software development process were collated and recorded in the FMEA worksheet shown in Table 4.5. The FMEA worksheet was adapted from the USDOD (MIL-STD-1629, 1980) and reviewed in field study 1. The three parameters (O, S and D) ranked on a 1-to-10 scale were then multiplied and the product of these three parameters is called the Risk Priority Number (RPN), recorded by the FMEA process facilitator in the RPN column of the FMEA worksheet as shown in the Table 4.5.

Table 4 5: Sample FMEA Worksheet (Source: USDOD, MIL-STD-1629 1980)



Studied Stage:

Date:
Compiled by:

Identification Number	Failure Mode	Failure Causes	Failure Effect	Failure Detection Method	Severity Rating (1 - 10)	Severity Rating (1 – 10)	Occurrence Rating (1 – 10)	Detection Rating (1 – 10)	RPN

4.9.2 The FMEA process method implementation

The three parameters (O, S and D) were measured using the ranking scale 1 – 10 of Tables 4.4 – 4.6 and documented in their respective columns provided in the FMEA worksheet. The parameter O was rated when the team responded to the question: *how often does the cause of the failure mode occur?* Participants (10 most experienced experts from team that conducted study 1 and 2) were allowed to respond with justification. This was later thrown to the team for further analysis. The ranking figure (1 - 10) from Table 4.4 that had the majority acceptance from the team members was considered and recorded against the Occurrence ranking for the failure mode in question. This process was repeated to rank the Occurrence for all other failure modes across the five studied stages of the software development process.

Similar process followed to rank the failure effect i.e. “Severity”. To rank the severity of each failure mode in all the studied stages, the ranking was confirmed when team members responded to the question: *how severe is the effect of the failure mode to the project outcome?* And the team responses to the question: “*how well can the stated detection method specified against a failure mode detect the cause of the failure mode?” were* used to determine the perfect rating for the parameter D.

The FMEA worksheet was used to document all the previously assigned ratings conducted by the project team. The FMEA facilitator (in this case, the researcher) was responsible for the documentation. The facilitator also did the calculation for RPN value for each failure mode across the study stages. For the purpose of this study, the calculated RPN value was used to address three major issues: (1) to prioritize the risk associated with the potential failure modes in all the studied stages, (2) to classify the risk criticality of the associated potential failure mode, and (3) to assess each failure impact on the project outcome.

i. Prioritizing the risk in all the studied stages

Here, the value of the calculated RPN was used to prioritize the risk associated with the failure modes. The RPN was used to identify the failure mode that poses the highest threat to the success of the project. The higher the RPN of a failure mode the higher the risk posed by the affected failure mode.

ii. Classification of risk criticality

The risk criticality provides a useful insight to project managers on how the associated risk is to be attended to. The criticality classification standard of USDOD (MIL-STD-1629, 1980) was adopted for the study. The higher the RPN value the higher the criticality. The criticality levels are classified as: Normal, Semi-critical and Critical levels (USDOD MIL-STD-1629 1980).

Level 1 - Normal level: *This* level is obtainable when all of the three parameters of RPN (specifically, the 'severity' and 'occurrence') have rated figures of less than 5. Or when the calculated RPN value is very low (usually, when RPN < 70), in which the risk needs no corrective and preventive actions. However, in this case, the corrective/preventive action would be presented.

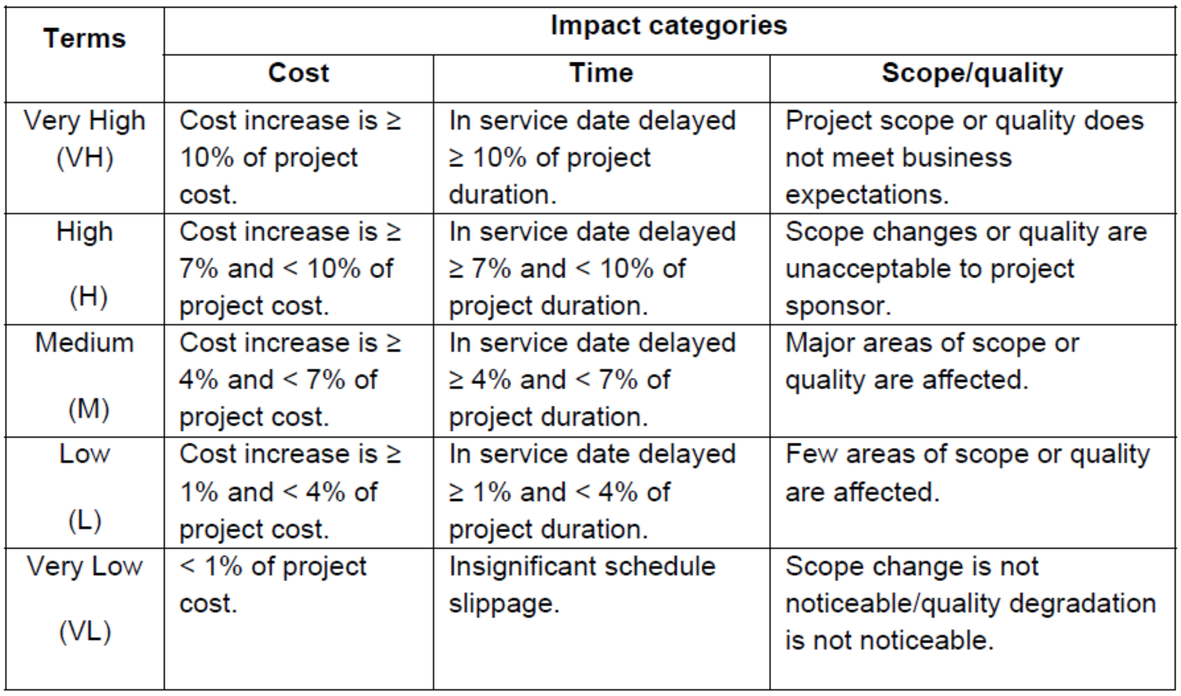
Level 2 - Semi-critical level: This level is obtainable when at least one out of the three parameters of the RPN (specifically, the 'severity' or 'occurrence') has a rated figure greater than 5 but RPN is relatively low (typically 70 < RPN < 140). In this case, corrective/preventive action is essential.

Level 3 - Critical level: This level is obtainable when at least two parameters of the three-RPN parameters (specifically, S, D and O) are highly rated, with the calculated RPN value being too high (usually RPN > 140). The consequence is that the risk must have a corrective/preventive action.

iii. Risk multiplier effects on the project outcome (cost, time & scope)

Understanding the dynamics of risk multiplier effects on software projects provides useful insight to project managers on how the occurring risk factor influences the project outcome. In this research, the risk impact category scale of the PMI (2008) extracted from Abdelgawad and Fayek (2010) was implemented to estimate the multiplier effects of the risk associated with potential failure modes on the project outcome (cost, time & scope). Table 4.6 presents the risk impact category scale of the PMI (2008) extracted from Abdelgawad and Fayek (2010) that was used in this research.

Table 4.6 presents the risk impact category scale of the PMI (2008) extracted from Abdelgawad and Fayek (2010)



Terms	**Impact categories**		
	Cost	**Time**	**Scope/quality**
Very High (VH)	Cost increase is ≥ 10% of project cost.	In service date delayed ≥ 10% of project duration.	Project scope or quality does not meet business expectations.
High (H)	Cost increase is ≥ 7% and < 10% of project cost.	In service date delayed ≥ 7% and < 10% of project duration.	Scope changes or quality are unacceptable to project sponsor.
Medium (M)	Cost increase is ≥ 4% and < 7% of project cost.	In service date delayed ≥ 4% and < 7% of project duration.	Major areas of scope or quality are affected.
Low (L)	Cost increase is ≥ 1% and < 4% of project cost.	In service date delayed ≥ 1% and < 4% of project duration.	Few areas of scope or quality are affected.
Very Low (VL)	< 1% of project cost.	Insignificant schedule slippage.	Scope change is not noticeable/quality degradation is not noticeable.

The risk criticality levels (Normal, Semi-critical and Critical levels) prescribed by the USDOD MIL-STD-1629 (1980) were used to describe the risk terms in the first column of the Table.4.6. The impact categories column describes their corresponding multiplier effects on cost, time and quality of the project as presented in the table. Risk factors with RPN value < 70 are categorized as Very high (VH) and High (H) level risks. Risks that have RPN value range of 70 < RPN < 140 are termed Medium (M) level risks, while those risks that have RPN value < 70 are categorized as Low (L) and or Very Low(VL) risks.

4.9.3 The Field study 4 conclusions

The FMEA process is a structured approach conducted to discover potential failures that may occur in project development. The method is also deployed to explore the various possible failure causes; assess the likely failure effects and suggest effective control mechanisms to detect failure occurrences and effects. The group dynamic in which team members are gathered from different backgrounds and functional departments for the conduct of the FMEA makes it a widely acceptable method of lowering the possibility of failure occurrence in project development. Also, the inclusion of detection method aimed at controlling failure occurrence factor adds more value to the analysis compared to the other traditional approaches (such as risk matrix) that employ duo factors of (Occurrence and Severity).

4.10 Field study 5: The FMEA method interactive study with SE students

Background

The introductory and training aspect of undergraduate course in SE programme is an interactive course that presents a step-by-step method for conducting FMEA for the purpose of software project risk management process. It begins by explaining the evolution and capabilities of the FMEA as a reliable and proactive risk preventive model. The session defines the FMEA responsibilities and its procedural requirement for the conduct of risk management process. The training also covers the implementation process. The interactive session was packaged with lesson materials, practice exercises, FMEA-process-service examples and free editable FMEA excel template. The interactive course was conducted to provide an opportunity for the trainee to learn important skills on FMEA application in risk management, which would help them in becoming professional risk managers. Before the commencement of the interactive session, the researcher, who is one of academic staff of Computer Science department of the institution where the interactive study was carried out also serves as the implementer of the interactive study informed the

Head of Department (HOD) of Computer Science and the Directorate of Undergraduate of the institution about the proposed research. The approval was communicated through the said HOD to the researcher as presented in Appendix 1C.

4.10.1 The FMEA interactive study design

4.10.1.1The course curriculum

The course content on which this interactive study was based is the university undergraduate curriculum of the course titled: Software Engineering with course code "COSC 403". At the end of the course, students should be able to explain the basic software engineering principles and software design concepts. Students are also expected to be able to use the design principles to create an efficient software design, and to acquire the skills required to develop qualitative and reliable software that are well designed, well tested and well documented. The course is also aimed at establishing students understanding of software development stages and models. The indicative student study hours include class lecture/tutorials/lab practical: 8 hours per week (150 hours in 18 weeks) on average two hours per week. There is directed independent study: four hours per week at a minimum. The course is evaluated by continuous assessment (CA) and examination weighting (40/60) with overall pass mark: (>=45).

4.10.1.2 Interactive study session

The interactive study was conducted for a period of four weeks extending the entire COSC 403 duration to 22 weeks. The interactive session was packaged with lesson materials and extensive practical exhibitions. The interactive study contents and objectives include to: describe the purpose of the FMEA process, Define the properties of FMEA, Define the FMEA terms: failure mode, failure effect, failure cause, severity, occurrence, and detection, Apply FMEA methods: to brainstorm potential failures, Assess risk of failure, determine areas that need action in software development projects, List at least three benefits of utilizing FMEA, explain when FMEA should and should not be performed in software projects, and complete a sample Process FMEA. The interactive study class has been updated to be consistent with the reviewed tables (from Field study 1): the probability of occurrence table (Table

4.2), the severity table (Table 4.3), the detection table (Table 4.4) and the FMEA worksheet (Table 4.5).

The interactive study also requires students to complete the practical work assignment that will be assessed by the researcher. Feedback was also provided on all practical works. After the training session, participants (students) were grouped into six groups, each group having six students minimum and maximum of 10 students. Each group was given the chance of selecting a software development stage of their choice, explore and identify all the potential failure modes, analyse their possible cause and effect, assess the probability of occurrence, severity and detection, and suggest possible method to detect the failure occurrence. All groups were mandated to use the reviewed tables of Field study 1 as directed by the researcher for their practical documentation. A total of 50 students participated in this study.

4.10.1.3 Structure of the FMEAEQ

The survey questionnaire (FMEAEQ) is a one-page document containing two sections with a total of twenty-two questions (22). The questions are self-designed by the researcher using the relevant information from the web and reviewed academic literatures concerning the FMEA procedural requirements and usage. Questions from the questionnaire require from the participating SE students to either agreeing or disagreeing with the statement asserting the usability and applicability of using FMEA for the conduct of risk management in software development projects. Most of the questions contained in the questionnaire are fairly close-ended.

Section 1 of the FMEAEQ asked about participants' personal information. In Section 2, for each of the listed assertion statement on FMEA usability and applicability, participants were asked to tick (√) the most appropriate response by either agreeing or disagreeing to the statement in the columns provided. All the items in the FMEAEQ were reviewed by three research experts from the Computer Science (CS) education department and two from Measurements and Evaluation from a public University in Nigeria, and changed until Instructions therein clearly stressed an answer. The questionnaires were pilot-tested by four SE students (not participating students) before the actual distribution. The reliability of the data depends mainly on the design quality of the FMEAEQ and its administration.

4.10.2 Interactive study method implementation

4.10.2.1 FMEA worksheet documentation

The FMEA worksheet is a documentation tool used to record the FMEA proceedings and outcome. In this study, the worksheet's documentation guide was handed out to students at the start of the interactive study. This guide explains the steps to follow to complete the worksheet. It also helps the participating students to understand how to conduct FMEA process on their selected software development process stage and to maximize their profile. The primary temptation is to guide students on how to use the FMEA method in project risk management and to reward those students who are making satisfactory progress in regular practice. The practical exercise engages students in regular risk management activities, and tracks their progress by evidences seen and observed.

4.10.2.2 FMEA method evaluation

After the practical exercise, a survey was conducted using the FMEA Evaluation Questionnaire (FMEAEQ) to evaluate the overall layouts of how the risk assessment was conducted over the stages of the software development process using the FMEA method in their practical exercises. Interested students were given the consent form to complete (see Appendix 2A). 53 students signified interest, however, 50 out of the 53 that completed and returned the consent form turned out to participate in the survey. The participating students were handed with the hard copies of the questionnaire (FMEAEQ) to complete after their practical work and return same to the survey taker after completion. The FMEAEQ (see Appendix B4) enquires from the participating students, information about the: FMEA concept comprehension, timing, feasibility, procedures, FMEA method usage, its strengths and weaknesses as a tool to be used in software project risk management.

However, the researcher was only interested in the assessment of the FMEA method ease of use and its usefulness as a tool for software project risk management. Hence, sections concerning students' self-evaluations (% satisfaction) of FMEA method ease of use and usefulness of the FMEA method to their practical assignment extracted from the completed questionnaires were processed and analysed. The reason is due to the fact that FMEA alignment with project management standards and other reviews have already been discussed extensively in the Field study 1. The results of the processed data on students' self-evaluations (% satisfaction) of FMEA method ease of use and usefulness as a tool for software project risk management is discussed in chapter five.

4.10.3 The Field study 5 conclusions

The FMEA interactive study introduces undergraduate SE students to FMEA process application and engages them in a practical training on how to use the method to manage the risks in software development projects. Undergraduate SE students were chosen because it is assumed that some of them would be the future professional risk managers. Hence, engaging SE students in a practical exercise concerning risk management activities, and tracking their progress by evidence formed the basis of the FMEA interactive study.

The interactive study was conducted to provide an opportunity for the prospective risk managers to learn important skills on FMEA application in risk management. Thus, addressing the challenge of inadequate training facing risk managers of software projects and provide feedback about the methods' concept, ease of use, usefulness, strengths and weaknesses.

The next section presents the summary of this chapter.

4.11 Chapter summary

The chapter objective was to provide a clear focus on the research design of this study with special attention to the paradigm and methodology applied to answer the research questions. CR was found to be the most appropriate term for the research epistemology. The study adopted a mixed-methods research strategy and a deductive research approach was justified as the suitable approach for the study. The chapter described how the five field studies were conducted and the instrumentation designs for the study with separate sections explaining how various data were collected and analysed for valid decision making. Finally, the chapter presents the criteria for risk classification, using internationally accepted standards.

The next chapter presents the analysis of the results obtained in the study.

CHAPTER FIVE

Presentation of Results and Analysis

This chapter presents the result analysis of both qualitative and quantitative data collected from the research. The analysis is organized by first presenting the qualitative data before analysing the quantitative data. The qualitative data are collected from the focus group discussions conducted in Field study1 and Field study 2 while quantitative data are data collected from the four sets of questionnaires (FMEAAQ, FMEOPOQ, PFMEOPOQ and the FMEAEQ) and the FMEA worksheet across the five conducted field studies. The chapter begins by discussing the method used to analyse the qualitative data presented in section 5.2. The data analysis of the focus group discussions obtained from Field study 1 and those of Field study 2 are presented in section 5.3 and section 5.4. Section 5.5 presents the analysis of the FMEAAQ data obtained from Field study 1 while data analysis of FMEOPOQ used in Field study 2 is presented in section 5.6. The analysis of PFMEOPOQ data obtained from Field study 3 is presented in section 5.7. Section 5.8 is a presentation of the analysis of FMEA Worksheet data obtained from the FMEA process conducted in Field study 4. In section 5.9, the analysis of FMEAEQ data obtained from Field study 5 is done to evaluate the perceived ease of use and the perceived usefulness of the FMEA. The comparative procedures and analysis that were conducted using the Analysis of Variance (ANOVA) testing techniques to test the research hypothesis are presented in section 5.10 while *a summary of the entire chapter follows in section 5.11.*

5.1 Analysing the qualitative data (focus group discussion)

The method used to analyse the qualitative data collected in the field studies 1 and 2 conducted in this research is based on the Miles and Huberman (1994) method of analysing qualitative data. According to Miles and Huberman, the qualitative data analysis consists of "three concurrent flows of activity: data reduction, data display, and conclusion drawing/verification" (Miles & Huberman 1994, p. 10). The three flow of activity by Miles and Huberman is considered appropriate as a result of the small sizes of participants and data captured during the discussion. Thus, the need to apply customized software for the qualitative data analysis is not required.

Data reduction: By data reduction, researchers collate and compare data, contrast, sort, and order data (Miles & Huberman 1994). In this study, the discussions were tape-recorded and documented. The documentation was then reduced to a transcribed document by removal of comments and contributions that are not related to the theme of discussion before the data analysis. The transcription was double-checked by hearing the recording of the discussant comments and contributions for checking any missing data or misinterpretation. Extra care was also put in place when reducing the data and summarizing the transcription.

Data display: Data display is an extension of data reduction to provide "an organized, compressed assembly of information that permits conclusion drawing" (Miles & Huberman1994, p. 9). It can also be an extension of a piece of text or a diagram, chart, or matrix that provides a new way of arranging and thinking about the more textually embedded data. A data display, in any form of arrangement, allows the analyst to permutate enough data that will initiate discern systematic patterns and interrelationships (Miles & Huberman 1994). In this study, Miles and Huberman (1994) recommendation of using tables to display data was adopted. The summarized transcripts are arranged and presented in a table created using Microsoft Word application. The table, containing rows and columns are arranged to discern systematic patterns and interrelationships. A row represents the identified probable failure mode while a column describes the characteristics of failure (such as: causative factor, likely effect and detection method).

Conclusion drawing and verification step: This refers to the strategies employed to analyse and interpret the transcribed data. According to Miles and Huberman (1994, p.11), “Conclusion drawing involves stepping back to consider what the analysed data mean and to assess their implications for the questions at hand. Verification, integrally linked to conclusion drawing, entails revisiting the data as many times as necessary to cross-check or verifies these emergent conclusions”. In this activity stage, the summarized data that contains only points, comments and justifications that are relevant to the discussion theme are documented. The new transcript document is later compared with the initial paper document and recorded documents to reconcile misinterpretation and search for any missing information or relevant points. In this case, the reconciled transcript is presented in a tabular form for easy analysis.

The subsequent section describes the method used to analyse the focus group discussions' data obtained from the Field study 1 conducted in this research.

5.2 Analysing focus group discussions' data obtained from Field Study 1

As earlier mentioned in section 3.8.4, the exploratory method implementation process, three different focus group discussions were conducted in a three-day meeting with software development practitioners selected from five different software companies in Nigeria. The first focus group meeting discussed the FMEA standard features, the second meeting discussed FMEA adherence analysis with standard project risk management frameworks and the third reviewed the existing tables of (occurrence, severity and detection) and FMEA worksheet that will be used to conduct the FMEA process.

The results of the three meetings discussion were paper documented by the research assistants in the notes provided and in the recording tapes 'at real time' during each session. Both documented sources were later transcribed verbatim into a document for final analysis. The transcribed document contains only issues, comments and recommendations raised during the discussion that are relevant to the theme of discussion. Every relevant point raised during the discussion session was documented against the author. Authors' names were not mentioned in the document. However, all the 15 participants' positions are labelled alphabetically as (A, B, C, …., O) respectively and used same to represent the author.

All relevant points and comments regarding the FMEA standard features raised by the participants during the first focus group meeting were described and summarized with each unique point raised, numbered, and matched against the author for traceability and easy analysis. The next section (section 5.2.1) describes the FMEA standard features as discussed during the first focus group meeting.

5.2.1 The FMEA standard features

The first focus group discussion meeting session discussed the FMEA standard features. The discussion was coordinated to follow a predefined structure so that the discussion is issue-based and the session stays focused on the agreed theme. Three questions were designed to serve as the discussion guide. The first question that stimulated the discussion is: "*How important are the FMEA procedures for conducting project risk management?"* Discussant responses to the question show that the FMEA standard procedures are reliable tool for risk assessment and risk control. As a reliable risk assessment tool, Mr. B, for instance, said: "*the FMEA procedures can efficiently identify possible risk in any proposed project, measure the magnitude of the risk and prioritize the risk using the values of the calculated RPN values. This will assist managers to set out reliable risk control mechanism for the proposed project"*. Similarly, as reliable risk control tool, Mr. J, for instance, said: "*FMEA procedures are reliable supportive tools to establish preventive and corrective strategies using the RPN values to control failure occurrences of software projects*".

Further analysis of the above comments and other comments from discussants reveal that FMEA standard procedures are reliable risk assessment and risk control method. Also, risk assessment and risk control are two components of important software project risk management method.

Discussants' responses to the question: "What project risk management stage does each procedure align with?" appear to be the same across the board. This was demonstrated by attitude of other discussant after Mr A's response: "the gathered FMEA team to brainstorm unknown risk as a first stage of FMEA process aligns with the risk identification stage in the PMBOK requirement, also assign different rating to severity, occurrence and detection for each failure in the FMEA process align with risk analysis in the PMBOK requirement. Computing the RPN value using the severity, occurrence and detection ranking to determine the risk magnitude and prioritize the risk conforms to risk prioritization and classification. Using the value of the RPN value to devise a suitable risk control strategy conforms to the risk control stage". This response was greeted with absolute applause and a carouse response from others that "that is perfect, he has said it all".

The discussants' responses to the question: "How the FMEA procedures should be prioritized" show that the FMEA procedural steps as arranged and used in the research of Georgieva (2010) 'Conducting FMEA over the software development process' was unanimously recommended as the adequate model for applying the FMEA process. Their main concern is that the research of Georgieva (2010) provided a technical step by step procedure to conduct FMEA process in software development process; hence, the Georgieva's prioritized steps is the appropriate model.

5.2.2 FMEA adherence analysis with popular project risk management models

Here, the discussants' responses to the question: "How well do the FMEA procedures satisfy the procedural requirement of prominent standard project risk management models (PMBOK, SEI-CRM and Boehm)?" show that all commentators made references to the gains from the previous workshop on the FMEA procedures and requirements of the popular risk management models. There appears to be a common response from all that responded to the question asked, which reinforces the confidence of high level of adherence between the FMEA procedural steps and requirements frameworks of the three popular risk management models (PMBOK, SEI-CRM and Boehm).

5.2.3 Reviewing the tables of occurrence, severity, detection and the FMEA worksheet

At the beginning of the reviews, discussants were provided with two different copies of tables of rating scales for the risk variables that will be used to support the FMEA process: Probability of Occurrence, Severity and Detection tables, which were developed by Abdelgawad (2010) and the other adopted from USA Department of Defence (1980) Military Standard. Copies of FMEA worksheet were also provided for reviews. The purpose was to review and analyse the possibility of their adoption for the study. The analysis of the discussants' responses to the question "What is the feasibility of using the construct measure tables of Occurrence, Severity and Detection and the FMEA worksheet by Abdelgawad (2010) or that of DOD (1980) standard tables for measure of Occurrence, Severity and Detection of failure modes in software project risk management and to document FMEA process?" revealed that discussants see the metric tables developed by Abdelgawad (2011) as most suitable measuring construct that will support the FMEA process for software project risk management and recommended the FMEA worksheet by USDOD (MIL-STD-1629, 1980) as the most suitable worksheet to document the FMEA process.

The participants expressed their views by indicating the five linguistic terms used for the rating, i.e., —very low (VL), low (L), medium (M), high (H), and very high (VH) are simple and convenient rating scales for risk managers to implement for their risk assessment. Hence, the construct measure tables of Occurrence, Severity and Detection by Abdelgawad (2011) presented in Table 4.2 to Table 4.4 and the FMEA worksheet by USDOD (MIL-STD-1629 1980) presented in Table 4.5 were recommended to be used in the FMEA process.

The following section describes the method used to analyse the focus group discussions' data obtained from the Field study 2 conducted in this research.

5.3 Analysing focus group discussions' data obtained from Field Study 2

Only one focus group discussion was conducted in the Field study 2 with the following objectives:

- to identify the probable failure modes in all the studied stages of the software development process;
- to suggest the likely causative factor(s) of the identified failure mode; and
- to suggest suitable symptom(s) that is/are coherent with each failure event.

The analysis of the focus group discussion of Field study 2 adopted the same method of data analysis used in Field study1. To achieve the stated objectives, the discussants, first, reviewed the work of Abubakar and Lawal (2020) together with Georgieva (2010) as secondary research techniques to garner useful information on the agenda for the discussion. This was then followed by brainstorming on each of the studied stages of the software development process sequentially, beginning from the first to the fifth stage (i.e., Planning, Requirement Analysis, Design, Implementation and Coding, Installation and Maintenance), to identify the probable failure modes that can influence project outcome. After completing the failure identification process in the stage1 (planning stage), the discussants then embarked on highlighting the likely cause(s) of the individual probable failure mode identified in the concerned stage and then selection of the most appropriate symptoms, which the discussants believed are coherent for detecting the failure occurrence early enough in the concerned stage. The same process was applied to the remaining stages of the software development process. The discussion sessions for the five stages were audio-taped and paper documented.

The Miles and Huberman (1994) method of analysing qualitative data was applied to analyse the transcribed data obtained from Field study 2. The analysis results yielded a distribution of the identified risky failure modes in their respective SDLC stages, their likely causative factor(s) and the failure symptom(s) that can be used to detect likely occurrence of the identified failure mode in all the studied stages were identified. The result of data analysis of focus group discussion engaged in Field study 2 is presented in Appendix 3A.

The following section describes the method used to analyse the quantitative data collected in all the Field studies conducted in this research.

5.4 Analysing the quantitative data (questionnaire and worksheet)

This section describes method used to analyse the quantitative data collected in all the field studies conducted in this research. As already mentioned in the preceding chapter (Chapter 4), four questionnaires (FMEAAQ, FMEOPOQ, PFMEOPOQ and FMEAEQ) were designed and used for data collection in the field studies 1, 2, 3 and 5 respectively. Descriptive statistics was used to analyse the questionnaires data and the followings statistical tools were used for the analysis of the questionnaires' data:

i. Frequency and percentage distribution

The frequency and percentage distribution table were constructed and used to present a summary of the discussants profile. The percentage is the term used to describe the qualitative relations of individual responses equated to one-hundredth of the total frequency.

ii. The Mean

The mean was used in this research to describe the perceptions of the respondents on each indicator of the questionnaire data. The mean was interpreted using the Liker Scale of I - 5 ranking concept. The decision rule set for the statistics as used for analysing and interpreting the questionnaire data is that any mean score above 2.5 is interpreted 'High' or positive consideration and the mean scores that are less than 2.5 are interpreted 'Low' or given negative interpretation as the case may be.

iii. Grand Mean (GM)

The GM is the mean of the means of several subsamples (Everith 2002). The GM was used in the field studies 1 and 5 to quantify the perception of the respondents of the surveyed population and provide the basis for the general interpretation of the research theme.

iv. *Analysis of variance (ANOVA) comparative analysis*

The ANOVA is a testing technique for comparative analysis that compares means of a continuous variable in two or more independent comparison groups. The ANOVA implementation in this study systematically compares the variability within potential failure modes' effect in stages of the software development process on the project outcome (cost, time and scope). The hypotheses were tested using the F-Table at 0.05 level of significance. The ANOVA is appropriate for handling both large and small number samples and its capability to compare means of a continuous variable in two or more independent comparison groups.

v. MS Excel 2007 and Statistical Packages for the Social Sciences (SPSS)

MS Excel 2007 and the SPSS were the main computer application packages used for the statistical process of questionnaire data. All the data gathered from FMEAAQ, PFMEQ and FMEAEQ were analysed to frequency, mean and standard deviation distributions using these packages. However, some other simple calculations were processed manually.

vi. *The rating scales*

In the FMEA process that was conducted in field study 4, the rating tables of: Occurrence, Severity and Detection (see Table 4.2 to Table 4.4) were used to rate the risk variables used to determine the risk magnitudes and the FMEA worksheet documentation (see Table 4.5).

5.5 Analysing the FMEAAQ data obtained from Field Study 1

As earlier described in the previous chapter (Chapter 4) of this thesis, the FMEAAQ is composed of five main sections: the focus group discussant profile, the level of importance of FMEA standard procedures, the FMEA adherence to PMBOK model, the FMEA adherence to SEI model and the FMEA adherence to Boehm's model. The FMEAAQ data were analysed according to the sequence of these five sections. The objectives of the FMEAAQ data analysis include: to describe the discussants' profile and to establish a definitive conclusion on whether FMEA model conforms to the international standards of prominent risk management models.

5.5.1 Analysing focus group discussant profile

This part of the FMEAAQ asks discussants about their personal information. Table 5.1 shows the frequency and percentage distribution of the focus group discussants' profile.

Table 5 1:: Focus group discussants' profile

Personal Information	Count	Percent (%)
Age:		
30-39 years	6	40
40 years and above	9	60
Gender:		
Male	12	80
Female	3	20
Professional Background:		
Programmer	4	26.7
Software Engineer	3	20
System Analyst	3	20
Software manager	4	26.7
Domain Experts	1	6.7
Other	-	-
Software Projects Successfully Developed:		
2 -5	8	53.3
6-10	6	40
11 and above	1	6.7
Years of experience in current job:		
<5 Year	2	13.4
6-10 Years	8	53.3
>10 Years	5	33.3
Experience in Software Development:		
<5 Year	-	-
5-10 Years	7	46.7
>10 Years	8	53.3

As shown in Table 5.1, the total of a 15-member team of software development practitioners participated in the focus group discussion of Field study 1. The analyses of the discussants profile show that amongst the participating team of experts, a total of 12 (80%) experts are male and 3 (20%) are female. Also, the analysis of the discussants' professional background shows that 4 (26.7%) are programmer, 3 (20%) are software engineer, 3 (20%) are system analyst, 4 (26.7%) are software manager and 1 (6.7%) is a domain expert. Analysing the number of successfully completed software projects, the result of the analysis shows that 8 (53.3%) of the discussant s have successfully completed between 2 to 5 projects while a total of 6 (40%) of the discussant completed between 6 to 10 projects, and only 1(6.7%) amongst them completed projects that are above 10.

Furthermore, results presented in the table show that 8 (53.3%) of the discussants spent between 6-10 years in their current job, 5 (33.3%) have spent between 5-10 years and 2 (13.4%) have spent less than 5 years on their current job. A further analysis on the years of experience spent in software development business shows that 7(46.7%) of the participating software development experts have less than 10 years of experience in the software development business while 8 (53.3%) have over 10 years of experience in . at least one of the following software application areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control. This profile information of the participating experts will assist in providing further guidance to future research in this field.

5.5.2 Analysing discussant's perception on the level of importance of FMEA standard procedures

Table 5.2 presents the analysis of discussants' perception on the level of importance of the FMEA standard procedures for the conduct of software project risk management.

Table 5 2: Level of importance of FMEA standard procedures for the conduct of project risk management

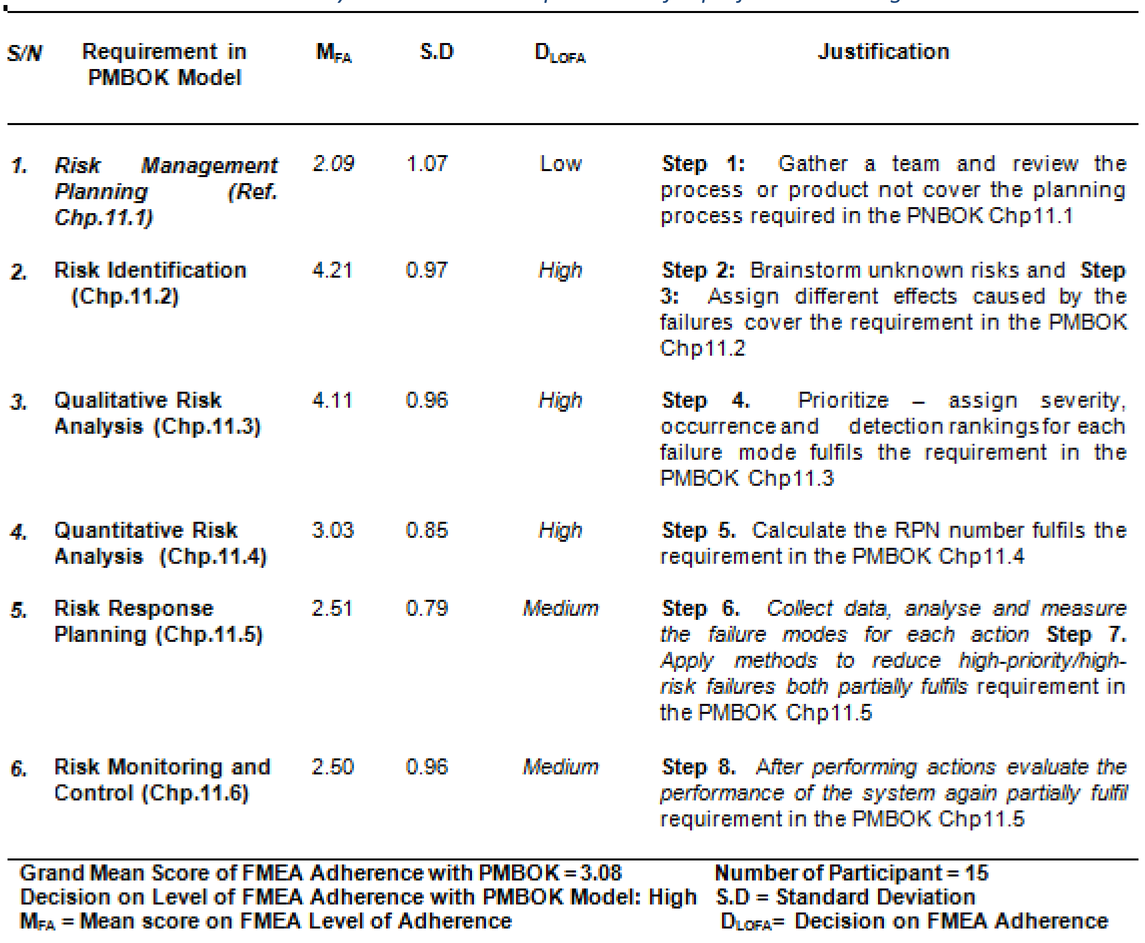
Step	FMEA Standard Procedures	Mean Score of FMEA Standard Procedures	Standard Deviation	Decision on Level of Importance of FMEA Standard Procedures
1.	Gather a team and review the process or product	4.253	0.76	High
2.	Brainstorm unknown risks	4.255	0.79	High
3.	Assign different effects caused by the failures	4.223	0.77	High
4.	Assign severity, occurrence and detection rankings for each failure mode	4.110	0.76	High
5.	Calculate the RPN value for each risk	4.228	0.78	High
6.	Collect data, analyze and measure the failure modes for each action	3.501	0.85	High
7.	Apply methods to reduce high-priority/high-risk failures	4.110	0.77	High
8.	After performing actions evaluate the performance of the system again	3.501	0.83	High

The analysis of results presented in the Table 5.2 shows that there is a common opinion from the discussant about the FMEA standard procedures listed for the survey. It is clear from the table that discussants agree that FMEA procedures can be adopted for the process of software project risk management. In the Table 5.2, all the listed FMEA procedures had mean values above 2.5, which marks the study decision rule: 'Highly Important''. It seems that applying the FMEA procedures for the conduct of software project risk management is feasible. The result is also in support of literatures suggestions of Georgieva (2010) that FMEA process is a reliable methodology for the conduct of risk assessment across the stages of software development process.

5.5.3 Analysing discussants' perception of FMEA level of adherence to PMBOK

Table 5.3 presents the analysis of the focus group discussants' perception on the level of adherence of the FMEA standard procedures with the PMBOK's requirements for project risk management.

Table 5 3: FMEA adherence analysis to PMBOK's requirements for project risk management



S/N	Requirement in PMBOK Model	M_{FA}	S.D	D_{LOFA}	Justification
1.	*Risk Management Planning (Ref. Chp.11.1)*	2.09	1.07	Low	**Step 1:** Gather a team and review the process or product not cover the planning process required in the PNBOK Chp11.1
2.	Risk Identification (Chp.11.2)	4.21	0.97	*High*	**Step 2:** Brainstorm unknown risks and **Step 3:** Assign different effects caused by the failures cover the requirement in the PMBOK Chp11.2
3.	Qualitative Risk Analysis (Chp.11.3)	4.11	0.96	*High*	**Step 4.** Prioritize – assign severity, occurrence and detection rankings for each failure mode fulfils the requirement in the PMBOK Chp11.3
4.	Quantitative Risk Analysis (Chp.11.4)	3.03	0.85	*High*	**Step 5.** Calculate the RPN number fulfils the requirement in the PMBOK Chp11.4
5.	Risk Response Planning (Chp.11.5)	2.51	0.79	*Medium*	**Step 6.** *Collect data, analyse and measure the failure modes for each action* **Step 7.** *Apply methods to reduce high-priority/high-risk failures both partially fulfils* requirement in the PMBOK Chp11.5
6.	Risk Monitoring and Control (Chp.11.6)	2.50	0.96	*Medium*	**Step 8.** *After performing actions evaluate the performance of the system again partially fulfil* requirement in the PMBOK Chp11.5

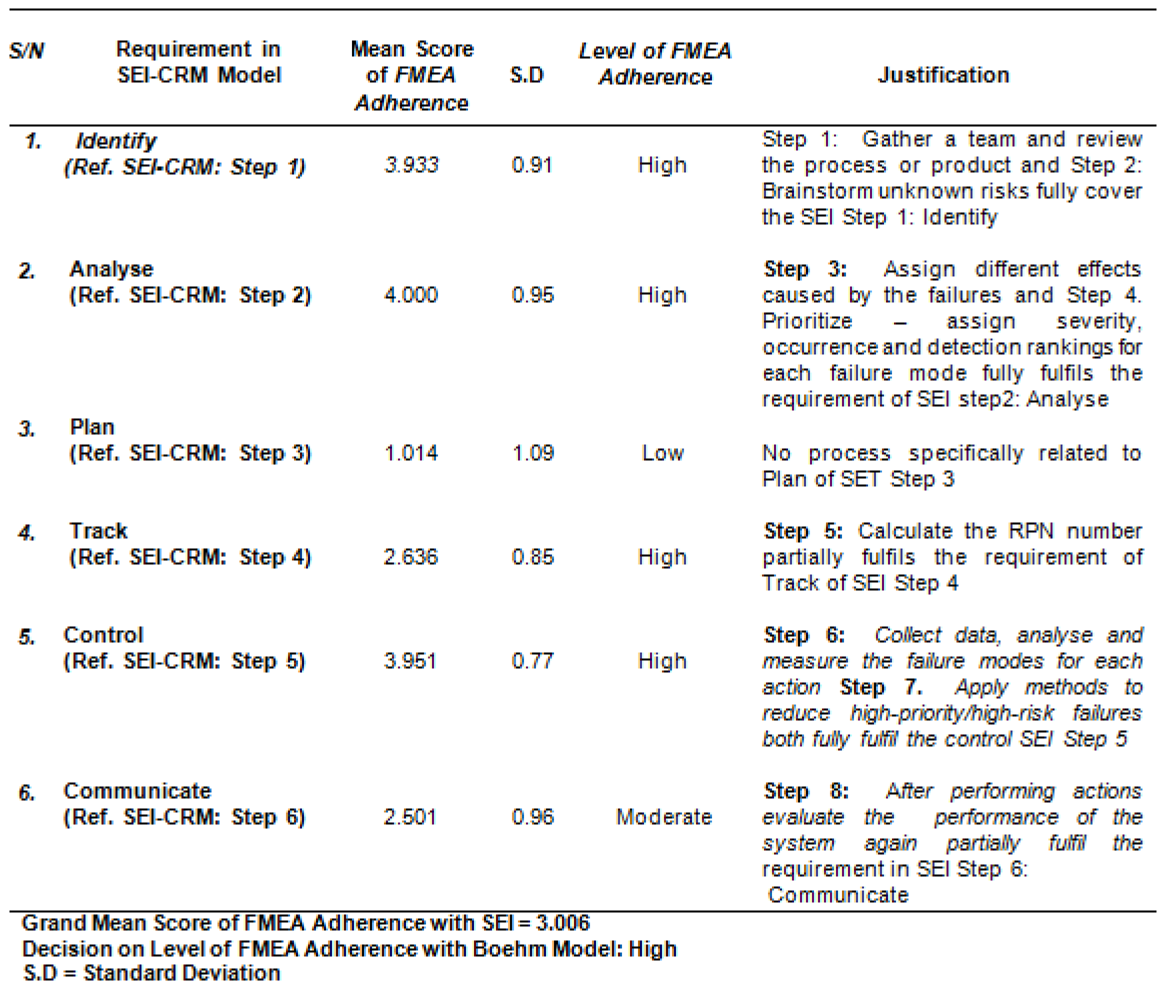
Grand Mean Score of FMEA Adherence with PMBOK = 3.08 Number of Participant = 15
Decision on Level of FMEA Adherence with PMBOK Model: High S.D = Standard Deviation
M_{FA} = Mean score on FMEA Level of Adherence D_{LOFA} = Decision on FMEA Adherence

The data in the Table 5.3 yielded a Grand Mean (GM) score of *3.08*. This result shows that the methods and all the procedure processes of FMEA model have a high level of adherence with the project risk management activities in the 11th chapter of PMBOK Guide (PMI 2004).

5.5.4 Analysing discussants' assessment of FMEA level of adherence to SEI-CRM model

Table 5.4 presents the analysis of data collected from the section 4 of the FMEA adherence questionnaire. This section of the questionnaire surveyed the perception of the discussants on the level of adherence of the FMEA standard procedures to the SEI-CRM requirements for project risk management.

Table 5 4: FMEA Adherence analysis to SEI's requirements for software project risk management



S/N	Requirement in SEI-CRM Model	Mean Score of *FMEA Adherence*	S.D	*Level of FMEA Adherence*	Justification
1.	***Identify (Ref. SEI-CRM: Step 1)***	3.933	0.91	High	Step 1: Gather a team and review the process or product and Step 2: Brainstorm unknown risks fully cover the SEI Step 1: Identify
2.	**Analyse (Ref. SEI-CRM: Step 2)**	4.000	0.95	High	**Step 3:** Assign different effects caused by the failures and Step 4. Prioritize – assign severity, occurrence and detection rankings for each failure mode fully fulfils the requirement of SEI step2: Analyse
3.	**Plan (Ref. SEI-CRM: Step 3)**	1.014	1.09	Low	No process specifically related to Plan of SET Step 3
4.	**Track (Ref. SEI-CRM: Step 4)**	2.636	0.85	High	**Step 5:** Calculate the RPN number partially fulfils the requirement of Track of SEI Step 4
5.	**Control (Ref. SEI-CRM: Step 5)**	3.951	0.77	High	**Step 6:** *Collect data, analyse and measure the failure modes for each action* **Step 7.** *Apply methods to reduce high-priority/high-risk failures both fully fulfil the control SEI Step 5*
6.	**Communicate (Ref. SEI-CRM: Step 6)**	2.501	0.96	Moderate	**Step 8:** *After performing actions evaluate the performance of the system again partially fulfil the* requirement in SEI Step 6: Communicate

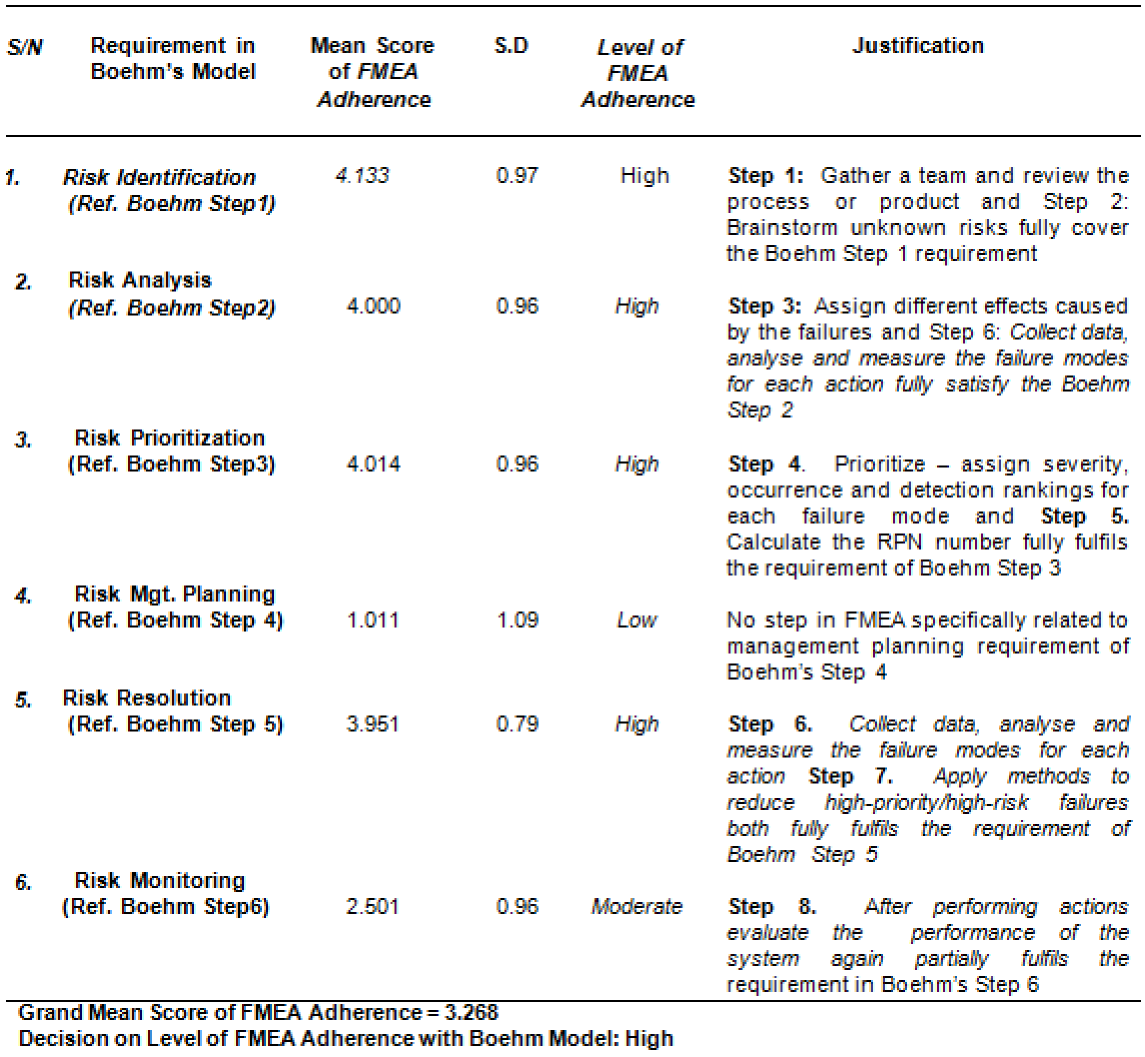
Grand Mean Score of FMEA Adherence with SEI = 3.006
Decision on Level of FMEA Adherence with Boehm Model: High
S.D = Standard Deviation

The data in the Table 5.3 yielded a Grand Mean (GM) score of *3.006*. This result shows that the methods and all the procedure processes of FMEA model have a high level of adherence to the SEI-CRM software project risk management requirement.

5.5.5 Analysing discussants' assessment of FMEA level of adherence to Boehm model

Table 5.5 presents the analysis of data collected from the section 5 of the FMEA adherence questionnaire. This section of the questionnaire surveyed the perception of the discussant on the level of adherence of the FMEA standard procedures to the Boehm requirements for project risk management.

Table 5 5: Adherence analysis between Boehm's requirements for software project risk management and FMEA



S/N	Requirement in Boehm's Model	Mean Score of *FMEA Adherence*	S.D	*Level of FMEA Adherence*	Justification
1.	***Risk Identification (Ref. Boehm Step1)***	*4.133*	0.97	High	**Step 1:** Gather a team and review the process or product and Step 2: Brainstorm unknown risks fully cover the Boehm Step 1 requirement
2.	Risk Analysis ***(Ref. Boehm Step2)***	4.000	0.96	*High*	**Step 3:** Assign different effects caused by the failures and Step 6: *Collect data, analyse and measure the failure modes for each action fully satisfy the Boehm Step 2*
3.	Risk Prioritization (Ref. Boehm Step3)	4.014	0.96	*High*	**Step 4.** Prioritize – assign severity, occurrence and detection rankings for each failure mode and **Step 5.** Calculate the RPN number fully fulfils the requirement of Boehm Step 3
4.	Risk Mgt. Planning (Ref. Boehm Step 4)	1.011	1.09	*Low*	No step in FMEA specifically related to management planning requirement of Boehm's Step 4
5.	Risk Resolution (Ref. Boehm Step 5)	3.951	0.79	*High*	**Step 6.** *Collect data, analyse and measure the failure modes for each action* **Step 7.** *Apply methods to reduce high-priority/high-risk failures both fully fulfils the requirement of Boehm Step 5*
6.	Risk Monitoring (Ref. Boehm Step6)	2.501	0.96	*Moderate*	**Step 8.** *After performing actions evaluate the performance of the system again partially fulfils the* requirement in Boehm's Step 6

Grand Mean Score of FMEA Adherence = 3.268
Decision on Level of FMEA Adherence with Boehm Model: High

The analysis of data in the Table 5.5 yielded a GM score of 3.268. This result reveals that FMEA model conforms to Barry Boehm's stages and principles for software project risk management which fundamentally comprised two steps: risk assessment and risk control.

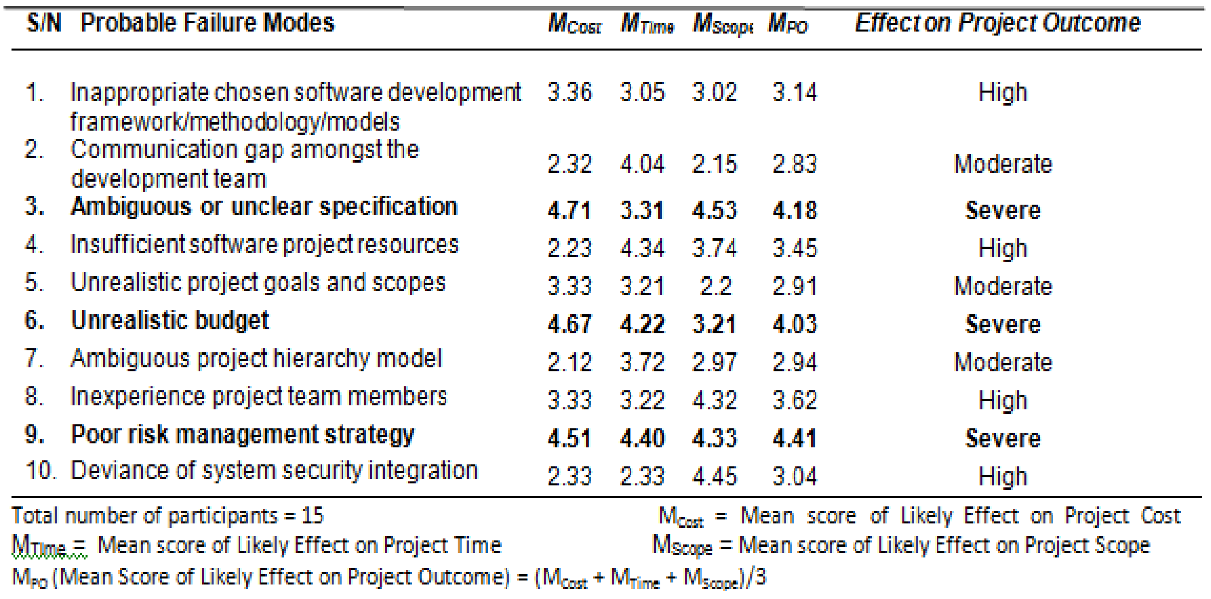
5.6 Analysing the FMEOPOQ data obtained from Field study 2

The FMEOPOQ asked discussants to assess the likely effect, which the listed probable failure modes will have on software project outcome (cost, time & quality/scope). The FMEOPOQ is composed of five sections. The sections are organized to sequentially cover the studied stages of the software development process (i.e., Planning, Requirement Analysis, Design, Implementation and Coding, Installation and Maintenance). The FMEPOQ data will be analysed according to the sequence of these five sections. The objective of the analysis of FMEOPOQ data is to evaluate the discussants views on the probable failure modes' effect on software project outcome (cost, time & scope/quality).

5.6.1 Analysing discussants' perception of probable failure modes' effect on project outcome in planning stage

Section 1 of the FMEOPOQ raised questions that were used to survey the perception of the discussants on the effect of the probable failure modes on software project outcome (cost, time & scope/quality) in the planning stage of the software development process. Table 5.6 shows the result analysis of effect of the probable failure mode in planning stage of the software development process on project outcome (cost, time & scope/quality) as perceived by discussants.

Table 5 6: Mean response of discussants' perceptions on adverse effect of probable failure modes on project outcome (cost, time & scope) in planning stage



S/N	Probable Failure Modes	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	*Effect on Project Outcome*
1.	Inappropriate chosen software development framework/methodology/models	3.36	3.05	3.02	3.14	High
2.	Communication gap amongst the development team	2.32	4.04	2.15	2.83	Moderate
3.	**Ambiguous or unclear specification**	**4.71**	**3.31**	**4.53**	**4.18**	**Severe**
4.	Insufficient software project resources	2.23	4.34	3.74	3.45	High
5.	Unrealistic project goals and scopes	3.33	3.21	2.2	2.91	Moderate
6.	**Unrealistic budget**	**4.67**	**4.22**	**3.21**	**4.03**	**Severe**
7.	Ambiguous project hierarchy model	2.12	3.72	2.97	2.94	Moderate
8.	Inexperience project team members	3.33	3.22	4.32	3.62	High
9.	**Poor risk management strategy**	**4.51**	**4.40**	**4.33**	**4.41**	**Severe**
10.	Deviance of system security integration	2.33	2.33	4.45	3.04	High

Total number of participants = 15 M_{Cost} = Mean score of Likely Effect on Project Cost

M_{Time} = Mean score of Likely Effect on Project Time M_{Scope} = Mean score of Likely Effect on Project Scope

M_{PO} (Mean Score of Likely Effect on Project Outcome) = $(M_{Cost} + M_{Time} + M_{Scope})/3$

Table 5.6 indicates that all the 10 listed probable failure items have M_{PO} rating above 2.5; hence, are perceived by experts as threat to software project outcome. However, the degree of threat posed by individual failure item varies. For instance, three out of the 10 listed probable failure modes have M_{PO} rating above 4.00 (poor risk management strategy (4.41), ambiguous or unclear specification (4.18) and unrealistic budget (4.03) and are perceived to likely pose adverse or severe threat to project outcome. Four of them have their rating as 3.99 ≤ M_{PO} ≥ 3.00 (inexperience project team members (3.62), insufficient software project tools and resource (3.45), inappropriate chosen software development framework/methodology/models (3.14) and deviance of system security integration (3.04) and are therefore, perceived as having the possibility of posing high threat to project outcome; while the remaining three listed probable failure modes have their rating as 2.99 ≤ M_{PO} ≥ 2.50 (ambiguous project hierarchy model (2.94), unrealistic project goals and scopes (2.91) and communication gap amongst the development team (2.83) and are perceived to likely pose moderate threat to software project outcome.

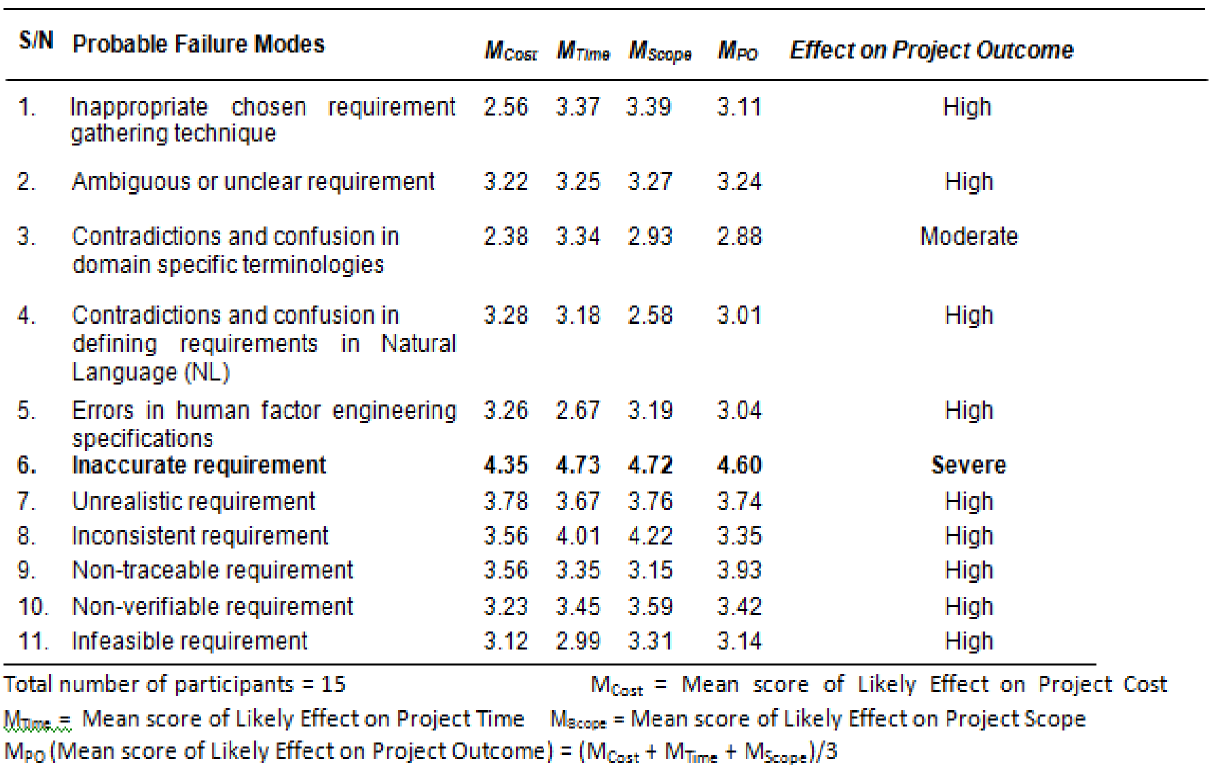
A further analysis of the results in the Table 5.6 shows that the probable failure mode: ambiguous or unclear specification will pose the highest threat to project cost and project quality having M_{cost} = 4.71 and M_{scope} = 4.18respectively while the failure: poor risk management strategy will pose the highest threat to the project outcome in general

(M_{PO} = 4.41) and specifically to the project time (M_{Time}= 4.40) as indicated in the Table 5.6.

5.6.2 Analysing discussants' perception of probable failure modes' effect on project outcomes in the requirement analysis stage

Table 5.7 presents the analysis of discussants' perception on the effect of probable failure modes on project outcomes in the requirement analysis stage.

Table 5 7: Mean response of discussants' perceptions on adverse effect of probable failure modes on project outcome (cost, time & scope) in requirement analysis stage



S/N	Probable Failure Modes	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	Effect on Project Outcome
1.	Inappropriate chosen requirement gathering technique	2.56	3.37	3.39	3.11	High
2.	Ambiguous or unclear requirement	3.22	3.25	3.27	3.24	High
3.	Contradictions and confusion in domain specific terminologies	2.38	3.34	2.93	2.88	Moderate
4.	Contradictions and confusion in defining requirements in Natural Language (NL)	3.28	3.18	2.58	3.01	High
5.	Errors in human factor engineering specifications	3.26	2.67	3.19	3.04	High
6.	**Inaccurate requirement**	**4.35**	**4.73**	**4.72**	**4.60**	**Severe**
7.	Unrealistic requirement	3.78	3.67	3.76	3.74	High
8.	Inconsistent requirement	3.56	4.01	4.22	3.35	High
9.	Non-traceable requirement	3.56	3.35	3.15	3.93	High
10.	Non-verifiable requirement	3.23	3.45	3.59	3.42	High
11.	Infeasible requirement	3.12	2.99	3.31	3.14	High

Total number of participants = 15 M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Time} = Mean score of Likely Effect on Project Time M_{Scope} = Mean score of Likely Effect on Project Scope
M_{PO} (Mean score of Likely Effect on Project Outcome) = (M_{Cost} + M_{Time} + M_{Scope})/3

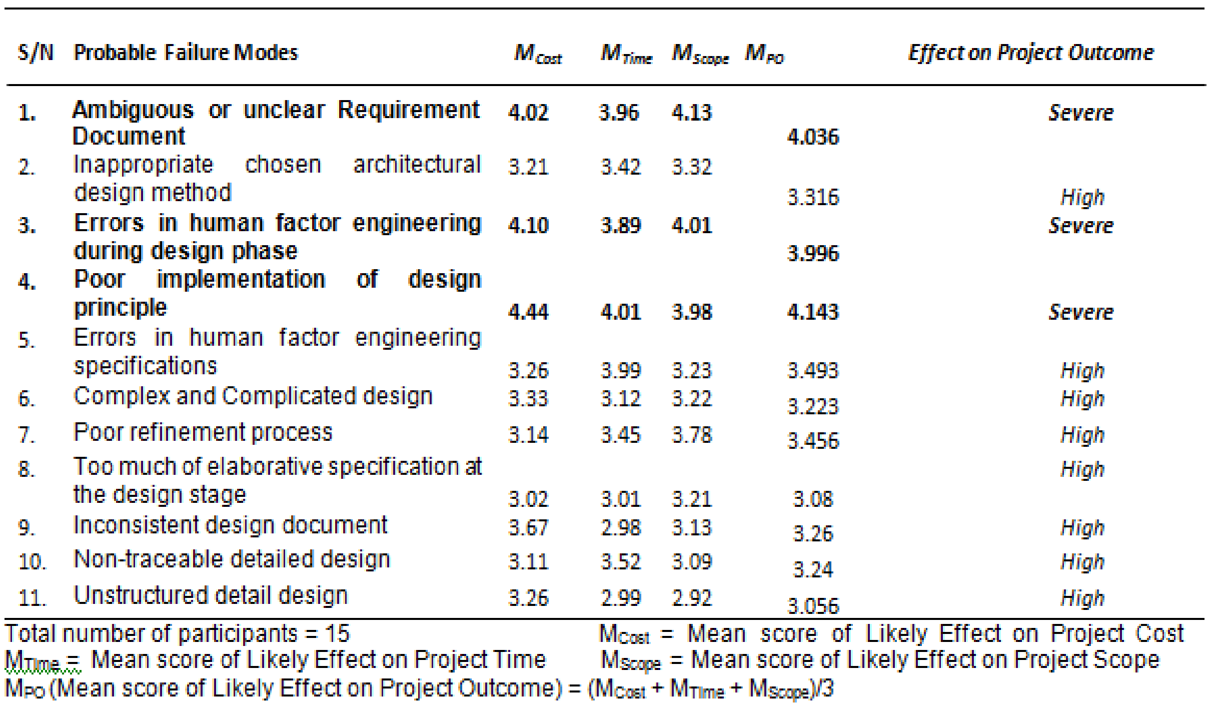
The data presented in Table 5.7 shows that all the 11 probable failure modes identified in the requirement analysis stage of the software development process have M_{PO} rating greater than 2.5. This result indicates that all the identified failure modes have influence on the software project outcomes in terms of cost, time and project scope or quality. The table reveals that probable failure mode: inaccurate requirement (M_{PO} = 4.60) will pose severe effect on project outcome and the failure mode: contradictions and confusion in domain specific terminologies (M_{PO} = 2.88) will pose the least damage to the project outcome. The remaining identified failure modes have M_{PO} ranking range of 3.99 ≤ M_{PO} ≥ 3.00 (non-traceable requirement (3.93), unrealistic requirement (3.74), non-verifiable requirement (3.42), inconsistent requirement (3.25), ambiguous or unclear requirement (3.24), infeasible requirement (3.14), errors in human factor engineering specifications (3.04) and ambiguity nature of requirements in natural language requirement (3.01).

The result analysis from the Table 5.7 also reveals that the probable failure mode: inaccurate requirement will pose the highest effect on project cost, time and quality with $M_{cost,}$ M_{time} and M_{scope} rankings: 4.35, 4.73 and 4.72 respectively while the failure: contradictions and confusion in domain specific terminologies will have the least effect on project cost,(M_{cost} = 2.38) and the failure: contradictions and confusion in defining requirements in natural language will have the least effect on project scope (M_{scope} = 2.58).

5.6.3 Analysing discussants' perception of probable failure modes' effect on project outcome in the design stage

Table 5.8 presents the result analysis of discussants' perception on effect of probable failure modes on project outcome in the design stage.

Table 5 8: Mean response of discussants' perceptions on adverse effect of probable failure modes on project outcome (cost, time and scope) in design stage



S/N	Probable Failure Modes	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	Effect on Project Outcome
1.	**Ambiguous or unclear Requirement Document**	**4.02**	**3.96**	**4.13**	**4.036**	***Severe***
2.	Inappropriate chosen architectural design method	3.21	3.42	3.32	3.316	*High*
3.	**Errors in human factor engineering during design phase**	**4.10**	**3.89**	**4.01**	**3.996**	***Severe***
4.	**Poor implementation of design principle**	**4.44**	**4.01**	**3.98**	**4.143**	***Severe***
5.	Errors in human factor engineering specifications	3.26	3.99	3.23	3.493	*High*
6.	Complex and Complicated design	3.33	3.12	3.22	3.223	*High*
7.	Poor refinement process	3.14	3.45	3.78	3.456	*High*
8.	Too much of elaborative specification at the design stage	3.02	3.01	3.21	3.08	*High*
9.	Inconsistent design document	3.67	2.98	3.13	3.26	*High*
10.	Non-traceable detailed design	3.11	3.52	3.09	3.24	*High*
11.	Unstructured detail design	3.26	2.99	2.92	3.056	*High*

Total number of participants = 15 M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Time} = Mean score of Likely Effect on Project Time M_{Scope} = Mean score of Likely Effect on Project Scope
M_{PO} (Mean score of Likely Effect on Project Outcome) = (M_{Cost} + M_{Time} + M_{Scope})/3

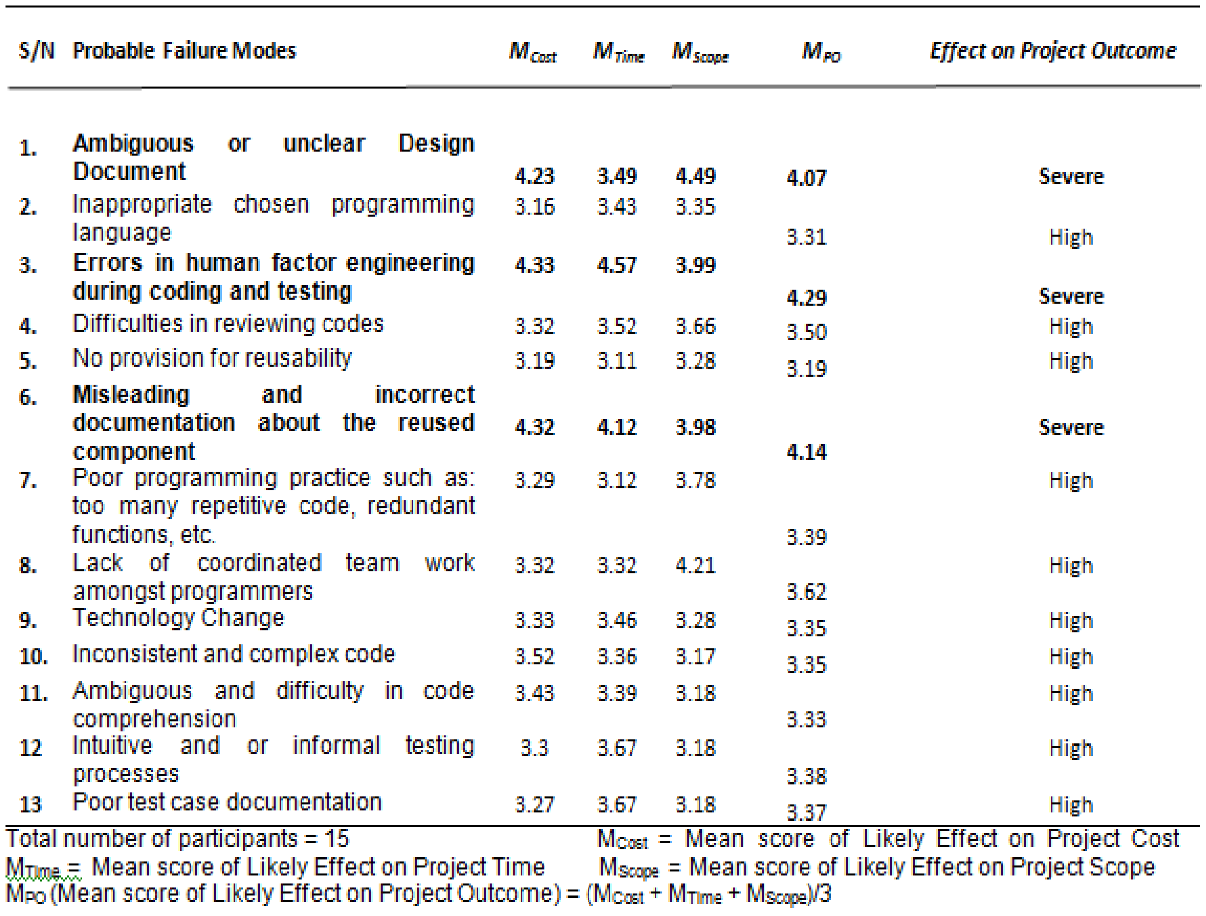
The result analysis of data presented in Table 5.8 indicates that all the 10 failure items identified in the design stage of software development process have the high potential of influencing software project outcome (having the M_{PO} rating range as: $3.99 \leq M_{PO} \geq 3.00$). The results show that three out of the 10 identified failure probable failure modes have a severe potential of damaging the project outcome with the failure: *poor implementation of design principle* having the highest effect ($M_{PO} = 4.44$), followed by the failure: *errors in human factor engineering during design phase* ($M_{PO} = 4.10$) and then the failure: *ambiguous or unclear requirement document* ($M_{PO} = 4.02$).

A further analysis of the results in the Table 5.8 shows that failure: *poor implementation of design principle* has the highest effect of project cost and time with M_{cost} and M_{time} ratings of 4.44 and 4.01 respectively while the failure: *ambiguous or unclear requirement document pose* the highest threat to the project scope with M_{scope} rating of 4.04.

5.6.4 Analysing discussants' perception of probable failure modes' effect on project outcome in the implementation and coding stage

Table 5.9 presents the result analysis of discussants' perception on effect of probable failure modes on project outcome in the implementation and coding stage of software development process.

Table 5 9: Mean response of discussants' perceptions on adverse effect of probable failure modes on project outcome (cost, time & scope) in implementation and coding stage



S/N	Probable Failure Modes	*M_{Cost}*	*M_{Time}*	*M_{Scope}*	*M_{PO}*	*Effect on Project Outcome*
1.	**Ambiguous or unclear Design Document**	**4.23**	**3.49**	**4.49**	**4.07**	**Severe**
2.	Inappropriate chosen programming language	3.16	3.43	3.35	3.31	High
3.	**Errors in human factor engineering during coding and testing**	**4.33**	**4.57**	**3.99**	**4.29**	**Severe**
4.	Difficulties in reviewing codes	3.32	3.52	3.66	3.50	High
5.	No provision for reusability	3.19	3.11	3.28	3.19	High
6.	**Misleading and incorrect documentation about the reused component**	**4.32**	**4.12**	**3.98**	**4.14**	**Severe**
7.	Poor programming practice such as: too many repetitive code, redundant functions, etc.	3.29	3.12	3.78	3.39	High
8.	Lack of coordinated team work amongst programmers	3.32	3.32	4.21	3.62	High
9.	Technology Change	3.33	3.46	3.28	3.35	High
10.	Inconsistent and complex code	3.52	3.36	3.17	3.35	High
11.	Ambiguous and difficulty in code comprehension	3.43	3.39	3.18	3.33	High
12	Intuitive and or informal testing processes	3.3	3.67	3.18	3.38	High
13	Poor test case documentation	3.27	3.67	3.18	3.37	High

Total number of participants = 15 M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Time} = Mean score of Likely Effect on Project Time M_{Scope} = Mean score of Likely Effect on Project Scope
M_{PO} (Mean score of Likely Effect on Project Outcome) = (M_{Cost} + M_{Time} + M_{Scope})/3

The data presented in Table 5.9 shows that all the 13 listed probable failure modes in design stage of software development process have high potential of influencing the project outcome with each of them having M_{PO} rating above 3.0. The result analysis shows that three out of the 13 identified failures will pose severe damage to the project outcome (errors in human factor engineering during coding and testing (M_{PO} = 4.29), misleading and incorrect documentation about the reused component (M_{PO} = 4.14) and ambiguous or unclear design document (M_{PO} = 4.04)).

Furthermore, the table shows that the failure: Errors in human factor engineering during coding and testing have the highest M_{PO} rating while the failure: No provision for reusability has the lowest M_{PO} rating. In addition, the failure: misleading and incorrect documentation about the reused component have the highest effect on project cost (M_{cost} = 4.32) and project M_{time}.

5.6.5 Analysing discussants' perception of probable failure modes' effect on the project outcome in the installation and maintenance stage

Table 5.10 presents the result analysis of discussants' perception on effect of probable failure modes on project outcome in the installation and maintenance of software development process.

Table 5 10: : Mean response of discussants' perceptions on adverse effect of probable failure modes on project outcome (cost, time and scope) in installation and maintenance stage

S/N	Probable Failure Modes	M_{Cos}	M_{Time}	M_{Scope}	M_{PO}	Effect on Project Outcome
1.	Experiencing difficulties during installation	3.33	3.23	212	2.89	Moderate
2.	System not installed correctly due to change in environment	3.23	2.38	2.29	2.63	Moderate
3.	New set of requirements are emerging	3.12	3.56	2.99	3.22	High
4.	Persisting difficulties in using the systems	3.23	3.34	2.56	3.04	High
5.	Expected functionalities are omitted	3.45	2.76	3.12	3.11	High
6.	Misleading and incorrect documentation about the system operation	4.18	4.23	3.96	4.12	**Severe**
7.	Several faults and errors emerging later during acceptance testing	3.23	3.41	3.21	3.28	High
8.	Adequate testing not conducted	3.54	3..23	3.18	3.36	High
9.	Difficulties experienced by software engineers in fixing reported problems	3.48	2.98	3.13	3.19	High
10.	Difficulties in system maintainability	3.45	3.45	3.39	3.43	High
11.	Change on one component of the new system having adverse effect on the rest of the system	4.35	4.99	3.89	4.41	**Severe**
12	Quest for system change is inevitable due to: System being slow for the current loading and concern about security measures	3.33	*3.119*	3.23	3.22	High

Total number of participants = 15
M_{Time} = Mean score of Likely Effect on Project Time
M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Scope} = Mean score of Likely Effect on Project Scope
M_{PO} (Mean score of Likely Effect on Project Outcome) = $(M_{Cost} + M_{Time} + M_{Scope})/3$

The result analysis of data presented in Table 5.10 indicates that all the 12 failure items identified in the design stage of software development process have the potential of influencing software project outcome (having the M_{PO} rating range as: $3.00 \leq M_{PO} \geq 2.5$). The results show that two out of the 12 identified failure modes are severe (Change on one component of the new system having adverse effect on the rest of the system (M_{Po} = 4.41) and the failure: system not installed correctly due to change in environment with (M_{PO} = 2.29) have the least.

The table also indicates that the failure: change on one component of the new system having adverse effect on the rest of the system pose the highest threat to project cost and project time, while the failure: experiencing difficulties during installation pose the least threat to project cost and the failure: system not installed correctly due to change in environment poses the minimum threat to the project time and scope.

5.7 Analysing the PFMEOPOQ data obtained from Field study 3

The PFMEOPOQ consists of two sections. Section 1 asked about participants' personal information. The left column of section 2 sought for participants' views on the '*Level of Acceptance/Agreement'* of the listed probable failure modes as a potential risk factor in the software development process stages. On the right-hand column of section 2, participants were asked to circle the responses that best describes the '*Likely Effect'* of the listed probable failure mode on software project outcome (cost, time and quality/scope).

The PFMEOPOQ data will be analysed according to the sequence of its two constituent sections. It begins with the analysis of participants' personal information, then the participants' responses to the level of acceptance of the listed probable failure mode as potential failure mode and then the analysis of their responses to the likely effect of the studied failure mode on project outcome. The main objective of this analysis is to validate the findings of previous field studies.

5.7.1 Analysing participants' personal information

Table 5.11 shows the frequency and percentage distribution of the participants' personal information.

Table 5 11: Participants' personal information

Personal Information	Count	Percent (%)
Age:		
25-30 years	12	15.6
30-39 years	26	33.8
40 years and above	39	50.5
Total	77	
Gender:		
Male	53	68.8
Female	24	31.2
Total	77	
Professional Background:		
Programmer	20	26
Software Engineer	22	28.6
System Analyst	13	16.9
Software manager	14	18.2
Domain Experts	8	10.4
Other	-	-
Total	77	
Software Projects Successfully Developed:		
2 -5	18	23.4
6-10	36	46.8
11 and above	23	29.9
Total	77	
Years of experience in current job:		
<5 Year	12	15.6
6-10 Years	38	49.4
>10 Years	27	35.1
Total	77	
Experience in Software Development:		
<5 Year	-	-
5-10 Years	37	48.1
>10 Years	40	51.9
Total	77	

As shown in Table 5.11, a total of 77 software development practitioners and researchers in SE around the world participated in this survey. The analysis of the participants' personal information shows majority of the participants 39 (50.5%) are 40 years and above. Also, 26 (33.8%) fall within the age group (30-39) years and the remaining 12 (15.6%) fell within the age group of 25-30 years. The analysis also reveals that a total of 53 (68.8%) participants are male and 24 (31.2%) are female. Furthermore, the analysis of the participants' professional background shows that 20(26%) are programmers, 22 (28.6%) are software engineers, 13 (16.9%) are system analysts, 14 (18.2%) are software managers and 8 (10.4%) are domain experts. Analysing the number of successfully completed software projects, the result of the analysis shows that 18 (23.4%) of the participants' have successfully completed between 2 to 5 projects while 36 (23.4%) completed between 6 -10 projects and only 23 (29.9%) of the participating software development professionals completed projects that are above 10.

Furthermore, results presented in the Table 5.11 show that 38 (49.4%) of the participants have spent between 6 -10 years in their current job, 27 (35.1%) spent between 6-10 years and 12 (15.6%) spent less than 5 years in their current job. A further analysis on the years of experience spent in software development business shows that 37 (48.1%) spent less than 10 years in the software development business while 40 (51.9%) have spent over 10 years of experience in at least one of the following software applications areas: security, safety critical systems, tracking, sorting, banking, education, real time applications and stock control the business. The implication of this analysis is to provide further guidance to future research in this field.

5.7.2 Analysing participants' views on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'potential failure modes' in the Stage1 (planning) and their likely effect on project outcome

Table 5.12 shows the results of data analysis of participants' opinion on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' and their Likely Effect on project outcome in the planning stage.

Table 5 12: Mean response of experts and researchers in se on level of acceptance of probable failure modes as potential failure mode in the Stage1 (planning) and their likely effect on project outcome (cost, time & scope)

S/N	Probable Failure Modes	M_{LoA}	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	*Failure Status*	*Effect*
1.	Inappropriate chosen software development framework/methodology/models	3.88	2.98	3.11	3.44	*3.18*	PFM	High
2.	Communication gap amongst the development team	3.11	2.10	3.10	3.64	*2.95*	PFM	Moderate
3.	Ambiguous or unclear specification	3.53	3.34	3.54	3.33	*3.40*	PFM	High
4.	Insufficient software project tools and resources	3.23	3.22	3.21	3.11	*3.18*	PFM	High
5.	Unrealistic project goals and scopes	2.89	2.91	3.02	3.15	*3.03*	PFM	High
6.	Unrealistic budget	3.24	4.10	3.22	2.78	*3.37*	PFM	High
7.	Ambiguous project hierarchy model	3.02	3.11	2.21	2.25	*2.58*	PFM	Moderate
8.	Inexperience project team members	2.89	2.91	3.02	3.54	*3.16*	PFM	High
9.	**Poor risk management strategy**	**4.05**	**4.56**	**4.82**	**4.23**	***4.54***	**PFM**	**Severe**
10.	Deviance of system security integration	2.81	2.45	3.21	3.51	*3.06*	PFM	High

Total number of participants = 77 *M_{LoA} = Mean score of Level of Acceptance*
M_{Cost} = Mean score of Likely Effect on Project Cost M_{Time} = Mean score of Likely Effect on Project Time
M_{Scope} = Mean score of Likely Effect on Project Scope PFM = potential failure mode

In Table 5.12, the M_{LOA} ranking of all the listed probable failure modes is above 2.5. This result indicated that all the enquired participants agreed that the failure statuses of all the listed probable failure modes are potential failure modes. Similarly, the M_{PO}

ranking of all the listed probable failure modes is above 2.5. This result also indicates that the enquired software experts and researchers to be potential threat to software project outcome perceive all the identified probable failure modes in the planning stage of the software development process. The table shows that the proportion of threat posed by the failures varies. For instance, the failure: *poor risk management strategy* has the highest M_{PO} ranking as 4.54, hence, considered to pose severe threat to the project outcome. Also, two out the 10 listed probable failure modes (*communication gap amongst the development team and ambiguous project hierarchy model*) are considered to pose moderate threats with M_{PO} ranking of 2.95 and 2.58.

Furthermore, the failure: *poor risk management strategy is considered by the participants to have the highest to project cost, time and project scope while the failure:* communication gap amongst the development team has the least effect of project cost and the failure: Ambiguous project hierarchy model is considered to have the least effect on project time and scope.

5.7.3 Analysing participants' views on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' in the Stage2 (requirement specification and analysis) and their likely effect on project outcome

Table 5.13 presents the results of data analysis of participants' opinion on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' and their Likely Effect on project outcome in the requirement specification and analysis stage.

Table 5 13: Mean response of software development professionals on level of acceptance of probable failure modes as potential failure mode in the Stage2 (requirement specification and analysis) and their likely effect on project outcome (cost, time & scope)

S/N	Probable Failure Modes	M_{LoA}	M_{Cost}	M_{Time}	M_{Scope}	Failure Status	Effect
1.	Inappropriate chosen requirement gathering technique	3.45	3.57	3.33	3.76	PFM	High
2.	Ambiguous or unclear requirement	3.56	3.12	3.29	3.31	PFM	High
3.	Contradictions and confusion in domain specific terminologies	2.98	3.43	2.99	3.06	PFM	High
4.	Contradictions and confusion in defining requirements in Natural Language (NL)	3.28	3.32	2.56	3.55	PFM	High
5.	Errors in human factor engineering specifications	3.26	2.99	3.12	3.68	PFM	High
6.	**Inaccurate requirement**	**3.65**	**4.73**	**4.72**	**4.67**	**PFM**	**Severe**
7.	Unrealistic requirement	3.78	3.56	3.78	3.65	PFM	High
8.	Inconsistent requirement	3.56	4.01	4.21	3.34	PFM	High
9.	Non-traceable requirement	3.66	2.98	3.13	3.24	PFM	High
10.	Non-verifiable requirement	3.55	3.45	3.39	3.22	PFM	High
11.	Infeasible requirement	3.62	2.99	3.31	3.28	PFM	High

Total number of participants = 77
M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Scope} = Mean score of Likely Effect on Project Scope
M_{LoA} = *Mean score of Level of Acceptance*
M_{Time} =Mean score of Likely Effect on Project Time
PFM = potential failure mode

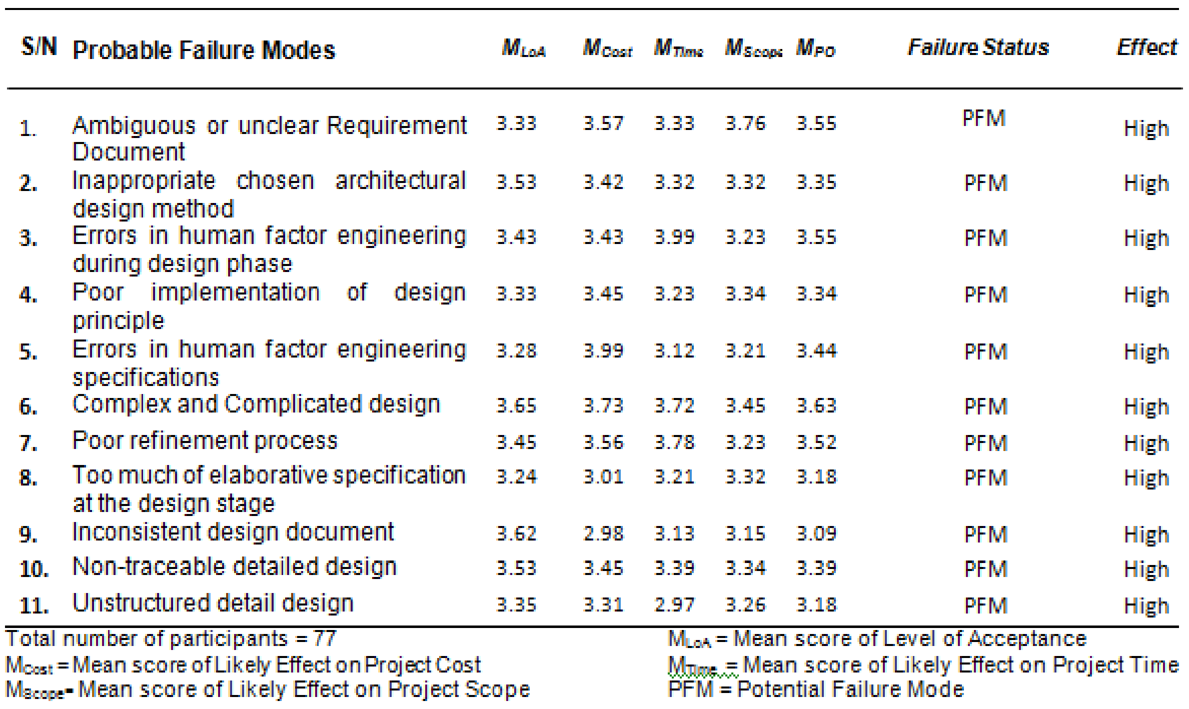
Result presented in Table 5.13 shows that all the listed probable failure modes are agreed to be potential failure modes by the consulted participants having the M_{LOA} rating above 2.5. The listed failures are also considered by the enquired researchers to possess the potential of damaging the project outcome at varying degree. As shown in the table, the failure: *Inaccurate requirement* is considered to *pose* the highest threat to the project outcome with M_{PO} rating of 4.67 while the failure: *Contradictions and confusion in domain specific terminologies Contradictions and confusion in domain specific terminologies* is considered to pose the least threat to the project outcome with M_{PO} rating of 3.06.

The result in the Table 5.13 also shows that the failure: *inaccurate requirement* is considered to *pose* the highest threat to the project cost, duration and quality of the project in the requirement specification stage with the M_{cost}, M_{time} and M_{scope} ratings of 4.73, 4.72 and 4.67 respectively. The failure: non-traceable requirement is considered to pose the least threat to project cost.

5.7.4 Analysing participants' views on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' in the Stage3 (design) and their likely effect on project outcome

Table 5.14 presents the results of data analysis of participants' opinion on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' and their 'Likely Effects' on the project outcome in the design stage.

Table 5 14: Mean response of software development professionals on level of acceptance of probable failure modes as potential failure modes in the Stage 3 (Design) and their likely effect on project outcome (cost, time and scope)



S/N	Probable Failure Modes	M_{LoA}	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	*Failure Status*	*Effect*
1.	Ambiguous or unclear Requirement Document	3.33	3.57	3.33	3.76	3.55	PFM	High
2.	Inappropriate chosen architectural design method	3.53	3.42	3.32	3.32	3.35	PFM	High
3.	Errors in human factor engineering during design phase	3.43	3.43	3.99	3.23	3.55	PFM	High
4.	Poor implementation of design principle	3.33	3.45	3.23	3.34	3.34	PFM	High
5.	Errors in human factor engineering specifications	3.28	3.99	3.12	3.21	3.44	PFM	High
6.	Complex and Complicated design	3.65	3.73	3.72	3.45	3.63	PFM	High
7.	Poor refinement process	3.45	3.56	3.78	3.23	3.52	PFM	High
8.	Too much of elaborative specification at the design stage	3.24	3.01	3.21	3.32	3.18	PFM	High
9.	Inconsistent design document	3.62	2.98	3.13	3.15	3.09	PFM	High
10.	Non-traceable detailed design	3.53	3.45	3.39	3.34	3.39	PFM	High
11.	Unstructured detail design	3.35	3.31	2.97	3.26	3.18	PFM	High

Total number of participants = 77
M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Scope}= Mean score of Likely Effect on Project Scope
M_{LoA} = Mean score of Level of Acceptance
M_{Time} = Mean score of Likely Effect on Project Time
PFM = Potential Failure Mode

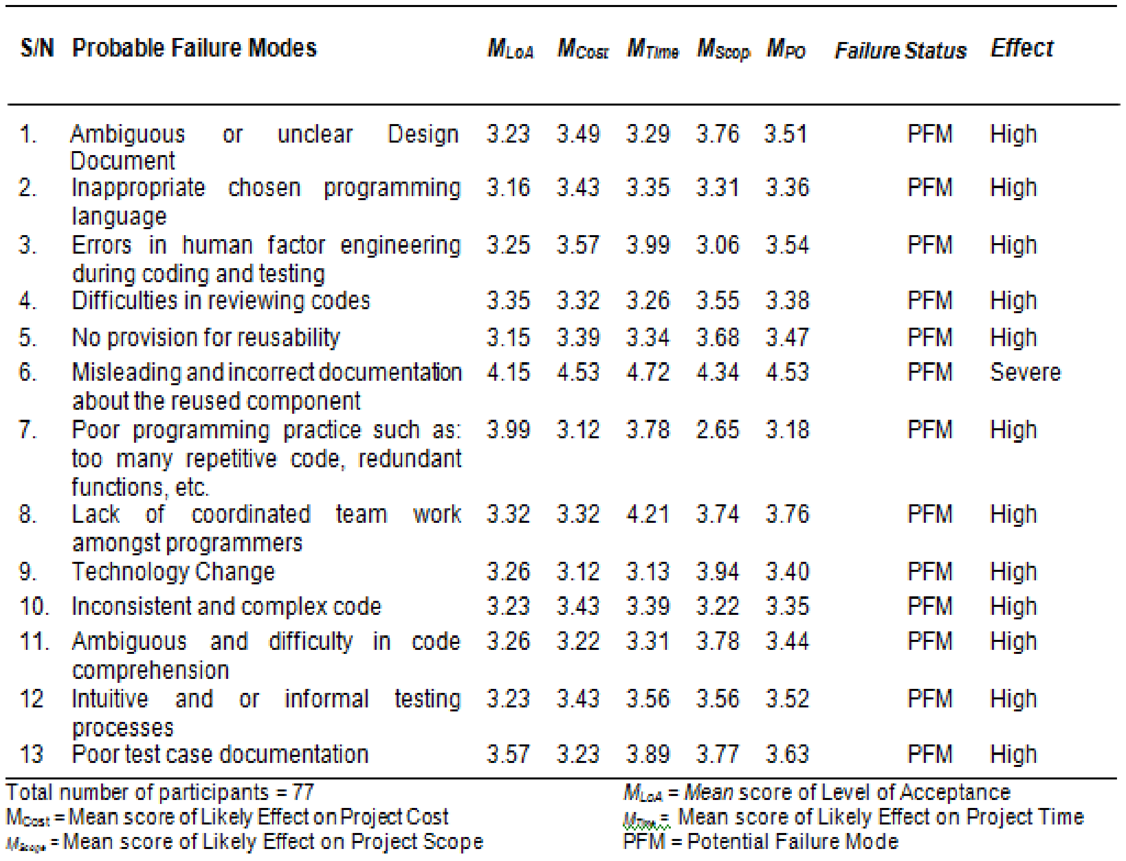
The result analysis of data presented in Table 5.14 shows that all the listed probable failure modes in the design stage of the software development process are considered to be potential failure modes in the stage by the enquired experts and researchers with each failure having a M_{LOA} rating above 2.5. Similarly, the results also indicate that each of the listed probable failure modes is considered to be a threat to software project outcome with each of the failure having an M_{PO} rating range of $3.99 \leq M_{PO} \geq 3.00$.

Furthermore, the result analysis of data presented in Table 5.14 shows that all the probable failure modes will pose nearly same quantum of threat to software project cost, duration and quality as shown by their individual M_{PO} rating.

5.7.5 Analysing participants' views on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' in the Stage4 (implementation and coding) and their Likely Effect on project outcome

Table 5.15 presents the results of data analysis of participants' opinion on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' and their Likely Effect on project outcome in the implementation and coding stage.

Table 5 15: Mean response of software development professionals on level of acceptance of probable failure modes as potential failure modes in the Stage 4 (Implementation and Coding) and their likely effect on project outcome (cost, time and scope)



S/N	Probable Failure Modes	M_{LOA}	M_{Cost}	M_{Time}	M_{Scop}	M_{PO}	Failure Status	Effect
1.	Ambiguous or unclear Design Document	3.23	3.49	3.29	3.76	3.51	PFM	High
2.	Inappropriate chosen programming language	3.16	3.43	3.35	3.31	3.36	PFM	High
3.	Errors in human factor engineering during coding and testing	3.25	3.57	3.99	3.06	3.54	PFM	High
4.	Difficulties in reviewing codes	3.35	3.32	3.26	3.55	3.38	PFM	High
5.	No provision for reusability	3.15	3.39	3.34	3.68	3.47	PFM	High
6.	Misleading and incorrect documentation about the reused component	4.15	4.53	4.72	4.34	4.53	PFM	Severe
7.	Poor programming practice such as: too many repetitive code, redundant functions, etc.	3.99	3.12	3.78	2.65	3.18	PFM	High
8.	Lack of coordinated team work amongst programmers	3.32	3.32	4.21	3.74	3.76	PFM	High
9.	Technology Change	3.26	3.12	3.13	3.94	3.40	PFM	High
10.	Inconsistent and complex code	3.23	3.43	3.39	3.22	3.35	PFM	High
11.	Ambiguous and difficulty in code comprehension	3.26	3.22	3.31	3.78	3.44	PFM	High
12	Intuitive and or informal testing processes	3.23	3.43	3.56	3.56	3.52	PFM	High
13	Poor test case documentation	3.57	3.23	3.89	3.77	3.63	PFM	High

Total number of participants = 77
M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Scope} = Mean score of Likely Effect on Project Scope
M_{LOA} = *Mean* score of Level of Acceptance
M_{Time} = Mean score of Likely Effect on Project Time
PFM = Potential Failure Mode

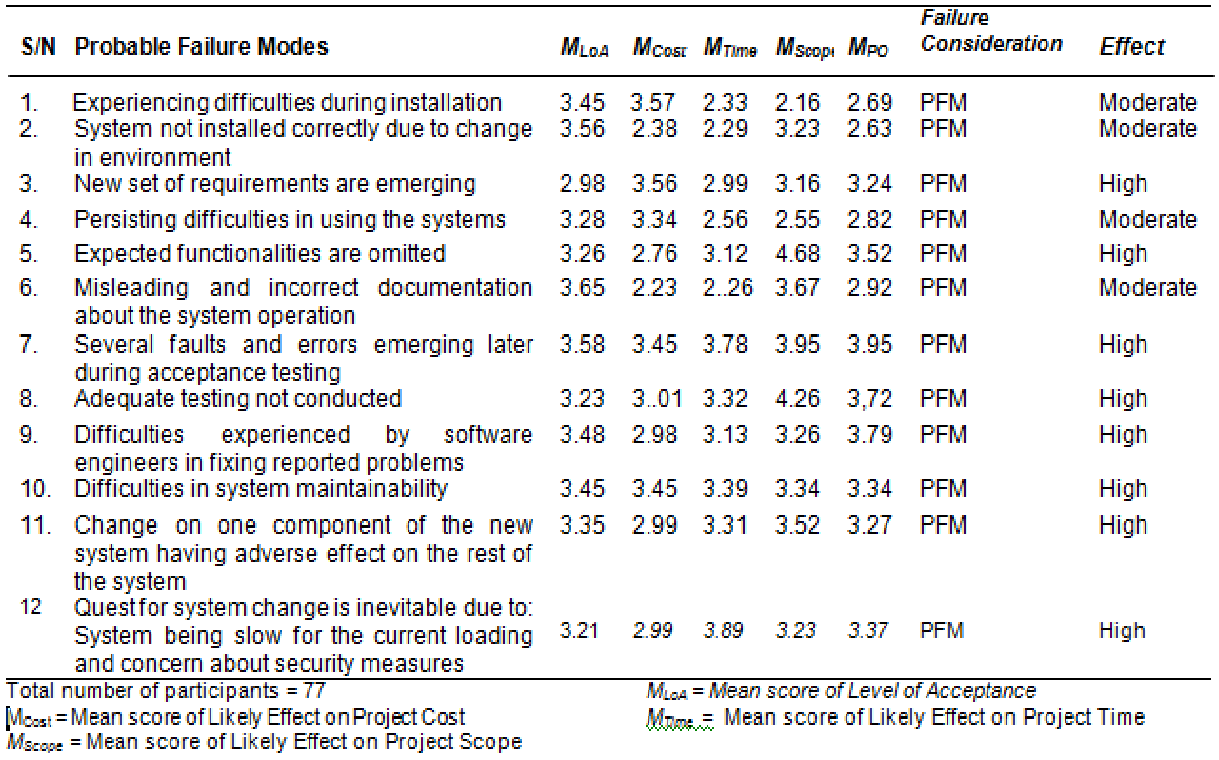
The results presented in Table 5.15 show that the M_{LOA} ratings for all probable failure modes in the implementation and coding stage are greater than 2.5. Thus, they are all considered to be potential failure modes in the stage. The results also indicate that all the listed failure modes in the stage are considered by the enquired experts to be potential threat to the software project outcome with each of the failure mode having M_{PO} rating above 3.00. Furthermore, the failure: *Misleading and incorrect documentation about the reused component* is considered to have the highest rating for M_{PO}, M_{cost}, M_{time} and M_{scope}.

Thus, the failure is considered to have the possibility of posing severe threat to project cost, duration of the project quality and to the overall project outcome. However, the failure: *Poor programming practice such as: too many repetitive code, redundant functions, etc.* is considered to pose the least threat to the project quality with M_{scope} rating of 2.65.

5.7.6 Analysing participants' views on the 'Level of Acceptance/Agreement' of the listed probable failure modes as 'Potential Failure Modes' in the Stage5 (Installation and maintenance) and their Likely Effect on project outcome

Table 5.16 presents the results of data analysis of participants' opinion on the 'Level of Acceptance/Agreement' of the listed probable failure modes as a 'Potential Failure Modes' and their Likely Effect on project outcome in the installation and maintenance stage.

Table 5 16:: Mean response of software development professionals on level of acceptance of probable failure modes as potential failure mode in the Stage5 (installation and maintenance) and their likely effect on project outcome (cost, time and scope)



S/N	Probable Failure Modes	M_{LoA}	M_{Cost}	M_{Time}	M_{Scope}	M_{PO}	*Failure Consideration*	*Effect*
1.	Experiencing difficulties during installation	3.45	3.57	2.33	2.16	2.69	PFM	Moderate
2.	System not installed correctly due to change in environment	3.56	2.38	2.29	3.23	2.63	PFM	Moderate
3.	New set of requirements are emerging	2.98	3.56	2.99	3.16	3.24	PFM	High
4.	Persisting difficulties in using the systems	3.28	3.34	2.56	2.55	2.82	PFM	Moderate
5.	Expected functionalities are omitted	3.26	2.76	3.12	4.68	3.52	PFM	High
6.	Misleading and incorrect documentation about the system operation	3.65	2.23	2..26	3.67	2.92	PFM	Moderate
7.	Several faults and errors emerging later during acceptance testing	3.58	3.45	3.78	3.95	3.95	PFM	High
8.	Adequate testing not conducted	3.23	3..01	3.32	4.26	3,72	PFM	High
9.	Difficulties experienced by software engineers in fixing reported problems	3.48	2.98	3.13	3.26	3.79	PFM	High
10.	Difficulties in system maintainability	3.45	3.45	3.39	3.34	3.34	PFM	High
11.	Change on one component of the new system having adverse effect on the rest of the system	3.35	2.99	3.31	3.52	3.27	PFM	High
12	Quest for system change is inevitable due to: System being slow for the current loading and concern about security measures	3.21	*2.99*	*3.89*	*3.23*	*3.37*	PFM	High

Total number of participants = *77*
M_{Cost} = Mean score of Likely Effect on Project Cost
M_{Scope} = Mean score of Likely Effect on Project Scope
M_{LoA} = *Mean score of Level of Acceptance*
M_{Time} = Mean score of Likely Effect on Project Time

The result analysis of data presented in Table 5.16 shows that all the probable failure mode identified in the installation and maintenance stage have M_{LOA} rating above 2.5. Thus, they are agreed to be potential failure modes in the stage. The result also shows that the M_{PO} rating for all identified probable failure mode is above 2.5. This implies that they are all considered potential threat to the project outcome.

Further analysis of the result as presented in the table shows that the failure: Several faults and errors emerging later during acceptance testing (M_{PO} = 3.95) will pose the highest threat to the overall project outcome while the failure: *System not installed correctly due to change in environment* (M_{PO} = 2.63) will pose the least threat to the overall project outcome in the stage. Also, participants perceived the failure: *New set of requirements are emerging* (M_{cost} = 3.56) to pose the highest damage to the project cost while the failure: *Experiencing difficulties during installation* is perceived to pose the least threat to the project duration and to the quality of the project (M_{time} = 2.33, M_{scope} = 2.16).

5.8 Analysing the FMEA data obtained from Field study 4

The main purpose for conducting field study 4 is to assess the risk factors associated with potential failure modes in the software development process using FMEA process. In the study, the FMEA worksheet adopted from USDOD (MIL-STD-1629, 1980) was used to document the assigned ratings and risk factors assessment in the FMEA process.

As earlier mentioned in the preceding chapter, the FMEA process was conducted to:

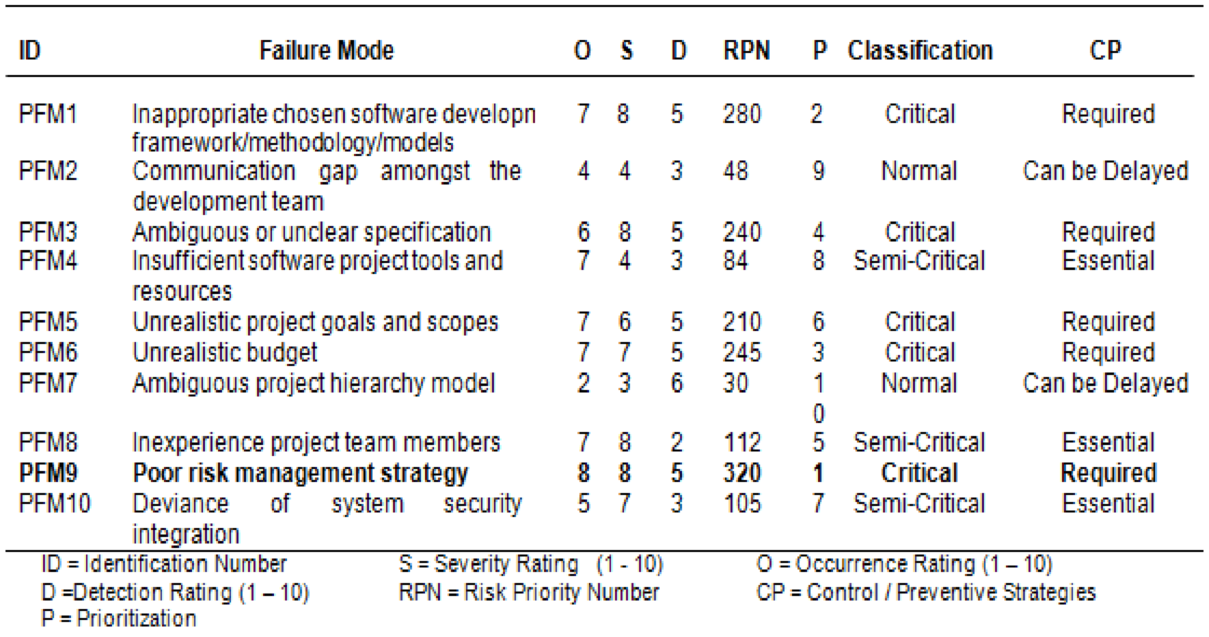
- assess the probability of occurrence for each potential failure mode
- assign severity rating for the listed potential failure mode
- assign detection rating for the listed failure
- calculate the RPN for the listed potential failure mode
- prioritize the risk associated with potential failure mode
- classify the risk, and
- suggest suitable control action for the risk

In order to attain the objective of the field study 4, the FMEA worksheet data will present data reflecting: the potential failure mode identity, the occurrence rating (O), severity rating (S), the detection rating (D), risk priority number (RPN) values, prioritization, classification of the risks and control action suitable for each risk.

5.8.1 Analysing the risk factors associated with potential failure modes in the planning stage

Table 5.17 presents the result of data analysis of FMEA worksheet document for the FMEA process conducted in planning stage.

Table 5 17: FMEA worksheet documenting the assessment of risk factors associated with potential failure modes in the planning stage



ID	Failure Mode	O	S	D	RPN	P	Classification	CP
PFM1	Inappropriate chosen software developn framework/methodology/models	7	8	5	280	2	Critical	Required
PFM2	Communication gap amongst the development team	4	4	3	48	9	Normal	Can be Delayed
PFM3	Ambiguous or unclear specification	6	8	5	240	4	Critical	Required
PFM4	Insufficient software project tools and resources	7	4	3	84	8	Semi-Critical	Essential
PFM5	Unrealistic project goals and scopes	7	6	5	210	6	Critical	Required
PFM6	Unrealistic budget	7	7	5	245	3	Critical	Required
PFM7	Ambiguous project hierarchy model	2	3	6	30	10	Normal	Can be Delayed
PFM8	Inexperience project team members	7	8	2	112	5	Semi-Critical	Essential
PFM9	**Poor risk management strategy**	**8**	**8**	**5**	**320**	**1**	**Critical**	**Required**
PFM10	Deviance of system security integration	5	7	3	105	7	Semi-Critical	Essential

ID = Identification Number S = Severity Rating (1 - 10) O = Occurrence Rating (1 – 10)
D =Detection Rating (1 – 10) RPN = Risk Priority Number CP = Control / Preventive Strategies
P = Prioritization

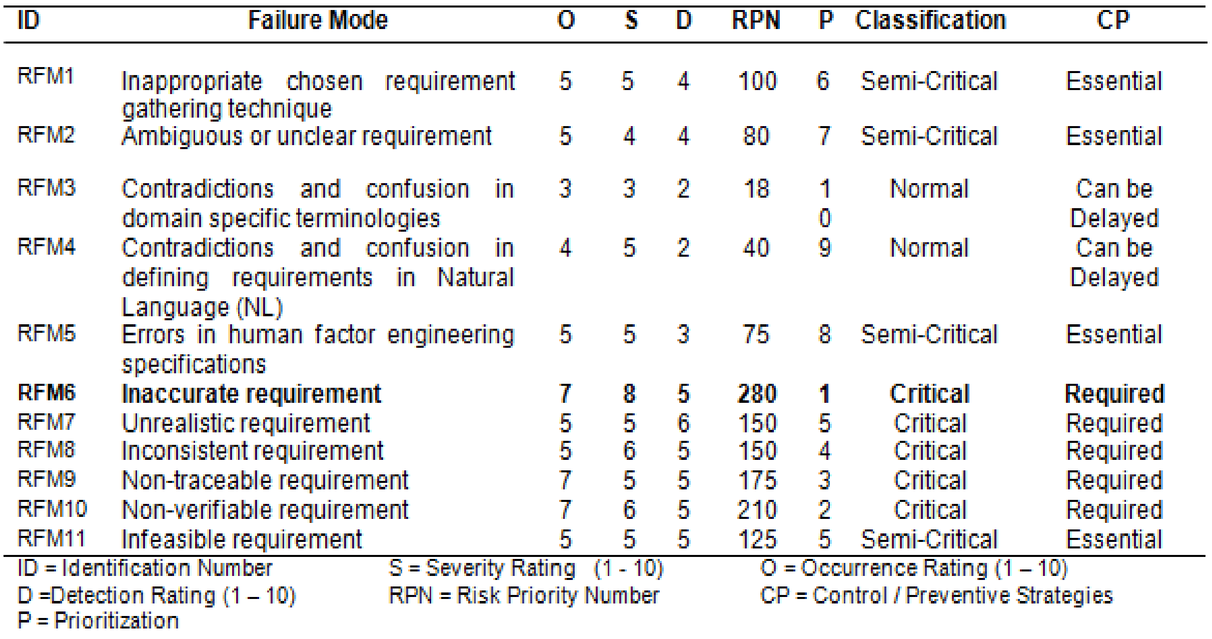
Result of data analysis of FMEA worksheet presented in Table 5.17 shows that the potential failure mode with ID PFM9 has the highest RPN of 320. Thus, the characteristics of the failure PFM9 are: it poses the highest threat to the project outcome in the planning stage. It is prioritized as 1, it is classified as *critical* risk in the stage, hence*, a control action is required* to mitigate its effect on project outcome, and it deserves more attention than any other failure modes in the stage. Other potential failure modes that are classified critical are failures with ID: PFM1 (RPN = 280), PFM6 (RPN = 245), PFM6 (RPN = 210) and PFM3 (RPN = 210) and are prioritized 2,3 and 4 respectively as shown in the Table 5.17.

The result also indicates that the potential failures with ID: PFM18 (RPN = 112), PFM10 (RPN = 105) and PFM4 (RPN = 84) are prioritized 5,6 and 7, and are classified *semi-critical. Thus, control action is essential to mitigate their effect on project outcome.* Potential FM with ID PFM2 (RPN = 48) and PFM7 (RPN = 30) are classified *Normal risk* in the stage; hence, control action can be delayed. These potential failure modes are prioritized 9 and 10 respectively. Implication of this result is that little resources are expected to mitigate these risks and attentions to their associative risk can be delayed.

5.8.2 Analysing the risk factors associated with potential failure modes in the requirement specification analysis stage

Table 5.18 presents the result of data analysis of FMEA worksheet document for the FMEA process conducted in requirement specification analysis stage.

Table 5 18: FMEA Worksheet documenting the assessment of risk factors associated with potential failure modes in the requirement specification analysis stage



ID	Failure Mode	O	S	D	RPN	P	Classification	CP
RFM1	Inappropriate chosen requirement gathering technique	5	5	4	100	6	Semi-Critical	Essential
RFM2	Ambiguous or unclear requirement	5	4	4	80	7	Semi-Critical	Essential
RFM3	Contradictions and confusion in domain specific terminologies	3	3	2	18	10	Normal	Can be Delayed
RFM4	Contradictions and confusion in defining requirements in Natural Language (NL)	4	5	2	40	9	Normal	Can be Delayed
RFM5	Errors in human factor engineering specifications	5	5	3	75	8	Semi-Critical	Essential
RFM6	**Inaccurate requirement**	**7**	**8**	**5**	**280**	**1**	**Critical**	**Required**
RFM7	Unrealistic requirement	5	5	6	150	5	Critical	Required
RFM8	Inconsistent requirement	5	6	5	150	4	Critical	Required
RFM9	Non-traceable requirement	7	5	5	175	3	Critical	Required
RFM10	Non-verifiable requirement	7	6	5	210	2	Critical	Required
RFM11	Infeasible requirement	5	5	5	125	5	Semi-Critical	Essential

ID = Identification Number S = Severity Rating (1 - 10) O = Occurrence Rating (1 – 10)
D =Detection Rating (1 – 10) RPN = Risk Priority Number CP = Control / Preventive Strategies
P = Prioritization

Result of data analysis of FMEA worksheet presented in Table 5.18 shows that the potential failure modes with ID RFM6 has the highest RPN of 280. Thus, the failure RFM9 attributes are: it poses the highest threat to the project outcome in the requirement specification stage, it is prioritized as 1 (meaning: it requires the first attention), it is classified as *a critical* risk in the requirement specification stage of the software development process, hence*, a control action is required* to mitigate its effect on project outcome, and it deserves more attention than any other failure mode in the stage. Other potential failure modes that are classified critical are failures with ID:

RFM10 (RPN = 210), RFM9 (RPN = 175), RFM8 (RPN = 150) and RFM7 (RPN = 150) and are prioritized 2, 3 and 4 respectively as shown in the Table 5.8.1.

The result also indicates that the potential failures with ID: RFM1 (RPN = 100), RFM2 (RPN = 80) and RFM5 (RPN = 75) are prioritized 5, 6 and 7, and are classified *semi-critical. Thus, control action is essential to mitigate their effect on project outcome.* Potential failure modes with ID PFM4 (RPN = 40) and PFM3 (RPN = 18) are classified *Normal risk* in the SDLC stage; hence, control action can be delayed. These potential failure modes are prioritized 9 and 10 respectively. Implication of this result is that little resources are expended to mitigate these risks and attention to their associative risk is delayed.

5.8.3 Analysing the risk factors associated with potential failure modes in the design stage

Table 5.19 presents the result of data analysis of FMEA worksheet document for the FMEA process conducted in design stage.

Table 5 19:: FMEA Worksheet documenting the assessment of risk factors associated with potential failure modes in the design stage

ID	Failure Mode	O	S	D	RPN	P	Classification	CP
DFM1	Ambiguous or unclear Requirement Document	5	6	5	150	4	Critical	Required
DFM2	Inappropriate chosen architectural design method	7	5	5	175	3	Critical	Required
DFM3	Errors in human factor engineering during design phase	5	5	4	100	6	Semi-Critical	Essential
DFM4	Poor implementation of design principle	6	7	5	210	2	Critical	Required
DFM5	**Errors in human factor engineering specifications**	**7**	**6**	**5**	**255**	**1**	**Critical**	**Required**
DFM6	Complex and Complicated design	5	5	5	125	5	Semi-Critical	Essential
DFM7	Poor refinement process	7	5	5	175	3	Critical	Required
DFM8	Too much of elaborative specification at the design stage	5	5	3	75	8	Semi-Critical	Essential
DFM9	Inconsistent design document	5	5	2	50	9	Normal	Can be Delayed
DFM10	Non-traceable detailed design	6	5	3	90	7	Semi-Critical	Essential
DFM11	Unstructured detail design	5	6	3	90	7	Semi-Critical	Essential

ID = Identification Number S = Severity Rating (1 - 10) O = Occurrence Rating (1 – 10)
D =Detection Rating (1 – 10) RPN = Risk Priority Number CP = Control / Preventive Strategies
P = Prioritization

Result of FMEA worksheet data analysis presented in Table 5.19 shows that four potential failure modes with ID: DFM5 (RPN = 255), DFM4 (RPN = 210), DFM7 (RPN

= 175) and DFM1 (RPN =175) are of *critical* status. The degrees of their criticality are in sequence. This indicates that the four potential failure modes sequentially require more developmental resources and more attention than other potential failure modes in the same stage. The table also shows that a potential failure mode with ID DFM5 has the highest RPN of 255. Thus, the failure DFM5 attributes are: it poses the highest threat to the project outcome in the design stage, it is prioritized and ranked 1 (meaning: it requires the highest attention and resources); it is classified as *a critical* risk in the design stage, hence*, a control action is required* to mitigate its effect on project outcome; and it deserves more attention than any other failure mode in the stage.

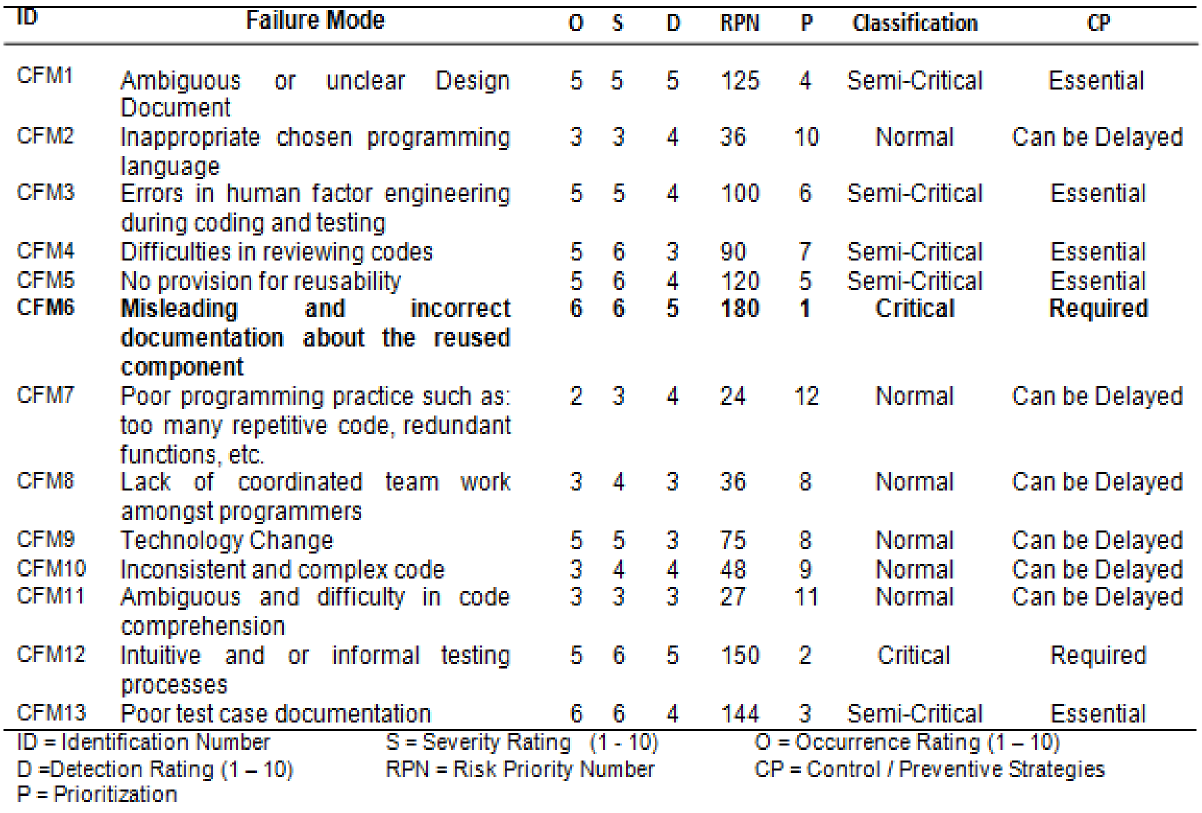
The result also indicated that the potential failures with ID: DFM6 (RPN = 125), DFM3 (RPN = 100), DFM10 (RPN = 90) and DFM11 (RPN = 90) are prioritized 5, 6 and 7 respectively, and are classified *semi-critical. Thus, control action is essential to mitigate their effect on project outcome.*

Results in the table show that only one potential failure mode with ID DFM9 (RPN = 50) is classified *Normal risk* in the software development process stage; hence, its control action can be delayed. The DFM9 failure is prioritized 9. Implication of this result is that little resources are expected to be expended to mitigate its associative risks and attention required to mitigate the risks can be delayed.

5.8.4 Analysing the risk factors associated with potential failure modes in the implementation and coding stage

Table 5.20 presents the result of data analysis of FMEA worksheet document for the FMEA process conducted in implementation and coding stage.

Table 5 20: FMEA worksheet documenting the assessment of risk factors associated with potential failure modes in the implementation and coding stage



ID	Failure Mode	O	S	D	RPN	P	Classification	CP
CFM1	Ambiguous or unclear Design Document	5	5	5	125	4	Semi-Critical	Essential
CFM2	Inappropriate chosen programming language	3	3	4	36	10	Normal	Can be Delayed
CFM3	Errors in human factor engineering during coding and testing	5	5	4	100	6	Semi-Critical	Essential
CFM4	Difficulties in reviewing codes	5	6	3	90	7	Semi-Critical	Essential
CFM5	No provision for reusability	5	6	4	120	5	Semi-Critical	Essential
CFM6	**Misleading and incorrect documentation about the reused component**	**6**	**6**	**5**	**180**	**1**	**Critical**	**Required**
CFM7	Poor programming practice such as: too many repetitive code, redundant functions, etc.	2	3	4	24	12	Normal	Can be Delayed
CFM8	Lack of coordinated team work amongst programmers	3	4	3	36	8	Normal	Can be Delayed
CFM9	Technology Change	5	5	3	75	8	Normal	Can be Delayed
CFM10	Inconsistent and complex code	3	4	4	48	9	Normal	Can be Delayed
CFM11	Ambiguous and difficulty in code comprehension	3	3	3	27	11	Normal	Can be Delayed
CFM12	Intuitive and or informal testing processes	5	6	5	150	2	Critical	Required
CFM13	Poor test case documentation	6	6	4	144	3	Semi-Critical	Essential

ID = Identification Number S = Severity Rating (1 - 10) O = Occurrence Rating (1 – 10)
D =Detection Rating (1 – 10) RPN = Risk Priority Number CP = Control / Preventive Strategies
P = Prioritization

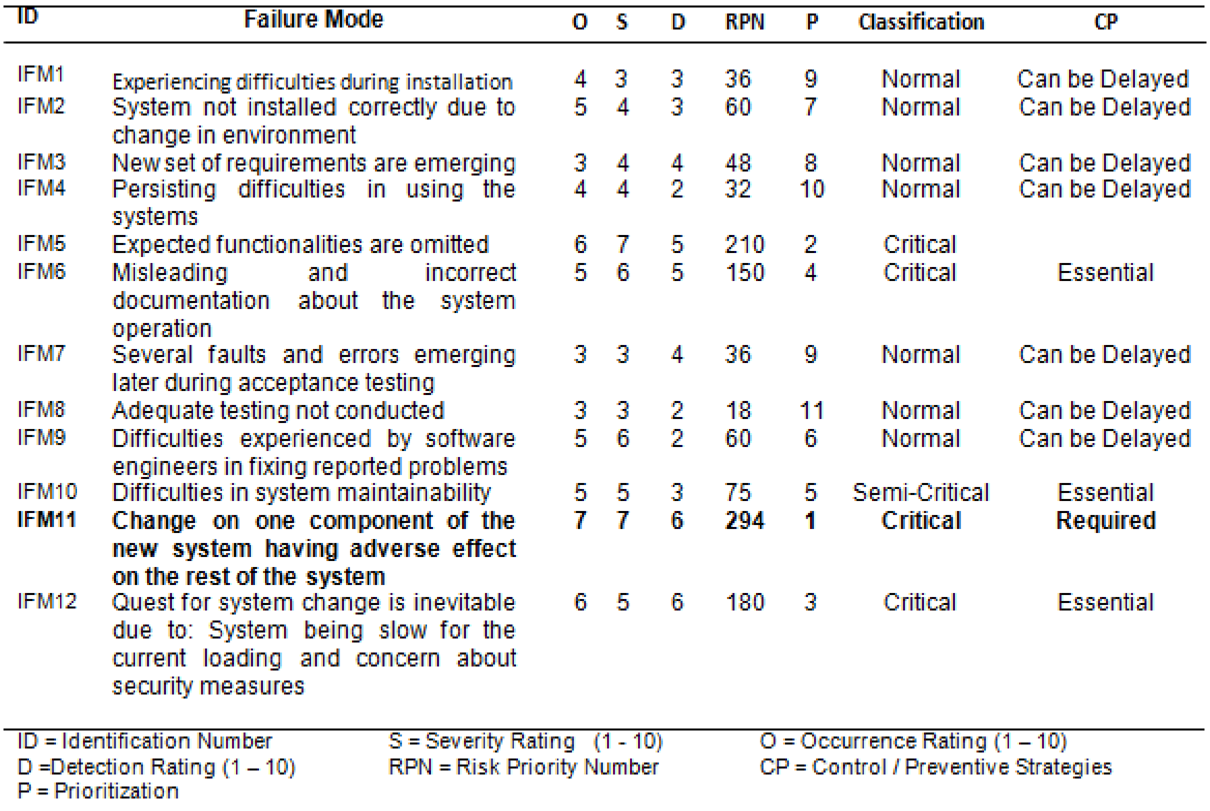
Result presented in Table 5.20 shows that two potential failure mode with ID: CFM6 (RPN = 180) and CFM12 are classified as *critical risk* in the stage. The failures CFM6 and CFM12 are prioritized 1 and 2 respectively. This indicates that the failures CFM6 and CFM12 require more developmental resources and more attention than other potential failure modes in the same stage. Thus, control action is required to mitigate their effect on project outcome in the installation and coding stage of the software development process.

Also, potential failure modes from the Table 4.20 that are classified semi-critical are failures with ID: CFM13 (RPN = 144), CFM1 (RPN = 125), CFM5 (RPN = 120) and CFM4 (RPN = 90). They are prioritized 3, 4 and 5 respectively, indicating that their control action is essential. In addition, potential failure mode with ID: CFM7 has the least risk impact on project outcome as shown in the table with the RPN = 24.

5.8.5 Analysing the risk factors associated with potential failure modes in the installation and maintenance stage

Table 5.21 presents the result of data analysis of FMEA worksheet document for the FMEA process conducted in installation and maintenance stage.

Table 5 21: FMEA worksheet documenting the assessment of risk factors associated with potential failure modes in the installation and maintenance stage



ID	Failure Mode	O	S	D	RPN	P	Classification	CP
IFM1	Experiencing difficulties during installation	4	3	3	36	9	Normal	Can be Delayed
IFM2	System not installed correctly due to change in environment	5	4	3	60	7	Normal	Can be Delayed
IFM3	New set of requirements are emerging	3	4	4	48	8	Normal	Can be Delayed
IFM4	Persisting difficulties in using the systems	4	4	2	32	10	Normal	Can be Delayed
IFM5	Expected functionalities are omitted	6	7	5	210	2	Critical	
IFM6	Misleading and incorrect documentation about the system operation	5	6	5	150	4	Critical	Essential
IFM7	Several faults and errors emerging later during acceptance testing	3	3	4	36	9	Normal	Can be Delayed
IFM8	Adequate testing not conducted	3	3	2	18	11	Normal	Can be Delayed
IFM9	Difficulties experienced by software engineers in fixing reported problems	5	6	2	60	6	Normal	Can be Delayed
IFM10	Difficulties in system maintainability	5	5	3	75	5	Semi-Critical	Essential
IFM11	**Change on one component of the new system having adverse effect on the rest of the system**	**7**	**7**	**6**	**294**	**1**	**Critical**	**Required**
IFM12	Quest for system change is inevitable due to: System being slow for the current loading and concern about security measures	6	5	6	180	3	Critical	Essential

ID = Identification Number	S = Severity Rating (1 - 10)	O = Occurrence Rating (1 – 10)
D =Detection Rating (1 – 10)	RPN = Risk Priority Number	CP = Control / Preventive Strategies
P = Prioritization		

Result of FMEA worksheet data analysis presented in the Table 5.21 shows that four potential failure modes with ID: IFM11 (RPN = 294), IFM5 (RPN = 210), IFM6 (RPN = 150) and IFM6 (RPN = 150) are classified *critical risks;* hence, *a control action is required* to mitigate its effect on project outcome. The degrees of their criticality are in sequence. They are prioritized 1, 2, 3 and 4 respectively in the stage. This indicates that the four potential failure modes sequentially require more developmental resources and more attention than other potential failure modes in the same stage.

The result also indicates that only one potential failures with ID: IFM10 (RPN = 75) is prioritized 5 and classified as *semi-critical risk* in the stage. Thus, its control action is essential. However, the failures with ID: IFM9 (RPN = 60), IFM2 (RPN = 60), IFM3 (RPN = 48), IFM1 (RPN = 36) and IFM4 (RPN = 32) are prioritized 6, 7, 8, 9 and 10 respectively, and are classified *Normal risks* in the stage. Thus, control action to mitigate their associated risks effect can be delayed. In addition, minimum developmental resources are expected to be expended to mitigate their associated risks.

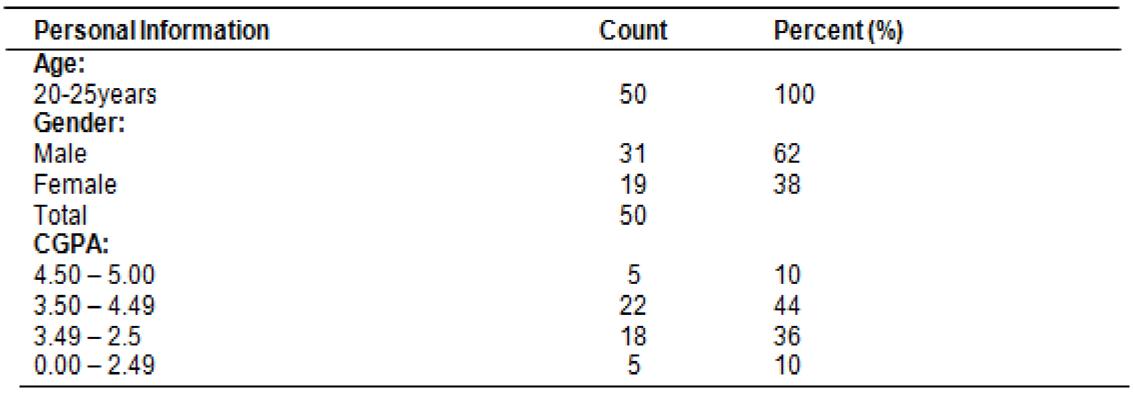
5.9 Analysing the FMEAEQ data obtained from Field Study 5

The FMEAEQ is a questionnaire-based evaluation instrument that was used to survey the opinion of undergraduate software engineering students on the ease of use (usability) and the usefulness (applicability) of the FMEA method for the conduct of software project risk management. The FMEAEQ is a page document that required undergraduate SE engineering students to self-evaluate how certain gains from the FMEA interactive study in field study 5 helped them in their software project risk management coursework. The FMEAEQ data that describes students' profile and students' perceived ease of use and perceived usefulness of the FMEA will be analysed. The objective of the FMEPOQ data analysis is to determine the students' percentage (%) satisfaction derived from the FMEA method application and its ease of use on their practical project coursework of Field study 5.

5.9.1 Analysing participating students' profile

Table 5.22 presents the profile of the participating undergraduate SE students that took part in the FMEA interactive study in Field study 5.

Table 5 22: SE students' profile



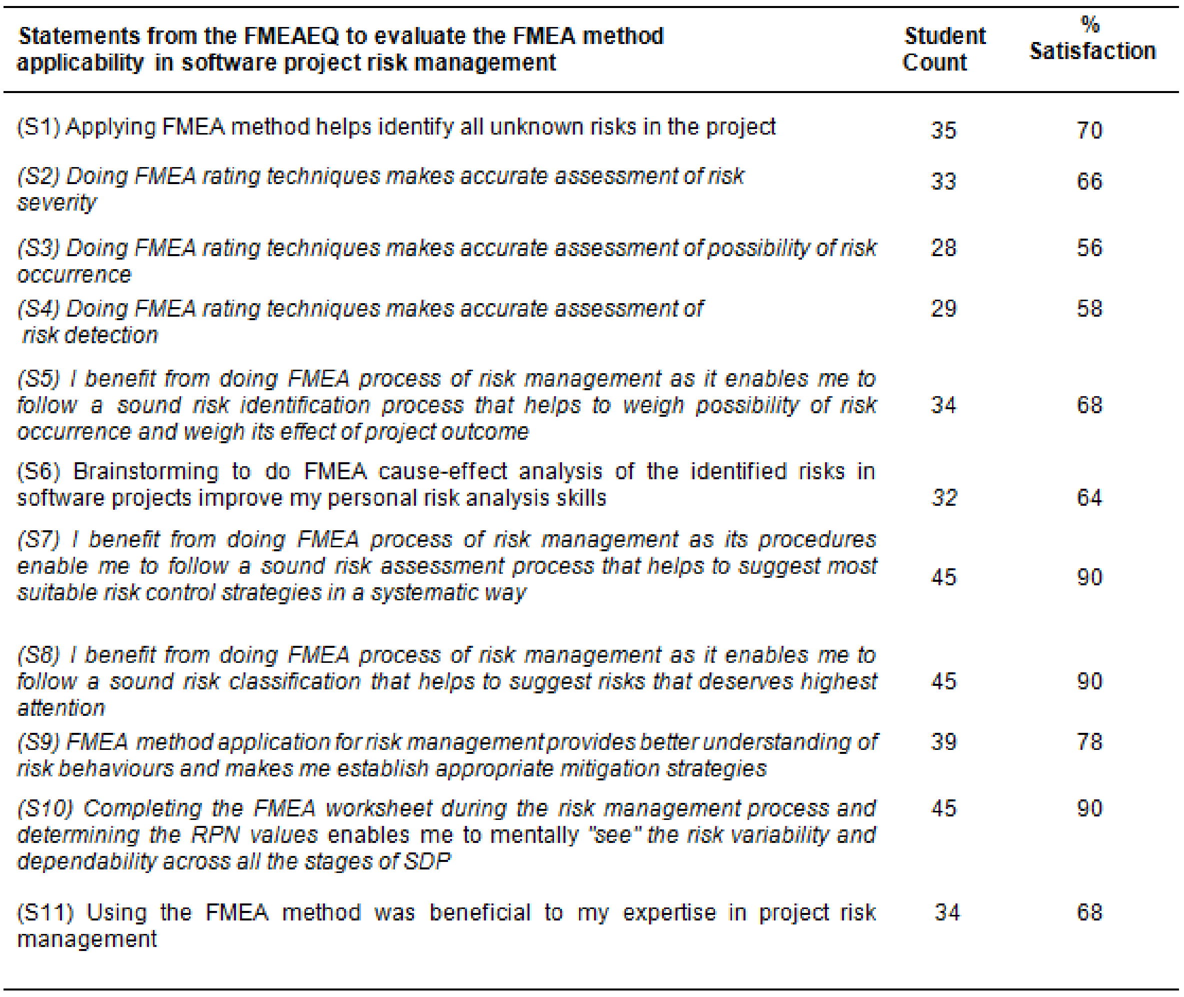
Personal Information	Count	Percent (%)
Age:		
20-25years	50	100
Gender:		
Male	31	62
Female	19	38
Total	50	
CGPA:		
4.50 – 5.00	5	10
3.50 – 4.49	22	44
3.49 – 2.5	18	36
0.00 – 2.49	5	10

Result analysis of data presented in Table 5.22 shows that 31 (62%) of the participating students that completed and returned the FMEAEQ were male and 19 (38%) were female. In summary, male student respondents dominated female student respondents. All the participating students are from the age category of 20 – 25. The table also shows that 5(10%) of the participating students were doing very well in their academics with CGPA of 4.50 – 5.00, 22 (44%) had CGPA of 3.50 – 4.99, 18 (36%) had CGPA of 2.5 – 3.49, and 5 (10%) fell within CGPA of 0.00 – 2.49.

5.9.2 Analysing students' self-evaluation of FMEA method usefulness for the conduct of software project risk management process

Table 5.23 shows the results of the FMEAEQ data on students' self-evaluations (% satisfaction) of FMEA usefulness in their project risk management practical coursework.

Table 5 23: Results of students' self-evaluation of FMEA method usefulness for their software project risk management coursework



Statements from the FMEAEQ to evaluate the FMEA method applicability in software project risk management	Student Count	% Satisfaction
(S1) Applying FMEA method helps identify all unknown risks in the project	35	70
(S2) Doing FMEA rating techniques makes accurate assessment of risk severity	33	66
(S3) Doing FMEA rating techniques makes accurate assessment of possibility of risk occurrence	28	56
(S4) Doing FMEA rating techniques makes accurate assessment of risk detection	29	58
(S5) I benefit from doing FMEA process of risk management as it enables me to follow a sound risk identification process that helps to weigh possibility of risk occurrence and weigh its effect of project outcome	34	68
(S6) Brainstorming to do FMEA cause-effect analysis of the identified risks in software projects improve my personal risk analysis skills	32	64
(S7) I benefit from doing FMEA process of risk management as its procedures enable me to follow a sound risk assessment process that helps to suggest most suitable risk control strategies in a systematic way	45	90
(S8) I benefit from doing FMEA process of risk management as it enables me to follow a sound risk classification that helps to suggest risks that deserves highest attention	45	90
(S9) FMEA method application for risk management provides better understanding of risk behaviours and makes me establish appropriate mitigation strategies	39	78
(S10) Completing the FMEA worksheet during the risk management process and determining the RPN values enables me to mentally "see" the risk variability and dependability across all the stages of SDP	45	90
(S11) Using the FMEA method was beneficial to my expertise in project risk management	34	68

Result analysis of data presented in the Table 5.23 shows that students are most satisfied with statements S7, S8 and S10. Thus, applying FMEA method for the conduct of software project risk management enables prospective software managers

(SE students) to: (i) *follow a sound risk assessment process that helps to suggest most suitable risk control strategies in a systematic way; (ii) follow a sound risk classification strategy that helps to suggest risks that deserves highest attention; and (iii) complete the FMEA worksheet during the risk management process and determine the RPN values, which* enables students to mentally *"see" the risk variability and dependability across all the stages of the software development process*.

In all, the level of satisfaction, which students derived from applying FMEA method for the conduct of software project risk management in their software risk management practical coursework as shown in the Table 5.23 shows that the FMEA method has a positive impact on their understanding of basic FMEA procedures, its method implementation and the FMEA worksheet documentation. In addition, students' comments that were gathered from the questionnaires suggested that though the FMEA method application in software project risk management is time consuming, it has helped students tremendously to follow a sound risk management process which helps identify and assess software project risks in a systematic way, and that the effort is worthwhile.

5.9.3 Analysing students' self-evaluation of ease of use of the FMEA method for the conduct of software project risk management process

Table 5.24 shows the results of the FMEAEQ data on students' self-evaluations (% satisfaction) of FMEA ease of use in their project risk management practical coursework.

Table 5 24: Results of students' self-evaluation of ease of use of the FMEA method for their software project risk management coursework

Statements from the FMEAEQ to evaluate the FMEA method applicability in software project risk management	Student Count	% Satisfaction
(S1) The FMEA standard guidelines are straight forward and efficiently enables me to identify, analyse, prioritize and control risks in software projects	46	92
(S2) Using the FMEA method was beneficial to my risk management skills in SE	27	54
(S3) The FMEA method improved my ability to conduct effective software project risk management	38	76
(S4) Using the FMEA method was beneficial to my risk management skills in SE	29	58
(S5) I think my understanding of the key FMEA standard procedures has improved because of the adherence analogy with other risk assessment models used in the lectures and training	34	68
(S6) FMEA method aids effective risk management process as it is faster and subjective practices via documentation of activities	40	80
(S7) It is easy to determine risk magnitude using the modified probability theory: RPN = Occurrence *Severity*Detection	46	92
(S8) I benefit from doing FMEA process of risk management as it provides easy guidelines that helps to suggest most suitable risk control strategies in a systematic way	46	92
(S9) The FMEA method aids my understanding of how software project risk can be prevented and controlled in software projects	44	88
(S10) The FMEA method is time consuming but clearly supports my understanding and practice	29	58
(S11) The FMEA method aids my understanding of how risk can be anticipated before it occurs and have it controlled after occurrence in software projects	31	62

Result analysis of data presented in the Table 5.24 shows that students are most satisfied with statements S1, S7 and S10. Thus, the usability of FMEA method for the conduct of software project risk management enables students to: (i) *follow efficient and straight forward FMEA guidelines*, which enable them to identify, analyse, prioritize and control risks in software projects;*(ii) easily determine the risk magnitude using the modified probability theory: RPN = Occurrence**Severity*Detection; and *(iii) use simple unambiguous guidelines that helps to suggest most suitable risk control strategies in a systematic way.*

Lastly, the results presented in the Table 5.24 indicate that students' perceived ease of the FMEA method for the conduct of project risk management was impressive. On the average, the percentage (%) satisfaction derived from FMEA usability for the conduct of risk management as in the Table 5.23 is 74.6%. Thus, students appreciate the ease of use of FMEA method for the conduct of risk management in software projects.

5.10 Hypothesis testing

This section describes the findings of comparative procedures and analysis that was conducted using the Analysis of Variance (ANOVA) testing techniques to test the research hypothesis. The ANOVA is selected for the comparative analysis because it is a test of hypothesis that is appropriate to compare means of a continuous variable in two or more independent comparison groups. The ANOVA implementation in this study systematically examines the variability within groups being compared and also examines the variability among the groups being compared. The ANOVA statistics helps the researcher understand how the variables in the formal research questions link together, with a null hypothesis for the test that the means of the different variables are equal. If there is a statistically significant result, then it means that there is no significant relationship amongst the compared variables

5.10.1 Effect of potential failure modes in the software development process on software project outcome (cost, time & scope/quality)

The following hypotheses are postulated to compare the mean scores' effect of potential failure modes in all the studied stages of software development process on software project outcome (cost, time & scope/quality):

H_{01}: There is no significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality).

H_1: There is significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality).

5.10.1.1 Effect of potential failure modes in the software development process on project cost

Table 5.25 shows the result of the ANOVA test procedures comparing the mean scores of the potential failure modes' effect in the studied stages of the software development process on the project cost.

Table 5.25: ANOVA procedure to compare the mean scores of potential failure effect in the studied stages of the software development process on project cost

Source of Variation	Sums of Squares (SS)	Degree of Freedom (DF)	Means of Squares (MS)	F
Between Treatments	3.8	4	0.95	
Error (or Residuals)	13	52	0.25	3.8
Total	16.8	56		

Critical Value = 2.56 from F Table at α=0.05

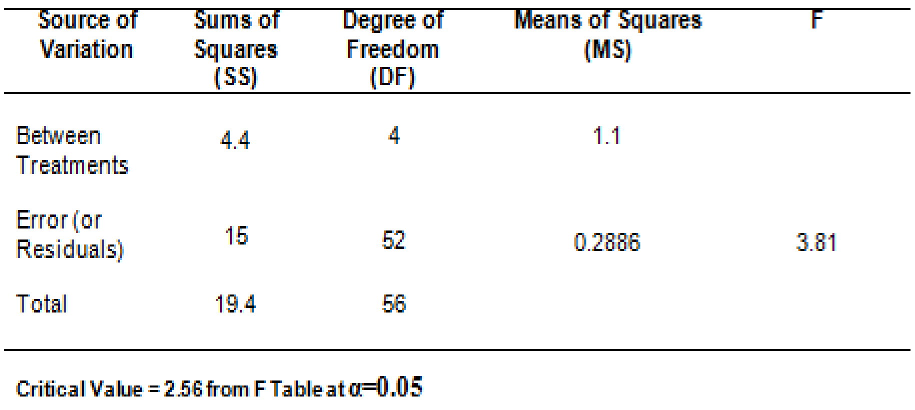
The F value was computed to statistical determine if there is a significant variation between the mean scores of the comparing variable groups. The critical value is derived from the F Table at α=0.05 using the computed F value. If the F value at α=0.05 is less than the critical value, then the result reveal no statistically difference evidence between the mean scores of the compared variable groups. Else, if the F value is greater than the critical value, there is a significant variation between the mean scores of the comparing groups.

The analysis of data in the Table 5.25 yielded an F value (F = 3.8) and the critical value (critical value = 2.56). Thus, $H0_1$ on cost is rejected and alternative hypothesis H_1 accepted because F > critical value.. This result reveals statistically significant evidence at α=0.05 to show that there is statistically significant difference between the mean scores of the potential failure modes' effect in all the studied stages of the software development process on the project cost. In addition, the result indicates that M_{cost} ratings of the potential failure modes in each of the studied stages of the software development process are above 3.0, and are perceived by software development practitioners to pose an unequal degree of threat to project cost across the stages of software development process.

5.10.1.2 Effect of potential failure modes in the software development process on project duration

Table 5.26 shows the result of the ANOVA test procedures comparing the mean scores of the potential failure modes' effect in the studied stages of the software development process on the project duration.

Table 5.26: ANOVA procedure to compare the mean scores of potential failure effect in the studied stages of the software development process on project duration



Source of Variation	Sums of Squares (SS)	Degree of Freedom (DF)	Means of Squares (MS)	F
Between Treatments	4.4	4	1.1	
Error (or Residuals)	15	52	0.2886	3.81
Total	19.4	56		

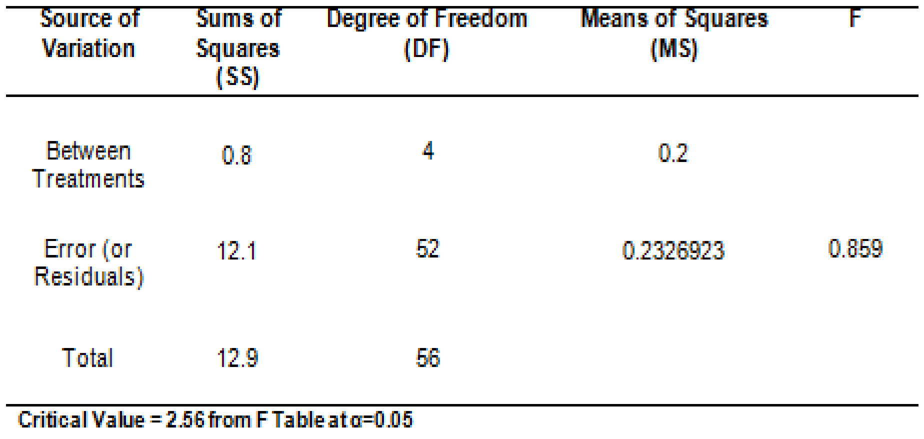
Critical Value = 2.56 from F Table at α=0.05

The analysis of data in the Table 5.26 yielded an F value (F = 3.81) and the critical value of (critical value = 2.56). Thus, H_{01} on time is rejected and alternative hypothesis H_1 accepted because the F > critical value. This result reveals a statistically significant statistically significant difference between the mean scores of the potential failure modes' effect in all the studied stages of the software development process on the project duration at $\alpha = 0.05$. In addition, the result indicates that the mean scores on project schedule (M_{time} rating) of the potential failure modes in each of the studied stages of the software development process are above 3.0 and are perceived by software development practitioners to be severe but pose unequal degree of threat across the studied stages of the software development process to software project duration.

5.10.1.3 Effect of potential failure modes in the software development process on project scope

Table 5.27shows the result of the ANOVA test procedures comparing the mean scores of the potential failure modes' effect in the studied stages of the software development process on the project scope.

Table 5.27: ANOVA procedure to compare the mean scores of potential failure effect in the studied stages of the software development process on project scope



Source of Variation	Sums of Squares (SS)	Degree of Freedom (DF)	Means of Squares (MS)	F
Between Treatments	0.8	4	0.2	
Error (or Residuals)	12.1	52	0.2326923	0.859
Total	12.9	56		

Critical Value = 2.56 from F Table at α=0.05

The analysis of data in the Table 5.27yielded an F value of (F = 0.86) and the critical value of (critical value = 2.56). Thus, H_{01} on scope is accepted and alternative hypothesis H_1 rejected because the F < critical value. This result reveals a statistically significant evidence at $\alpha = 0.05$ to show that there is no statistically significant difference between the mean scores of the potential failure modes' effect in all the studied stages of the software development process on the project duration. In addition, the result indicates that the mean scores on project scope (M_{scope} rating) of the potential failure modes in each of the studied stages of the software development process is above 3.0, hence, the effects are perceived by software development practitioners to be high and pose equal degree of threat to the quality of software projects across the stages of software development process.

5.10.2 FMEA adherence with other popular project risk management models

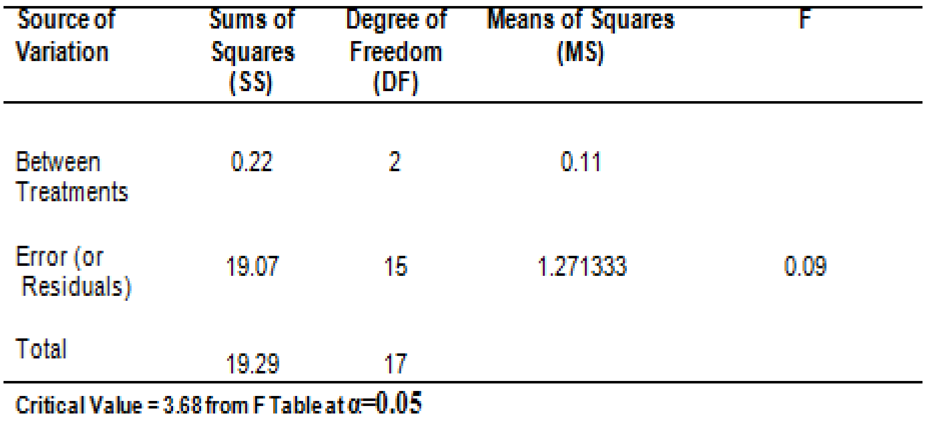
The following hypotheses are postulated to compare the FMEA Adherence to other popular project risk management frameworks:

H_{02}: There is no significant difference between the mean scores of FMEA adherence to the other popular project risk management models.

H_2: There is significant difference between the mean scores of FMEA adherence to the other popular project risk management models.

Table 5.28 shows the result of the ANOVA test procedures comparing the mean scores of the FMEA Adherence to the other popular project risk management models.

Table 5 2825: ANOVA test procedure to compare the mean scores of FMEA adherence to the other popular project risk management models



Source of Variation	Sums of Squares (SS)	Degree of Freedom (DF)	Means of Squares (MS)	F
Between Treatments	0.22	2	0.11	
Error (or Residuals)	19.07	15	1.271333	0.09
Total	19.29	17		

Critical Value = 3.68 from F Table at $\alpha=0.05$

The analysis of data in the Table 5.28 yielded an F value of 0.09 and the critical value of 3.68. The F value was computed to statistical determine if there is a significant variation between the mean scores of the comparing groups. If the F value is less than the critical value, then the result reveal no significant difference between the mean scores of the compared variable groups and vice versa.

Analysis of data in the Table 5.28 showed that the F value is less than the critical value. Thus, H_{02} is accepted and alternative hypothesis H_2 rejected because $0.09 < 3.68$. This result reveals a statistically significant evidence from F Table at $\alpha=0.05$ to show that there is no statistically significant difference between the mean scores of the FMEA adherence to the three studied popular project risk management frameworks (PMBOK, SEI and Boehm). This result reveals that FMEA model conforms to the standard principles and practices for managing the risk in software development projects.

5.10.3 Efficiency of the FMEA method for the conduct of software project risk management

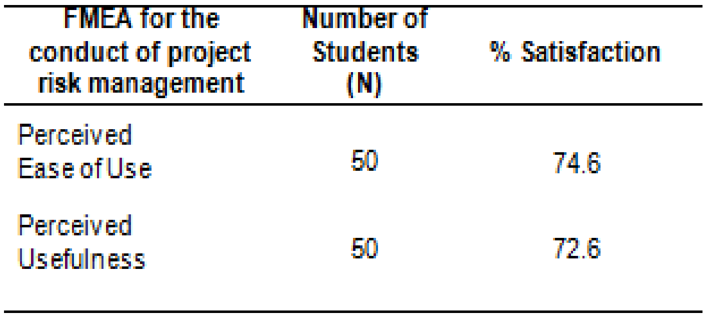
As indicated in Figure 3.5 (conceptual model for assessing risk factors in software projects using FMEA), the efficiency of using the FMEA for risk factor assessment in software projects is a measure of both the *perceived ease of use* and the *perceived usefulness* of the FMEA for the task of risk management. The following hypotheses are postulated to determine the level of efficiency of the FMEA method for the conduct of software project risk management:

H_{03}: FMEA method is not efficient for the conduct of software project risk management.

H_3: FMEA method is efficient for the conduct of software project risk management.

Table 5.29 shows the summarized results of the FMEAEQ data on percentage (%) satisfaction on the FMEA's ease of use and usefulness as perceived by undergraduate software engineering students in their project risk management practical coursework. The decision rule in this case is that a percentage (%) satisfaction that is not greater than (i.e., $\geq$) 50% is significant. Thus, if % satisfaction $\geq$ 50%, H_{03}will be rejected and the alternative hypothesis H_3 will be accepted.

Table 5 26: % satisfaction on the FMEA's ease of use and usefulness as perceived by undergraduate software engineering students in their project risk management practical coursework



FMEA for the conduct of project risk management	Number of Students (N)	% Satisfaction
Perceived Ease of Use	50	74.6
Perceived Usefulness	50	72.6

A cursory look at the results presented in the Table 5.29 indicated that the % satisfaction derived from FMEA ease of use for the conduct of risk management as perceived by student managers is 74.6%. Thus, the perceived ease of use is significant. This result shows that students' behavioural use as a measure of perceived ease of use of FMEA method for the conduct of risk management in software projects is convenient.

Also, the analysis of percentage (%) satisfaction of the perceived usefulness in the Table 4.29 yielded an average of 72.6%. Thus, the satisfaction derived from FMEA usefulness for the conduct of risk management as perceived by student managers is convincingly significant.

Based on the level of satisfaction, which students derived from applying FMEA method for the conduct of software project risk management in their software risk management practical coursework as shown in the Table 5.29, there is a statistically significant evidence to show percentage (%) satisfaction for both perceived ease of use and usefulness is $\geq$ 50%. Thus, Ho_5 is rejected and the alternative hypothesis H_5 is accepted.

5.11 Chapter summary

This chapter presented the rigorous processes adopted for analysing the qualitative and the quantitative data generated from all the field studies conducted in this research. The focus was on assessing the risk factors associated with potential failure modes in all the studied stages of the software development process. The Miles and Huberman (1994) method of analysing qualitative data was used to analyse the qualitative data collected in the research while the quantitative data were analysed using descriptive statistics and the FMEA worksheet. The analyses of data generated from the five field studies conducted were organized by first presenting analysis of the qualitative data before presenting the quantitative ones.

Two different themes emerged from the analysis of the qualitative data through explorative studies, post-mortem, and focus group discussions. The analysis was based on the research questions outlined in Section 1.3. The first theme was concerned with the possibility of applying the FMEA for the conduct of risk assessment in software development projects, where findings revealed that software practitioners opined that FMEA has the potential for risk management in software projects. The second theme was about the identification and assessment of probable failure modes in the 5 studied stages of software development process. Participants suggested the probable failure modes that can affect project outcome, their causes, and their associated risks effect in the 5 studied stages of the software development process.

Four themes emerged from the analysis of the quantitative data through series of survey studies using questionnaires (FMEAAQ, FMEOPOQ, PFMEOPOQ and the FMEAEQ) and FMEA worksheet as research instruments. The analysis was based on the research hypothesis outlined in Section 1.4. Theme 1 presented participants' views about the FMEA model conformity with other popular project risk management models (PMBOK, SEI and Boehm), where findings showed a high level of FMEA adherence to standard project risk management frameworks. Theme 2 was about assessing the effect of risk associated with potential failure modes in software development process on the project outcome (cost, time and scope). The third theme was about using the FMEA method for the conduct of software project risk management, where findings revealed the individual risk's severity, priority and criticality. The fourth theme was concerned with the assessment of students' behavioural usage of FMEA for the conduct of software project risk management in their software engineering course, where findings revealed a significant percentage (%) satisfaction for conducting software project risk management using the FMEA method as perceived by student developers.

The next chapter discusses the key findings and conclusions drawn from this research including expected academic, policy and practical contributions of the study to scholarship.

CHAPTER SIX

Conclusion and Recommendations

This chapter presents the key findings of this research and summarizes its contributions to the existing body of knowledge. The research overview is presented in section 6.0. Sections 6.1 and 6.2 present the summary of the conclusion and contributions of the research findings respectively. In section 6.3, the limitations and assumptions of the study are presented while section 6.4 is the presentation of the scientific, methodological and personal reflections on the study. Section 6.5 discusses the recommendations for future academic research and the concluding remarks about the study follows suit in section 6.6, while section 6.7 summarizes the chapter.

6.0 Research overview

The main objective of this research is to assess the risk factors associated with potential failure modes in the software development process using the FMEA method. Risk assessment, according to Boehm (1991), involves four major activities: Risk identification, Risk analysis, Risk prioritization and Risk classification. To achieve the research main objective, this research is conducted to cover the four activities of risk assessment using the FMEA method.

The research problem was informed by lack of sufficient information on the risk factors associated with potential failure modes in the software development process to support managers in determining the best risk control mechanism to be deployed to mitigate the risk of this dimension. Also, lack of evidence in the literature on the application of safety and reliability engineering tools (such as: FMEA) for the conduct of software project risk management motivated this study. Reflecting the problem and the rationale for the study, the main research question for the study was formulated as: ‘What are the effects of risk factors associated with potential failure modes in the software development process on the software project outcome? (cf. section 1.3).

The following sub-research questions were defined in support of the investigations of the main research question (cf. section 1.3).

1. What potential failure modes in the software development process do software development experts consider to be risky to software project outcome? (Chapters 4, 5)
2. What factors are considered by software development experts to be the likely causes of the identified potential failure modes? (Chapters 4, 5)
3. What failure symptom(s) can be used to detect likely occurrence of the potential failure mode in the software development process? (Chapters 4, 5)
4. What multiplier effects do risks associated with potential failure modes in the process of software development pose on the project outcome (cost, time & scope)? (Chapters 4, 5)
5. To what extent is the FMEA feasible and practicable for the conduct of software project risk management? (Chapters 4, 5)
6. What risk response strategies are considered appropriate actions for managers to address the threats posed by the identified potential failure modes in the software development process? (Chapters 4, 5)

This chapter is the concluding chapter of the thesis and it presents an overview of the entire study, reflection on the findings, recommendations, and limitations of the study. The chapter also provides suggestions for further studies that can be used as a foundation by future researchers to advance research in the field. Finally, conclusions are drawn from the findings.

During the execution of this research, five field studies were carried out sequentially to address the identified problems and to provide answers to the research questions using both qualitative and quantitative research methods. The following section is the presentation of the summary of the key findings and the contributions to existing body of knowledge.

6.1 Summary of the research findings and conclusions

The findings and conclusions of this research are summarized based on: (i) the research questions, and (ii) the research hypotheses.

6.1.1 Research findings and discussions based on the research questions

Research question 1:

<u>What potential failure modes in the process of software development do software development experts consider to be risky to software project outcome?</u>

Literature review revealed different research contributions that focused mainly on risk factors associative dimensions in software projects to help improve existing risk management practices (e.g., Menezes et al. 2018; Vahidnia et al. 2016; Hijazi et al. 2014; Wallace et al. 2004; Boban et al. 2003, Boban et al. 2003, Schmidt et al. 2001; Houston 2000; Cule et al. 1998). After evaluating these efforts and based on the suggestion of Barki et al. (1993), a new risk factor associative dimension was proposed by extending the existing risk factor dimension to include risk factors associated with potential failure modes in the software development process. The need to redirect software project managers' attention to other pertinent risk dimensions that can influence software project outcome in the software development projects was accepted by the participating software practitioners during a focus group discussion in Field study 2 of this research.

The first step in the risk assessment process is to identify and understand the risk factor itself, so that appropriate control measures can be implemented. Thus, identifying the risky potential failure modes in the software development process as the first task in this study aligns with the actual practice of assessing risk factors in software development projects as reported by Boehm (2004; 1991) and as agreed by the participating software development practitioners during the workshop and focus group discussion. The main challenge in this task however, is that there is no evidence of authoritative list of risk factors available in the literature to assist project managers and other software practitioners to comprehend the nature of risks that are typically associated with potential failure modes in the software development process.

Thus, this risk identification task was conducted towards addressing this problem by adopting combined risk identification methods: brainstorming and focus group discussion to authoritatively identify risky potential failure modes in the software development process that can influence the software project outcome. The idea of applying the FMEA that adopts brainstorming and focus group discussion as risk identification method was collectively agreed by the participating discussants. Moreover, the *FMEAAQ* questionnaire results show that the FMEA procedural requirements adhered perfectly to the formal project risk management frameworks. This reinforces the decisions of the discussant to use the FMEA method as a tool for the risk assessment. The focus group discussion method was designed to elicit and organise the opinions of the participating software development practitioners on the subject matter during a focus group discussion in Field studies 1, 2, and 3 as discussed in chapters 4 and 5.

The diffusion of innovation theory was applied using its communication channels to spread the new idea of risk factors of this dimension (i.e., the risk factors associated with potential failure modes in the software development process) amongst the participating software experts. A detailed conceptual framework employed during the risk identification and analysis was presented in chapter 3 (section 3.2). The outcome of this task involving the identification of risk type, the risk causal factor(s) and their method of detection is presented in Appendix 3A.

Specifically, in addressing RQ1, the result analysis of the Field study 2 revealed a more than 50 probable failure modes in the process of software development that are perceived by software experts as potential threat to the project outcome as presented in Appendix 3A. The results also indicated that the proportion of threat posed on project outcome by individual failure item varies, interdependent to each other and are unevenly distributed across the development stages with early stages having higher numbers of failure modes that are potentially risky to the project outcome than the later stages of the SDLC.

Furthermore, the results of the *FMEAOPOQ* questionnaire used in a validation case study in Field study 3 (see Tables 4.12 to 4.16) confirm all the identified probable failure modes in the Field study 2 to be potentially risky to the project outcome. The results also revealed that the failure modes: *Poor risk management strategy,* *Inaccurate requirement, Misleading and incorrect documentation about the reused component* are considered of having potential of *posing* the highest threat to the project outcome. This finding agrees with the conclusion of Wellington (2018) and Castsoftware (2015) that singled out effective risk management processes and accurate requirement specification as key to project success.

Further interactions with the participating software experts on the implications of these findings reveal that software experts considered that the occurrence of the identified potential failure modes in the software development process, if not appropriately controlled, will cause software projects to have budget overruns, or be developed behind schedule and unable to deliver features originally specified. These findings are an extension of the existing risk factor dimensions in software projects (e.g., Menezes et al. 2018; Vahidnia et al. 2016; Hijazi et al. 2014; Wallace et al. 2004), thus, having a significant implication on the current practice of software project risk management.

An in-depth study of the current practice of software project risk management shows that project managers and their risk management teams rely more on their intuitive ability to identify a comprehensive set of risk factors, creating mitigation and/or contingency plans to manage these risks. Other reports from literature confirmed that most project managers lack the knowledge and understanding of risks that are associated with the manner and description, including both the pre and post conditions in which failure occurs in the software development process. Consequently, their intuitiveness on software risk management typically focuses on factors that contribute to the likelihood of project failures rather than on the magnitude of loss should failure occur. These practices however tend to ignore other important associative risk dimensions that can compromise the success of software development projects. The consequence of this problem is that managers will be characterised with ignorance of the possible threats of failures on the software project outcome. This in turn, misguides project managers and other practitioners to make risky decisions that can impact negatively on the software project outcome.

Providing information on failure modes that are potentially risky to software project outcome will empower project managers with a reusable, intuitive boost to be able to probe risk levels and to evaluate the priorities of risks. This helps the risk managers and their team to determine remedial actions to minimize the risk of the failure. It will also improve their managerial attitude through sound knowledge of the risks as per their dominant risk outcomes and evaluate their associated impacts quantitatively and continuously in all the stages of the project lifecycle.

This study contributes to theory and practice in SE by bringing to light another important risk factor dimension that will help to improve the existing risk management practices in software projects as suggested by Barki et al. (1993). The study also provides useful opportunities for future research in SE and practical purposes. A knowledge contribution of this study was the identification of the risky potential failure modes in the software development process, which reinforced the findings of Abubakar and Lawal (2020) and Georgieva (2010) as discussed in chapters 4 and 5 and presented in the Appendix 3A.

Research question 2:

<u>What factors are considered by software development experts to be the likely causes of the identified potential failure modes?</u>

The outcome of the focus group study sessions in the Field study 2 as discussed in section 5.3, produced mainly qualitative information that describes the theme of the study: the probable failure modes in the software development process; their likely causes and the failure detection methods. The result analysis of the focus group study is presented in Appendix 3A. Thus, the third column with the heading: '*Likely Causes'* in the table presented in the Appendix 3A provides multiple likely causative factors of the identified potential failure modes in all the studied stages of the software development process as considered by the participating software practitioners and thus, answered the research question 2.

Analysis of the findings showed that human engineering factor as well as other factors relating to the development process, hardware and software resources were the dominant factors that are likely responsible for the failure occurrences in the software development process as presented in the third column *'Likely causes'* in Appendix 3A. This discovery agrees with the findings of Georgieva (2007). It was also discovered that causality relationship and inter-dependency exist amongst the potential failure modes across the studied stages, which also aligns with the findings of Hijazi et al. (2014b). In addition to this, the study also discovered that some of the identified potential failure modes have multiple causative factors. As an example, the potential failure mode: *Inconsistent and complex code* can result from multiple causative factors such as: Human engineering and development process factors: when best programming practices are not followed and or due to inadequate programming skills by programmers (see failure mode in SN 10 under Implementation and Coding stage in Appendix 3A. This finding corroborates the findings of previous research on risk factors in software development projects (e.g., Menezes et al. 2018; Wallace et al. 2004 & Schmidt et al. 2001).

Findings from previous research on risk factors in software development projects (e.g., Menezes et al. 2018; Hijazi et al. 2014; Wallace et al. 2004 & Schmidt et al. 2001) revealed that causative agents of the identified risk factors were not given the needed attention and their effects on the project outcome were not reported in literature. Also, while the identification and assessment of risk factors at various dimensions, which most previous research provided, empower managers to probe risk levels, previous research did not sufficiently account for what factor combination and ordered scenarios culminated in the failures.

However, lack of information about the relative importance of risk factors associated with the potential failure modes and the ordered scenarios leading to the failure occurrences in the software development process contributes to managers' ignorance of their possible threat. This in turn, misdirects project managers and their risk management team to make undesirable decisions that have a negative impact on the software project outcome. This study is an extension of previous research on risk factors in software development projects (e.g., Menezes et al. 2018; Vahidnia et al. 2016; Hijazi et al. 2014; Wallace et al. 2004; Boban et al. 2003, Boban et al. 2003, Schmidt et al. 2001; Houston 2000; Cule et al. 1998) by including the causative agents of the identified risk factors. The study has contributed rich empirical grounding information on other important risk factors and their causative factors that will guide and assist project managers on risk factor management and decision making in software project management.

Research question 3:

<u>What failure symptom(s) can be used to detect the likely occurrence of the potential failure mode in the software development process?</u>

Every software project manager across the globe at one time or more has witnessed failures of software projects in their career. Interestingly, research reports from literatures prove that proper identification of the project risk factors and application of appropriate risk management response strategies to control the risks can effectively minimize the rate of project failures. However, interactions with focus group discussants on the implication of the techniques used by managers for failure detection revealed that the reliance on the managers' technical thinking style/approach (intuitivism) and the various testing techniques (e.g. system and user integration tests) as the primary model for addressing project risk factors is no longer sufficient to minimize the risk effects on the software project outcomes. This finding agrees with the conclusion of Castsoftware (2015) that reported that managers intuitive approach and testing technique to eliminate software project risks amount to nearly 90% of the factors that cause project failures.

Castsoftware also found that both techniques ignore less apparent issues that capture the failure symptoms to detect the failure occurrence in the process. Thus, inclusion criteria for the failure detection mechanism in the software development process to prevent project failures have not been given adequate attention in the software risk factor related research.

The application of Miles and Huberman (1994) method for the analysis of the qualitative data obtained from the focus group study sessions of Field study 2 resulted in the table listing that describes the nature of potential failure modes that exist in the software development process, as presented in Appendix 3A. The result analysis shows that the table is segmented into 5 stages of the software development process and each table consisting of 4 columns. The fourth column with the heading: '*Detection Method'* contains the failure symptom(s) in detecting the likely occurrence of the potential failure mode in all the 5 studied stages of the software development process as perceived by the sampled practitioners and thus, answers research question 3.

The findings from this study revealed that the failure symptom(s) exhibited in detecting the likely occurrence of the failure are functions of the causative factor as opined by software development experts and presented in the fourth column 'Method of detection' in Appendix 3A. It was also found that some potential failure modes can be easily detected at earlier stage while some can only be detected at later stages of the development process. Software experts also perceived that detecting failure occurrences at earlier stages in the development process can save developers from reworking tasks. This finding agrees with the opinions of Jayson, Xiaoqing and Irem (2007) and Liliana and Eila (2002). This practice will ensure that software projects are developed within the allocated budget and the specified time frame, thus confirming the findings of Mitrabinda and Durga (2011).

This study contributes to the existing knowledge by hypothesizing what factor combination and ordered scenario led to the failure in all the stages of the SDLC as well as their detection mechanisms. Providing Information on failure detection mechanism will assist risk managers to proactively set an effective combative action to control the effect of the risks on the project outcome as presented in Appendix 3A.

Research question 4:

<u>What multiplier effects do risks associated with potential failure modes in the process of software development pose on the project outcome (cost, time & scope)?</u>

The perceptions of software development practitioners on the multiplier effects of risk factors associated with the identified potential failure modes in the software development process on software project outcome (cost, time and quality) was investigated and analysed using empirical research that surveyed software experts' opinion in Nigeria (Field study 2) and complementing it with another research that surveyed software experts' opinion on the same subject from around the world (Field study 3). The multiplier effects of the potential failure modes in all the studied stages of software development process on project outcome (cost, time and scope) as perceived by software experts in Nigeria and around the world were presented in tables (5.6 – 5.10) and tables (5.12 – 5.16) respectively.

Sections 5.6 and 5.7 presented valid evaluation of software experts' perception on the multiplier effects of risk factors associated with the identified potential failure modes in the software development process on software project outcome (cost, time and scope). Analyses of the findings reveal that the multiplier effects of the identified potential failure modes in the software development process on the project outcome (cost, time and scope) vary across the studied stages of the SDLC and are classified as severe, high and moderate as shown in the tables 5.6 to 5.12.

Furthermore, using the impact category scale of Project Management Institute (PMI 2004) to estimate the level of impact of the identified potential failure mode on project outcome (cost, time & scope) as perceived by the participating software development experts, the resulting potential failure modes' multiplier effects on the project outcome (cost, time & scope) are:

- For potential failure modes with severe threat, cost increase is ≥ 7% and < 10% of project cost, In service date delayed is ≥ 7% < 10% of project duration and scope changes are unacceptable to project sponsor
- For potential failure modes with high threat, cost increase is ≥ 4% and < 7% of project cost, in-service date delayed is ≥ 7% & < 10% of project duration and scope changes are unacceptable to project sponsor
- Cost increase is < 1% of project cost, insignificant schedule slippage and quality degradation is not noticeable

In addition to this, the result analysis of the FMEA process conducted to assess the risk factors associated with potential failure modes in the software development process in Field study 4 is discussed in section 5.8.1 to section 5.8.5. The results provided a comprehensive analysis of how identified potential failure modes in the studied stages of the software development process during the previous Field studies 1, 2 and 3 were assessed (i.e., prioritized and classified), thus provided insight on the main research question. The results obtained from the FMEA process of Field study 4 (see section 5.8.1 to section 5.8.5) showed similar multiplier effects of the identified risky potential failure modes on the project outcome as perceived by the participating software experts in the survey studies of Field studies 2 and 3.(see sections 5.6 and section 5.7).

Research question 5:

<u>To what extent is the FMEA method feasible and practicable for the conduct of software project risk management?</u>

The research efforts in SE on the possibility of applying safety and reliability engineering techniques (including the FMEA) for software project risk management have been comprehensively reviewed in the literature. For instance, researchers have used the FMEA for software risk assessment and to eliminate potential failure across system development lifecycle (e.g. Khaiyum and Kumaraswamy 2014; Gupta et al. 2012; Mitrabinda & Durga Prasad 2011; Hassan et al. 2005). Others used it to conduct risk assessment at the requirement stage based on a multiple experts' knowledge (Cornford & Feather 2001). Thus, reports from the literature reveal successful outcome of application of the reliability engineering techniques (including the FMEA) in SE. However, reviewed literature showed no evidence of adherence of procedural requirements of the reliability engineering tools (including the FMEA) with formal project risk management frameworks creating major challenge to managers who are searching for reliable tool for their risk management business, thus, the RQ 4: *To what extent is FMEA method feasible and practicable for the conduct of software project risk management?*

One unique benefit of using FMEA method for conducting project risk management is that it follows simple and unambiguous steps to identify, analyse, prioritize, classify and suggest suitable response strategies to control project risks (SW-Quality 2006). This benefit was explained in an interactive study with student developers of Field study 5. The student developers acknowledged the ease of use and the usefulness of the FMEA for the conduct of risk management in software development projects. Another important benefit of the FMEA is the procedural requirement that is in strong alignment with the formal project risk management frameworks, which is evidently proven by FMEAAQ results in Field study 1.

The FMEA process design and method implementations provided in the Field studies 4 and 5 explained in details how the risk factors associated with the identified potential failure modes in the software development process can be assessed using the FMEA method. The FMEA process design is discussed in section 4.9.1 and outlined the team size and the criteria for rating scales for the 'Occurrence', 'Severity' and 'Detection' as well as FMEA worksheet documentation as presented in the Tables 4.2, to 4.5 respectively. Similarly, the FMEA method implementation that explained the prioritization and classification criteria are discussed in section 4.92.

Indeed, the application of the FMEA method for the risk assessment in software projects and benefits derived did not largely appeal to the participating discussants at the initial stage. However, it did appeal to the student developers in the interactive case studies after applying FMEA method in their SE risk management practical course. The FMEAEQ questionnaire results show that the initial attitude of some of the discussants and the student developers have positively changed after applying the FMEA method for risk assessment process during the FMEA process and interactive case studies. This justifies the feasibility and the practicality of the FMEA method for the conduct of risk assessment in software development projects is practicable and reliable. It can also be argued that the FMEA method is practicable and reliable based on the results of the focus group discussions and survey study with software practitioners (Field studies 1 & 2); the FMEA process study with software experts (Field study 4) and the FMEA interactive study with student developers (Field study 5). However, these findings show that this conclusion cannot be generalised as it portends only the perceptions of software developers. It is logical to assume that different software project stakeholders (e.g., software users) would view risk and the method used to assess its effects differently, and the differences in their evaluation would provide a ground for further investigations.

The literature reviewed on the traditional FMEA process and its application in the SE as presented in sections 2.7 and 2.8, provided the steps required for the use of the FMEA method for project risk analysis and informed the FMEA application in this study. The trialability element of the DIT theory was applied to address the fourth research question of the study. The implementation involved the risk assessment activities conducted following the 8 steps of the procedural requirements of the FMEA for the process of software project risk assessment as proposed by Georgieva, (2010), and discussed in the Chapters 2 and 4, with practical implications analysed in Chapter 5. The 8 procedural steps of the FMEA as implemented in this study are:

1. Select experience team and review the process,
2. Think about unknown risks,
3. Rank severe effects of the failures.
4. Rank severity, occurrence and detection for each failure mode.
5. Compute the RPN value for each risk.
6. Determine the risk priority and classify the risk for failure modes for each action.
7. Deduce appropriate control action to minimize the risk effect,
8. Repeat the risk management process from start and evaluate the performance of the system again.

The main finding of this study is that the FMEA procedural requirement was found to conform to the popular project risk management frameworks; hence, it was possible to identify, assess, prioritise and classify the risk factors associated with potential failure modes in the software development process using the FMEA method as discussed in sections 4.9 and 5.8. The finding also showed that the application of the FMEA method with the adoption of risk criticality standard of USDOD (MIL-STD-1629, 1980) for the conduct of software project risk management as discussed in section 5.8 assist in determining the remedial actions to minimize the risk associated with the failure. Thus, it provided useful insight to project managers on how the associated risk is to be attended to.

It was also found that student developers that were trained to use the FMEA method for a project risk management course in one of their software engineering courses acknowledged the ease of use and the applicability of the FMEA method for the conduct of software project risk management. This satisfied the requirements of TAM for application of an efficient technology for the conduct of project risk management as discussed in section 5.9.

One theoretical contribution of this study is the modification of the existing risk theory to estimate the risk magnitude in this study using the RPN value as specified by the FMEA theory. The risk concept/theory as conceptualized in the literature, which viewed risk as: Risk = Probability of Occurrence * Impact of Loss (Boehm 1991, 1989; Charette 1989, 1996) is narrower than the nature of the problem in practice as required in software engineering. Based on this gap, it is apparent that current risk estimation practices lack a context-based framework that can adequately address the unforeseeable problems before their occurrences in the software development process. In view of this prevailing challenge, a modified risk theory was considered in this research that will integrate the ranking of symptoms of detecting the unforeseeable failure that may occur by including the '*Detection*' variable in the existing theory. Hence, the use of Risk Priority Number (RPN) to quantify the risk magnitude was considered.

The RPN was computed using the risk factor variables as (RPN = Probability of occurrence * Impact of Loss * Detection). The variable: the probability of occurrence of the identified failure mode is determined using the probability table, assessing the impact of loss, using the impact table, and assessment of the level of failure detection and control is determined using the detection/control table. The RPN value was used to rank and classify the risk factors associated with the failure modes. The ranking was followed by suggestion of risk response plan against each of the identified failure modes using the criticality classification standard suggested by the USDOD (MIL-STD-1629, 1980). The inclusion of the detection variable in the risk estimation theory as used in this study addresses the identified gap in the previous estimation theory.

The modified risk theory and the adopted criticality classification standard used in this study provided a novel risk estimation practice. The practice provided a proactive risk analysis culture using a reliable risk estimates that is capable of identifying and preventing a failure before its occurrence. The estimate can also determine the failure mode that poses the highest threat to the success of the project. The higher the RPN value, the higher the risk posed by the concerned failure mode. Similarly, the criticality classification standard adopted for the study as suggested by the USDOD (MIL-STD-1629, 1980) provided useful insight to project managers on how to attend to the associated risk and more importantly, it provided useful insight on how to distribute management resources across the software development process.

Research question 6:

What risk response strategies are considered appropriate actions for managers to address the threats posed by the identified potential failure modes in the software development process?

The worksheet document presented in the Tables 5.17 – 5.21 indicated the appropriate response strategy that managers will follow to mitigate the risk. The effective strategies as indicated in the tables can be immediate, required, and essential or delayed, depending on the risk classification of the identified potential failure modes. Addressing the RQ5 of this study involved the use of the calculated RPN value to classify the risk criticality for each of the identified potential failure modes and the RPN value is used to determine the appropriate control strategy to counter the effect of the risk. The analysis was based on the criticality classification standard of USDOD (MIL-STD-1629 1980) as explained in Chapter 4 and the appropriate strategies outlined in Chapter 5.

The risk criticality standard of the USDOD (MIL-STD-1629 1980) provided criteria for risk prioritization and classification using the RPN value (see Chapter 2). The criticality classification standard of the USDOD (MIL-STD-1629, 1980) was adopted in this study as criteria used to suggest appropriate strategies for controlling the risk effect as explained in Chapter 4 and outlined in Chapter 5. The results of the FMEA process conducted in the Field study 4 implemented the criticality classification standard of the USDOD (MIL-STD-1629, 1980) to suggest suitable response strategy to address the effects of the identified potential failure modes in this research, thus, answering research question 5.

Analysis of the results obtained revealed that the failure mode with RPN < 70 is considered to pose a threat whose risk does not require corrective and preventive actions. Hence, the corrective/preventive action can be delayed. Similarly, failure modes that have at least one out of the three parameters of the RPN (specifically, the ‘severity’ or ‘occurrence’) with a rated figure greater than 5 but RPN relatively low (typically 70 < RPN < 140) are considered semi-critical threats, thus, corrective/preventive action is essential. Also, failure modes that are highly rated with at least two parameters of the three RPN parameters (specifically, S, D and O) and have the calculated RPN value being too high (usually RPN > 140) are considered critical risks and must have a corrective/preventive action.

The main theoretical and practical contribution of this research study is the structured process with which the FMEA method was conducted, which was different from the previous research to suggest best risk control mechanism to mitigate against risk effects. The method is oriented towards a continuous, proactive and team-oriented approach and adopted formal standards as criteria for the most appropriate decision for the risk assessment and control in all the stages of the SDLC.

Based on the literature as reinforced by the findings of this study and practical experiences on the FMEA applications, the FMEA method is justified to be a competent risk assessment tool that can reliably estimate the risk magnitude of the potential failure modes in the software development process; assess the magnitude of their effects on software project outcomes (cost, time and quality/scope); classify their criticality; and suggest realistic response to mitigate the risk effect. The findings from the FMEA process conducted in the Field study 4 justifies this as it provided strategic information on how project managers can apply the FMEA method to carry out software project risk assessment and determine what best risk control mechanism to be deployed to mitigate the risk.

6.1.2 Research findings and discussions based on the research hypotheses

Research hypothesis 1:

Effect of potential failure modes in the software development process on software project outcome (cost, time and scope/quality)

$H_{01:}$ There is no significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality).

H_1: There is significant difference between the mean scores' effect of potential failure modes across the stages of the software development process on the project outcome (cost, time & scope/quality).

The results of the ANOVA test procedures comparing the mean scores of the potential failure modes' effect in the studied stages of the software development process on the project outcome to test the research hypothesis 1 were discussed in section 5.10.1. The results revealed the various forms of effects of potential failure modes in the software development process on the project outcome variables as follows:

<u>Project Cost:</u> The analysis of data in the Table 5.25 revealed a statistically significant evidence at α=0.05 to show that there was statistically significant difference between the mean scores of the potential failure modes' effect in all the studied stages of the software development process on the project cost. Thus, the effects of potential failure modes in the software development process are perceived by software development practitioners to be severe and pose a significant but unequal degree of threat to project cost across the stages of software development process.

<u>Project Duration:</u> The analysis of data in the Table 5.26 revealed a statistically significant evidence at α=0.05 to show that there was a statistically significant difference between the mean scores of the potential failure modes' effect in all the studied stages of the software development process on the project duration. Thus, the effects of potential failure modes in the software development process are perceived by software development practitioners to be severe and pose a significant but unequal degree of threat to project duration across the stages of software development process.

<u>Project Scope/Quality:</u> The analysis of data in the Table 5.27 revealed a statistically significant evidence at α=0.05 to show that there was no statistically significant difference between the mean scores of the potential failure modes' effect in all the

studied stages of the software development process on the project scope/quality. Thus, the effects of potential failure modes in the software development process are perceived by software development practitioners to be severe and pose a significant and equal degree of threat to quality of the project across the stages of software development process.

These findings showed that the risks associated with the potential failure modes pose varying degrees of threat to software project outcome. These threats are classified as critical, semi-critical and normal risk, depending on the RPN values and the rating scales of Occurrence and Severity of the potential failure modes. The risk with highest RPN poses the highest threat to software project outcomes. Understanding the nature of the threats posed by the risky potential failure modes on the project outcomes will assist risk managers in deciding which risk to give the first priority of attention and the next in order of severity. Moreover, having adequate knowledge on nature of threat posed by these risks on the project outcomes will support project managers in making the best allocations of risk resources in project management.

Research hypothesis 2:

FMEA adherence to other popular project risk management frameworks

H_{02}: There is no significant difference between the mean scores of FMEA adherence and the other popular project risk management frameworks.

H_2: There is significant difference between the mean scores of FMEA adherence and the other popular project risk management frameworks.

The results of the ANOVA test procedures comparing the mean scores of the FMEA adherence with the other popular project risk management models to test the research hypothesis 1 were discussed in section 5.10.2. The result showed that there was a statistically significant evidence from F Table at $\alpha = 0.05$ to show that there was no statistically significant difference between the mean scores of the FMEA Adherence to the three studied popular project risk management frameworks (PMBOK, SEI & Boehm). This result confirmed that FMEA model conformed to the standard principles and practices for managing the risk in software development projects. This result agrees with the findings of Souza and Cabral (2008).

The possibility of applying the FMEA method in SE was discussed extensively in literatures including detailed case application of FMEA method in software projects. The adherence case study conducted by Souza and Cabral (2008) revealed that FMEA standard procedures are of high level of adherence to a popular project risk management framework of PMBOK and hence, recommended to be a powerful tool for conducting project risk management. Also, the findings of exploratory case studies involving focus group study and survey study in Field study 1, clearly demonstrated that FMEA method has the adequate procedural requirements to effectively conduct software project risk management.

Research Hypothesis 3:

Efficiency of the FMEA method for the conduct of software project risk management

$H_{03:}$ FMEA method is not efficient for the conduct of software project risk management.

$H_{3:}$ FMEA method is efficient for the conduct of software project risk management.

As discussed in section 5.10.3, the efficiency of using the FMEA for risk factor assessment in software projects is a measure of both the *perceived ease of use* and the *perceived usefulness* of the FMEA for the task of risk management. The result analysis of the FMEAEQ data on % satisfaction on the FMEA’s ease of use and usefulness as perceived by undergraduate SE students in their project risk management practical coursework indicated that the % satisfaction derived from FMEA ease of use and usefulness for the conduct of risk management as perceived by student managers is significant (as discussed in section 5.10.3).

The findings of FMEA interactive study with undergraduate SE students in Field study 5 proved that the idea of applying FMEA method in software risk management is perceived to be quite practicable, acceptable, viable, simple, usable, convenient and user-friendly. This result showed that students' behavioural usage of FMEA method as a measure of perceived ease of use and usefulness for the conduct of risk management in software projects is resourceful and effective. Thus, the FMEA method is an efficient tool for the conduct of risk management task in software development projects.

- The analysis of results presented in section 5.10 indicates that FMEA behavioural use in terms of ease of use, the FMEA method is convenient, simple and user-friendly for conducting software project risk management.
- Results discussed in section 5.10 indicated that in terms of usefulness, FMEA method is practicable, efficient and resourceful for the conduct of software project risk management.

The FMEA method provides a novel alternative tool for the conduct of software project risk management that overcomes the limitations of the P-I theory for risk estimation. It also builds upon the existing software projects risk management attempts in literatures as demonstrated in this research. Furthermore, the application of the FMEA for software risk assessment as used in this study creates a link between the FMEA theory and the risk criticality classification standard as suggested by the USDOD (MIL-STD-1629 1980) and as used in this research. This may contribute to improving the theory of software project risk management through expanding the conventional perception of software project risk factors dimensions to include potential failure modes dimension and integrate the FMEA as alternative tool for risk management. In this regard, this research has presented the concept of *"software risk management using the FMEA method*' that assesses the risk effects on the software project outcomes and reflects the level of efficiency to which the software projects risk management is conducted using the FMEA method.

6.2 Summary of the research contributions to knowledge

Despite the narrow scope and limitations of this study, it is interesting to establish that the findings that emerged from the study contribute significantly and positively to the discourse in the SE research domain. Specifically, this study has made significant contributions to the existing body of knowledge in the theory and practice of software project risk management, preparing and supporting project managers for the application of safety and reliability engineering tools for the business of risk management in software projects in the following ways.

Firstly, with regard to theory and practice in SE, the study provides empirical baseline data and information that can be used as a guideline for best practices and theory formulation for software improvement initiatives and support. This is to develop more reliable software products through effective project management. In this study, a promising conceptual framework was designed by extracting seven core constructs from the original DIT model (Rogers 2003) and the TAT model (Davies 1989) to spread the idea of using the FMEA with software practitioners and student developers to explore other important risk factor dimension and conduct risk factor assessment in software projects and the subsequent measure of actual behavioural use of the tool. The use of a combined model (DIT and TAM) to investigate the integration of the FMEA in software project risk management using both qualitative and quantitative research using the case study and survey designs is a novel methodological contribution in the field of software engineering research.

Secondly, this study contributes to theory and practice in SE by bringing to light an important risk factor dimension that will help to improve the existing risk management practices in software projects. The extension of the existing risk factor dimension to include the risk factors associated with potential failure modes in the software development process is a novel initiative. The study contributes to knowledge in the area of risk factor research by showing the pattern of distribution of risk factors associated with potential failure modes in the software development process. The distribution patterns showed that the risky potential failure modes in the SDLC are unevenly distributed across the stages of the SDLC with early stages having higher numbers of failure modes that are potentially risky to the project outcome than the later stages of the SDLC as summarized below:

- Stage 1: Planning – Ten (10) potential failure modes were identified in this stage with potential failure mode: *poor risk management strategy* perceived to pose the highest threat to project outcome and the potential failure mode: *communication* *gap between developers* perceived of posing the lowest damage to product outcome (see Table 5.6).

- Stage 2: Requirement Specification analysis – Eleven (11) potential failure modes were identified in this stage. The potential failure mode: *inaccurate requirement* was perceived by experts to be capable of posing the highest threat to project outcome while the failure mode: *contradictions and confusion in domain specific terminologies* posing the lowest threat to the project outcome (see Table 5.7).

- Stage 3: Design - Eleven (11) potential failure modes were identified in this stage with the potential failure mode: *poor implementation of design principle* considered to have the highest level of threat in the stage and the failure mode: *too much of elaborative specification at the stage* perceived to have the lowest threat 5(see Table 5.8).

- Stage 4: implementation and coding - Thirteen (13) potential failure modes were identified in this stage with the potential failure mode: *Errors in human factor engineering during coding and testing* perceived to have the highest threat in the stage while the failure mode: *no provision for reusability* was perceived the failure mode with the lowest threat in the stage (see Table 5.9).

- Stage 5: installation and maintenance – Twelve (12) potential failure modes were identified in this stage with the potential failure mode: *change on one component of the new system having adverse effect on the rest of the system* perceived of having the highest threat to the project outcome in the stage and the failure mode: *system not installed correctly due to change in environment* was perceived as failure mode with the lowest potential damage to the project outcome in the stage (see Table 5.10).

In addition, the research findings showed that the causative agents of the risky potential failure modes are not mutually exclusive. A number of them are resulted from multiple causative factors ranging from human engineering, hardware engineering and development process factors. In addition to this, the findings also revealed that the magnitude of the threat posed on project outcome by the individual failure item are interdependent to each other and that the magnitude is higher at the early stages of the development process as shown in the Figure 6.1.

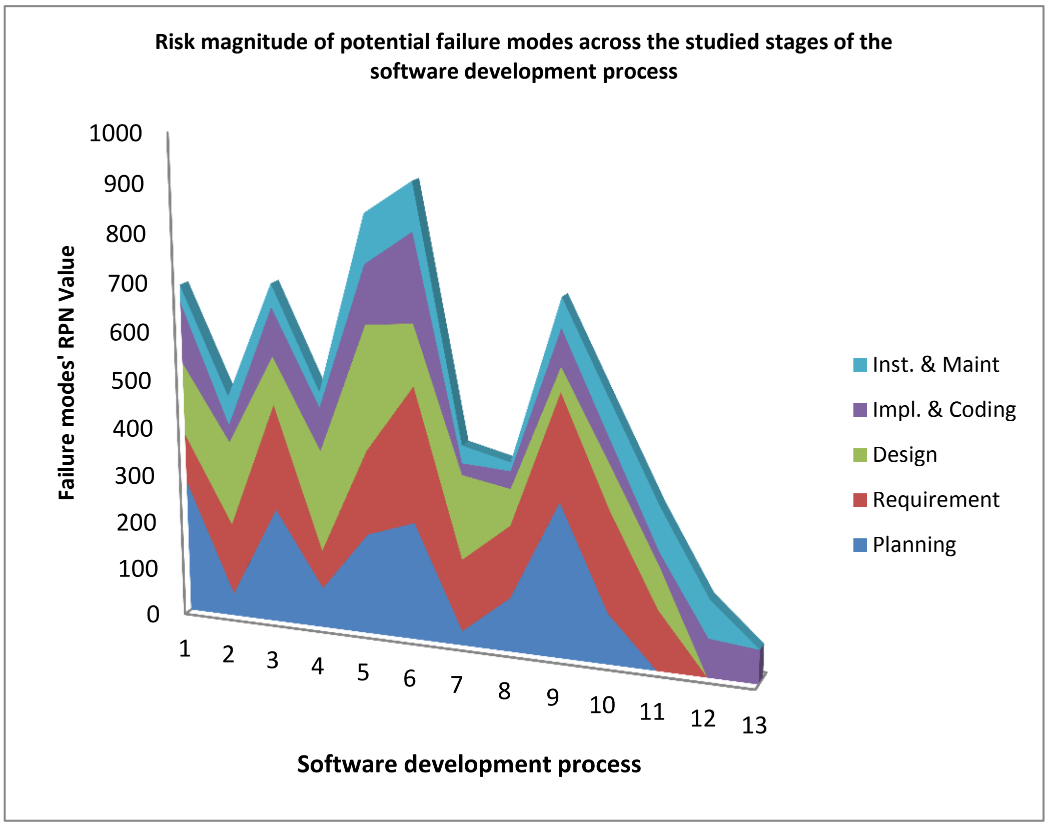
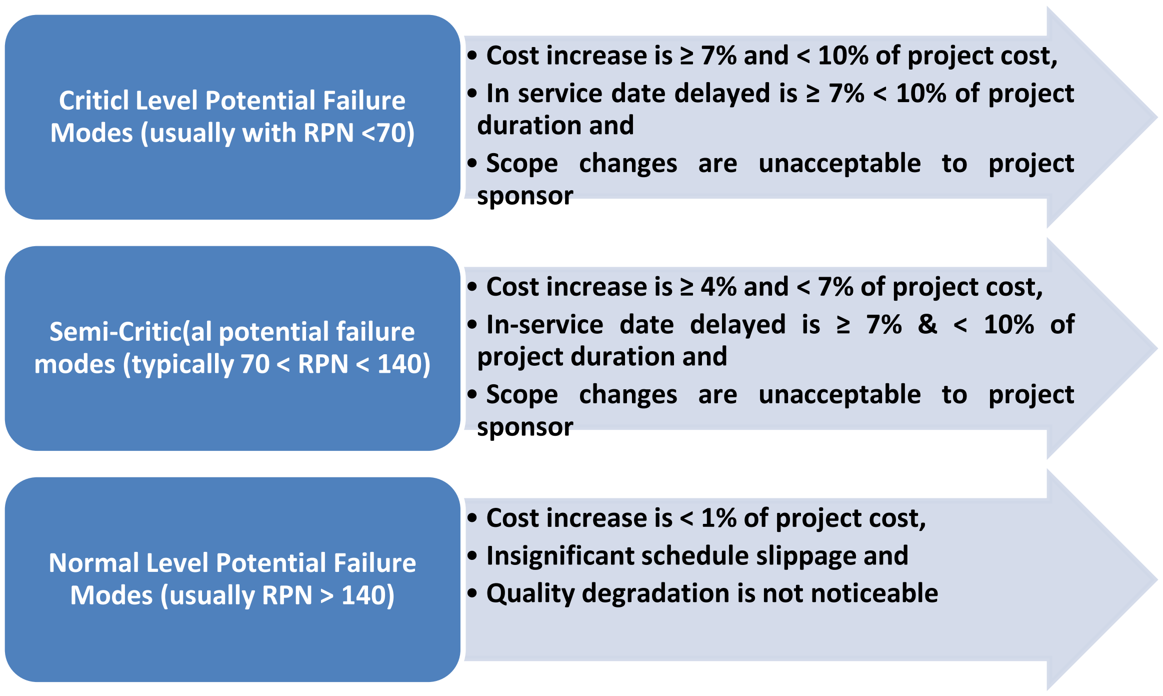


Figure 6.1: Risk magnitude of potential failure modes across the studied stages of the SDLC

The figure shows that the potential failure modes considered to be risky to the project outcome do not remain the same across the stages of the SDLC. It evolves in a dynamic pattern with early stages of the SDLC tend to be dominated with the highest magnitude of risky pattern incidents than the later stages. The study contributes to knowledge in the area of risk factor research in software engineering by extending the existing research in risk factor dimensions to include the risk factors associated with potential failure modes in the software development process. This study contributes rich empirical grounding information that will guide and assist project managers on risk factor management and decision making in software project management.

Thirdly, the manner in which this study adopted a modified risk theory to estimate the risk multiplier effect on project outcome using the RPN value as specified by the

FMEA theory is novel. The modified risk theory integrates the criticality classification standard suggested by the USDOD (MIL-STD-1629 1980) for risk prioritization and classification of risks and adopted the risk impact category scale of the PMI (2008) to interpret the multiplier effects of the classified risks on the project outcome (cost, time & quality). The RPN value of the identified risks are then interpreted using the combined standards of the USDOD (MIL-STD-1629 1980) and the impact category scale of the PMI (2008) to suggest appropriate risk response strategies that will mitigate the multiplier effects on the project. Figure 6.2 presents the likely multiplier effects of classified risky potential failure modes in the SDLC on the project outcome using the impact scale of the PMI (2008).



Criticl Level Potential Failure Modes (usually with RPN <70)
• Cost increase is ≥ 7% and < 10% of project cost,
• In service date delayed is ≥ 7% < 10% of project duration and
• Scope changes are unacceptable to project sponsor
Semi-Critic(al potential failure modes (typically 70 < RPN < 140)
• Cost increase is ≥ 4% and < 7% of project cost,
• In-service date delayed is ≥ 7% & < 10% of project duration and
• Scope changes are unacceptable to project sponsor
Normal Level Potential Failure Modes (usually RPN > 140)
• Cost increase is < 1% of project cost,
• Insignificant schedule slippage and
• Quality degradation is not noticeable

Figure 6.2: Likely multiplier effects of classified risky potential failure modes in the SDLC on the project outcome

This practice provided useful insight to project managers on how the associated risk is to be prioritized, classified and managed. More importantly, it suggests the most prudent way managers can used to distribute management resources across the software development process.

Fourthly, , the study justified the use of the FMEA method as an efficient tool for conducting risk management practice in SE. Figure 6.3 presents the correlation pattern of the procedural requirements of the FMEA model with the requirement features of the prominent project risk management models PMBOK, SEI & Boehm).

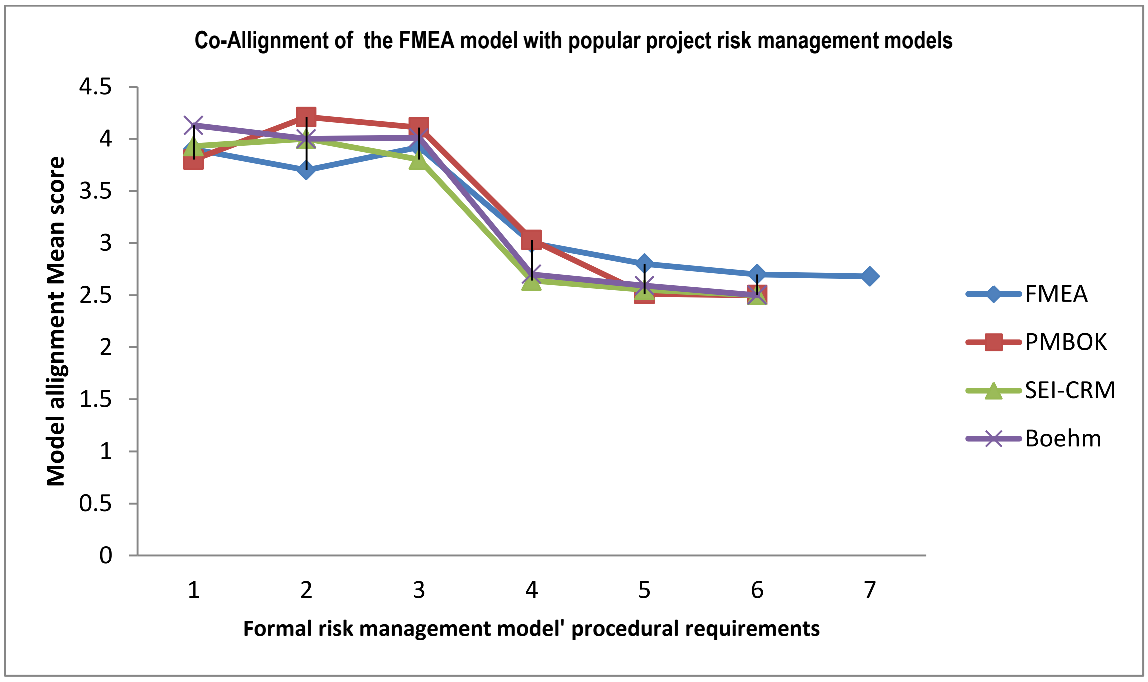


Figure 6.3: Co-Alignments of the FMEA model with popular project risk management models

Figure 6.3 shows that the procedure requirements of FMEA model have a high level of adherence to the formal software project risk management requirement (PMBOK, SEI & Boehm). This study contribute to the existing knowledge by providing empirical evidence that confirms the FMEA model as a competent tool for the conduct of software project risk management process. in addition to this, the structured and formalized processes under which the FMEA method was used in this study to assess the risk factors associated with potential failure modes in the software development process is novel.

Figure 6.4: Modified FMEA process used for the conduct of risk management in this study

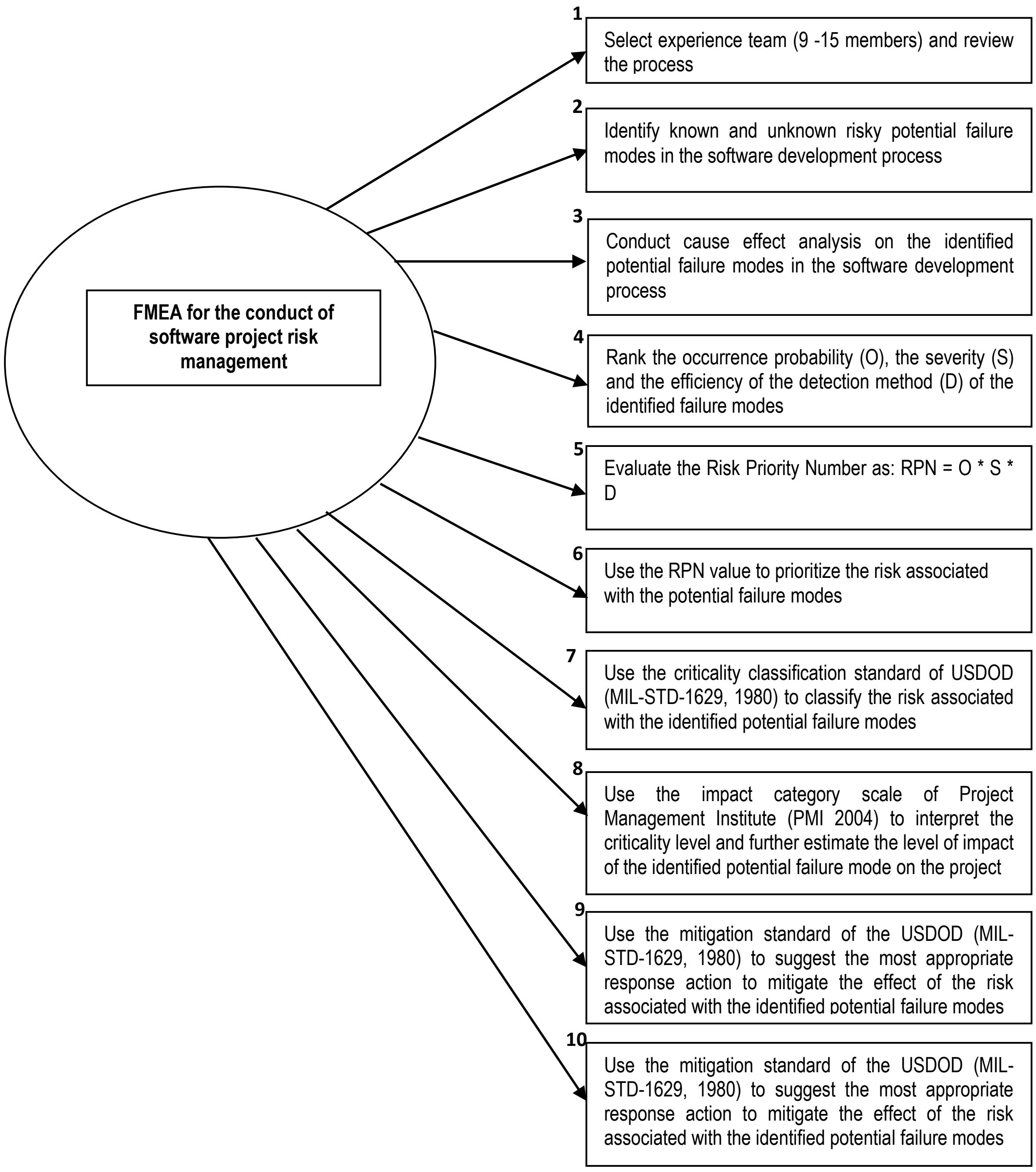

FMEA for the conduct of software project risk management
1
Select experience team (9 -15 members) and review the process
2
Identify known and unknown risky potential failure modes in the software development process
3
Conduct cause effect analysis on the identified potential failure modes in the software development process
4
Rank the occurrence probability (O), the severity (S) and the efficiency of the detection method (D) of the identified failure modes
5
Evaluate the Risk Priority Number as: RPN = O * S * D
6
Use the RPN value to prioritize the risk associated with the potential failure modes
7
Use the criticality classification standard of USDOD (MIL-STD-1629, 1980) to classify the risk associated with the identified potential failure modes
8
Use the impact category scale of Project Management Institute (PMI 2004) to interpret the criticality level and further estimate the level of impact of the identified potential failure mode on the project
9
Use the mitigation standard of the USDOD (MIL-STD-1629, 1980) to suggest the most appropriate response action to mitigate the effect of the risk associated with the identified potential failure modes
10
Use the mitigation standard of the USDOD (MIL-STD-1629, 1980) to suggest the most appropriate response action to mitigate the effect of the risk associated with the identified potential failure modes

Figure 6.4: Modified FMEA process used for the conduct of risk management in this study

Lastly, the manner in which the FMEA method interactive study with students was organized and conducted provided opportunity for the prospective risk managers to learn important skills required for a method implementation for the conduct of software project risk management (see section 4.10.1.2: The interactive study session and section 4.10.2.1: The interactive method implementation). Thus, addressing the challenge of inadequate training facing risk managers of software development projects

The next section presents the limitations and assumptions of the study.

6.3 Limitations of the Study

Even though the findings that emerged from this study have been justified to contribute significantly to the discourse in the software engineering research domain, the research has some limitations. The research limitations are summarized as follows:

- Risk factors in software projects and software project risk management are two emerging research areas in SE. Both of them are complex dimensions which dominate research areas in SE. Going by the aim of this research, it is clear that this study is only restricted to one risk dimension. Thus, it may not have captured every aspect of risk factor dimension in software development projects. This limitation was mitigated by reporting a comprehensive literature study about software risk factor dimensions that was conducted and software experts and researchers in the field were consulted for inputs.
- The use of the FMEA method for project risk management needs to be tested in different software projects and industries outside the research participants' application area of specialization in order to further validate the method. The continuous risk approach will make it possible to test the repeated risk management results against the previous ones and analysis will be conducted to validate the findings. The replication of the study in different projects across different industries and a wider coverage would add to the validity of the findings and for further empirical support for related studies. However, this limitation was mitigated by conducting an empirical study involving wide range of software practitioners around the world for inputs on related studies of which findings were used for validatory purposes (see Field study 3).
- Companies outside Nigeria were not considered in this research. If some companies outside of Nigeria were considered it would have made the research far more applicable. However, the threat of this limitation to the study validity was mitigated with wide range of experts participation in the study,

- Though the table of scales for rating risk variables (occurrence, severity and detection) used in the FMEA process of this study were modified in natural language and validated (see Tables 4.2 - 4.4), there is room for further improvement of the construct so as to improve their reliability level and improve their capacity to simplify the variance associated with the variables they are developed to measure. In this research, the tables used for rating risk variables (occurrence, severity and detection) as used in the FMEA process were modified in natural language, this simplify their applicability and suitability for their assessment purposes. Large parts of the findings in this research were derived based on surveying software development practitioners only. Thus, the research findings are limited to practitioners' perspectives and cannot be generalised to the user side. There is the need to extend the research to investigate the efficacy of the research contributions and the validity of its findings from a software users' perspective.
- The research used four different questionnaires, which were administered through real time distribution and through e-mail. The use of the online method was cancelled due to some technical challenges, which has negative effect on the number of responses. The response rate could have been more. Specifically, the cancellation of online questionnaires made it impractical for some software exerts to respond to the e-mailed questionnaires even after their initial agreement to participate in the research. The use of the online questionnaires could have generated a wider spectrum of software experts' opinion and constructive feedback.
- Time and distance constraints were part of the challenges encountered from the beginning to the end of this research. It took the researcher a longer time to convene focus group meetings in the Nigerian capital city of Abuja and participants around the world were contacted through e-mail due to distance barriers.
- The interactive study with students (Field study 5) took quite some time resulting into some fatigue challenges as expressed by some participating students. This challenge was however addressed, by integrating the interactive study as part of the SE curriculum with regulated timeframe.

- The use of students developers in the studies, with none or limited real world experience pose a validity threat to the study. This threat was mitigated by engaging the students in an interactive training on the application of the FMEA for the conduct of software project risk assessment.

- The student developers that validated the usability and applicability of the FMEA method commented in the spaces provided in the FMEAEQ that the FMEA method was time consuming. This study was not able to investigate the average time required to apply the method.

The following section presents the reflections on the study.

6.4 Reflection on the study

This section presents the scientific, methodological and personal reflections on the study.

6.4.1 Scientific and technological reflection

The study was conducted to assess the risk factors associated with potential failure modes in the software development process that can influence the software project outcomes using the FMEA method. The study conducted on IS research revealed that interpretivism, constructivism, critical realism, positivism and more recently pragmatism formed the dominant research philosophical paradigms used in the IS research. An in-depth study of these paradigms revealed that critical realism would be the most appropriate to foreground the application of the FMEA method for the assessment of risk factors associated with potential failure modes in the software development process (see Chapter 4).

Furthermore, an in-depth investigation into the techniques, practice and instruments for identifying, assessing and controlling risks in the field of risk management in SE informed the research area of this study. The theoretical framework, as presented by Grant & Osanloo (2013), informed the theoretical framework of this study (section 3.1). The adoption of the FMEA method is suitable for the conduct of the risk assessment and informed by the conformity of method's procedural requirements with the procedural standards of formal project risk management frameworks (e.g., PMBOK, SEI and Boehm) as revealed by the findings of the Field study 2 (see sections 5.5.2 to 5.5.5). The literatures on risk factors in software project management and the application of safety and reliability engineering techniques for risk analysis in project management informed the study and added to the empirical resources in these domains.

The clarifications and justifications provided in this section provide evidence that this study was conducted based on the methods and principles of scientific and technological research.

6.4.2 Methodological reflection

The analyses of responses generated from the question (Q1) were used to reflect on the methodology for this study.

Q1: '*Was the selected research design and methodology the most appropriate to provide answers to the research questions and hypotheses?'*

The study was conducted to assess factors associated with potential failure modes in the software development process using the FMEA method. Risk assessment according to Boehm (2004) involves four major activities: Risk identification, Risk analysis, Risk prioritization and Risk classification. The risk assessment activities were conducted in a 5 field studies following a mixed-method research: qualitative and quantitative.

The field studies include an exploratory case study that involved a workshop and a method introduction study of the feasibility of integrating the FMEA method in software project risk management, which was followed by a focus group study, then survey study with selected experts and researchers in software engineering, the FMEA application process and the FMEA interactive studies. The field studies were conducted using four questionnaires, three paper and one e-mailed, five semi-structured focus group interviews, and two validation case studies. Furthermore, the study investigated the procedural requirements of the applied FMEA method in relation to the requirements of the formal project risk management frameworks (such as PMBOK, SEI and Boehm).

Procedures and guidelines incorporated in the FMEA procedural framework as suggested by Tay and Lim (2006) were utilised for the execution of the study. The FMEA is a proactive approach to failure prevention that anticipates and plans for problems or failures before they occur. It involves structured brainstorming techniques to identify and analyse potential failure modes in a project rate; rank, prioritize and classify the risk in a project and suggest appropriate actions to mitigate the risks. The use of FMEA has been a productive, resourceful, successful, reliable technique in software risk assessment methodologies as evidently reported in the literatures reviewed (e.g., Gupta et al. 2012; Mitrabinda & Durga 2011).

A modified risk theory adopted from the FMEA theory that included the detection factor was adopted for the risk assessment, the risk effects were interpreted and prioritized using their computed RPN values and the risk classification was conducted using the risk criticality standard proposed by the USA DOD standard, (MIL-STD-1629, 1980). The research hypotheses were tested using the ANOVA test procedures. The FMEA was adjudged to be a cost-effective tool that can significantly reduce the cost of risk analysis in software projects (SW-quality 2016). Findings from literature reviews on the FMEA integration in the SE, as presented in Chapter 2, together with the reflection and interpretation thereof, contributed to the assumptions that motivated the application of the FMEA theory for the risk assessment in this study. The study was based on a critical realism philosophical paradigm. This indicates that the research followed the typical stages of mixed-method research while applying important considerations of critical realism epistemology by engaging in a critical dialogue with software development practitioners to use the FMEA method to identify, analyse, assess, prioritise and classify the risk associated with the potential failure modes in the software development process. The applied FMEA method was evaluated for its ease of use and applicability for the conduct of risk management in software development projects by student developers, which is one of the TAM evaluation methods suggested by Davis (1989).

Based on the above clarifications, it can be concluded that suitable research design and appropriate methodology were applied to address the research problem.

6.4.3 Personal reflection

My personal reflection on the study revolves around the assumptions I hold as the researcher that the successful completion of the research study relied heavily on the following:

- a comprehensive and systemic literature review that led to a clearly stated and focused research problem;
- a realistic and detailed study plan with specific objectives;
- an appropriate field study design, instruments used for data collection and the study objectives inked to specific due dates; and
- motivation, discipline and perseverance.

I also have the assurances that my level of motivation and the success of this study were dependent on the contribution and cooperation of the research participants and the guidance and knowledge of my skilled and experienced supervisors. This study has increased my knowledge on the following research components: theoretical and conceptual frameworks, research design and methodologies and philosophical paradigms.

As a software developer, I have gained vast knowledge of other risk factor dimensions in software projects and application of safety and reliability engineering tools for the conduct of risk management. Specifically, my knowledge of the application of the FMEA for the practice of risk management in software development projects is fairly significant. Reflecting on the study stages and processes, it was a very rich, intense and rewarding experience.

The next section presents the recommendations for future academic research.

6.5 Recommendation for future academic research

Improving the practice of risk management in software development projects is a pertinent concern in modern software development organizations. As a result of assessment of risk factors associated with potential failure modes in the software development process, further research questions for future academic research could include:

- **To what extent are other reliability engineering techniques (such as the Failure Tree Analysis (FTA), Functional Failure Analysis (FFA), Bi-Directional Safety Analysis (BDSA), etcetera) be combined with the FMEA to improve the practice of software practicable for project risk management?**

This study integrated the FMEA for the conduct of project risk management in SE. Although, the findings from the literatures studied provided a sound motivation for the application of FMEA method for project risk management in software projects, further scientific research could be extended on the combination of the FMEA with other popular safety and reliability engineering tools (such as Failure Tree Analysis (FTA), Functional Failure Analysis (FFA), Bi-Directional Safety Analysis (BDSA), etcetera) that have record in literatures, to determine their efficacy and applicability in project risk management. This idea will further strengthening the use of of the reliability engineering techniques for the conduct of software project risk management.

- **Researching other software stakeholders (e.g., software users) perception on the application of the FMEA to analyse the potential failure modes' effect on the software product**

Another potential area of research could be capturing software users' opinion on the failure occurrence and effect on software products. This involves evaluating software project users' perception of probability of occurrence of potential failure modes in the software development process and their effect on project outcome and comparing their perceptions with other stakeholders of same software project development. It is logical to assume that different project stakeholders would view risk and risk effect differently and the differences in their evaluation would provide a ground for further investigation for improving the quality of software product.

- **Investigating the potential failure modes - risks' inter-dependability in all the stages of the software development process using the FMEA**

This research has adopted multidimensional approach to identify the potential failure modes in the SDLC and used the FMEA to assess the effect of their associated risks on the project outcome. The assessment is limited to individual risk evaluation within the software development process. The research did not establish a framework that will track risk inter-dependability across the studied stages of the software development process. Information about risk inter-dependability will assist risk managers to be aware of other factors that could have an influence on the existence of risks in another stage of the software development process and tracing corresponding effects on project outcome.

- **How to use the FMEA to model risks and develop appropriate risk management strategies?**

This study has proved that the nature of risk associated with potential failure modes varied across the stages of the software development process. Future research should focus on how to use the FMEA for risk modelling and developing appropriate risk management techniques for each of the risks associated with potential failure modes in all the stages of the software development process. This idea could further help software project managers to re-direct their risk management strategies more appropriately.

- **To what extent are the constructs tables used for risk variable ratings in the FMEA models for software risk management project practicable in other fields outside SE?**

Three measurement constructs were validated and used for risk factors rating (Occurrence Table, Severity Table and Detection Table) in the FMEA process (see Tables 4.2 – 4.4). These measures are validated in both practice and theory. These measurement constructs provided guides for rating the probability of potential failure modes' occurrence, rates the likely effect of project outcomes and rates the possibility of the detection method to detect failure early enough in the software development process. The product of the resulting ratings will determine the risk magnitude of the potential failure modes. The existence of validated and reliable measures will enable numerous future researchers to modify these constructs for other perspectives.

- **To what extent is the FMEA method practicable for the conduct of project risk management outside the software industries?**

This study indicated that the FMEA method is an effective project risk management tool. Findings from the study show that FMEA has the capacity to identify, assess, prioritize and classify risks in software projects. Thus, its application can go beyond software projects. Hence, it is worth investigating the efficacy and practicality of the FMEA models and methodologies in other fields such as construction projects, manufacturing, investment and policy making projects.

- **How to objectively validate the FMEA method in software project risk management**

Research in this area can be conducted in two ways: first, the efficiency of the FMEA method for the conduct of software project risk management can be determined using a continuous risk management process. This involves reassessment of the risk for the second time in all the stages of the software development process after the implementation of the suggested management strategies to control the risk. The results will then be compared and analysed to determine if the risk value for each failure mode in the software development process has been reduced. The findings will then be analysed to determine the efficiency of the method.

Secondly, the accuracy of the results obtained in FMEA process for the conduct of software project management can be improved using other validation methods such as longitudinal case studies. By this method, the FMEA process can be applied for risk management activities in different software projects and industries in order to determine accuracy of the method.

- **Evaluating the perceptions of experienced software project managers on FMEA practicality and suitability for the conduct of software project risk management**

The results used to evaluate the ease of use and usefulness of the FMEA method for software project risk management were generated from undergraduate SE student's perspective, who are assumed to be prospective software project managers. Hence, the results are limited to the student context and cannot be generalised to the experienced managerial side. However, further research is required to investigate the practicability and usability of the FMEA method from experienced project managers' perspectives.

The next section presents the concluding remarks about the study.

6.6 Concluding remarks

The study was concerned with the assessment of the risk factors associated with potential failure modes in the software development process. The risk assessment was conducted using the FMEA theory and supported by a critical research philosophy. During the conduct of this research, a combination of case study and survey designs were adopted to collect and analyse data for the study. The conceptual framework used for the risk assessment was formulated using the seven fundamental constructs extracted from the DIT and TAM in relation to the study.

This study provided insight into another dimension of risk factors in software development projects and proposed a pioneering initiative of a proactive project risk management practice in SE that provided information about the 'risky potential failure modes in the software development process, and their likely effects on the software project outcome'. The study also demonstrated how the FMEA method can be applied for the conduct of risk assessment in software projects. Specifically, the study demonstrated how to use the FMEA method to identify risk, analyse, prioritize, classify risks and suggested appropriate risk response strategies to control the risk effect. The study finally demonstrated how to evaluate the ease of use and the applicability of the FMEA method for the conduct of risk assessment in software development projects.

The next section presents the chapter summary.

6.7 Chapter summary

This chapter has presented an overview of the entire research report and provided evidences that the research questions were answered. Furtherance to this, the chapter presented a summary of key findings and highlighted some of the major contributions that the study made to the existing body of knowledge. Given that the study is bounded by time and scope, the research recommendations and suggestions for further research were presented in this chapter. In addition to this, the chapter highlighted what the limitations of the study are and reflected on the study.

Conclusions from the findings as presented in this chapter suggest that project managers and their risk management team will conduct effective risk management in software project development when they have sufficient information on other important risk factor dimensions (such as potential failure modes in the software development process). Also, managers will develop positive managerial attitude by building confidence on the application of reliability engineering tools when they are well informed on how to apply the reliability engineering tools for project risk management. To sum it up, student developers seem to acknowledge the ease of use and the applicability of the FMEA method for the conduct of risk assessment in software development projects.

REFERENCES

Abdelgawad, M. & Fayek, A. R. (2010). Quantitative assessment of horizontal directional drilling project risk using fuzzy fault tree analysis. *CRC, Innovation for Reshaping Construction Practice,* 373, p. 1274-1283.

Abubakar, M. M. & Lawal, B. (2020). Exploring the Potential Failure Modes in the Software Development Process. *International Journal of Science for Global Sustainability,* 6(3), 94-98. Available at: link.gale.com/apps/doc/A644501391/AONE?u=anon~f5f28630&sid=googleScholar&xid=fa31255a.

AIPM & KPMG (2021). AIPM and KPMG project management survey report. Available at: https://assets.kpmg/content/dam/kpmg/au/pdf/2021/kpmg-aipm-project-management-survey-2021.pdf

Almeida, F.; Simões, J.; Lopes, S. Exploring the Benefits of Combining DevOps and Agile. *Future Internet*, 14(63). 2-14. https://doi.org/10.3390/fi14020063

Anis, S. (2009), Diffusion of Innovations: Theoretical Perspectives and Future Prospects of Diffusion Studies. Available at: http://www.nmconf.uob.edu.bh/download/english_article/010.pdf

Arnuphaptrairong, T. (2011). Top Ten Lists of Software Project Risks: Evidence from the Literature Survey. Proceeding of the international multi conference of engineers and computer scientist, March 2011.

Appukkutty, K., Ammar, H. H. & Goseva-Popstojanova, K. (2005). Software Requirement Risk assessment using UML. In the Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications. 112-115.

Asadi, Z. (2015). An investigation of risk management strategies in projects. *Marketing and Branding Research*, 2(2015), 89-100.

Aven, T. (2016). Risk assessment and risk management: Review of recent advances on their foundation. *European Journal of Operational Research,* 253 (1), 1-13

Bannerman, P. (2008). Risk and risk management in software projects: A reassessment. *Journal of Systems and Software*, 81(12), 2118–2133.

Barki, H., Rivard, S., & Talbot, J. (1993). Toward an assessment of software development risk. *Journal of Management Information Systems*, 10(2), 203-225. http://dx.doi.org/10.1080/07421222.1993.11518006.

Barki, H., Rivard, S. & Talbot, J. (2001). An integrative contingency model of software project risk management. *Journal of Management Information Systems*, 17(4), 37-69.

Balzer R., Litoiu M., Müller H.A., Smith, D., Storey M., Tilley S.R. & Wong K. (2004). 4th International Workshop on Adoption-Centric Software Engineering (ACSE 2004). In *Proceedings of the 26th International Conference on Software Engineering (ICSE 2004)*, 748–774. Edinburgh, United Kingdom.

Bilal, M., Gani, A., Liaqat, M., Bashir, N. & Malik, N. (2020). Risk assessment across life cycle phases for small and medium software projects, *Journal of Engineering Science & Technology*, 15(1), 572–588.

Boehm, B. W. (1988). A spiral model of software development and enhancement. IEEE Computer, 21(5), 61-72. http:// dx.doi.org/10.1109/2.59.

Boehm, B. W. (1991). Software risk management: principles and practices. *IEEE Software*, 8(1), 32-41. http://dx.doi. org/10.1109/52.62930.

Boehm, B. & Turner, R. (2003). Using risk to balance agile and plan-driven methods. *IEEE Computer*, 36(6), 57-66. Available at: http://dx.doi.org/10.1109/MC.2003.1204376.

Boehm, B. W. (1979). R&D Trends and Defense Needs. In P. Wegner (Ed.), *Research Directions in Software Engineering, 44-86*. Cambridge, Massachusetts MIT Press.

Boehm, B. W. (1983). Seven Basic Principles of Software Engineering. *Journal of systems and software* 3(1), 3 – 24.

Boehm, B. W. (1989). *Risk Management Practices: The Six Basic Steps. Software Risk Management* 115-147. IEEE Computer Society Press, Piscataway, NJ, USA

Boehm, B. W. (1991). Software Risk Management: Principles and Practices. *IEEE Software 8(1),* 32-41.

Boehm, B. W. (1991). Software Risk Management: Principles and Practices. *IEEE Softw.,* 8(1), 32--41.

Brown, A., Fleetwood, S. & Roberts, J. M. (2002). The marriage of critical realism and Marxism: Happy, unhappy or on the rocks? In A. Brown, S. Fleetwood, & J. M. Roberts (Eds.), Critical realism and Marxism, 1–22. London; New York: Routledge.

Capers, J. (2006). Social and Technical Reasons for Software Project Failures. CROSSTALK. The *Journal of Defence Software Engineering*, 12(3), 4 – 9.

Carbone, T. & Tippett, D. (2004). Project Risk Management Using the Project Risk FMEA, *Engineering Management Journal*, 16(4), 28-35

Castsoftware, (2015). Report on Software Quality and Developer productivity. Available at: www.castsoftware.com/blog/software-quality-and-developer-productivity-together-improve-efficiency

CAST (2015). Software Quality and Developer Productivity: Together Improve Efficiency. *Software Intelligence Pulse.* Retrieved October, 2016 from www.castsoftware.com/blog/software-quality-and-developer-productivity-together-improve-efficiency

Cerpa, N. & Verner, J. M. (2009). Why Did Your Project Fail? *Communications of the ACM,* 52(2009) 130-134.

CHAOS Report (2020). Project success quick reference card based on CHAOS 2020: Beyond Infinity Overview. January'2021, QRC by Henny Portman

Charette, R. N. (2005). Why software fails. *IEEE Spectrum*, 42(9), 42-49. http://dx.doi.org/10.1109/ MSPEC.2005.1502528.

Charette, R. N. (1996). *Software Engineering Risk Analysis and Management.* McGraw-Hill Software Engineering Series.

Charette, R. N. (1989). *Software Engineering Risk Analysis and Management.* Intertext, New York.

Chin K.S., Wang Y. M., Poon G. K. K. & Yang, J. B. (2009). Failure mode and effects analysis using a group-based evidential reasoning approach. *Comput Oper Res,* 2009(36), 1768–79.

Cornford, S. L. & Feather, M. S. (2001). Dpd: A Tool for Lifecycle Risk Management. In the Proceeding of IEEE Aerospace Conference. 441-451.

Creswell, J. W. (2013). Qualitative inquiry and research design: Choosing among five approaches. Thousand Oaks, CA: Sage.

Creswell, J. W., & Plano Clark, V. L. (2011). Choosing a mixed methods design. *Designing and conducting mixed methods research*, 53-106.

Creswell, J. W., & Plano Clark, V. L. (2007). *Designing and conducting mixed methods research*, California: Sage Publications.

Cule, P., Schmidt, R., Lyytinen, K. & Keil, M. (2000). Strategies for heading off IS project failure. *Information Systems Management*, 17(2), 65-73. http://dx.doi.org/1 0.1201/1078/43191.17.2.20000301/31229.8.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS quarterly*, 319–340.

Davis, F. D. (1985). *A technology acceptance model for empirically testing new end-user information systems: Theory and results* (Doctoral dissertation, Massachusetts Institute of Technology).

Danermark B., Ekström, L., Jakobsen, L., Karlsson, J. C. (2002). *Explaining society: critical realism and the social sciences*. London and New York: Routledge; 2002.

Deephouse, C., Mukhopadhyay, T., Goldenson, D. R., Kellner, Marc I. (2005). Software Processes and Project Performance. *Journal of Management Information Systems (Winter 1995-96) 12(3),* 185-203.

Denzin, N. K. & Lincoln, Y. S. (2011). The Sage handbook of qualitative research. Thousand Oaks: Sage.

De Villers, M. R. (2005). Three approaches as pillar for interpretive Information Systems research: development research, action research and grounded theory. In j. Bishop & D. Kourie. *Research for a Changing World: Proceedings of SAICSIT 2005: 142-145. ACM International Conference Proceedings Series.*

de Wet, B. & Visser, J.K. (2013). An Evaluation of software project risk management in South Africa. *The South African journal of Industrial Engineering*, 24(1).

Dehlinger, J., & Lutz, R. (2004) Software Fault Tree Analysis for Product Lines. Proc. 8th IEEE International Symposium on High Assurance Systems Engineering (HASE'04), March 24-26, 2004, Tampa, Florida, 12- 21.

Department of Defense, USA, (1980). US MILITARY STANDARD, MIL-STD-1629A. Procedures for Performing a Failure Mode, Effect and Criticality Analysis,

Eisenhart, M. (1991). Conceptual frameworks for research circa 1991: Ideas from a cultural anthropologist; implications for mathematics education researchers. Paper presented at the Proceedings of the Thirteenth Annual Meeting North American Paper of the International Group for the Psychology of Mathematics Education, Blacksburg, Virginia, USA.

El-Gohary, H. (2012). Factors affecting E-Marketing adoption and implementation in tourism firms: An empirical investigation of Egyptian small tourism organizations. *Tourism Management, 33*(5), 1256-1269. http://dx.doi.org/10.1016/j.tourman.2011.10.013

Everith, B.S. (2002). *Cambridge Dictionary of Statistics.* 2nd Edition, CUP.

Flávio, R. S. & Sandro, C. (2008). FMEA and PMBOK applied to project risk management. *Journal of Information Systems and Technology Management*. 5(2), 347-364.

Feather, M., Cornford, S., Dunphy, J. & Hicks, K. (2002). A Quantitative Risk Model for Early Lifecycle Decision Making. In the 6th World Conference on Integrated Design and Process Technology.

Glass, R. L. (2006). The Standish Report: Does It Really Describe a Software Crisis? Reconsidering the relevancy of a frequently cited report on software project failures. *Communications of the ACM*, 49 (8), p. 15 -16.

Grant, C. & Osanloo, A. (2015). Understanding, selecting, and integrating a theoretical framework in dissertation research: creating the blueprint for your "house". *Administrative Issues Journal Education Practice and Research. 4(2), 12-26.*

Geneca (2011). Why a majority of business and IT teams anticipate their software development projects will fail. Retrieved September, 2015 form https://www.geneca.com/why-up-to-75- of-software-projects-will-fail/

Georgieva, K. (2010). Conducting FMEA over the Software Development Process. *ACM SIGSOFT Software Engineering Notes*, 35(3),1-3. Retrieved September, 2015 form http://doi.acm.org/10.1145/10.1145/1764810.1764819

Georgieva, K. (2007). The Incompleteness of the Risk Assessment Methods. Proceedings of the 11th IEEE International Workshop on Rapid System Prototyping (RSP 2007). IEEE Computer Society.

GitLab (2019). 2019 Global Developer Report: DevSecOps. Retrieved October, 2019 from https://about.gitlab.com/developer-survey/#pdf.

Giuseppe, A. (2017). Comparative research about high failure rate of IT projects and opportunities to improve. *PM World Journal*, 6(2), www.pmworldjournal.net

Grunske, L. & Han, J. (2008). A Comparative Study into Architecture-Based Safety Evaluation Methodologies using AADL's Error Annex and Failure propagation Models. 2008 11th IEEE High assurance Systems Engineering Symposium

Gupta, S., Gogate V. V. & Gupta, A. (2012). Software Failure Analysis in Requirement Phase. Proceedings of ISEC '12, Feb. 22-25, 2012 Kanpur, UP, India.

Gupta, D. (2008). Software risk assessment and estimation model. Proceedings of The International Conference on Computer Science and Information Technology, 963–967.

Gusmão, C. M. G. (2007). *Um modelo de processo de gestão de riscos para ambientes de múltiplos projetos de desenvolvimento de software* (Tese de doutorado), Centro de Informática, Universidade Federal de Pernambuco, Recife.

Harrison, H., Birks, M., Franklin, R., & Mills, J. (2017). Case Study Research: Foundations and Methodological Orientations. *Forum: Qualitative Social Research*, 18(1), Art. 19. Retrieved September 2018 from http://nbn-resolving.de/urn:nbn:de:0114-fqs1701195

Hansen M. O. (1989). Survey of Available Software-Safety Analysis Techniques. 1989 Proceedings of IEE Annual Reliability and Maintainability Symposium. Ford Aerospace Corporation; Colorado Springs. 46 – 49.

Hassan, A., Goseva-Popstojanova, K., Ammar, A. (2005). UML Based Severity Analysis Methodology. In Proceedings International Conference Reliability and Maintainability Symposium RAMS 2*005.*

Hassan, A., Goseva-Popstojanova, K. & Ammar, H. (2003). Methodology for Architecture Level Hazard Analysis, A Survey. ACS/IEEE Intl. Conference on Computer Systems and Applications (AICCSA 2003), Tunis, Tunisia, 68 – 70.

Hijazi, H., Alqrainy, S., Muaidi, H. & Khdour, T. (2014a). Risk Factors in Software Development Phases. *European Scientific Journal,* 10(3), 213 - 232

Hijazi, H., Alqrainy, S., Muaidi, H., & Khdour, T. (2014b). Identifying Causality relationship between Software projects Risk Factors. *International Journal of Software Engineering and Its Applications*, 8(2), 51-58.

Hopkin, P. (2018). Fundamentals of Risk Management: Understanding, Evaluating and Implementing Effective Risk Management. New York: Kogan Page Publishers.

Houston, D. (2000). A Software Project Simulation Model for Risk Management, Ph.D. Thesis, Arizona State University, Tempe AZ,

Huang, S.J. & Han, W. (2007). An Empirical Analysis of Risk Components and Performance on Software Projects. *Journal of Systems and Software*, 80(1), 42-50.

Huang, S., J. & Han, W., (2008). Exploring the relationship between software project duration and risk exposure: A cluster analysis. Information & Management, 45(3), 175-182. http://dx.doi.org/10.1016/j.im.2008.02.001.

International Business Machines (2003). IBM. Rational Software Corporation. *Rational unified process: best practices for software development teams* (Rational Software White Paper, TP026B, Rev 11/01: IBM). Cupertino, CA: Rational Software Corporation. Recuperado em 12 de setembro de 2009, de http://www. ibm.com/developerworks0/rational/library

International Organization for Standardization (2009). ISO 31.000:2009 - risk management - principles and guidelines on implementation. Genebra: ISO. Recuperado em 21 setembro de 2009, de http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43170.

International Organization for Standardization – ISO. (2003). *ISO 10.006:2003 – quality management systems - guidelines for quality management in projects*. Genebra: ISO.

International Organization for Standardization – ISO. (1999). *ISO/IEC 15.504-5:1999 – Information technology - Software process assessment — Part 5: An assessment model and indicator guidance*. Genebra: ISO.

International Standard Organization (1995). ISO/IEC 12207.

Iversen, J. H., Mathiassen, L., & Nielsen, P. A. (2004). Managing risk in software risk in software process improvement: An action approach. *Management Information System quarterly*. 28(3), 221-228

Jayson, P. V., Xiaoqing, L. & Irem Y. T. (2007). Risk assessment in early software design based on the software function-failure design method. 31st Annual International Computer Software and Applications Conference (COMPSAC 2007), Miami, 2007

Jiang, J., James, J. & Klein, G., (2000). Software Development Risks to Project Effectiveness, *Journal of Systems and Software 52(1),* 3.

Joan-Pastor, J. & Ramon-Roy, N. (2005). Implementing and improving the SEI Risk Management method in a university software project. *IEEE Latin America Transactions,* 3(1), 90 – 97

Jørgensen, M. & Moløkken-Østvold, K. (2006). How large are software cost overruns? A review of the 1994 CHAOS report. *Information and Software Technology*, 48 (4), 297-301.

Kanchanatanee, K., Suwanno, N. & Jarernvongrayab, A. (2014). Perceived Ease of Use and Perceived Compatibility on Intention to Use E-Marketing. *Journal of Management Research,* 6(3), 3-13.

Keil, M., Li, L., Mathiassen, L. & Zheng, G. (2008). The influence of checklists and roles on software practitioner risk perception and decision-making. *Journal of Systems and Software*, 81(6), 908-919. http://dx.doi.org/10.1016/j. jss.2007.07.035.

Keil, M., Wallace, L., Turk, D., Dixon-Randall, G. & Nulden, U. (2000). An investigation of risk perception and risk propensity on the decision to continue a software development project. Journal of Systems and Software, 53(2), 145-157. http://dx.doi.org/10.1016/S0164-1212(00)00010-8

Kerzner, H. (2003). *Project Management: A Systems Approach to Planning, Scheduling, and Controlling*, (8th ed). John Wiley & Sons, Hoboken.

Khaiyum, S. & Kumaraswamy, Y. (2014). An effective method for the identification of potential failure modes of a system by integrating FTA and FMEA. *Adv. Intell. Systs. Comput*. 24(8), 679–686.

KPMG (2019). The Future of Project Management: Global Outlook 2019. Australian Institute of Project management survey 2019. Retrieved December, 2019 from www.aipm.com.au/resources/reports/future-of-project-management-2019.

KPMG (2011). Most business experience business failure. Global IT Project Management Survey. KPMG, Australia.

Sarigiannidis, L. & Chatzoglou, P.D. (2011). Software Development Project Risk Management: A New Conceptual Framework. *Journal of Software Engineering and Applications*, 2011(4), 293-305 doi:10.4236/jsea.2011.45032.

Larman, C. & Basili (2003) Iterative and Incremental Development: A Brief History. *Computer*, 36, 47–56

Lehtinen, T. O. A., Mäntylä, M. V., Vanhanen, J., Itkonen, J., & Lassenius, C. (2014). Perceived Causes of Software Project Failures – An Analysis of their Relationships. *Information and Software Technology*. Retrieved January, 2017 from http://www.journals.elsevier.com/information-and-software-technology/

Lehtinen, T. O. A., & Mäntylä, M. V. (2011). What are problem causes of software projects? Data of root cause analysis at four software companies, ESEM '11 Proc. of the 2011 International Symposium on Empirical Software Engineering and Measurement, 388-391.

Leite, L., Rocha, C., Kon, F., Milojicic, D.; Meirelles, P. A. (2019). Survey of DevOps Concepts and Challenges. *ACM Comp. Surv*., 52, 127–162.

Leveson, N. (1995). *Safeware: System Safety and Computers*. Addison-Wesley, 1995.

Liliana, D. & Eila, N. (2002). A survey on software architecture analysis methods. IEEE Transactions on software engineering, 28(7), 638-653.

Liu, H., Chen, Y; You, J. & Li, H. (2016). Risk evaluation in failure mode and effects analysis using fuzzy digraph and matrix approach*. Journal of Intelligent manufacturing.*27(3), 805-816.

Livari, J. R. (1996). Analysing information systems development: A comparison and analysis of eight IS development approaches. *Information Systems* 21 (7), 551-575.

Lutz, R. & Nikora, A. (2007). Failure Assessment. Jet Propulsion Laboratory/Caltech and Iowa State University publication.

Lutz, R. & Shaw, H. (1999). Applying Adaptive Safety Analysis Technique, In Proceedings of 10th International Symposium on Software Reliability Engineering, 42-49.

Lutz, R. & Woodhouse, R. (1997). Requirements Analysis Using Forward and Backward Search, *Annals of Software Engineering*. 3(1), 459-475.

Luz, W. P., Pinto, G & Bonifácio, R. (2018). Building a collaborative culture: A grounded theory of well succeeded DevOps adoption in practice. In Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Oulu, Finland, McFarlan, F. W. (1981). Portfolio Approach to Information Systems. Harvard Business Review 59(5), 142-150.

McKinsey & Company (2012). Delivering large-scale IT projects on time, on budget, and on value. A survey report on large scale IT projects. Retrieved September 2016 from www.mckinsey.com/business-functions/mckinsey-digital/our-insights/delivering-large-scale-it-projects-on-time-on-budget-and-on-value

Menezes, J., Gusmão, C. & Moura, H. (2018). Risk factors in software development projects: a systematic literature review. *Software Qual. Journal,* 27(2009)**,** 1149–1174, https://doi.org/10.1007/s11219-018-9427-5.

Meyer, C. B. (1997). A case in case study methodology. *Field Methods*, 13(4), 329-352.

Microsoft. (2002). *Microsoft Solutions Framework: MSF Risk Management Discipline v. 1.1*. Recuperado em 18 de setembro de 2009, de http://www.microsoft.com/msf

Miles, M. B. & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook* (2nd ed.). Thousand Oaks, California: Sage Publications.

MIL-STD-882D. Department of Defence, United States of America. (2000). Standard Practice for System Safety, Military Standard.

MIL-STD-1929A (1980), Military Standard- Procedures for performing a failure mode effects and criticality analysis, 2, US Department of Defense, Washington DC, USA.

Mitrabinda, R. & Durga Prasad, M. (2011). A Novel Methodology for Software Risk Assessment at Architectural Level Using UML Diagrams. *SETLabs Briefings,* 9(4), 41-60.

Mnkandla, E. (2012). Assessing a Methodology's Project Risk Management Competence. *Journal of Contemporary Management*. 9, 279-299.

Momani, A. M. & Jamous, M. M. (2017). The Evolution of Technology Acceptance Theories. *International Journal of Contemporary Computer Research (IJCCR)*, 1(1), 51-58.

Na-Allah, D. M. (2019). Understanding Teachers' Beliefs and Experiences of ICT Integration in Five South African Paperless High Schools. PhD thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy to the Faculty of Humanities, University of the Witwatersrand, Johannesburg, South Africa July, 2019.

Na-Allah, D. M. (2018). Understanding the factors that impact on the integration of new technologies (ICTs) in South African paperless classrooms. A paper presented at the European Conference on Education, ECE2018, Jurys Inn Brighton Waterfront Hotel, Brighton & Hove, UK, 29 June–1 July 2018.

NASA (2004). *Software Safety Standard, NASA-STD 8719-13B*, July, 2004.

Ngulube, P., Mathipa, E. R. & Gumbo, M. T. (2015). Theoretical and conceptual frameworks in the social and management sciences. *Addressing Research Challenges: Making Headway for Developing Researchers*, 2(5), 43–66.

Neuman, W. L. (2002). Social Research Methods. *Qualitative and Quantitative approaches (4th ed.). Boston: Allyn and Bacon.*

Neves S. M. Silva C. E. S. (2016). Risk management applied to software development projects in incubated technology-based companies: Literature review, classification and analysis. *Gestão & Produção*, 23(4), 788–814.

Neves, S. M., Silva, C. E. S., Salomon, V. A. P., Silva, A. F. & Sotomonte, B. E. P. (2014). Risk management in software projects through Knowledge Management techniques: cases in Brazilian Incubated Technology-Based Firms. *International Journal of Project Management*, 32(1), 125-138.

Nidumolu, S., (1995). The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable. *Information Systems Research,* 191-219.

Nogueira, J. C. (2000). Formal model for risk assessment in software projects. A PhD Thesis submitted in partial fulfilment of the requirements for the degree of doctor in philosophy in software engineering to naval postgraduate school Monterey, California, September, 2000.

Nogueira, J.C., Luqi, & Berzins, V. (2000). A Formal Risk Assessment Model for Software Evolution. SEKE 2000. Chicago, July 2000.

Ozarin, N. & Siracusa, M. (2003). A Process for Failure Modes and Effects Analysis of Computer Software, In Proceedings of Annual Reliability and Maintainability Symposium, 365-370.

Pasha, M., Qaiser, G. & Pasha, U. (2018). Critical Analysis of Software Risk Management Techniques in Large Scale Systems. *IEEE Access,* 6(2018), 12412 - 12424*,* DOI 10.1109/ACCESS.2018.2805862A.

Petersen, K. & Wohlin, C. A. (2009). Comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. Journal of System Software, 82, 1479–1490.

Pinder, J., Wilkinson, S. J. & Demack, S., (2003). A Method for Evaluating Workplace Utility. *Property Management,* 21(4),218 – 229.

Pressman, R. (1997). *Software Engineering: A practitioners' approach, 4th edition,* McGraw-Hill Higher Education.

Project Management Institute PMI. (2017). A Guide to the Project Management Body of Knowledge (PMBOK® Guide) (sixth edition.). Project Management Institute, EUA.

Project Management Institute – PMI, (2008). *A guide to the Project Management Body of Knowledge (PMBOK@Guide)*. Pennsylvania: PMI.

Project Management Institute (2004). A guide to the project management body of knowledge (PMBOK® Guide). Project Management Institute, Pennsylvania.

Project Management Institute, (2000). A guide to the project management body of knowledge (PMBOK® Guide). Project Management Institute, Pennsylvania.

Rajapakse, R. N., Zahedi, M., Babar, M.A., Shen, H.(2022). Challenges and solutions when adopting DevSecOps: A systematic review. *Inf. Soft Tech*., 141, 106700.

Riegler, A. (2012). Constructivism, In L. L'Abate (Ed.), *Paradigms in theory Construction*, 12(3), 235–255.

Reifer, D. J. (1979). Software Failure Modes and Effects Analysis, *IEEE Transactions on Software Engineering*, 247-249,

Ridder, H. G. (2017). The theory contribution of case study research designs. *Bus Res*10,281–305. https://doi.org/10.1007/s40685-017-0045-z

Ridder, H.G. (2016). *Case study research. Approaches, methods, contribution to theory. Sozialwissenschaftliche Forschungsmethoden*, vol. 12. München/Mering: Rainer Hampp Verlag.

Robinson, L., (2009), A Summary of Diffusion of Innovations. Available at: http://www.enablingchange.com.au/Summary_Diffusion_Theory.pdf

Rogers, E. M. (2003). *Diffusion of innovations* (5th ed.). Free Press.

Rogers, E. M. (1995). Diffusion of Innovations: Modifications of a model for telecommunications. *Die Diffusion von Innovationen in der Telekommunikation*, *17*, 25–38.

Ropponen, J. & Lyytinen, K., (2000). Components of software development risk: How to address them? A project manager survey. *IEEE Transactions on Software Engineering*, 26 (2), 98–112.

Ropponen, J. & Lyytinen, K., (1997). Can software risk management improve system development: an exploratory study? *European Journal of Information Systems*, 6(1), 41-50.

Ropponen, J. & Lyytinen, K. (2000). Components of software development risk: how to address them? A project manager survey. *IEEE Transactions on Software Engineering*, 26(2), 98-112. http://dx.doi. org/10.1109/32.841112.

Sarigiannidis, L. & Chatzoglou, P. D. (2011). Software Development Project Risk Management: A New Conceptual Framework. *Journal of Software Engineering and Applications*, 2011(4), 293-305 doi:10.4236/jsea.2011.45032.

Saunders, M., Lewis, P. & Thornhill, A. (2009). *Research Methods for Business Students.* Edinburgh, UK: Pearson Education Limited.

Sayer, A. (1992). Methods in social science: A realist approach. London: Routledge.

Scapens, R. W. (1990). Researching management accounting practice: the role of case study methods. *The British Accounting Review,* 22(3)**,** 259-281.

Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001). Identifying Software Project Risks: An International Delphi Study, *Journal of Management Information Systems*, 17(4), 5-36.

Schunk, D. (2000). *Learning Theories*. *An Educational Perspective*. (3rd ed.). New Jersey: Prentice-Hall.

Scotland, J. (2012). Exploring the philosophical underpinnings of research: Relating ontology and epistemology to the methodology and methods of the scientific, interpretive, and critical research paradigms. *English Language Teaching*, *5*(9), 9.

Selby, R. (2007). Software Engineering Barry Boehm's Lifetime Contributions to Software Development, Management, and Research. John Wiley, Chichester.

Shahzad, B., Ullah, I., & Khan, N. (2009). Software risk identification and mitigation in incremental model. Proceedings of the International Conference on Information and Multimedia Technology, 366–370.

Shore, J. & Warden, S. (2021). The Art of Agile Development; O'Reilly Media: Newton, MA, USA.

Softrel.com. Available at softrel.com

SOFTEX. (2006). Associação para Promoção da Excelência do Software Brasileiro. –*Melhoria de processo do software brasileiro, versão 1.1*. Brasília: Softex. Recuperado em 19 de setembro de 2009, de www.softex.br

Software Engineering Institute. (2006). CMMI® for development. staged representation, version 1.2, technical report (06tr008). Pittsburgh: Software Engineering Institute, Carnegie Mellon University. http://www.sei.cmu.edu/reports/06tr008.pdf

Software Engineering Institute (1996). Software Risk Management. Technical Report CMU/SEI-96-TR012.June, 1996.

Sommerville, I. (Ed.). (2006). *Software Engineering*. Addison Wesley.

Souza, S. F. & Cabral, S. (2008). FMEA and PMBOK applied to project risk management. Journal of Information System and Technology Management, 5(2), 347-364

Stahl, D., Mårtensson, T. & Bosch, J. (2017). Continuous Practices and DevOps: Beyond the Buzz, What Does it All Mean? In Proceedings of the 43rd Euro micro Conference on Software Engineering and Advanced Applications (SEAA), Vienna, Austria

Standards Australia & Standards New Zealand. (2004). AS/NZS 4360:2004: risk management. Sydney: Standards Australia; Standards New Zealand.

Standish Group Report (2015). Chaos Report. The Standish Group International.

Standish Group International, (1999). Chaos: A Recipe for Success. The Standish Group International.

Svitla system (2022). Modern methodologies of software development. Available at: https://svitla.com/blog/modern-methodologies-of-software-development

Taroun, A. (2012). Decision Support System (DSS) for Construction Project Risk Analysis and Evaluation via Evidential Reasoning (ER), A PhD Thesis submitted to the University of Manchester – UK for the degree of Doctor of Philosophy in the Faculty of Humanities September, 2000.

Tay, K. M., & Lim, C.P. (2006). Fuzzy FMEA with a Guided Rules Reduction System for Prioritization of Failures. *International Journal of Quality & Reliability Management*, 23(5), 1047-1066.

The Standish Group. (2000). *Extreme chaos*. Recuperado em 13 de julho de 2009, de www.standishgroup.com/ sample_research/PDFpages/extreme_chaos.pdf

Thomas, S. (2008). A study on software development project risk, risk management, project outcomes and their inter-relationship. PhD thesis submitted to the school of management studies Cochin university of Science and Technology for the award of the degree of doctor of philosophy 2008.

Tolfo, C., Wazlawick, R. S., Ferreira, M. G & Forcellini, F. A. (2021). Agile methods and organizational culture: Reflections about cultural levels. *Journal of Software Maintenance*. 23, 423–441.

Trzeciak, M. (2021). Sustainable Risk Management in IT Enterprises. *Risks* 9(135), 2-20. Available at: https://doi.org/10.3390/risks9070135

Ulum, O. G. (2016). Epistemology in qualitative educational research: a review of published articles. Journal of Education and Humanities: Theory and Practice 7(13), 19-28.

USA Department of Defence (1980) Military Standard, Procedure for Performing a Failure Mode, Effects and Criticality Analysis. Department of Defence, Washington DC, 27-33.

Vahidnia, S., Özgür, O. T. & Askerzadeet, I. N. (2016). An evaluation study of general software project risk based on software practitioners' experiences. *International Journal of Computer Science & Information Technology (IJCSIT),* 8(6). 221-229.

Vucovich, J.P., Stone, R.B., Liu, X. & Tumer, I.Y. (2007). Risk Assessment in Early Software Design Based on the Software Function-Failure Design Method. In COMPSAC '07: Proceedings of the 31st Annual International Computer Software and Applications Conference. IEEE Computer Society.

Wallace, L. (1999). *The development of an instrument to measure software project risk. PhD t*hesis (January 1999), Georgia University.

Wallace, L. & Keil, M. (2000). Software project risks and their effect on outcomes. *Communications of the ACM 47(4),* 68 – 73.

Wallace, L., Keil, M, & Rai, A. (2004). Understanding software project risk: a cluster analysis. *Information & Management*, 42(1), 115-125.

Weber, R. (2004). Editor's Comments. The rhetoric of positivism versus interpretivism: A personal view. *MIS Quarterly,* 28(1), 3-4.

Wellingtone, (2008). The State of Project Management Survey 2018, UK: Wellingtone PPM. https://www.wellingtone.co.uk/wp-content/uploads/2018/05/The-State-of-Project-Management-Survey-2018-FINAL.pdf

Wikipedia, (2015). Wikipedia Encyclopdia [Online]. "Failure causes" Available at: HYPERLINK, Retrieved October, 2015 from http://en.wikipedia.org/wiki/Failure_Mode

Yin, R.K. (2014). *Case study research. Design and methods*, 5th ed. London, Thousand Oaks: Sage Publications.

Yin, R. K. (2009). *Case study research: design and methods* (4 ed.). Thousand Oaks: Sage Publications, Inc.

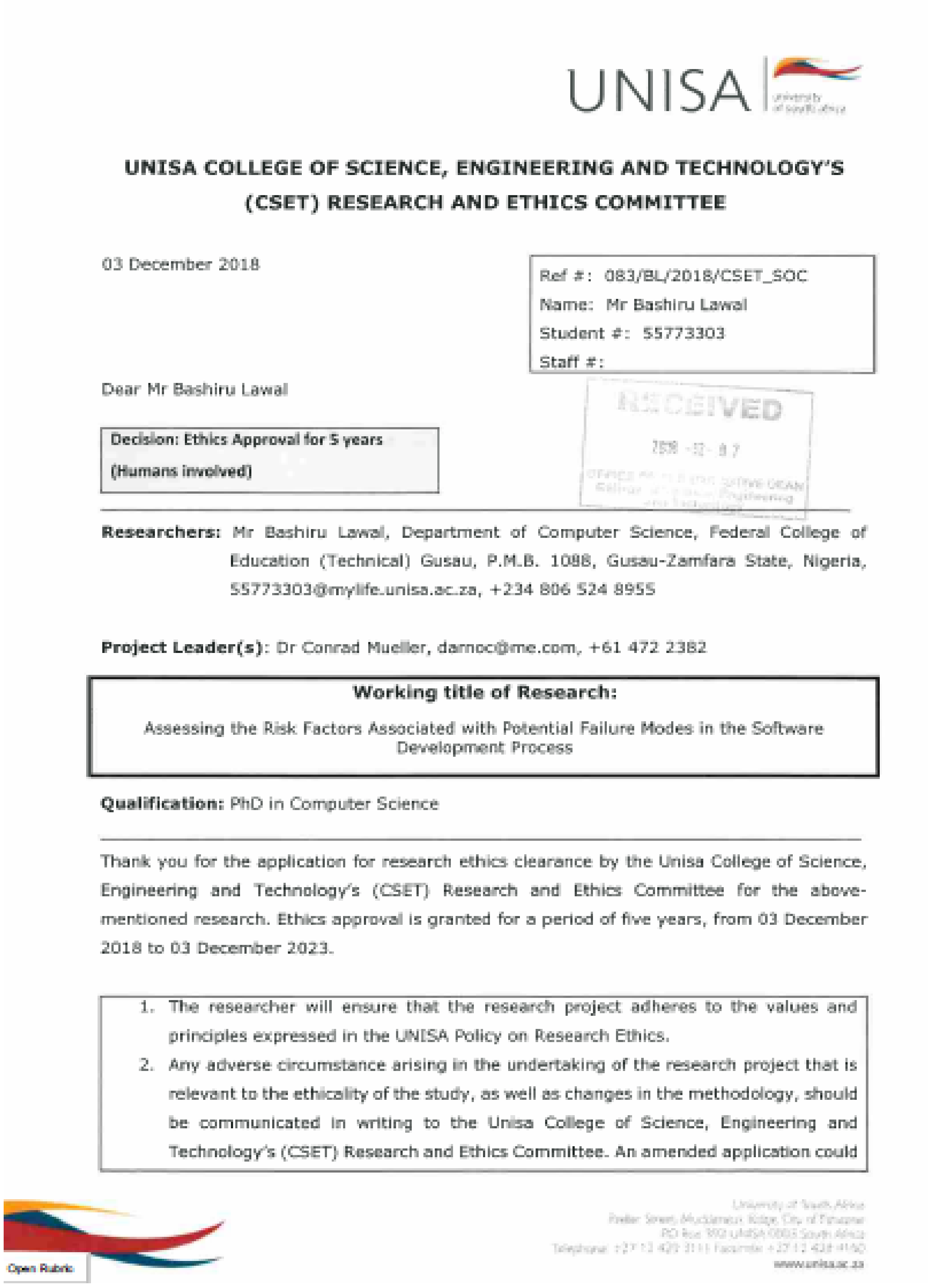
Yusuf, O. M. (2018). The ups and downs of software project managements. West African Journal of Project Management, 4(6), 125-133.

Zardari, S. (2009). Software risk management. In Proceedings of the 2009 of the International Conference on Information Management and Engineering (ICIME) 375-379.

Zhao, H., You, J., & Liu, H. (2017). Failure mode and effect analysis using MULTIMOORA method with continuous weighted entropy under interval-valued intuitionistic fuzzy environment. Soft Computing, 21(2), 5355-5367.

APPENDIX

APPENDIX 1A: UNISA ETHICS CLEARANCE CERTIFICATE



UNISA university of south africa

UNISA COLLEGE OF SCIENCE, ENGINEERING AND TECHNOLOGY'S (CSET) RESEARCH AND ETHICS COMMITTEE

03 December 2018

Ref #: 083/BL/2018/CSET_SOC
Name: Mr Bashiru Lawal
Student #: 55773303
Staff #:

RECEIVED
2018 -12- 07
OFFICE OF THE EXECUTIVE DEAN College of Science, Engineering and Technology

Dear Mr Bashiru Lawal

Decision: Ethics Approval for 5 years (Humans involved)

Researchers: Mr Bashiru Lawal, Department of Computer Science, Federal College of Education (Technical) Gusau, P.M.B. 1088, Gusau-Zamfara State, Nigeria, 55773303@mylife.unisa.ac.za, +234 806 524 8955

Project Leader(s): Dr Conrad Mueller, darnoc@me.com, +61 472 2382

Working title of Research:

Assessing the Risk Factors Associated with Potential Failure Modes in the Software Development Process

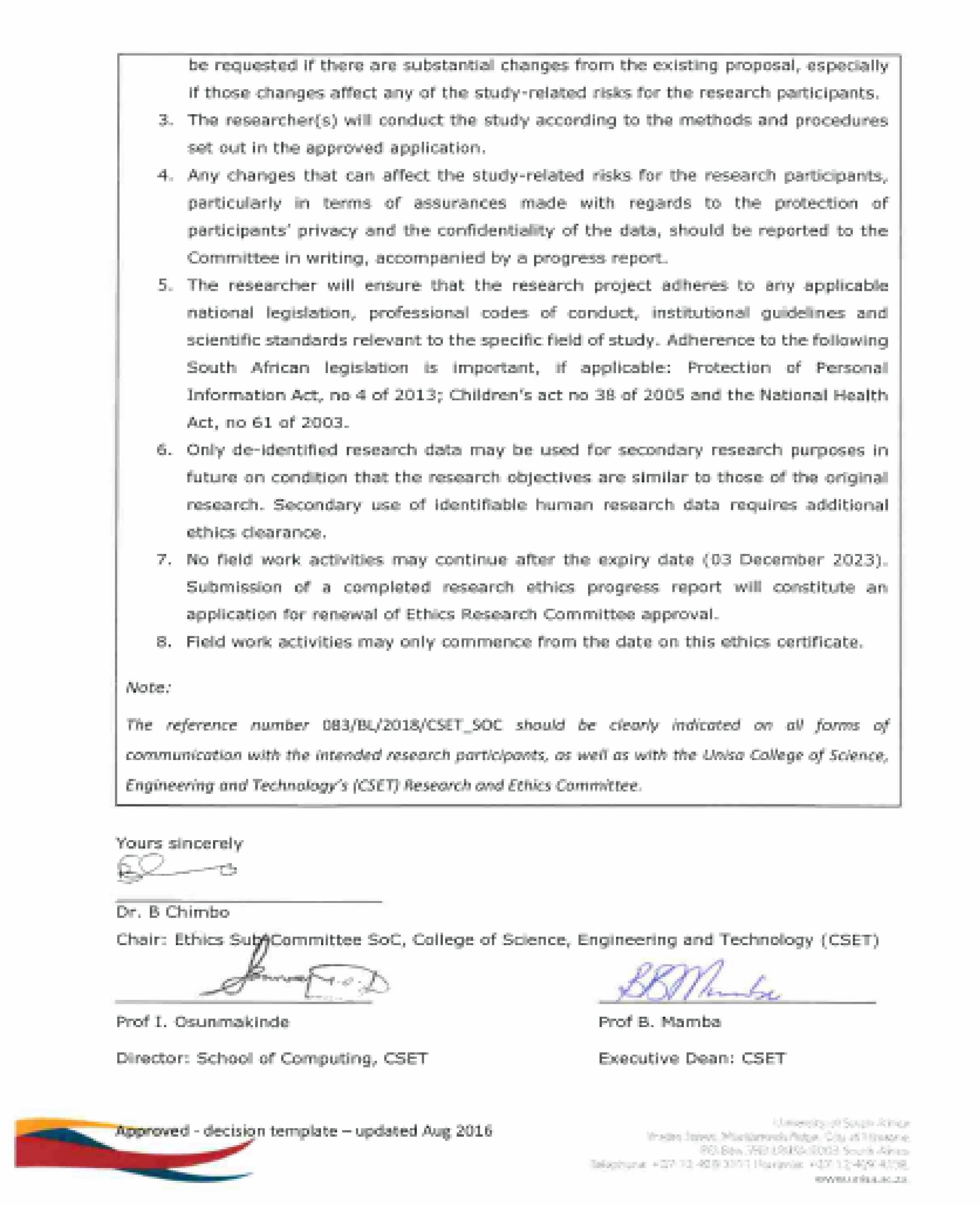
Qualification: PhD in Computer Science

Thank you for the application for research ethics clearance by the Unisa College of Science, Engineering and Technology's (CSET) Research and Ethics Committee for the above-mentioned research. Ethics approval is granted for a period of five years, from 03 December 2018 to 03 December 2023.

1. The researcher will ensure that the research project adheres to the values and principles expressed in the UNISA Policy on Research Ethics.
2. Any adverse circumstance arising in the undertaking of the research project that is relevant to the ethicality of the study, as well as changes in the methodology, should be communicated in writing to the Unisa College of Science, Engineering and Technology's (CSET) Research and Ethics Committee. An amended application could

Open Rubric

University of South Africa
Preller Street, Muckleneuk Ridge, City of Tshwane
PO Box 392 UNISA 0003 South Africa
Telephone: +27 12 429 3111 Facsimile: +27 12 429 4150
www.unisa.ac.za



be requested if there are substantial changes from the existing proposal, especially if those changes affect any of the study-related risks for the research participants.

3. The researcher(s) will conduct the study according to the methods and procedures set out in the approved application.
4. Any changes that can affect the study-related risks for the research participants, particularly in terms of assurances made with regards to the protection of participants' privacy and the confidentiality of the data, should be reported to the Committee in writing, accompanied by a progress report.
5. The researcher will ensure that the research project adheres to any applicable national legislation, professional codes of conduct, institutional guidelines and scientific standards relevant to the specific field of study. Adherence to the following South African legislation is important, if applicable: Protection of Personal Information Act, no 4 of 2013; Children's act no 38 of 2005 and the National Health Act, no 61 of 2003.
6. Only de-identified research data may be used for secondary research purposes in future on condition that the research objectives are similar to those of the original research. Secondary use of identifiable human research data requires additional ethics clearance.
7. No field work activities may continue after the expiry date (03 December 2023). Submission of a completed research ethics progress report will constitute an application for renewal of Ethics Research Committee approval.
8. Field work activities may only commence from the date on this ethics certificate.

Note:

The reference number 083/BL/2018/CSET_SOC should be clearly indicated on all forms of communication with the intended research participants, as well as with the Unisa College of Science, Engineering and Technology's (CSET) Research and Ethics Committee.

Yours sincerely

[illegible]

Dr. B Chimbo

Chair: Ethics SubCommittee SoC, College of Science, Engineering and Technology (CSET)

[illegible]

Prof I. Osunmakinde

Director: School of Computing, CSET

[illegible]

Prof B. Mamba

Executive Dean: CSET

Approved - decision template – updated Aug 2016

University of South Africa
Preller Street, Muckleneuk Ridge, City of Tshwane
PO Box 392 UNISA 0003 South Africa
Telephone: +27 12 429 3111 Facsimile: +27 12 429 4150
www.unisa.ac.za

APPENDIX 1B: CONSENT FORM

I, (First name and last name): **………………………………………………………**

This part to be completed by software practitioners:

Area of specialization: **…………………………………………..**

Address: **……………………………………………………………………..**

..……......

This part to be completed by students:

Registration Number: **…………………………………………..**

State that I have not been under any pressure to participate in this survey study, and have voluntarily participated in it.

I realise that findings of the study will be used for research purposes and that findings will be published.

I also consented to the terms and conditions that all my personal profile will be handled with confidentiality.

Signed: ……………………………………………………..

Date: ……………………………………………………………

APPENDIX 1C: ETHICAL APPROVAL

The exploratory case study (Field study 1); the focus group study (Field study 2) and the FMEA process for assessing risk (Field study 3) were conducted at a conference centre of a software industry in Abuja – Nigeria and the studies carry the approval of the Head of Research and the CEO of the software industry in Nigeria where the studies were conducted. Also, the FMEA interactive study (Field study 5) meets the approval of the Director Undergraduate School and the Head of Department Computer Science to conduct the study with undergraduate SE students as participants. However, terms and conditions that: no explicit reference should be made to the subject group, industry participants' name, school's name, faculty of the university where the studies have been carried out and that all information concerning the participants should remain confidential.

APPENDIX 2A: THE FOCUS GROUP DISCUSSION PROTOCOL

Introduction

This document describes the organization of the programme of events of the focus group discussion sessions. It contains two sections. Section I: The focus Group Demographic Detail Questionnaire, which asks about the status of the prospective participants of the focus group. Section II: The Focus Group Guide, which describes sets of protocols/activities to be observed during the focus group plenary. Also contained in the section II is a list of the topics to be covered/ kinds of questions to be asked that will stimulate information-rich cases of the subject matter.

SECTION I: FOCUS GROUPDEMOGRAPHIC DETAILS QUESTIONNAIRE

Please answer the following questions in the spaces provided, circle or tick the most appropriate options.

1. Age:……………………………………………………………………………

2. Are you: (please tick as necessary) □ Male □ Female

3. What is your professional background?

□ Programmer
□ Software Engineer
□ System Analyst
□ Software manager
□ Domain Experts
□ Other: (please describe) ________________________________

4. How many software development projects have you involved in the last months (approximately)? _______________

5. How many years of experience have you had in this current job?

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

6. Experience in Software Development (optional):

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

Thank you for taking the time to complete this questionnaire.

SECTION II: FOCUS GROUP DISCUSSION GUIDE

Facilitator's welcome address (The Principal Researcher's Address):

Welcome and thank you for volunteering to take part in this focus group discussion. You have been asked to participate, as your point of view is important. I realise you are busy and I appreciate your time. Thanks for taking the time to join us to talk about the topic: '**Assessing the Risk Factors Associated with the Potential Failure Modes in the Software Development Process'** My name is Bashiru Lawal, I am a PhD candidate in the department of Information Systems from the famous University of South Africa and assisting me is Stephen Omodara. We're both from academia. .

Introduction:

This focus group discussion is designed to document your current thoughts on two issues: first – the possibility of using the FMEA method for the conduct of software project risk management and secondly, your feelings about the manner which a failure can occur in each stage of the SDLC during the development process of software project of any kind. Software development process or the SDLC is a structure imposed on the development of a software system. This structure is in stages and the stages vary depending on the type of software model process to be employed in the development of software project. For the purpose of our discussion, we will focus on the five stages that are common to most software development projects. These stages are: Planning, Requirements Analysis and Definition, Design, Implementation and Coding, Installation and Maintenance Stage. The 5 stages are modelled from the prescribed International Standard ISO/IEC 12207 (ISO/IEC 1995) by (Hijazi, et al., 2014).

In furtherance to the above, the discussion will be of two sessions. In the first session, the researcher will engage a team of selected software development experts in a focus group discussion to brainstorm and explore the possibility of integrating the FMEA methodology in software project management and review its associated rating tables. To explain further, the discussion will focus on a risk management methodology and use four well-known sets of concepts: PMBOK, the Software Engineering Institute (SEI)'s software project risk management model, the Boehm's risk management model and the FMEA as a multiple case study for comparative analysis. The discussion will use both the secondary research and formal qualitative research approach to establish the standard features of FMEA for project risk management. Secondly, it will use a collective/multiple case study creating theory on the procedural requirement' adherence of FMEA and the three standard project risk management models (PMBOK, SEI-CRM, and Boehm) are related. Thirdly, it will use the later approach to review the existing rating tables of (Occurrence, Severity and Detection) as used in traditional FMEA process for project management.

General Information

There are no wrong answers but rather differing points of view. Please feel free to share your point of view even if it differs from what others have said. Keep in mind that we're just as interested in negative comments as positive comments, and at times the negative comments are the most helpful.

Once again you were invited because you have participated in some sort of software development projects involving one of these studied stages of the SDLC, so you are familiar with software project development process and the stages of the SDLC.

You have probably noticed the microphone. We are tape recording the session because we don't want to miss any of your comments. People often say very helpful things in these discussions and we can't write fast enough to get them all down. We will be on a first name basis tonight, and we won't use any names in our reports. You may be assured of complete confidentiality. Each session of the focus group discussion will take no more than three hours of a two-working day per week for the period of three weeks.

Anonymity:

Despite being taped, I will like to assure you that the discussion will be anonymous. The tapes will be kept safely in a locked facility until they are transcribed word for word, then they will be destroyed after about five years of concluding this research. The transcribed notes of the focus group will contain no information that would allow individual subjects to be linked to specific statements. You should try to answer and comment as accurately and truthfully as possible. I and the other focus group participants would appreciate it if you would refrain from discussing the comments of other group members outside the focus group. If there are any questions or discussions that you do not wish to answer or participate in, you do not have to do so; however please try to answer and be as involved as possible.

Ground rules

- The most important rule is that only one person speaks at a time. There may be a temptation to jump in when someone is talking but please wait until they have finished.
- There are no right or wrong answers
- You do not have to speak in any particular order
- When you do have something to say, please do so. There are many of you in the group and it is important that I obtain the views of each of you
- You do not have to agree with the views of other people in the group
- Does anyone have any questions? (Answers).

We've placed name cards on the table in front of you to help us remember each other's names. Let's find out some more about each other by going around the table. Tell us your name, your professional background and your organization.

Overall, the first focus group session will be conducted with the aim of clarifying the following main issues:

- What standard procedures are required for the conduct of FMEA that will make the technique suitable for project risk management tool?
- How well do the FMEA procedures satisfy the procedural requirement of prominent standard project risk management models (PMBOK, SEI-CRM and Boehm)?
- How feasible are the application of the existing tables of (Occurrence, Severity and Detection) as used in traditional FMEA in software project management?

During the second focus group session, the researcher will examine each of the stages/phases of the SDLC listed earlier, by brainstorming the processes to identify the important probable failure modes that can exist therein, identify their likely causes and suggest suitable detection method(s).

A failure mode is a characterization of the way a product or process fails. Rather than the simple description of symptoms that many product users or process participants might use, the term failure mode refers to a rather complete description, including the pre-conditions under which failure occurs, how the thing was being used, proximate and ultimate/final causes (if known), and any subsidiary or resulting failures that result. Over time, as more is understood about a failure, the failure mode evolves from a description of symptoms and outcomes (that is, effects) to a systematic and relatively abstract model of how, when, and why the failure comes about (that is, causes).

During this session, we will carefully explore and identify all the probable failure modes, examine the likely causes and suggest a suitable detection method(s) for the identified failure modes. The discussion will disclose whether the identified failures are reproducible or transient, and hypothesize what combination of conditions and sequence of events led to failure is part of the process of fixing design flaws. In the end, the discussion will proffer ways for improving future iterations in all the studied stages of the SDP.

Other guiding questions

- What are other factors you may consider as important risky failure modes from the stage under review?
- What do you think could be the root cause of the failure mode you suggested?

- When thinking back to what other practitioners and researchers say on effects of these risk factors on the success of software projects what are your own views on the effects of these failure modes, should it occur on a project?

What response plans were suggested by practitioners and or researchers to prevent the failure from occurring?

Are there other recommended actions that could have been introduced to minimize the effect of the failure?

Note: these set of questions would be repeated for the other four Stages of the SDLC (i.e. Requirement Analysis, Design; Implementation and Coding, Installation and Maintenance phase).

Concluding question

- ❖ Of all the things we’ve discussed today, what would you say are the most important issues you would like to express about the failure modes existence/occurrence in any phase of the SDLC?

Conclusion

- ❖ We have come to the end of all sessions of our focus group discussion. Thank you for participation.
- ❖ This has been a very successful discussion
- ❖ Your opinions will be a valuable asset to the study
- ❖ We hope you have found the discussion interesting
- ❖ Your opinions will be a valuable asset to the study
- ❖ Thank you and God bless

APPENDIX 3A: PROBABLE FAILURE MODES IN THE STUDIED STAGES OF THE SOFTWARE DEVELOPMENT PROCESS: THEIR LIKELY CAUSE(S) AND DETECTION METHODS

A. Planning Stage

S/N	Probable Failure Mode	Likely Cause(s)	Detection Method
1	Inappropriate chosen software development framework/methodology /models	• Can occur from human engineering factor, such an insufficient information on the proposed system (Abubakar & Lawal 2020)	• Persistence system failure
2	Communication gap amongst the development team	May occur from factors related to development process, such as: • Poor communication channel established amongst team members • If the documentation and results obtained by the chosen software model does not reach the concerned management, development or maintenance team (Abubakar & Lawal 2020)	• Team members cannot account for the current development updates
3.	Ambiguous or unclear specification	• Use of natural language for all project component specifications	• Misinterpretations of project specifications by developers
4.	Insufficient software project tools and resources	• May occur from f actors related to development process e.g., when project resources are not properly identified or • If the information and documentation about the selected methods and tools for the project are not enough to complete the required tasks (Abubakar & Lawal 2020)	• When there are always overruns of resources (Abubakar & Lawal 2020)
5.	Unrealistic project goals and scopes	• Caused by factors related to development process e.g., Project goals not clearly defined • Developmental resources are insufficient limitations of an inappropriate chosen framework or programming tools (Abubakar & Lawal 2020)	• When project managers find it difficult to specifically determine and state what the project is supposed to do
6.	Unrealistic budget	• Human and software engineering related factors, such as: standard cost framework not implemented in the budget; or when the estimated cost for the project exceeds the available budget (Abubakar & Lawal 2020)	• Project cost exceeds the budget
6.	Ambiguous project hierarchy model	• Dysfunctional hierarchy model for the system users	• The project hierarchy model not functional (Abubakar & Lawal 2020)
7.	Unrealistic project scheduling	• Can be caused by Human and/ or software engineering related factors. For example, if standard scheduling models not applied	• Project delay

8.	Inexperience project team members	• Human related factors, e.g., when qualified professionals are not selected for the project (Abubakar & Lawal 2020)	• Decisions made are always wrong
9	Poor risk management strategies	• Human related factors, e.g. if experience and qualified risk management team not appointed	• Project failure occurs
10.	Deviance of system security integration	• Hardware and software limited resources. For example, when no security application such as firewalls integrated with the system	• System vulnerable to attacks

B. Requirement Specification and Analysis Stage

S/N	Probable Failure Mode	Likely Cause	Detection Method
1	Inappropriate chosen requirement gathering technique	• Failure mar occur from human engineering factor: Standard requirement collection method not applied (Abubakar & Lawal 2020)	• Selected requirement gathering technique not working
2	Ambiguous or unclear requirement	• Use of natural language to specify requirements	• Misinterpretations of requirements amongst developers
3.	Contradictions and confusion in domain specific terminologies	• May result from development process: e.g., when domain-specific terminologies not clearly defined end-users	• Misinterpretations of domain terminologies by end-users (Abubakar & Lawal 2020)
4.	Contradictions and confusion in defining requirements in Natural Language (NL)	• May occur from software related factor: use natural language instead of formal representations of requirements	• Misinterpretations of requirements amongst developers
5.	Errors in human factor engineering specifications	• Human engineering factor: if operators are not properly trained	• Persistent errors associated with human operations
6.	Inaccurate requirement	• Human and software engineering factors: when requirement engineering principles not applied	• Vital requirements are not captured
7.	Unrealistic requirement	Failure can come from human and software engineering factors: when requirement engineering principles not applied (Abubakar & Lawal 2020)	• Vital requirements are not captured (Abubakar & Lawal 2020)
8.	Inconsistent requirement	Human and software engineering factors: when requirement engineering principles not applied	• When there is confusion in the requirement specification
9.	Non-traceable requirement	• Can occur from human engineering factors: When one-way directional transient method implemented	• Irrelevant requirements are captured
10.	Non-verifiable requirement	• Human and software engineering factors: when requirement engineering principles not applied	• Irrelevant requirements are captured

11.	Infeasible requirement	• Human and software engineering factors: when project scope beyond available resources	• Vital requirements are not captured

C. Design Stage

S/N	Probable Failure Mode	Likely Cause	Detection Method
1	Ambiguous or unclear Requirement Document	• Failure may occur from human engineering and development process: when requirement engineering principles not applied	• Vital requirement is difficult to trace from the requirement document
2	Inappropriate chosen architectural design method	• Wrong architectural design applied	• Misinterpretations of design by developers
3.	Errors in human factor engineering during design phase	• Failure may occur from human engineering and development process: when developers are not adequately trained in design modelling techniques	• Modelling associated problems are emerging (Abubakar & Lawal 2020)
4.	Poor implementation of design principle	• Human engineering and development process: When the design witnesses some design hitches such as tight coupling and low cohesion (Abubakar & Lawal 2020)	• Intra and intra modular problem emerging in the system design
5.	Errors in human factor engineering specifications	• Human engineering factor: Can occur due inadequate training amongst developers	• Human associated errors are evolving
6.	Complex and complicated design	• Human engineering and development process: As a result of poor design skills by the developers which result	• Design components disjoint not communicating perfectly with each other
7.	Poor refinement process	• Human engineering and development process: As a result of poor design skills by the developers which result (Abubakar & Lawal 2020)	• Complexity in design components emerging
8.	Too much of elaborative specification at the design stage	• Best practises not adopted	• Design components disjoint not communicating perfectly with each other (Abubakar & Lawal 2020)
9.	Inconsistent design document	• Human engineering and development process: As a result of poor design skills by the developers	• Misinterpretations of design by developers

10	Non-traceable detailed design	• Human engineering and development process: Best design practises not adopted	• Misinterpretations of design by developers
11.	Unstructured detail design	• Human engineering and development process: Best design practises not adopted	• Design components disjoint

D. Implementation and Coding Stage

S/N	Probable Failure Mode	Likely Cause	Detection Method
1	Ambiguous or unclear Design Document	• Human engineering and development process: Design principles not implemented • Developers were not part of documentation activities	• Relevant documents are difficult to trace by developers (Abubakar & Lawal 2020)
2	Inappropriate chosen programming language	• May occur from human engineering and development process: when programmers are devoid of adequate programming skills	• The selected PL for the system development not working
3.	Errors in human factor engineering during coding and testing	• Human engineering factor: when testers and developers are not well trained	• Human associated errors are evolving (Abubakar & Lawal 2020)
4.	Difficulties in reviewing codes	• Human engineering factor reviewers and testers are not: sufficiently trained • It can also occur as a result of undocumented codes (Abubakar & Lawal 2020)	• Testers and code reviewers are facing difficult challenges in reviewing the code
5.	No provision for reusability	May occur from development process: e.g., when procedural programming paradigm was applied	• Difficulty in maintaining the program
6.	Misleading and incorrect documentation about the reused component	• Human engineering factor: due to poor programming skills by the programmers	• Difficulty in maintaining the program
7.	Poor programming practice	• Human engineering factor: can occur due to programming skills by the programmers • Programmers not sufficiently trained (Abubakar & Lawal 2020)	• Repetitive codes, programs cannot compile due to errors,
8.	Lack of coordinated team work amongst programmers	Human engineering factor programmers not sufficiently trained	• Different code versions for the same component exist (Abubakar & Lawal 2020)

9.	Technology change	• May occur from hardware factor: occurs when the when need for new technologies arises.	• Compatibility problems due to system adaptation to new technology (Abubakar & Lawal 2020)
10.	Inconsistent and complex code	• Human engineering and development process factors: when best programming practices are not followed • Inadequate programming skills by programmers (Abubakar & Lawal 2020)	• Repetitive codes resulting into lengthy and multiple code versions for the same component existing
11.	Ambiguous and difficulty in code comprehension	• May occur from human engineering and development process factors: when best programming practices are not followed • Codes not well documented	Repetitive codes resulting into lengthy and multiple code versions for the same component existing
12.	Intuitive and or informal testing processes	• May occur from human engineering and development process factors: when programming best practices are not followed	• System devoid of users' need
13.	Poor test case documentation	• May occur from human engineering and development process factors: can occur when best testing practises not implemented (Abubakar & Lawal 2020)	• System and unit tests conducted cannot be traced

E. Installation and Maintenance Stage

S/N	Probable Failure Mode	Likely Cause	Detection Method
1	Experiencing difficulties during installation	This difficulty may arise from human engineering hardware and development process factors: • Insufficient training of • System complexity • Environmental factors	• System cannot install
2	System not installed correctly installed due to change in environment	• May occur from hardware related issue. This may occur due to: hardware advancement, and or • Due to long period and continuous deployment	• During installation following error messages are displayed: installation cannot be completed or incompatible system component
3.	New set of requirements are emerging	• This failure may arise f when new set of requirements not originally captured are inevitably required	• During installation error messages such as requirement A, B, C and so on, not recognized or not specified (Abubakar & Lawal 2020)

4.	Persisting difficulties in using the systems	• May occur from human engineering and hardware related issue. This may occur as a result of inability of end users to adapt to the new system • Undocumented operational directives (Abubakar & Lawal 2020)	• When system users are confronted with operational challenges
5.	Expected functionalities are omitted	This occurs as a result of • inaccurate requirement • miscommunications amongst the project stakeholders	• Error messages such as certain system unit missing
6.	Misleading and incorrect documentation about the system operation	• Best requirement and design practises not implemented (Abubakar & Lawal 2020)	• System operators cannot understand the operation manual
7.	Several faults and errors emerging later during acceptance testing	• May occur from human and development process related issue. For example, can occur as a result of poor risk management practises or best testing practices not followed (Abubakar & Lawal 2020)	• Acceptance testing fails
8.	Adequate testing not conducted	• Hardware challenges. If standard testing tools are not available or best practices not followed by testers	• System cannot meet the users need (Abubakar & Lawal 2020)
9.	Difficulties experienced by software engineers in fixing reported problems	This may occur from human engineering challenges. When: • Developers are not sufficiently trained • If problems are not well described (Abubakar & Lawal 2020)	• Attempted system faults repairs fail
10.	Difficulties in system maintainability	This may result from development related challenges. For example, when: • Not adopting suitable programming paradigm	• System cannot be reused
11.	Change on one component of the new system having adverse effect on the rest of the system	• Can occur when best programming practises are not implemented	• System upgrade impossible (Abubakar & Lawal 2020)
12.	Quest for system change is inevitable	• Can likely occur due to security breaches. For example, when: there is virus attack	• System operation malfunction
13.	Miscommunication between end users and product support team	• May occur if there is communication gap between the customers, managers and developers	• System not meeting users' need (Abubakar & Lawal 2020)

APPENDIX B1: FMEA ADHERENCE QUESTIONNAIRE (FMEAAQ)

UNISA university of south africa

FMEA ADHERENCE QUESTIONNAIRE (FMEAAQ)

Dear Participant,

I thank you for accepting to participate in this survey.

This questionnaire was prepared as a followed up to our previous exploratory case study. The aim of the exploratory study was to: (i) to identify and justify the FMEA standard procedures, which software development experts consider as important for the conduct of project risk management process, and (ii) to determine how well do the standard procedures of FMEA satisfy the procedural requirement of prominent standard project risk management models (PMBOK, SEI-CRM and Boehm). In order to improve the research design and build a better understanding of the study under view, I have developed a questionnaire, which I attach and hope that you will spare your time to complete. The purpose of this survey is to determine the FMEA standard procedures that are important for the conduct of software project risk management and to determine the level of adherence of the FMEA procedures to the requirement of standard project risk management models (PMBOK, SEI-CRM and Boehm).

Once again, you are invited to participate in this survey because of your active engagement and contribution in the last exploratory case study. Be assured that all information you provide will be kept strictly confidential and that any data that is obtained will be reported in aggregate format. There will be no circumstances in which any identifiers linking you to the study will be included in any report associated with this study.

It should take you not more than 25 minutes to respond to the survey items. There are no known risks associated with this study. Your participation will benefit your profession, software development practitioners globally.

If you decide to participate, you are free to skip any of the questions that may make you uncomfortable. Sharing your opinion by filling this questionnaire is voluntary and you are free to discontinue at any time without consequence.

Thank you for your participation.

University of South Africa
Preller Street, Muckleneuk Ridge, City of Tshwane
PO Box 392 UNISA 0003 South Africa
Telephone: +27 12 429 3111 Facsimile: +27 12 429 4150
www.unisa.ac.za

General Directions:

This questionnaire has three sections and consists of four printed pages. Most of the questions are fairly close-ended. So, please read and adhere strictly to the directions provided in each section and tick (√) or circle the most appropriate response when answering the questions.

Your response is highly appreciated.

Section 1: Personal Information

Directions: Please answer the following questions in the spaces provided, circle or tick the most appropriate options.

1. Age…………………………………………………………………………

2. Are you: (please tick as necessary) □ Male □ Female

3. What is your professional background?
   - □ Programmer
   - □ Software Engineer
   - □ System Analyst
   - □ Software manager
   - □ Domain Experts
   - □ Other: (please describe) ________________________________

4. How many software development projects have you involved in the last months (approximately)? _______________

5. How many years of experience have you had in this current job?
   - □ <1 Year □ 1-2 Years
   - □ 2-5 Years □ 5-10 Years
   - □ >10 Years

6. Experience in Software Development (optional):
   - □ <1 Year □ 1-2 Years
   - □ 2-5 Years □ 5-10 Years
   - □ >10 Years

Section 2: Level of importance of FMEA standard procedures for the conduct of software project risk management

Directions: For each of the following FMEA standard procedures listed in this section, **circle** the response in the **right column** that best describes your view on the level of **IMPORTANCE** of the listed FMEA procedures for the conduct of project risk management.

Sample Question:

FMEA Standard Procedure	**Level of Importance**				
	VH	**H**	**M**	**L**	**VL**
Step 3. Assign different effects caused by the failures	**5**	**4**	**3**	[✓ **2**]	**1**

This person believes that the FMEA standard procedure of "Step 3 - Assign different effects caused by the failures" is of low importance for the conduct of software project risk management

VH = Very High (5), H = HIGH (4), M = Moderate (3), L = Low (2) and VL = Very Low (1)

FMEA Standard Procedures	**Level of *Importance***				
	VH	**H**	**M**	**L**	**VL**
Step 1. Gather a team and review the process or product	**5**	**4**	**3**	**2**	**1**
Step 2. Brainstorm unknown risks	**5**	**4**	**3**	**2**	**1**
Step 3. Assign different effects caused by the failures	**5**	**4**	**3**	**2**	**1**
Step 4. Prioritize – assign severity, occurrence and detection rankings for each failure mode	**5**	**4**	**3**	**2**	**1**
Step 5. Calculate the RPN number (RPN = Occurrence * Severity * Detection)	**5**	**4**	**3**	**2**	**1**
Step 6. Collect data, analyse and measure the failure modes for each action	**5**	**4**	**3**	**2**	**1**
Step 7. Apply methods to reduce high-priority/high-risk failures	**5**	**4**	**3**	**2**	**1**
Step 8. After performing actions evaluate the performance of the system again	**5**	**4**	**3**	**2**	**1**

Section 3: Extent of FMEA Adherence with popular project risk management standard frameworks/models (such as: PMBOK, SEI-CRM and Boehm)

Directions: For each of the following models' requirement of the project risk management framework (PMBOK, SEI and Boehm) listed in this section; **circle** the response in the **right column** that best describes your view on the level of **ADHERENCE** of the FMEA procedural requirement with the listed project risk management models' requirement.

A. Level of Adherence between PMBOK Model and FMEA Model

SN	Requirement in PMBOK Model	Level of *Adherence with FMEA Model*				
		VH	H	M	L	VL
1	**Risk Management Planning (Ref. Chp.11.1)**	5	4	3	2	1
2	**Risk Identification (Chp.11.2)**	5	4	3	2	1
3	**Qualitative Risk Analysis (Chp.11.3)**	5	4	3	2	1
4	**Quantitative Risk Analysis (Chp.11.4)**	5	4	3	2	1
5	**Risk Response Planning (Chp.11.5)**	5	4	3	2	1
6	**Risk Monitoring and Control (Chp.11.6)**	5	4	3	2	1

B. Level of Adherence between SEI Model and FMEA Model

SN	Requirement in SEI-CRM Model	Level of *Adherence with FMEA Model*				
		VH	H	M	L	VL
1	***Identify (Ref. SEI-CRM: Step1)***	5	4	3	2	1
2	**Analyse (Ref. SEI-CRM: Step2)**	5	4	3	2	1
3	**Track (Ref. SEI-CRM: Step4)**	5	4	3	2	1
4	**Plan (Ref. SEI-CRM: Step3**	5	4	3	2	1
5	**Control (Ref. SEI-CRM: Step5)**	5	4	3	2	1
6	**Communicate (Ref. SEI-CRM: Step6)**	5	4	3	2	1

C. Level of Adherence between Boehm Model and FMEA Model

SN	Requirement in Boehm Model	Level of *Adherence with FMEA Model*				
		VH	H	M	L	VL
1	***Risk Identification (Ref. Boehm Step1)***	5	4	3	2	1
2	**Risk Analysis *(Ref. Boehm Step2)***	5	4	3	2	1
3	**Risk Prioritization (Ref. Boehm tep3)**	5	4	3	2	1
4	**Risk Mgt. Planning (Ref. Boehm Step4)**	5	4	3	2	1
5	**Risk Resolution (Ref. Boehm Step5)**	5	4	3	2	1
6	**Risk Monitoring (Ref. Boehm Step6)**	5	4	3	2	1

Thank you for taking the time to complete this questionnaire

APPENDIX B2: FAILURE MODE EFFECT ON PROJECT OUTCOME QUESTIONNAIRE

UNISA university of south africa

FAILURE MODES EFFECT ON PROJECT OUTCOME QUESTIONNAIRE (FMEOPOQ)

Dear Participant,

I thank you again for accepting to participate in this survey.

The main purpose of this survey study is to assess the effects of potential failure modes in the software development process on software project outcome (cost, time and quality). Your invitation to participate in this survey came as a result of your participation in the previous exploratory study and the focus group discussions on the survey theme. Thus, your gains from the previous discussions and your professional background is expected to assist you make accurate and logical perception on the theme of the survey. In order to improve the research design and build a better understanding of the study under view, I have developed a questionnaire, which I attach and hope that you will spare your time to complete.

Be assured that all information you provide will be kept strictly confidential. Any data that is reported will be reported in aggregate format. Under no circumstances will any identifiers linking you to the study will be included in any report associated with this study. It should take you not more than 25 minutes to respond to the survey items. There are no known risks associated with this study. Your participation will benefit your profession and software development communities in general.

If you decide to participate, you are free to skip any of the questions that may make you uncomfortable. Sharing your opinion by filling this questionnaire is voluntary and you are free to discontinue at any time without consequence.

I would greatly appreciate your taking the time to complete the questionnaire.

Thank you for your participation.

General Directions:

This questionnaire has t*wo* sections and consists of six printed pages. Most of the questions are fairly close-ended. So, please tick (√) or circle the most appropriate response when answering the questions.

Your response is highly appreciated.

Section 1: Personal Information

Please answer the following questions in the spaces provided, circle or tick the most appropriate options.

1. Age……………………………………………………………………………

2. Are you: (please tick as necessary) □ Male □ Female

3. What is your professional background?
   - □ Programmer
   - □ Software Engineer
   - □ System Analyst
   - □ Software manager
   - □ Domain Experts
   - □ Other: (please describe) ________________________________

4. How many software development projects have you involved in the last months (approximately)? _______________

5. How many years of experience have you had in this current job?

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

6. Experience in Software Development (optional):

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

Section 2: Assessment of likely effect of probable failure modes on software project outcome (cost, time and quality/scope)

Directions: For each of the following listed probable FMs in their corresponding software development process stages, on the **right-hand** column, **circle** the response that best describes the **Likely Effect** of the listed FM on **software project outcome (cost, time and quality/scope)**.

Sample Question:

Software Development Process Stage: Installation and Maintenance	Adverse Effect on Project outcome: Cost	Adverse Effect on Project outcome: Time	Adverse Effect on Project Outcome: Quality/Scope
New set of requirements are emerging	VH H M L VL	VH H M L VL	VH H M L VL
	5 4 3 2 [✓ 1]	5 4 3 2 1	5 [✓ 4] 3 2 1

SA = Strongly Agree (5), A = Agree (4), M = Moderate (3), D = Disagree (2),and SD = Strongly Disagree (1)
VH = Very High (5), H = HIGH (4), M = Moderate (3), L = Low (2) and VL = Very Low (1)

This participant that made this submission viewed that *during the **Installation and Maintenance Stage** of the SDP, the listed probable FM* **"New set of requirements are emerging"** *has a significantly **low effect on the project cost,** a **high effect on the project quality/scope but no idea of its effect on software quality.***

(A) Stage 1: Planning

	Software Development Process Stage 1: Planning	**Adverse Effect on Project Outcome: Cost**					**Adverse Effect on Project Outcome: Time**					**Adverse Effect on Project Outcome: Quality/Scope**				
S/N	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	Inappropriate chosen software development framework/methodology/models	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
2.	Communication gap amongst the development team	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
3.	Ambiguous or unclear specification	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
4.	Insufficient software project tools and resource	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
5.	Unrealistic project goals and scopes	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
6.	Unrealistic budget	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
7.	Ambiguous project hierarchy model	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
8.	Inexperience project team members	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
9.	Poor risk management strategy	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
10.	Deviance of system security integration	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**

See next page for stage 2 of the software development process

(B) Stage 2: Requirement Specification and Analysis

	Software Development Process Stage 2: Requirement Specification and Analysis	Adverse Effect on Project outcome: Cost					Adverse Effect on Project outcome: Time					Adverse Effect on Project outcome: Quality/Scope				
S/N	**Probable Failure Modes**	VH	H	M	L	VL	VH	H	M	L	VL	VH	H	M	L	VL
1.	Inappropriate chosen requirement gathering technique	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	Ambiguous or unclear requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	Contradictions and confusion in domain specific terminologies	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	Contradictions and confusion in defining requirements in Natural Language (NL)	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	Errors in human factor engineering specifications	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	Inaccurate requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	Unrealistic requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	Inconsistent requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	Non-traceable requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	Non-verifiable requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
11.	Infeasible requirement	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

See next page for stage 3 of the software development process

(C) Stage 3: Design

	Software Development Process Stage 3: Design	**Adverse Effect on Project outcome: Cost**					**Adverse Effect on Project outcome: Time**					**Adverse Effect on Project outcome: Quality/Scope**				
S/N	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	Ambiguous or unclear Requirement Document	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
2.	Inappropriate chosen architectural design method	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
3.	Errors in human factor engineering during design phase	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
4.	Poor implementation of design principle	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
5.	Errors in human factor engineering specifications	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
6.	Complex and Complicated design	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
7.	Poor refinement process	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
8.	Too much of elaborative specification at the design stage	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
9.	Inconsistent design document	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
10.	Non-traceable detailed design	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	5	4	3	2	1
11.	Unstructured detail design	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	5	4	3	2	1

See next page for stage 4 of the software development process

(D) Stage 4: Implementation and Coding

	Software Development Process Stage 4: Implementation and Coding	Adverse Effect on Project outcome: Cost					Adverse Effect on Project outcome: Time					Adverse Effect on Project outcome: Quality/Scope				
S/N	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	Ambiguous or unclear Design Document	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
2.	Inappropriate chosen programming language	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
3.	Errors in human factor engineering during coding and testing	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
4.	Difficulties in reviewing codes	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
5.	No provision for reusability	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
6.	Misleading and incorrect documentation about the reused component	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
7.	Poor programming practice such as: too many repetitive code, redundant functions, etc.	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
8.	Lack of coordinated team work amongst programmers	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
9.	Technology Change	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
10.	Inconsistent and complex code	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
11.	Ambiguous and difficulty in code comprehension	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
12	Intuitive and or informal testing processes	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
13	Poor test case documentation	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**

See next page for stage 5 of the software development process

(E) Stage 5: Installation and Maintenance

	Software Development Process Stage 5: Installation and Maintenance	Adverse Effect on Project outcome: Cost					Adverse Effect on Project outcome: Time					Adverse Effect on Project outcome: Quality/Scope				
S/N	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	Experiencing difficulties during installation	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	System not installed correctly due to change in environment	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	New set of requirements are emerging	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	Persisting difficulties in using the systems	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	Expected functionalities are omitted	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	Misleading and incorrect documentation about the system operation	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	Several faults and errors emerging later during acceptance testing	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	Adequate testing not conducted	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	Difficulties experienced by software engineers in fixing reported problems	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	Difficulties in system maintainability	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
11.	Change on one component of the new system having adverse effect on the rest of the system	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
12.	Quest for system change is inevitable due to: System being slow for the current loading and concern about security measures	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

Thank you for taking the time to complete this questionnaire

APPENDIX B3: POTENTIAL FM EFFECT ON PROJECT OUTCOME QUESTIONNAIRE

POTENTIAL FAILURE MODES EFFECT ON PROJECT OUTCOME QUESTIONNAIRE (PFMEOPOQ)

Dear Prospective Participant,

My name is Bashiru Lawal and I am doing research with Professor Ernest Mnkandla and Professor Adéle Da Veiga towards the award of PhD in Information Systems at the famous University of South Africa, Pretoria. You have been specially selected to participate in the research titled: '**Assessing the Risk Factors Associated with the Potential Failure Modes in the Software Development Process'** after a purposeful random selection of the authors of articles and participants in panels at all the proceedings and workshops of the International Conferences on Software Engineering sponsored by ACM and IEEE CS between the periods of 2007 to 2017. The research is part of the requirement for the award of PhD in Information Systems at the famous University of South Africa, Pretoria – South Africa. You are selected because your point of view is important. I realize you are busy and I appreciate your time.

This questionnaire was prepared based on the discussions I have had so far during a focus group discussion with a team of software development practitioners, who were as well, purposefully selected from five software development industries across Nigeria. The aim of the discussion was to identify the probable failure modes in the SDP, which they considered as threat to successful completion of software development projects from software development professionals' point of view. In order to improve the research design and build a better understanding of the study under view, I have developed a questionnaire, which I attach and hope that you will spare your time to complete. The purpose of this questionnaire survey is to reassess and validate the findings of the focus group discussion.

Once again you were invited because you have participated in some sort of software development projects involving one of the stages of the software development process, so you are familiar with software project development process and the stages of the SDLC. All information you provide will be kept strictly confidential. Any data that is reported will be reported in aggregate format. Under no circumstances will any identifiers linking you to the study will be included in any report associated with this study.

It should take you not more than 25 minutes to respond to the survey items. There are no known risks associated with this study. Your participation will benefit your profession, software development practitioners and shall create valuable information through collection of authoritative list of risk factors associated with potential failure modes in the software development process that could threaten the survival of software development industries globally.

If you decide to participate, you are free to skip any of the questions that may make you uncomfortable. Sharing your opinion by filling this questionnaire is voluntary and you are free to discontinue at any time without consequence. However, your professional experiences and opinions are crucial to helping me make accurate and informed analysis of the study under view from software development practitioners' point of view. I would greatly appreciate your taking the time to complete the questionnaire. If you have any observation or contributions that will improve the quality of this research, please contact the address below.

Thank you for your participation.

Bashiru Lawal
Department of Information Systems
University of South Africa Pretoria – South Africa
Contact: email : 55773303@mylife.unisa.ac.za, blawal3119@gmail.comTel: +2348065248955

General Directions:

This questionnaire has t*wo* sections and consists of eight printed pages. Most of the questions are fairly close-ended. So, please tick (√) or circle the most appropriate response when answering the questions.

Your response is highly appreciated.

Section 1: Personal Information

Please answer the following questions in the spaces provided, circle or tick the most appropriate options.

1. Age………………………………………………………………………………

2. Are you: (please tick as necessary) □ Male □ Female

3. What is your professional background?
   - □ Programmer
   - □ Software Engineer
   - □ System Analyst
   - □ Software manager
   - □ Domain Experts
   - □ Other: (please describe) ________________________________

4. How many software development projects have you involved in the last months (approximately)? _______________

5. How many years of experience have you had in this current job?

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

6. Experience in Software Development (optional):

□ <1 Year	□ 1-2 Years
□ 2-5 Years	□ 5-10 Years
□ >10 Years	

Section 2: Level of Agreement of the listed probable failure modes as a potential risk factor in all the five studied stages of the software development process and Assessment o likely effect on the project outcome (cost, time and quality/scope)

Directions: For each of the following listed probable failure modes in their corresponding software development process stages, **circle** the response in the **left column** that best describes your view on the level of **Acceptance/Agreement** of the listed probable failure mode as a potential risk factor in the software development process stage. On the **right-hand** column, **circle** the response that best describes the **Likely Effect** of the listed probable failure mode on **software project outcome (cost, time and quality/scope)**.

Sample Question:

Could this be a potential failure mode in the software development process?	**Software Development Process Stage: Installation and Maintenance**	**Adverse Effect on Project Outcome: Cost**	**Adverse Effect on Project Outcome: Time**	**Adverse Effect on Project Outcome: Quality/Scope**
SA A M D SD	New set of requirements are emerging	**VH H M L VL**	**VH H M L VL**	**VH H M L VL**
5 [✓ 4] 3 2 1		**5 4 3 2 [✓ 1]**	**5 4 3 2 1**	**5 [✓ 4] 3 2 1**

SA = Strongly Agree (5), A = Agree (4), M = Moderate (3), D = Disagree (2),and SD = Strongly Disagree (1)
VH = Very High (5), H = HIGH (4), M = Moderate (3), L = Low (2) and VL = Very Low (1)

This participant that made this submission **Agreed** that *during the **Installation and Maintenance Stage** of the SDP, the listed probable failure mode* **"New set of requirements are emerging"** is *a potential failure mode in the stage and that the risk factor associated with the potential failure mode has a significantly **low effect on the project cost,** a **very high effect on the project duration of completion** and a **high consequence on the project quality/scope.***

Risk Factors associated with Potential Failure Modes across the studied stages of the Software Development Process

A. Stage 1: Planning

	Could this be a potential failure mode in the software development process?					Software Development Process Stage: Stage 1: Planning	Adverse Effect on Project outcome: Cost					Adverse Effect on Project outcome: Time					Adverse Effect on Project outcome: Quality/Scope				
S/N	**SA**	**A**	**M**	**D**	**SD**	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	5	4	3	2	1	Inappropriate chosen software development framework/methodology/models	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	5	4	3	2	1	Communication gap amongst the development team	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	5	4	3	2	1	Ambiguous or unclear specification	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	5	4	3	2	1	Insufficient software project tools and resource	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	5	4	3	2	1	Unrealistic project goals and scopes	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	5	4	3	2	1	Unrealistic budget	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	5	4	3	2	1	Ambiguous project hierarchy model	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	5	4	3	2	1	Inexperience project team members	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	5	4	3	2	1	Poor risk management strategy	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	5	4	3	2	1	Deviance of system security integration	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

B. Stage 2: Requirement Specification and Analysis

	Could this be a potential failure mode in the software development process?					Software Development Process Stage: Stage 2: Requirement Specification and Analysis	Adverse Effect on Project Outcome: Cost					Adverse Effect on Project Outcome: Time					Adverse Effect on Project Outcome: Quality/Scope				
S/N	**SA**	**A**	**M**	**D**	**SD**	**Probable Failure Modes**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**	**VH**	**H**	**M**	**L**	**VL**
1.	**5**	**4**	**3**	**2**	**1**	Inappropriate chosen requirement gathering technique	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
2.	**5**	**4**	**3**	**2**	**1**	Ambiguous or unclear requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
3.	**5**	**4**	**3**	**2**	**1**	Contradictions and confusion in domain specific terminologies	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
4.	**5**	**4**	**3**	**2**	**1**	Contradictions and confusion in defining requirements in Natural Language (NL)	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
5.	**5**	**4**	**3**	**2**	**1**	Errors in human factor engineering specifications	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
6.	**5**	**4**	**3**	**2**	**1**	Inaccurate requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
7.	**5**	**4**	**3**	**2**	**1**	Unrealistic requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
8.	**5**	**4**	**3**	**2**	**1**	Inconsistent requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
9.	**5**	**4**	**3**	**2**	**1**	Non-traceable requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
10.	**5**	**4**	**3**	**2**	**1**	Non-verifiable requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**
11.	**5**	**4**	**3**	**2**	**1**	Infeasible requirement	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**	**5**	**4**	**3**	**2**	**1**

C. Stage 3: Design

	Could this be a potential failure mode in the software development process?					Software Development Process Stage: Stage 3: Design	Adverse Effect on Project Outcome: Cost					Adverse Effect on Project Outcome: Time					Adverse Effect on Project Outcome: Quality/Scope				
S/N	SA	A	M	D	SD	**Probable Failure Modes**	VH	H	M	L	VL	VH	H	M	L	VL	VH	H	M	L	VL
1.	5	4	3	2	1	Ambiguous or unclear Requirement Document	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	5	4	3	2	1	Inappropriate chosen architectural design method	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	5	4	3	2	1	Errors in human factor engineering during design phase	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	5	4	3	2	1	Poor implementation of design principle	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	5	4	3	2	1	Errors in human factor engineering specifications	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	5	4	3	2	1	Complex and Complicated design	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	5	4	3	2	1	Poor refinement process	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	5	4	3	2	1	Too much of elaborative specification at the design stage	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	5	4	3	2	1	Inconsistent design document	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	5	4	3	2	1	Non-traceable detailed design	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
11.	5	4	3	2	1	Unstructured detail design	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

D. Stage 4: Implementation and Coding

	Could this be a potential failure mode in the software development process?					Software Development Process Stage: Stage 4: Implementation and Coding	Adverse Effect on Project Outcome: Cost					Adverse Effect on Project Outcome: Time					Adverse Effect on Project Outcome: Quality/Scope				
S/N	SA	A	M	D	SD	Probable Failure Modes	VH	H	M	L	VL	VH	H	M	L	VL	VH	H	M	L	VL
1.	5	4	3	2	1	Ambiguous or unclear Design Document	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	5	4	3	2	1	Inappropriate chosen programming language	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	5	4	3	2	1	Errors in human factor engineering during coding and testing	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	5	4	3	2	1	Difficulties in reviewing codes	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	5	4	3	2	1	No provision for reusability	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	5	4	3	2	1	Misleading and incorrect documentation about the reused component	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	5	4	3	2	1	Poor programming practice such as: too many repetitive code, redundant functions, etc.	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	5	4	3	2	1	Lack of coordinated team work amongst programmers	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	5	4	3	2	1	Technology Change	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	5	4	3	2	1	Inconsistent and complex code	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
11.	5	4	3	2	1	Ambiguous and difficulty in code comprehension	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
12	5	4	3	2	1	Intuitive and or informal testing processes	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
13	5	4	3	2	1	Poor test case documentation	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

E. Stage 5: Installation and Maintenance

	Could this be a potential failure mode in the software development process?					Software Development Process Stage: Stage 5: Installation and Maintenance	Adverse Effect on Project Outcome: Cost					Adverse Effect on Project Outcome: Time					Adverse Effect on Project Outcome: Quality/Scope				
S/N	SA	A	M	D	SD	**Probable Failure Modes**	VH	H	M	L	VL	VH	H	M	L	VL	VH	H	M	L	VL
1.	5	4	3	2	1	Experiencing difficulties during installation	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
2.	5	4	3	2	1	System not installed correctly due to change in environment	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
3.	5	4	3	2	1	New set of requirements are emerging	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
4.	5	4	3	2	1	Persisting difficulties in using the systems	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
5.	5	4	3	2	1	Expected functionalities are omitted	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
6.	5	4	3	2	1	Misleading and incorrect documentation about the system operation	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
7.	5	4	3	2	1	Several faults and errors emerging later during acceptance testing	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
8.	5	4	3	2	1	Adequate testing not conducted	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
9.	5	4	3	2	1	Difficulties experienced by software engineers in fixing reported problems	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
10.	5	4	3	2	1	Difficulties in system maintainability	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
11.	5	4	3	2	1	Change on one component of the new system having adverse effect on the rest of the system	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1
12.	5	4	3	2	1	Quest for system change is inevitable due to: System being slow for the current loading and concern about security measures	5	4	3	2	1	5	4	3	2	1	5	4	3	2	1

Thank you for taking the time to complete this questionnaire

APPENDIX B4: FMEA EVALUATION QUESTIONNAIRE (FMEAEQ)

Dear Students,

This questionnaire has been designed to help the researcher evaluate your perception on the ease of use and the practicability/applicability of the FMEA method used for the conduct of software project risk assessment during your software project risk assessment practical. All information you provide will be kept strictly confidential and under no circumstances will your individual identity or responses be mentioned or released during or after the research work.
Thank you for taking the time to fill this questionnaire thoughtfully.

Instructions

The questionnaire is a one-page document and consists of 2 sections. Most of the questions are close-ended. So, please tick (√) the most appropriate response when answering the questions.

Your response is highly appreciated.

Return this questionnaire to the survey taker when requested.

Section 1: Personal Information

Gender: ☐ Male ☐ Female
Age: ______________________

Your CGPA (please tick): ☐ 4.5 – 5.0 ☐ 3.5 – 4.49 ☐ 2.5 – 3.49 ☐ 2.49 – 0.0

Section 2: Usability and Practicability of FMEA Method in Software Project Risk Assessment

S/N	FMEA method's usability and applicability in software project risk management	Agree	Disagree
1	The FMEA standard guidelines are straight forward and efficiently enables me to identify, analyse, prioritize and control risks in software projects		
2	Using the FMEA method was beneficial to my risk management skills in SE		
3	Applying FMEA method helps identify all unknown risks in the project		
4	*Doing FMEA rating techniques makes accurate assessment of risk severity*		
5	*Doing FMEA rating techniques makes accurate assessment of possibility of risk occurrence*		
6	*Doing FMEA rating techniques makes accurate assessment of risk detection*		
7	*Doing FMEA rating techniques makes accurate assessment of possibility of risk occurrence*		
8	Using the FMEA method was beneficial to my risk management skills in SE		
9	*I think my understanding of the key FMEA standard procedures has improved because of the adherence analogy with other risk assessment models used in the* lectures and training		
10	*FMEA method aids effective risk management process as it is faster and subjective practices via documentation* of activities		
11	*It is easy to determine risk magnitude using the modified probability theory: RPN = Occurrence* *Severity*Detection		
12	Brainstorming to do FMEA cause-effect analysis of the identified risks in software projects improve my personal risk analysis skills		
13	*I benefit from doing FMEA process of risk management as its procedures enable me to follow a sound risk assessment process that helps to suggest most suitable risk control strategies in a systematic way*		
14	*I benefit from doing FMEA process of risk management as it enables me to follow a sound risk classification that helps to suggest risks that deserves highest attention*		
15	Using the FMEA method was beneficial to my expertise in project risk management		
16	*FMEA method application for risk management provides better understanding of risk behaviours and makes me establish appropriate mitigation strategies*		
17	*Completing the FMEA worksheet during the risk management process and determining the RPN values* enables me to mentally *"see" the risk variability and dependability across all the stages of SDP*		
18	The FMEA method aids my understanding of how software project risk can be prevented and controlled in software projects		
19	Using the FMEA method was beneficial to my expertise in project risk management		
20	The FMEA method is time consuming but clearly supports my understanding and practice		
21	*FMEA method aids effective risk management process as it is faster and subjective practices via documentation* of activities		
22	The FMEA method aids my understanding of how risk can be anticipated before it occurs and have it controlled after occurrence in software projects		

Thank you for taking the time to complete this questionnaire