# SOFTWARE PROCESS IMPROVEMENT: AN ASSESSMENT OF REQUIREMENT ENGINEERING PROCESS FOR IMPROVING SOFTWARE DEVELOPMENT IN SELECTED SOUTH AFRICA DEVELOPMENT FIRMS

BY

UCHECHUKWU DESMOND ANOKWURU
(STUDENT NUMBER: 23989645)

DISSERTATION SUBMITTED IN FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF SCIENCE (MSC.) IN COMPUTER SCIENCE

NORTH-WEST UNIVERSITY
YUNIBESITI YA BOKONE-BOPHIRIMA
NOORDWES-UNIVERSITEIT
MAFIKENG CAMPUS

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF MATHEMATICAL AND PHYSICAL SCIENCES
FACULTY OFAGRICULTURE, SCIENCE AND TECHNOLOGY
NORTHWEST UNIVERSITY, MAFIKENG CAMPUS

SUPERVISOR: PROFESSOR O.O. EKABUA

MARCH, 2014

## DECLARATION

I declare that this Research Project on **Software Process Improvement: An Assessment of Requirement engineering Process for Improving Software Development in Selected South Africa Development Firms** is my work, and has never been presented for the award of any degree in any university. All the information used has been dully acknowledged both in text and in the references.

Signature: [illegible] Date: 18/09/2014

**Uchechukwu Desmond Anokwuru**

## Approval

Signature: ____________________ Date: ____________________________

Supervisor: **Prof O.O. Ekabua**
Department of Computer Science
Faculty of Agriculture, Science and Technology
North West University, Mafikeng Campus
South Africa

## DEDICATION

This research project is dedicated to my late Parents who would have been delighted to see this day:

**John Anokwuru Kamalu**
**and**
**Pearl Olejurumaka Anokwuru**

# ACKNOWLEDGEMENTS

I wish to first express my gratitude and praise to the Almighty God, for making everything possible. I thank Him for giving me life, knowledge, wisdom and seeing me through to the successful completion of this research project and my programme.

I am immensely grateful and thankful to Prof O.O. Ekabua, my supervisor, for his support, advice, discussions and useful criticism while carrying out this research. I am also thankful for his support and encouragement to me in pursuing this degree.

I am very thankful to the Dean of the Faculty of Agriculture, Science and Technology, Prof Eno Ebenso for his support and encouragement.

I am also grateful to the lecturers and staff of the Department of Computer Science, North West University, Mafikeng Campus, for their help and support.

I also appreciate the help, support and encouragement of all my family and friends – Chinedu & Rafiat Anokwuru, Chinwendu Nwabeke, Emmanuel Anokwuru, Bassey Isong, Mr Gilbert Dzawo, Prof Kinsley Ayisi, Ms Dimakatso Tlaka, Mr Gideon Ledwaba, Nosipo Dladlu. Ifeoma Ohiaeri, Nnenna Eric-Nwoye, Chigozie Azunna

I am very grateful Mr and Mrs ZM Mahango for all the love, patience and support they granted to me and my family whilst undertaking this programme.

Finally, I wish to express my love and gratitude to my wife, Madipere Anokwuru, for her love, support and encouragement. And to my Kids (Chiamaka Pearl Anokwuru and Uchenna John Anokwuru) for inspiring me, even in the midst of their distractions. It would not have had any meaning without you. Thank you.

# ABSTRACT

Requirement engineering (RE) is a crucial step towards software process improvement. It is the first and critical phase in software development projects, and the main aim of RE process is the gathering of requirements, in order to meet system owners and system users specifications. More also, It involves a set of activities like system feasibility study, elicitation, analysis, negotiation, verification, validation, documentation and management of the requirements. RE process uses several methods and techniques to establish user and systems requirements. These techniques and methods are the tools that requirement engineers apply to gather the requirements. Furthermore, RE engineers face numerous problems in gathering these requirements. These limitations are due to the lack of knowledge and awareness of the results that can be obtained by using these techniques, as well as the ability of RE engineers to select appropriate techniques during RE process. Furthermore, this inability greatly affects the quality of software, and increases the production cost of software projects. In this thesis, we looked at the use of RE processes during software development projects, and focused on software development firms in South Africa. We made use of experimental case survey, whereby we used questionnaires in the gathering of our, data and further presented our results with the aid of bar and pie charts. More also, we embarked on the analysis and comparism of different techniques, and tools used during for RE process. This provided the platform for requirements engineers to know the characteristics of these techniques, the effectiveness of every technique and also, the popularity of specific techniques. Furthermore, it is important to know that selecting a particular technique depends on the type of application to develop, the years of experience of RE engineers as well as educational background of RE engineers. Based on our findings, we can also deduce that the analysis made in this thesis, can be used for the future development of new techniques in RE and subsequently, improving the level of software development process in South Africa and the world at large.

# Table of Content

## List OF Figures

## List of Tables

## List of Acronyms

| | |
|---|---|
| AHP | Analytic Hierarchy Process |
| ANSI/EIA | American Standard Institute/ Electric Industries Alliance |
| CMM | Capability Maturity Model |
| CMMI | Capabilty Maturity Model Integration |
| CO | Customer Online |
| FR | Functional Requirement |
| IEEE | Institute of Electrical and Electronics Engineers |
| ISO | International Standard Organization |
| IT | Information Technology |
| JAD | Joint Application Development |
| KPA | Key Process Area |
| KZN | KwaZulu Natal |
| MIL-STD | Military Standards |
| NFR | Non-Functional Requirement |
| QS | Quality System |
| RE | Requirement Engineering |
| RE-ELICIT | Requirement Elicitation |
| RI | Requirement Inspection |
| RT | Requirement Testing |
| SDL | Structured Development Language |

| | |
|---|---|
| SEI | Software Engineering Institute |
| SNLS | Structured Natural Language Specification |
| SPI | Software Process Improvement |
| SPICE | Software Process Improvement & Capability dEtermination |
| SPSF | Software Project Success Factor |
| SRS | Software Requirement Specification |
| SW-CMM | Software Capability Maturity Model |
| UCD | User Centred Design |
| UML | Unified Modelling Language |
| VPBV | View point Base Verification |
| WC | Western Cape |

# Chapter 1

# Introduction and Background

## 1.1 Introduction

For the past two decades, software development in South Africa has played a vital role in the management of business activities and socio-economic reforms existing today. For many small software companies, full-scale software process improvement (SPI) initiatives are often out of reach, due to prohibitive cost and lack of SPI knowledge. However, to compete in the global market, software developers must improve their productivity, time to market and customer satisfaction. Most software products developed today are over the shelf products which meet general requirements of users' needs but do not meet specific user requirements [1, 2]. Developing software applications that meet specific user requirements face many problems which have led to delay of many projects and also exceeding the budgets for such software development projects [2].

Software process improvement (SPI) is recognized as having the potential to improve competitiveness by increasing productivity, reducing costs, reducing defect and reducing rework, as well as improving time to market and customer satisfaction. Software process refers to all activities like software tools, methods and practices used in the whole lifecycle of software development that meet all of users' requirements and implement them into the exact required software product [2]. With the increasing complexity of software development, many software companies and research institutions focus their attentions on how to manage, and control the software development process, and also improve the production efficiency, to ensure producing reliable and economical software [3]. Most of, if not all software development projects, depend on adequate requirement engineering (RE) process to improve on software development processes.

RE covers the activities of requirement elicitation (capture, discovery, acquisition) analysis, specification and validation [4]. Most work in the RE field is on requirement representation or modeling techniques [5], but not on the assessment of RE processes. Whenever requirement engineers lack the knowledge of the performance and characteristics of different RE techniques, the activities related to requirement specification will fail, thus, leading to gathering of wrong requirements that makes wrong specification document [5]. A product developed with wrong specification document never meets the customers' expectation and more also, changing or adjusting requirements at the middle of a project development has a direct impact on project delay and increase in cost. Requirements are direct products of the RE process, whereby inadequacies in any of the processes can have severe consequences. These consequences are mainly due to the fact that problems in requirements filter down to design and implementation [6].

The importance of having an adequate RE process in place, which produces good enough requirements, can be considered as crucial to the successful development of products. These products can be in a custom-made or market-driven development effort [7]. There are however, clear indications that RE is lacking in industry since inadequacies in requirements is a major determinant in project failure [7]. As such, the increased attention given to the field of RE in research is understandable and crucial.

There are some instances of software failure in South Africa, and these failures are due to improper use of RE techniques. There was a case of the Health Information System (MEDICOM) implemented in Limpopo province South Africa in 1998 [8]. In this case, requirement elicitation did not cover multi-dimensional aspects of the system and also lack the mechanism to adequately understand user preferences. We also have the situation of the e-tolling system implemented in Gauteng (N1 Pretoria to Johannesburg). In this case, the socio-economic requirement was not adequately classified, as well as the complexity of the system. These are the few examples of the importance of selecting the appropriate RE techniques. The RE requests for every organization differ between organizations. The success of a software project is due to the application of RE during software development process.

## 1.2 Background Study

In today's software development world, RE is faced with lots of challenges which require urgent attention. These challenges could be organizational dependent or engineers dependent or the nature of RE process activity. In the perspective of RE process activity, RE poses one of the great challenges in both market-driven and bespoke software development, even though it is harder in market-driven development, since there is no existing specific customer [9]. There has been some research works already conducted on this topic from 1990, such as Basher Nuseibeh's paper "Requirements Engineering Road map" and "Requirements Engineering Techniques: Analysing the Gap between Technology Availability and Technology Use" by Hickey etal [10].

In RE, requirement identification is a key element, and several techniques exists that can be used to gather customer's needs, and achieve their business objectives. However, despite these techniques, the choice of a good technique is critical to the quality of the requirements [11]. This is due to the fact that, not all techniques are good or effective for all project situations. Hence, knowing which technique is effective, in terms of application, and function, as well as the situations, where by there will be a difference in the quality requirements gathered, which will in turn, facilitate development process and ultimately produce quality software product [12].

Therefore in this research, an experimental survey is the method we used in assessing the actual state of the RE processes, tools and techniques used in South African software industries. These processes and techniques are aligned to RE concepts to enable adequate classification during usage. This is because software development in South Africa is increasingly gaining momentum, and we need to improve the quality of the requirements, in order to ensure, that all requirements for a giving project, is gathered according to the need of the systems owners for that specific software development project. To achieve this goal, we perform a survey of some selected software development firms in South Africa in respect to their RE processes.

## 1.3 Problem Statement

RE process among South Africa software development companies need software re-engineering. Requirement engineers are facing many problems in requirements specification, due to lack of knowledge on results of methods, and inability to select appropriate methods. This lack of knowledge, reduces the perception on the importance of RE processes in every software development, and might be totally omitted during software development

South Africa has experienced some direct consequences of insufficient or lack of RE process, during software development, and this is why most software development is reviewed mid –way into implementation or does not meet user expectations or does not meet legislations after implementation. The correction cost, and post implementation cost is usually very high, and this might lead to project cancellation, especially when the cost of fixing the correction outweighs the benefits derived from the software product.

## 1.4 Research Questions

Considering that the knowledge and perception of the importance of RE is vital to the success of software development, this research seeks to address the following questions:

RQ 1. How can requirement engineers select adequate RE method from multiple available methods that will meet the characteristics of a specific type of project?

RQ 2. In the South African context, how can we design a framework that can be used by requirement engineers to adequately classify and select RE method during a specific type of software development project?

## 1.5 Motivation

Investing in software applications is a cost that most organization sees as very important towards providing competitive advantage all over the world. South Africa is no exception to this quest, and has moved towards software development as an important component in every business and government initiatives. Furthermore, as software development is on the increase in South Africa, the cost of inaccurately specifying requirement becomes a major setback in achieving the objective of any software development process. This lack of knowledge has led to the production of poor quality software, increased cost of production, scope creep and the worst case of total cancellation of the project.

A fundamental solution is to gain a proper understanding of RE techniques and develop a framework that can enable selection of appropriate RE techniques that are relevant and favorable to the socio-economic region of South Africa. This will be achieved by analyzing and comparing the different methods of RE processes, which will be useful to find the characteristics and performance of different RE techniques for specific type of software development project.

## 1.6 Research Goal

The main goal of this research is to analyze and compare the different methods of RE processes during software development in South Africa, as well as develop a framework that will enable developers' select adequate RE technique during software development.

## 1.7 Research Objectives

In achieving the research goal, we shall use the following objectives:

a) Analyze current or existing literature and techniques on general RE methods in RE processes.
b) Determine a set of project attributes influencing RE techniques effectiveness, which will serve as a back bone for experimental survey, for the purpose of classifying RE techniques focusing on South African software development environment.
c) Design a generic RE framework identification technique, as a proof of concept tool, to validate and enhance RE processes in South African software development environment.

## 1.8 Research Methodology

The research methods used in this project work are as follows:

a) **Literature Survey:** This method entails surveying the background of the area of interest and analyzing related works. Previous and existing RE techniques will be closely scrutinized.
b) **Experimental Survey:** This method will be used to understand informal processes. Furthermore, it will be used to collect data and analyze the results.
c) **Model Formulation:** The theoretical and practical knowledge gained from the survey will be used as a foundation for this research and to formulate our proposed model.
d) **Results Evaluation:** The result is presented in graphical formats, showing the effect of the selected variables and techniques.

## 1.9 Key Terminologies

a) **Requirements** - Requirements are a specification of what should be implemented; they are descriptions of how the system should behave, or of a system property or attribute
b) **Requirement Engineering** - Requirement engineering is a repeatable and systematic technique it is the branch of software Engineering concerned with the real world goals for Functions of and constraints on software systems.
c) **Requirement Elicitation** – this is the process of seeking, uncovering, acquiring and elaborating requirernents for a task
d) **Software process** – it refers to methods and practices used in the whole lifecycle of software development

## 1.10 Research Contribution

The main contribution of this research to academia and research community, is the assessment of various tools, and processes used during RE, to classify various software development projects within a South African perspective. This will give requirement engineers the platform to adequately implement RE techniques during software development projects

## 1.11 Thesis Summary

The remainder of this research project is organized as follows:

Chapter 2 is on review of related literature

Chapter 3 on analysis of findings in Requirement Engineering

Chapter 4 is on discussion of findings in requirement Engineering

Chapter 5 is about concluding chapter of this research report. A summary emphasizing this project's contributions is presented followed by recommendations. Suggestions on future work are also pointed out.

# Chapter 2

# Literature Review

## 2.1 Chapter Overview

RE affects software development deliverables extensively, both positively and negatively. The knowledge of RE process, increase the level of benefit derived from RE. This chapter analyses literature from various researchers on the pros and cons of RE, RE tools, RE processes and the effect RE on software process improvement within software development projects.

## 2.2 Software Process

Software process refer to all activities like software tools, methods and practices used in the whole lifecycle of software development, that meet all of user requirements and implement these requirements into the exact software product[13] . With the increasing complexity of software development, many software companies, and research institutions focus their attentions on how to manage and control the software development process, and also, improve the production efficiency to ensure production of reliable and economical software [14]. Many enterprises have introduced software quality standards such as ISO 9000 quality system (QS), software-capability maturity model/ capability maturity model integration (SW-CMM/CMMI) model into their daily activities for software quality assurance and process improvement [15].

## 2.3 Software Process Improvement Standards

The International Standards Organization's ISO 9000 standards and the Software Engineering Institute's Capability Maturity Model for Software (CMM) are two important models for software improvement.

### 2.3.1 ISO 9000 Standards

The rationale behind the ISO 9000 standards is that a disciplined organization with an accurately defined engineering process has a higher chance to manufacture products that consistently meet the purchaser's requirements, within deadlines, and available finance, than an inefficiently handled organization that lacks an engineering process [16]. ISO 9001 defines 'Quality systems -

models for quality assurance in design/implementation, manufacture, installation and servicing'. [17] A schematic perspective of the structure of the ISO 9000 standards is given in Figure 2.
The ISO 9001 standard from the ISO 9000 standard series, mainly deals with the requirements of management responsibilities, resources management, products realization, measurement analysis and improvement to produce stable software products meeting customer's requirements and applicable to certain laws and regulations [17].
ISO 9001 is a group of quality system requirements that is made up of twenty clauses that depict requirements for quality assurance in design, development, manufacture, installation and offering services that defines which facets of a quality system have to be available within an organization. Further details as to how these facets should be implemented and institutionalized are not supplied by ISO 9001 [18]. As ISO 9001 was developed to be applied in all sorts of industries, ISO 9000-3 was added specifically for software-development [18]:
The standards also encourage organizations using software process approach in establishing, implementing and improving the quality of the system to improve customer satisfaction, and to achieve the goal of continuous improvement in quality management system.

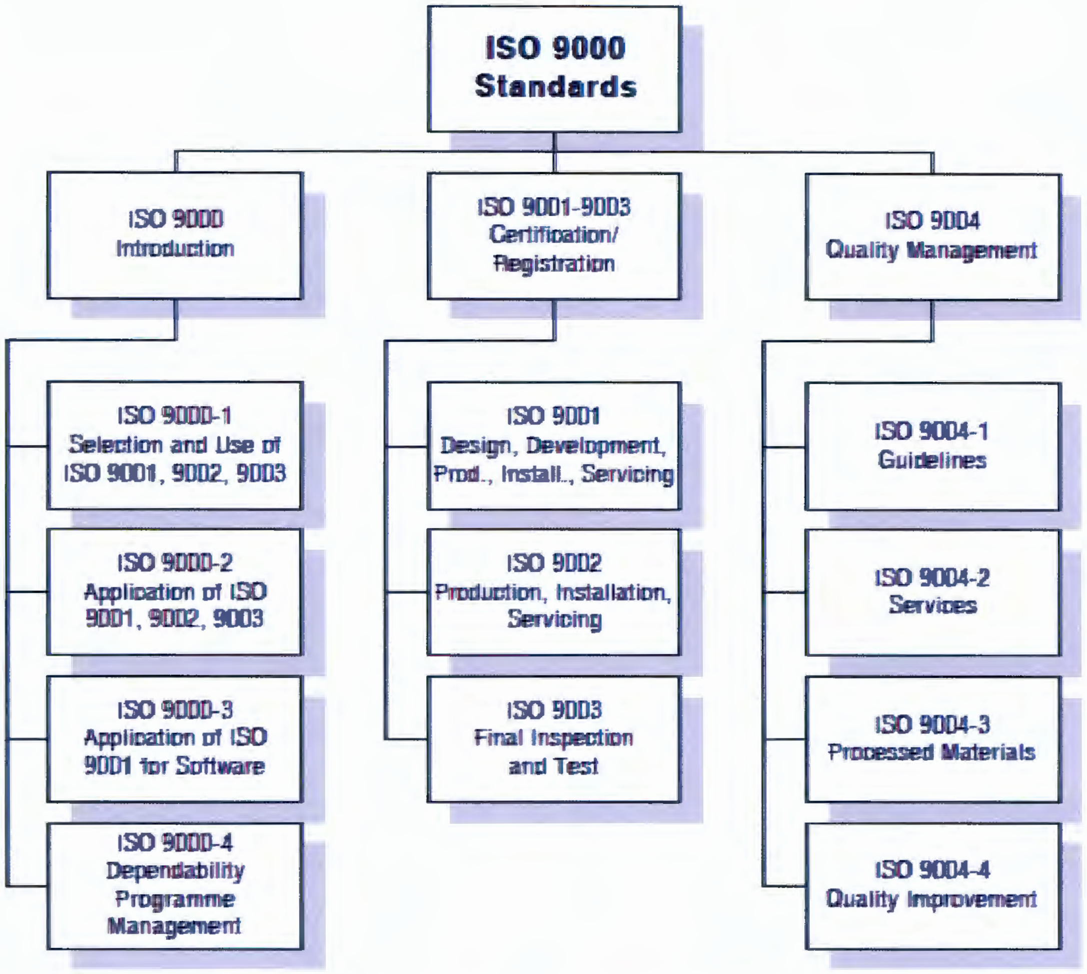

ISO 9000 Standards
ISO 9000 Introduction
ISO 9001-9003 Certification/ Registration
ISO 9004 Quality Management
ISO 9000-1 Selection and Use of ISO 9001, 9002, 9003
ISO 9000-2 Application of ISO 9001, 9002, 9003
ISO 9000-3 Application of ISO 9001 for Software
ISO 9000-4 Dependability Programme Management
ISO 9001 Design, Development, Prod., Install., Servicing
ISO 9002 Production, Installation, Servicing
ISO 9003 Final Inspection and Test
ISO 9004-1 Guidelines
ISO 9004-2 Services
ISO 9004-3 Processed Materials
ISO 9004-4 Quality Improvement

**Figure 2.1: The Structure of the ISO 9000 Standards**

### 2.3.2 Software Capability Maturity Model

SW-CMM includes a straightforward process path by defining certain key process areas (KPAs) on four different maturity levels (levels 2 to 5) [19]. But the SPI views of the small companies are not fully aligned with the SW-CMM maturity levels. The differences can, to some extent, be related to the actual process capability of the company.

The architectures mentioned above provide an effective framework for software process improvement for software enterprises to gradually improve the development process and improve the products quality with more specific aims. However, the ISO 9000 standards system which is mainly from the user point of view, controls the quality requirements of the systems, and sets the minimum goals of the system.

SW-CMM/CMMI system only tells what to do in a software process improvement, but does not provide key processes and the specific knowledge and skills required by those processes [19]. There are some limitations of it by introducing CMM model into small software enterprise because it puts more emphasis on project control than business decision making. At the same time the implementation of the CMM model system should build tailoring guideline in accordance with their conditions.

Capability Maturity Model, referred to in abbreviation as CMM, is a model designed and implemented by the Software Engineering Institute (SEI) and its main theme was founded on the best practices from the industry [19]. The initial objective of devising this modem was to evaluate the software development process of third-party suppliers of the United States' Department of Defense. The model aids those suppliers to boost their processes, as well. Organizations are backed up by the CMM to enhance the maturity of their software process via an evolutionary path, of five maturity levels (Figure 2.2), from 'ad hoc and chaotic' to 'mature and disciplined' managing styles.

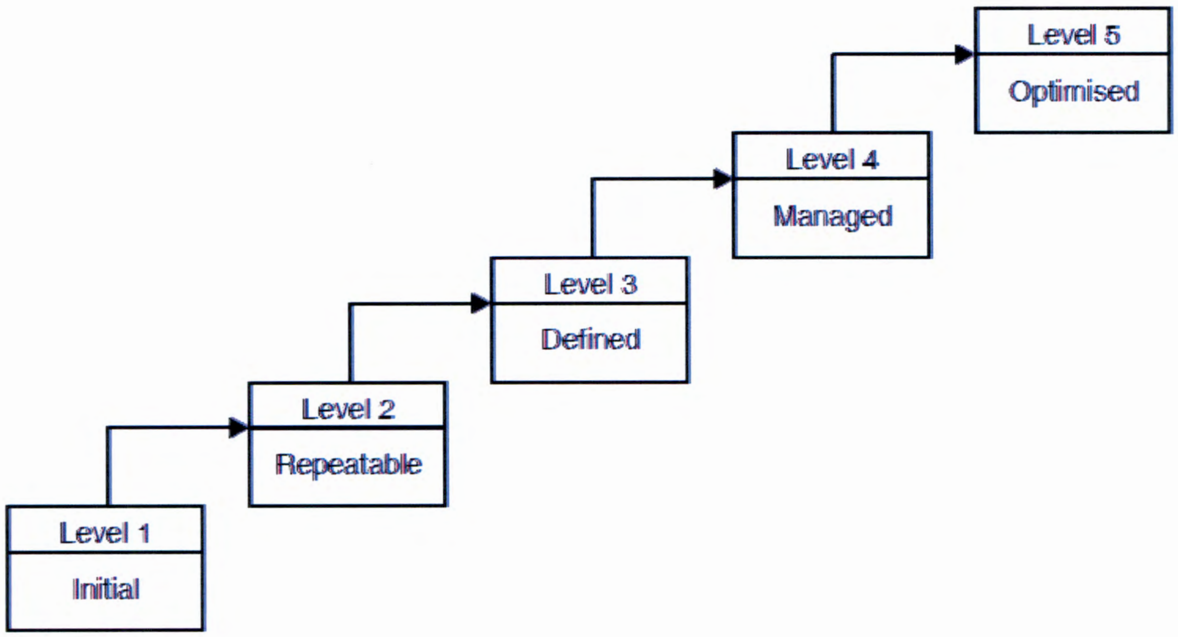


**Figure 2.2: CMM Capability Maturity Model**

Each CMM maturity level contains a set of Key Process Areas (KPA¡¦s) which describe the most important abilities for that level. The lists of KPA¡¦s of the CMM are:

a) Initial

The software process is described as ad hoc, occasionally or chaotic. Not many processes are defined, and fruitful achievement closely depends on individual effort and dedication.

b) Repeatable

Basis project-management process is founded to trace costs, time schemes and operations. The essential process discipline is out there to bring about the earlier successes on projects again with similar applications.

i. Requirements management
ii. Software project planning
iii. Software project tracking & oversight
iv. Software subcontract management
v. Software quality assurance
vi. Software configuration management

c) Defined

Software process, both for management and engineering activities, has to be documented, standardized, and blended into a standard software process for the organization. All projects should be required to use a widely-approved, well-tailored version of the organizations standard software process when composing and maintaining software.

i. Organization process focus
ii. Organization process definition
iii. Training program
iv. Integrated software management
v. Software product engineering
vi. Inter-group co-ordination
vii. Peer reviews

d) Managed

Meticulous evaluations of the software process and product quality are collected. Software process and products are both understood and controlled in terms of quantity.

i. Quantitative process management
ii. Software quality management

e) Optimizing

Persistent process improvement could be initiated by taking quantitative feedback from the process and from piloting novel ideas and creative technologies.

i. Defect prevention
ii. Technology change management
iii. Process change management

### 2.3.3 BOOTSTRAP

The BOOTSTRAP method is the consequence of a European project under the auspices of the European Strategic Programme for Research in Information Technology (ESPRIT). It furnished a new option for organizations that are enthusiastic to improve their software development process, and attain ISO 9001 certification, as it combines and drives forward the methods supplied by the CMM and the ISO 9000 quality standards [20].

The fundamental of the BOOTSTRAP method is set up by CMM. Exactly as the CMM, an evaluation is based on five maturity levels, but the BOOTSTRAP method applied a different criterion to evaluate an organizations' or projects' total strengths and flaws . The ISO 9000 quality standards (ISO 9001 and ISO 9000-3) have been directly built into the methodology owing to the face that they supply key hints for a company-wide quality system. The CMM does not contain such guideline. Furthermore, many European companies apply ISO 9000 as a primary quality standard. BOOTSTRAP can be used by organizations to determine well-preparedness for ISO 9001 certification [20]. BOOTSTRAP singles out three fields that identify the maturity of an organization. These fields include technology, methodology and organization.

Methodology is sub-categorized into a life-cycle dependent, life-cycle independent and process related area, of which the life-cycle independent area goes as far as to be divided into management, support and customer-supplier. For each field in this BOOTSTRAP tree, a number of processes are defined. Each process has a number of 'key-practices' that need to be targeted for that process. Adding to this, each process has a 'capability dimension', which identifies the current status of that process on a scale from 0 to 5. Unlike the CMM, quartiles between these levels are distinguished, which make it possible to evaluate on organization

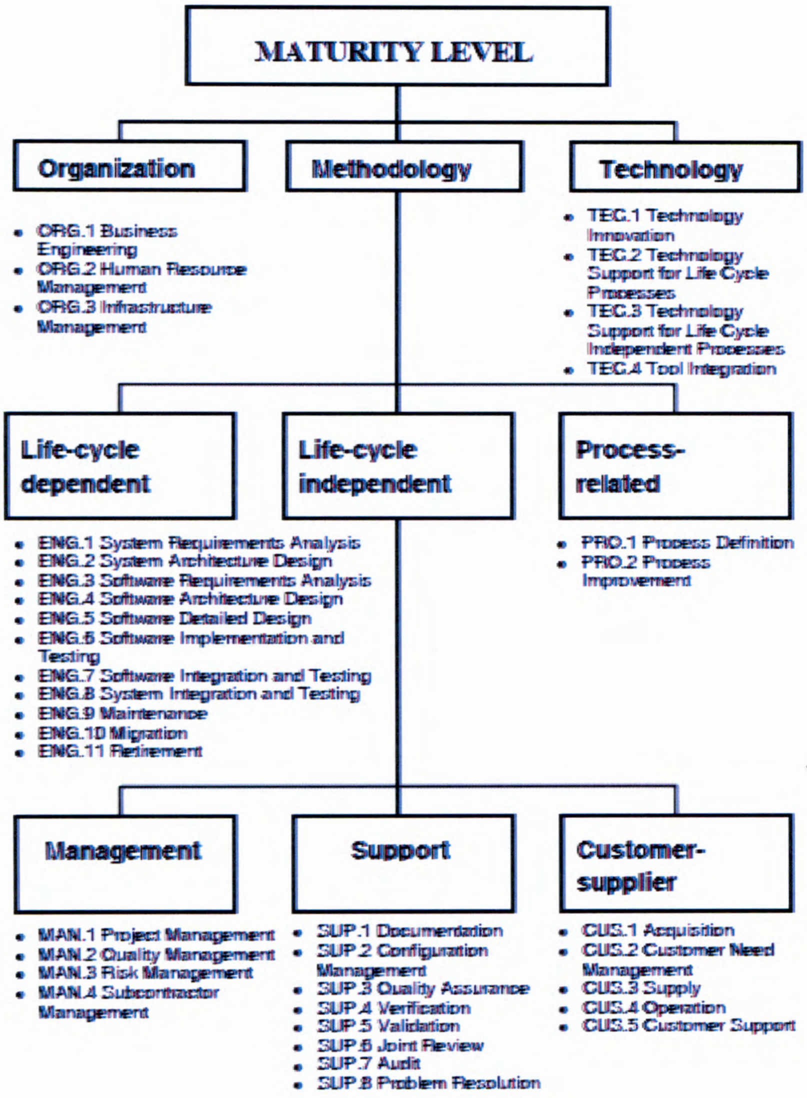

MATURITY LEVEL
Organization
Methodology
Technology
ORG.1 Business Engineering
ORG.2 Human Resource Management
ORG.3 Infrastructure Management
TEC.1 Technology Innovation
TEC.2 Technology Support for Life Cycle Processes
TEC.3 Technology Support for Life Cycle Independent Processes
TEC.4 Tool Integration
Life-cycle dependent
Life-cycle independent
Process-related
ENG.1 System Requirements Analysis
ENG.2 System Architecture Design
ENG.3 Software Requirements Analysis
ENG.4 Software Architecture Design
ENG.5 Software Detailed Design
ENG.6 Software Implementation and Testing
ENG.7 Software Integration and Testing
ENG.8 System Integration and Testing
ENG.9 Maintenance
ENG.10 Migration
ENG.11 Retirement
PRO.1 Process Definition
PRO.2 Process Improvement
Management
Support
Customer-supplier
MAN.1 Project Management
MAN.2 Quality Management
MAN.3 Risk Management
MAN.4 Subcontractor Management
SUP.1 Documentation
SUP.2 Configuration Management
SUP.3 Quality Assurance
SUP.4 Verification
SUP.5 Validation
SUP.6 Joint Review
SUP.7 Audit
SUP.8 Problem Resolution
CUS.1 Acquisition
CUS.2 Customer Need Management
CUS.3 Supply
CUS.4 Operation
CUS.5 Customer Support

**Figure 2.3: Bootstrap Model [25]**

### 2.3.4 SPICE

SPICE (Software Process Improvement & Capability Determination) is a widely recognized massive international action to raise a Standard for Software Process Assessment [ISO 15504 1998]. ISO 15504 is used as a reference framework for software process capability determination. It is based on other popular approaches, mainly on BOOTSTRAP, CMM and ISO 9001 [20].

Changes with respect to the CMM are:

a) a broader scope: processes which are directly related to the software development processes are also taken into account
b) a different architecture: levels are distinguished in all Key Process Areas, whereas specific Key Process Areas within the CMM are only of significance within a determined level
c) an integration of other SPI-models, such as ISO 9000, TickIT and Trillium [21]

Listed below are the strengths and weaknesses of SPI methodologies for embedded software

**Strengths**

a) Based on best practices
b) Provides a vision
c) Management tool for improvement
d) Changes are prescribed
e) Explicit priority to quality

**Weaknesses**

a) Product quality not addressed
b) Lack of measurement
c) No cost/benefit analysis included
d) Too generic
e) No project level support: mainly for large organizations
f) Continuation difficult
g) Dependency on individual managers
h) Phasing not logical
i) Improvement takes long
j) Risk for bureaucracy

### 2.3.5 Software Project Success Factor (SPSF) for the Organization

There are other factors involved in software project development. These factors include technical maturity of the organization, the involvement of engineering and managerial disciplines, organizational culture, application domain and specific characteristics of the project [22]. Therefore, different projects have different needs that require different RE process models. There are different components involved within RE during software projects we will be looking at these components next.

### 2.3.5.1 Business Rules

Business rules are statements that aim to influence or guide behavior and information in an organization and categorized to mandate, policies and guidelines [23]. They serve as guidance in helping the people manage their daily work but in certain case these business rules normally make the daily work more complex. National and international standards, regulations and laws impose restrictions on business practices to achieve societal goals such as improving corporate accountability in financial markets or ensuring the privacy of medical records in the health care industry.

### 2.3.5.2 Business Processes

Business Processes are set of activities that create values for a customer. Organizations are increasingly automating processes using work flow systems and are building elaborate management systems around processes [24]. Mature standards and regulations describe specific personnel responsibilities that cut across several business units and require comprehensive documentation to demonstrate how personnel decisions implement standards and regulations [16].

### 2.3.5.3 Stake Holder

Stakeholders are persons or organizations (legal entities such as companies, standards bodies) who have valid interest in the system. They are all those who are involved in a project and have some interest in the software to be developed, and may vary from one project to another [24]. They may be affected by it either directly or indirectly. Stakeholder can be categorized into two. The first being the user that use the systems and secondly, the user that do not use the systems. The user that uses the system is normally called the end-user. This user will give the requirement base on their concern. Normally, they want a system that is easy to use, easy to manage and easy to monitor [26]. The users that do not use the systems will give requirements based on scenario, experience, case study and bench marking. Software practitioners need the knowledge of RE, in gaining an understanding of the users' needs, identify the system owners' needs, analysing, categorizing and documenting RE processes.

#### 2.3.5.4 Developers

A developer is the IT personnel who act as a middle person between the stake holder and management. The developers align both the user needs and come out with the big picture and present to management and stake holder. They are the ones who use the requirements to build the actual systems. They have technical knowledge of the tools and technology to use in building the system [25]

#### 2.3.5.5 Technology

The development of software projects are tightly involved with the IT infrastructure, software and hardware. The success of the software projects are dependent on the current infrastructure and can map with the development of the software project [25]

## 2.4 Requirement Engineering (RE)

RE is the branch of software Engineering concerned with the real world goals for Functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families. The work in the RE is related to the analysis of the system boundaries and the system characteristics. RE is a repeatable and systematic technique. In each and every phase of the RE lifecycle, the requirements are analyzed and evaluated to find the consistency and completeness of the requirement [26]. The requirements that are gathered from this process are applicable to whole system and not only for a single component.

Requirements are a specification of what should be implemented; they are descriptions of how the system should behave, or of a system property or attribute. The aim of RE is to help to know what to build before system development starts to prevent costly rework [26]. RE is one of the most crucial steps in software development process.

There are many problems due to usage of wrong requirements . These problems include

a) Delayed and over budget projects
b) The product does not reach the intended purpose. The customers, who are actually paying for the system, are not satisfied.
c) The errors encountered in the development of the system, is the reason for the problems in using the system.

d) The continuous use of such system makes it error prone and thus increases the cost of the maintenance

### 2.4.1 Types of requirements

Requirement Engineering is a sub discipline of system engineering and software engineering that is concerned with determining the goals functions and constraints of hardware and software systems. A project needs to address three levels of requirements; business requirement, user requirement and system requirement which will come from different sources at different project stages [27].

#### 2.4.1.1 Business Requirements

The business requirements describe the task or business processes which the user will be able to perform with the product.

#### 2.4.1.2 User requirements

User requirements describe the expected services from the system, constraints on achieving them and the way the system provides the requirements [27]. It must be written in such a way, that it must be understandable by a person without technical experience and background knowledge. These requirements are generally defined using the external activities and behavior of the system and is never defined based on system design or implementation.

#### 2.4.1.3 System Requirements

System requirements provide the in-depth knowledge of the user requirements. System requirements are basic principles that should be followed to design the system architecture. The software engineer should analyze these requirements to know about what exactly has to be implemented and provided in the proposed system. This is very important and useful to make an agreement or the contract for the implementation of the system [27]. System requirements are classified into different types

#### 2.4.1.4 Functional Requirements

Functional Requirement (FR) is the requirements that are directly related to what the system must do. It describes the specific system behavior that must be carried out and can be found in a software requirements specification (SRS). These requirements mainly depend on the users of the system and the type of software that is developed. These are the functions/ services that define [27]

a) What the system is expected to provide
b) How the system should respond or react to particular input or situation
c) What the system should not do
d) Constraints on implementing the above said requirements

#### 2.4.1.5 Non-Functional Requirements (NFRs)

Non-Functional Requirement can be seen as how the software complies with what the software must do [28]. These requirements define the effectiveness of the function provided by the system. They are not provided by the system, but they affect the functions provided by the system. Any system that does not provide a reliable service and also security measures against any threats will not be considered as a success. These requirements form the basis of the quality of the system. For example, a banking system that satisfies every functional requirement, and does not provide any non-functional requirements is sure to fail. Thus, the failures of the non-functional requirements depend on the items listed below. These items are;

a) Safety and security measures provided by the system
b) Reliability and efficiency of the system
c) Portability and integrity of the system
d) Memory used and cost effectiveness of the system

## 2.5 Requirement Engineering Process

A process is organizing a set of activities; it is a continuous transformation of input to output activities. Requirements engineering is also an organized and structured process with the set of activities to transform into out followed elicitation, validation and maintaining the requirements [28]. Many researches have been done on requirement engineering process. RE has various phases during implementation. These phases are described below:

### 2.5.1 Requirement Elicitation

The RE process starts with requirement elicitation. Requirement elicitation is a process of seeking, uncovering, acquiring and elaborating requirements for developing a computer- based system [10]. The requirements elicitation process involves a set of activities that must allow for communication with all relevant stake holders [28].

Requirements elicitation comprises activities that enable the understanding of the goals, objectives, and motives for building a proposed software system. Elicitation also involves identifying the requirements that the resulting system must satisfy in order to achieve these goals. The requirements to be elicited may range from modifications to well-understood problems and systems (e.g., software upgrades), to hazy understandings of new problems being automated, to relatively unconstrained requirements that are open to innovation (e.g., mass-market software). As such, most of the research in elicitation focuses on technologies for improving the precision, accuracy, and variety of the requirements details [29]:

a) Techniques for identifying stakeholders help to ensure that everyone who may be affected by the software is consulted during elicitation.
b) Analogical techniques, like metaphors and personas help stakeholders to consider more deeply and be more precise about their requirements.
c) Contextual and personal RE techniques analyze stakeholders' requirements with respect to a particular context, environment, and perhaps individual user, to help ensure that the eventual system is fit for use in that environment.
d) Techniques for inventing requirements, like brainstorming and creativity workshops, help to identify nonessential requirements that make the final product more appealing.
e) Feedback techniques use models , model animations , simulation , and storyboards to elicit positive and negative feedback on early representations of the proposed system [31,33]

### 2.5.2 Requirement Modeling

In requirements modeling, a project's requirements or specification is expressed in terms of one or more models. Models are refined in several steps by adding details about functionalities and hardware characteristics. At each step, functional requirements can be traced to ensure a correct refinement [28]. In contrast to models developed during elicitation, late-phase requirements models tend to be more precise, complete, and unambiguous. The process of creating precise models helps to evoke details that were missed in the initial elicitation. The resulting (more complete) models can be used to communicate the requirements to downstream developers.

Modeling notations help to raise the level of abstraction in requirements descriptions by providing a vocabulary and structural rules that more closely match the entities, relationships, behavior, and constraints of the problem being modeled. Each modeling notation is designed to elicit or record specific details about the requirements, such as what data the software is to maintain, functions on the data, responses to inputs, or properties about data or behavior.

### 2.5.3 Requirement Analysis

This is the process of studying, determining, and documenting user needs and expectations of the software system to be designed that solves a particular problem. Most of the research in requirements analysis focuses on new or improved techniques for evaluating the quality of recorded requirements. Some analyses look for well-formed errors in requirements, where an "error" can be ambiguity, inconsistency, or incompleteness. Other analyses look for anomalies, such as unknown interactions among requirements, possible obstacles to requirements satisfaction or missing assumptions [28]. Both types of analyses reveal misunderstandings or questions about the requirements that usually call for further elicitation. Requirements analysis also includes techniques, such as risk analysis and impact analysis that help specifiers to better understand the requirements, their interrelationships, and their potential consequences, so that specifiers can make more-informed decisions. As other examples, prioritization, visualization, and analysis techniques help a manager to select an optimal combination of requirements to be implemented, or to identify acceptable off-the-shelf solutions.

### 2.5.4 Requirement Communication

The communication problems occur across different organizational units and the decisions are not documented, therefore it is hard to know who is accountable for decision

### 2.5.5 Requirement Negotiation

Inconsistencies and conflicts discovered during analysis need to be resolved. The analysts and stake holders consider the problematic requirements to try to reach a consensus about their resolution and hence reach acceptable "win" conditions for all stakeholders. These trade-offs may necessitate the elicitation of further requirement information.

### 2.5.6 Requirement Specification

Requirements expressed in a more precise way, sometimes as a documentation of the external behavior of the system

### 2.5.7 Requirement Documentation

Requirement documentation is an essential and integral part of the software requirements process [10]. The term 'documentation' refers to both the process of documenting the requirements and the resulting work product, the requirements document it. The purpose of requirements documentation is to communicate requirements between stakeholders and developers

### 2.5.8 Requirement Verification

Verification is described as the comparison of the system (or the developmental artifacts) with its requirements through the use of examinations, analysis, demonstrations, tests, or other objective evidence. Verification techniques can be used to prove that the software specification meets these requirements. Such proofs often take the form of checking that a specification model satisfies some constraint. For example, model checking [28], checks behavioral models against temporal-logic properties about execution traces; and model satisfiability checks that there exist valid instantiations of constrained object models, and that operations on object models preserve invariants [26].

### 2.5.9 Requirement Validation:

Validation is described as the comparison of the completed system (artefacts) with the intended mission or purpose of the system [29]. The purpose of the requirements validation is to certify that the requirements are an acceptable description of the system to be implemented. Inputs for the validation process are the requirements documents, or organizational standards, and organizational knowledge. Requirements validation ensures that models and documentation accurately express the stakeholders' needs. As such, validation usually requires stakeholders to be directly involved in reviewing the requirements artifacts.

### 2.5.10 Requirements Management

All experts agree on Requirements Management as a key of a good project management, and all actors are supposed to link their work to requirement. The goal of requirement management is to capture, store, disseminate, and manage information. Requirements management includes all activities concerned with change & version control, requirements tracing, AND requirements status tracking. Requirement traceability provides relationships between requirements design and implementation of a system in order to manage changes to a system.

Requirements management is an umbrella activity that comprises a number of tasks related to the management of requirements, including the evolution of requirements over time and across product families. Of particular interest are tools and techniques to ease, and partially automate, the task of identifying and documenting traceability links among requirements artifacts and between requirements and downstream artifacts. Also included are analyses that determine the maturity and stability of elicited requirements, so that the requirements most likely to change can be isolated [29]. Lastly, the basic management of requirements has become a challenge and has inspired research on techniques to organize large numbers of requirements that are globally distributed, and that are at different phases in development in different product variants.
Unlike the software engineering process, Requirement engineering is also made up of different activities that connect, interact and lead one another to form a whole requirement engineering life cycle. The life cycle is represented in the figure 2.4

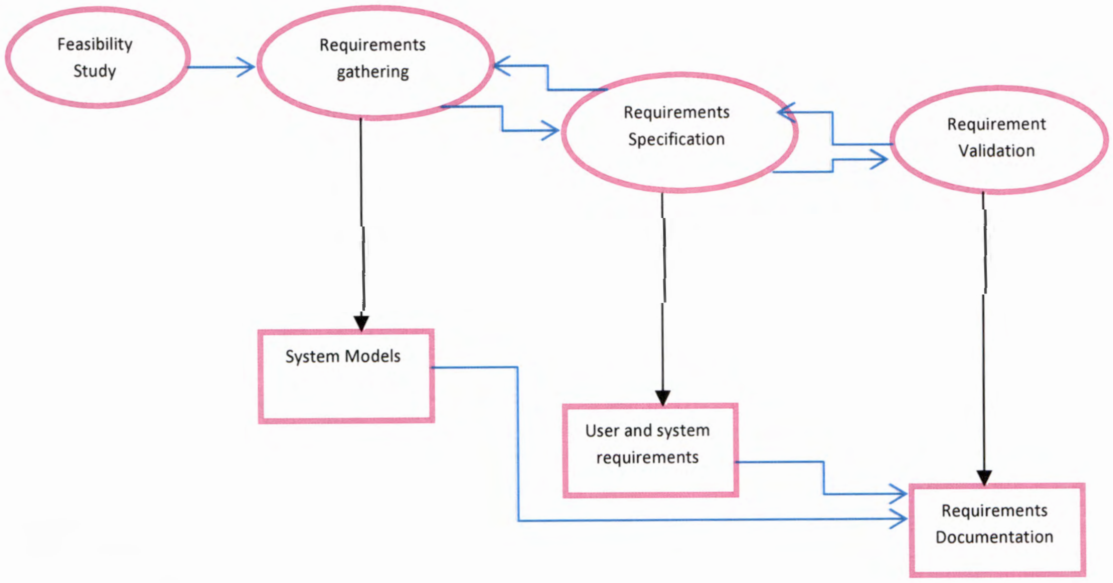

Feasibility Study
Requirements gathering
Requirements Specification
Requirement Validation
System Models
User and system requirements
Requirements Documentation

**Figure 2.4: Requirement Engineering Process**

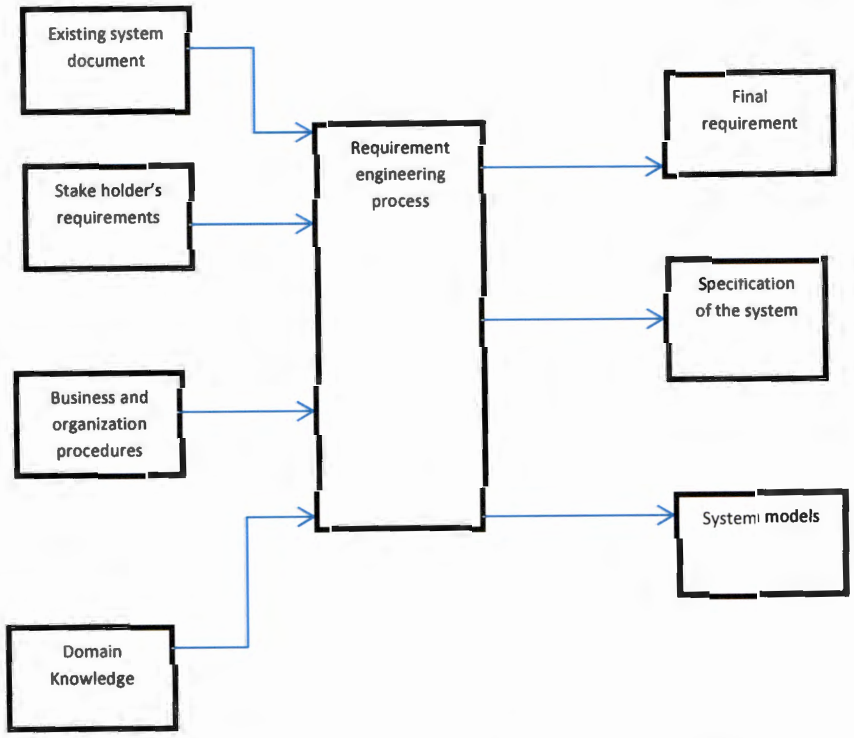
The detailed background theory of the activities showed in figure 2.4 describes planning and scheduling of the activities, inputs and outputs of every activity, tools used to perform each activity. The performance of the activities depends mostly on the people who are in the requirements engineering process; they will decide the major issues like where and when to perform the activities. The requirement engineers will decide the usage of the different available resources depending on the situation and the necessity.

The requirement engineering process is an input and output activity. This is shown in figure 2.5. It mainly depends on four elements to perform requirements engineering process. These elements are known as the inputs of the requirements engineering process and are listed below [29];

a) Existing system document
b) User and stake holders requirements
c) Organization and business procedures
d) Domain knowledge

Requirement engineering process have the under listed outputs;

a) Final requirements
b) Specification of system
c) System models



Existing system document
Stake holder's requirements
Business and organization procedures
Domain Knowledge
Requirement engineering process
Final requirement
Specification of the system
Systemı models

**Figure2.5: RE Process Input/output Process Representation**

## 2.6 Requirements Engineering Tools and Techniques

In RE, selection of pertinent tools and techniques in accordance with the type and complexity of the project is fundamental to eliciting requirement. This section outlines prominent RE tools and technique along with their role. These techniques are by and large classified into four main categories namely, classic/traditional techniques, cognitive techniques, modern and group elicitation techniques and contextual techniques. Each of these categories consists of a set of various techniques that are grouped together on the basis of their common characteristic and peculiarities [30].

### 2.6.1 Classic/Traditional Techniques

a) Interviews: Interviews is the common and popular method used by the requirement engineers to elicit system requirements and comprehend objectives of the system through verbal conversation with the stakeholders. Interviews could be structured or closed (i.e., in the form of predefined questions), semi-structured (i.e., a blend of predefine and unplanned questions) and unstructured or open (i.e., an informal interview that does not involve predefined questions). The first two approaches largely aim towards acquiring quantitative data, whereas the later approach attributes to understand user expectations through open discussions with the stakeholders and acquire qualitative data.

b) Surveys: The survey techniques are used to get large set of requirements from a larger population that may scattered on disparate geographical locations. Surveys collect information from large number of users and it is quite economical and rapid to analyze the data through planned surveys.

c) Questionnaires: The questionnaire is a method of requirement elicitation which is simple and requires lesser time and cost. To get precise results, the questionnaire should be clear, concise and structured to obtain genuine user requirements, objective and constraints. However, this technique lacks in the mechanism to seek users' clarification on the topic.

d) Task Analysis: This technique entails constructing top-down tasks hierarchy of the system to find out the knowledge used or required in the development of the system. Using this hierarchy, the task and sub-tasks are placed at different levels in a tree structure.

e) Domain Analysis: Domain analysis is used to gather early requirements and capture a bird eye view of the domain knowledge by investigating the existing applications and related documentation [30]. Usually this technique is used by the domain experts to study the domain area thoroughly. It is helpful in eliciting requirement from design documents, instruction manuals, templates and forms either used in the existing system or in the current business processes. Domain analysis also encompasses the domain knowledge and its reusable concepts and components. Mostly, this technique is used when project involves replacement or enhancement in the existing legacy system.

f) Introspection: Introspection is a preparatory step in requirement elicitation where requirement engineers use their experience and expertise to acquire requirements of the stakeholders in terms of their expectations towards the new system. However, this technique mainly necessitates requirement analysts to have a massive experience in this area. It is very effective when analysts are well-known of the domain and goal of the system as well as experts in business processes that users ordinarily perform.

### 2.6.2 Cognitive Techniques [31]

a) Card Sorting: In this technique, set of cards are sorted according to the name of domain entities by the customer/stakeholders along with the description of the criterion according to which the requirements elicitation cards are sorted. Card sorting helps in prioritizing most important requirements by ordering the cards. To make this technique more effective, it is important that all the essential entities are included in the process and requires that both the analyst and participants have sufficient knowledge of domain; otherwise, this technique produces wrong results. If domain knowledge is not well-known then group work is relatively more effective than the card sorting technique [31].

b) Class Responsibility Collaboration (CRC): It is a derived technique of card sorting and is used to represent software requirement in the form of classes where each class has its corresponding assigned responsibility to process the user requirements. CRC demonstrate relationship among classes and provides high level of abstraction. However, CRC cards are limited in delineating details about the software elicitation.

c) Laddering: Laddering technique aims at collecting clear answers for a series of questions from the stakeholders followed by arranging them in hierarchical order which is easy to understand and useful to prioritize the stakeholders' needs. The domain information of the stakeholder plays a very important role for the success of this technique. If requirements are too large then it becomes complex and hard to perform modifications like adding or deleting requirements anywhere in the ladder.

d) Repertory Grids: In repertory grid technique, the stakeholders are requested to build attribute and assign appropriate values to the set of specific domain entities on a grid and store requirements in the form of matrix. It involves element categorization, ordering the defined categories and assigning suitable variable with their corresponding values. Repertory grid is useful to identify similarity and differences between different information domains. Traceability also becomes easy through this technique. Since repertory grid is more meticulous than card sorting and lesser than laddering, therefore, the efficacy of repertory grids is inadequate to delineate specific distinctiveness for the complex requirements [31].

### 2.6.3 Modern and Group Elicitation Techniques

These modern and group elicitation techniques are part of the newer ways of elicitating requirements. They are described below as follows [32]:

a) Group Work: This technique is used to elicit the requirements of the system by inviting different stakeholders in a group meeting. This technique is effective to elicit requirements and resolving conflicts among the stakeholders by discussing all aspect of requirements with proper suggestions by the group members in a cooperative environment. However, it requires a lot of effort to conduct such meeting as it is always difficult to get hold of all the stakeholders at the same time.

b) Brainstorming: Brainstorming is used to generate numerous ideas in a shorter time span regardless of focusing on a specific issue through informal discussions amongst the participant of different stakeholder groups. It is mostly used in innovative type of projects where participants share their ideas on the basis of their experience and personal research about the project. The key disadvantage of brainstorming is that it cannot be effectively used to resolve major issues.

c) Joint Application Development (JAD): JAD is a business analysis approach for rapid decision making and to solve a problem quickly where a large number of stakeholders are engaged through open discussion. It is an agile approach used to elicit most of the requirements and their changeability. JAD is a structured approach where all steps, actions and roles of participants are defined for the session. Since the main goals of the system are already established before the stakeholders participate in discussion, therefore it differentiates from the brainstorming. The major focus of JAD session is to focus on the needs and desires of the business and users, but not on the technical issues. Due to its agile nature, sometimes it is incapable to validate the requirements. Also, it requires requirement engineers to possess vast experience and expertise.

d) Requirements Workshops: Requirement workshops are organized to elicit requirements for the project from the stakeholder. As compare to brainstorming and group meetings, the requirement workshops are able to provide a complete set of requirements but are relatively slow in the process of elicitation. As requirements elicited from this technique are collected after a multiple sessions, therefore, the resulted requirements are unchangeable. It is considered as a cost effective technique in terms of time and money and is feasible for only large and complex type of projects .

e) Protocol Analysis: In protocol analysis, the participants perform an activity to discuss the customer requirements while talking loudly. This technique facilitates active participation of all the key stakeholders. For targeted system, the protocol analysis can provide specific information and rationale of the processes to the analyst. Sometime talking through operation, this technique may not provide true picture of requirements and unable to completely represent the real processes.

f) Prototyping: Prototype is an early product version which is launched so that customers can get experience with it and can suggest their required requirements for the next version. This response/feedback is considered as additional requirements and helps to further investigate the possible solutions. Prototyping is a useful technique to develop novel applications and to build GUI interface. This technique is used with the combination of other requirement engineering techniques like interviews and JAD. Conversely, potential hazards in prototyping are that the user often resist changes if they had become used to a specific kind of the system as well as it is also expensive in terms of time and cost.

g) Use cases: This technique intends at defining the requirements by portraying complete flow of events to the stakeholders in the form of a story telling style. Use cases are informal and easy to use that help understanding the requirements and validating them with stakeholders.

h) Scenarios: Scenarios are used to find and prepare the narrative and detailed descriptions of current and future processes required for developing the software project. Scenarios are commonly used after collecting the initial requirements. Scenarios also define the actions and interactions between user and the system. Scenarios are useful to validate requirements and develop test cases

### 2.6.4 Contextual Techniques

a) Ethnography: In social context, ethnography is the study how people understand their problems and perceive their solutions. In requirement engineering context, the ethnography is a technique to find out how people discern their needs to be met from the software. This technique is useful for the collection of quality attribute like usability and efficiency from the peoples and these attributes are essentials for the success of project [3].

b) Observation/Social Analysis: Observation (also known as social analysis) is one of the types of ethnographic technique in which requirement engineer visits and observes the customer's environment where software services are required to be performed [6]. Alongside, the software engineer also observes the existing processes and it makes this technique more authentic as requirement engineers directly visits and observers the entire environment and verifies and validates the requirements. Observations are usually coined with other requirement engineering tools such as interviews and task analysis. However, observations become much expensive technique when huge travelling costs are involved [3, 7].

## 2.7 International standards for RE

Good quality products and services sometimes originated from organizations characterized by low maturity level, resulting from the relatively low number of IT best practices that had been implemented, whereas in other cases, organizations characterized by a higher maturity level generated disappointing results . Some of the related standards are described below [33]:

a) **ANSI/EIA-632**: The system design must meet stakeholder needs and expectations. There are 33 process requirement definitions in achieving good product .

b) **IEEE-1220**: Requirements analysis establishes system capabilities and product performance and defines the operational environments, human and system interfaces, physical characteristics, and other constraints that impact design solutions. The project team conducts various tradeoffs and risk analyses to identify and resolve conflicts, ideally resulting in a requirements baseline that balances an operational view, how system product serve the users a functional view, what the product do; and a design view design considerations

c) **ISO**: ISO 9001:2000 is a model which deals with requirements for quality management systems. ISO/IEC 15504-2, which focuses on defining requirements for performing an assessment. On the definition, ISO/IEC 15504-7 focuses on the processes used in developing a product or rendering a service. The value of introducing ISO-15288 as a means for defining the area of concern is that it recognizes the actual breadth of systems engineering practice through requirement analysis

d) **CMMI**: CMMI can be used to guide process improvement across a project, a division, or an entire organization. CMMI for development is a reference model that covers the development and maintenance activities applied to both products and services. In other words it focused Requirement development and Requirements Management. The purpose of CMMI for development is to help organizations improve their development and maintenance processes for both products and services

e) **MIL-STD-499C**: defines the interdisciplinary tasks that are required throughout a system life cycle to transform the customer needs into system solution. The process is through system requirements analysis and validation

## 2.8 Chapter Summary

This chapter analyzed literature related to this research to get a better understanding of the topic. Keywords to be used in the research were defined and various frameworks in RE was also reviewed. We also looked at the tools and processes of RE as it affect software development initiatives. Software process improvement methods were reviewed, as well as international standards guiding the use of software process improvement method and RE processes.

# Chapter 3

## Analysis of Findings in Requirement Engineering Processes

### 3.1 Chapter Overview

This chapter focuses on the methodology used during the course of this research thesis. It starts with a brief analysis of state-of-the-art on RE, as it relates to process improvement in software engineering. In addition, this chapter consists of methods used during experimental survey, the selection of the methods used in analyzing the data collected, and the methods used in presenting the results obtained, as reported in chapter 4.

### 3.2 State- of-art on RE Methods and Processes

Table 3.1 below summarizes the state of the art on RE methods with the aim of showing their contributions and their shortcomings.

**Table 3.1: Analysis of Previous work**

| Researcher | Contribution | Description |
|---|---|---|
| **Yudistira Asnar.et al. 2011 [28]** | Goal-driven risk assessment in requirements engineering | Their work extends the Tropos goal modeling formal framework proposing new concepts, qualitative reasoning techniques, and methodological procedures. |
| **Grzogorz Loniewski. et al. 2011 [29]** | An architecture-oriented model-driven requirements engineering approach | They created a methodological approach for Architecture-oriented Model- Driven Requirements Engineering. their work presented the classification of a process which is based on the OpenUP method, including its activities, roles, and work products |
| **Tao Yue. et al. 2011 [30]** | A systematic review of transformation approaches between user requirements and analysis models | They did a systematic review on existing literature works that transform textual requirements into analysis models, highlighted open issues, and provided suggestions on potential directions on textual requirements |
| **Andy J. Nolan. et al. 2011 [31]** | Managing requirements uncertainty in engine control systems development | Evaluates the impact of not managing uncertainties and describes how Rolls-Royce uses Requirements Uncertainty Analysis to reduce this impact. They further summarised the findings from an Six Sigma into requirements uncertainty and provided an overview of the technique now used to identify and monitor uncertainty through a project life |
| **Stefan Hallerstede. et al. 2013 [32]** | A Method and Tool for Tracing Requirements into Specifications | They described an incremental approach to requirements modelling and validation that incorporates formal and informal reasoning and created an approach to requirements tracing that delivers the necessary connection that links the reasoning to the system description. |
| **Martin Kost. et al. 2011 [33]** | Privacy Verification Using Ontologies | A comprehensive approach for privacy centred on requirement engineering, implementation, and verification and provided a systematic approach that better protects privacy in future information systems |
| **Golnaz Elahi . et al. 2010 [34]** | A vulnerability-centric requirements engineering framework: analyzing security attacks, countermeasures, and requirements based on vulnerabilities | A methodological framework for security requirements elicitation and analysis centred on vulnerabilities. This framework offers modelling and analysis facilities to assist system designers in analysing vulnerabilities and their effects on the system. it further proposes a qualitative goal model evaluation analysis for assessing the risks of vulnerabilities exploitation and analysing the impact of countermeasures on such risks. |
| **Edith Felix. et al. 2011 [35]** | Managing changes with legacy security engineering processes | Provided a solution for integrated methodology in which risk, security requirements and architectural solutions are addressed within the same tooling environment and changes can be easily propagated |
| **Donna H. Rhodes. et al. 2010 [36]** | Five aspects of engineering complex systems emerging constructs and methods | There work describes the use of the framework for structuring engineering methods and assessing the balance of methods under development with illustration of its use for descriptive purposes, and as applied to development of a comprehensive research portfolio for evolving advanced engineering methods. |

## 3.3 Experimental Approach

There are two main approaches that can be used to conduct this type of research. The two approaches are: a). qualitative and b). quantitative methods [34]. Either of these two methods or a combination of the two is acceptable. However, the research reported in this dissertation takes the quantitative approach through the use of questionnaires for the survey [35, 36].

### 3.3.1 Summary of Questionnaire

Table 3.2 shown below contains a summary of the questions used for this research

**Table 3.2: Summary of Questionnaire**

| | |
|---|---|
| 1 | **How long have you been working (Years of working Experience) : less than 3yrs ( ), 3 – 5yrs( ), more than 5yrs ( )** |
| 2 | How long have you been working in a software development organization: less than 3yrs(), 3 – 5yrs ( ), more than 5yrs ( ) |
| 3 | Do you use any process to obtain requirements in a project Yes ( ) No ( ) I do not Know ( ) |
| 4 | Do you like to carry out feasibility study before starting a new project? Yes ( ) No ( ) I do not Know ( ) |
| 5 | Do you like to carry requirements elicitation before going to start a project? Yes ( ) No ( )I do not Know ( ) |
| 6 | Have you ever used any requirement engineering methods in any form during your work activity? |
| 7 | From your perspective, is the requirements engineering important in any software development project?<br>Yes ( ) No ( )I do not Know ( ) |
| 8 | Tick the methods which are effective for Requirement engineering (your personal opinion, multiple selections allowed. Use '*' to indicate your selection)?<br>i. Requirement elicitation ( )<br>ii. Requirement analysis and negotiation ( )<br>iii. Requirement verification and validation ( )<br>iv. Requirement Documentation management ( ) |
| 9 | Tick the methods which are effective for Requirement Elicitation (your personal opinion, multiple selections allowed. Use '*' to indicate your selection)?<br>i. Interviews ( )<br>ii. Questionnaires ( )<br>iii. Observations ( )<br>iv. Social analysis ( )<br>v. Prototype ( )<br>vi. Scenario ( )<br>vii. Brain Storming ( )<br>viii. JAD ( )<br>ix. User centered design ( ) |

| | |
|---|---|
| 10 | Tick the popular methods which are used in your companies as a requirements analysis and negotiation tool (multiple selections allowed)?<br><br>i. AHP ( )<br>ii. Card Sorting ( )<br>iii. Designer Apprentice( )<br>iv. Goal Oriented Analysis ( )<br>v. Scenarios ( )<br>vi. Structured Analysis Development ( )<br>vii. System development Language ( )<br>viii. Contextual Enquiry ( )<br>ix. Fault Tree Analysis( )<br>x. Entity Relationship Diagram |
| 11 | Tick the popular methods which are used in Requirement Documentation your companies as a tool (multiple selections allowed)?<br><br>i. Structured Natural Language Specification( )<br>ii. Unified Modeling Language ( )<br>iii. System Development Language ( )<br>iv. User Story Card ( )<br>v. View Point Documentation ( )<br>vi. LOTOS ( )<br>vii. Service Definition Template ( )<br>viii. Document mining ( )<br>ix. State diagram ( ) |
| 12 | Tick the popular methods which are used in your companies as a requirements Verification and validation (multiple selections allowed)?<br><br>i. Request Inspection ( )<br>ii. Request Testing ( )<br>iii. Request Checklist ( )<br>iv. System Development Language ( )<br>v. View Point Base verification( )<br>vi. CO ( )<br>vii. Ethnography ( )<br>viii. Utility testing ( ) |

### 3.3.2 Participants and Sample Size

The survey was conducted among fifty professionals working in 10 different software development organizations in South Africa. These companies are situated in the four provinces namely western cape, Gauteng, Limpopo and KwaZulu-Natal as shown in Table 3.3. Emails were used for the distribution of the questionnaire because the respondents were located in four (4) different major provinces that are far apart from each other. Out of the fifty participants, forty five (45) answered the questions, while five (5) had not responded. Three (3) companies each, participated from Gauteng and Western-Cape Province while Limpopo and Kwa-Zulu Natal had two participating companies each. Table 3.3shows the distribution of participants according to their respective provinces

**Table 3.3: Participants according to Province**

| Province | Firms per province | No of questionnaires sent per province | Number of questionnaire received per province |
|---|---|---|---|
| **Limpopo** | 2 | 12 | 9 |
| **Gauteng** | 3 | 18 | 14 |
| **KZN** | 2 | 12 | 9 |
| **WC** | 3 | 18 | 13 |

In order to attain a dependable result, the number of total respondent becomes an important element. Due to the time, and cost constraint, we chose an appropriate size of 50 respondents was chosen. A larger number of about 100 respondents would have been more ideal, because a larger sample size will reduce the margin of error but was hindered by time and cost constraints.

**Table 3.4: No of Questionnaires Sent and Received based on Companies**

| Company List in each province | No of questionnaire sent | No of questionnaire returned |
|---|---|---|
| Gauteng A | 5 | 5 |
| Gauteng B | 5 | 4 |
| Gauteng C | 5 | 5 |
| W-Cape D | 5 | 4 |
| W-Cape E | 5 | 4 |
| W-Cape F | 5 | 5 |
| KZN G | 5 | 5 |
| KZN H | 5 | 4 |
| Limpopo I | 5 | 5 |
| Limpopo J | 5 | 4 |

Table 3.4 shows that all companies that participated in this survey received five (5) questionnaires each. The company Gauteng B returned four (4) questionnaires only, while the companies Gauteng A, and C returned all five (5) questionnaires. In Western Cape, the companies W-cape D, and W-Cape E returned four (4) questionnaires each, as well while the company W-Cape F returned five (5). The companies Limpopo J and the company KZN H

returned four (4) questionnaires each, while the companies Limpopo I and KZN G returned all five (5) questionnaires respectively. This shows that 45 out of 50 participants returned their various questionnaires. Consequently, we received 90% of all questionnaires sent to participants.

### 3.3.3 Questionnaire

The questions in the survey are aimed at gathering the perception on the various requirements engineering techniques, tools and problems. The questionnaire is divided into two sections. The first section describes the general information about the company while the second section describes the RE techniques used in the companies as shown in appendix A. Based on these sections, the project factors addressed by the questionnaire is discussed next, in this chapter.

#### 3.3.3.1 Educational Background

The participants were asked to indicate their educational background in terms of discipline. Fourteen (14) respondents ticked 'computer science', twenty (20) respondents, ticked 'Information technology', and three (3) ticked 'Business administration' while eight (8) respondents ticked 'others'. Category 'others' include any other professions involved in software development projects. This shows that most of the people working in software companies are familiar with software engineering methods and practices, which imply that they should be aware of RE engineering. This is shown in table 3.5

**Table 3.5: Educational Background**

| Educational Background | No of respondent |
|---|---|
| Computer Science | 14 |
| Business Administration | 3 |
| Information Technology | 20 |
| Others | 8 |
| Total Respondents | 45 |

#### 3.3.3.2 Years of Experience

The participants were asked to indicate the years of experience they have obtained while working, and they should also indicate the actual number of years, they have worked in a software development or information technology firm. The responses were categorized into less than 3 years, between 3 to 5 years and above 5 years. Of the forty five (45) respondents, Nineteen (19) respondents falls within the group of less than 3yrs, for the general work experience, while twenty four (24) respondents fall within this same group of IT experience in industries. Fifteen (15) respondents have worked between 3 to 5 years for general work experience while Thirteen (13) respondents have worked within information technology and software engineering sector. Eleven (11) respondents have above 5 years of experience within the general work, while nine (9) respondents have worked in IT/ SW engineering work environment.

Moreover, Table 3.6 shows that 24 respondents (53%) of those working have less work experience in software firms. At this rate, more than 50% of software development firms' work force is inexperienced in software development projects.

**Table 3.6: Years of Experience**

| Range of years (experience) | Number of participants (All work experience) | Number of participants (IT related Work experience) |
|---|---|---|
| Less than 3 years | 19 | 24 |
| 3 – 5 Years | 15 | 13 |
| Above 5 years | 11 | 9 |

### 3.3.3.3 Perception of the importance of RE

Knowing the importance of RE in software development increases the use of RE, and also improve the quality of software produced. The importance of RE is above average, as shown in table 3.7. This is because thirty (30) respondents indicated yes which is 67% of respondents, 11 respondents (24%) do not know if RE is important while only 4 respondents (9%) does not see the relevance of RE. This means that 33% of the respondents (addition of 24 % I don't know and 9 % of No) are not likely to use RE processes on their own, as they do not see the need or the relevance of using RE during software development projects.

Table 3.7: Importance of RE in Software Engineering Projects

| Type of Responses | Number of respondents |
|---|---|
| Yes | 30 |
| No | 4 |
| I do not know | 11 |

### 3.3.3.4 Popular techniques used in RE Elicitation

When asked to choose the most popular RE elicitation technique used within their organizations. Table 3.8 shows that forty one (41) respondents ticked questionnaires, thirty seven (37) respondents ticked interview and reused requirements respectively, thirty three (33) respondents ticked prototyping, thirty (30) respondents ticked social analysis and brain storming, twenty one (21) respondents ticked scenario, twenty (20) respondents ticked JAD and sixteen (16) respondents ticked UCD. This implies that questionnaire and reuse requirements are the most popularly used elicitation techniques while JAD and UCD are the least preferred requirement elicitation techniques.

Table 3.8: Popular Elicitation Techniques

| Respondents per tool | Actual respondents | Percentage (100%) |
|---|---|---|
| Interviews | 37 | 82 |
| Questionnaires | 41 | 91 |
| Social Analysis | 30 | 67 |
| Prototyping | 33 | 73 |
| Scenarios | 21 | 47 |
| Brain storming | 30 | 67 |
| JAD | 20 | 44 |
| UCD | 16 | 36 |
| Reused requirements | 37 | 82 |

### 3.3.3.5 Popular techniques used in RE Analysis and Negotiation

More also, table 3.9 shows the most popular requirement analysis and negotiation techniques, in use at their various firms. Forty two (42) respondents answered this question. Of the forty two (42) respondents, scenario is the most popular techniques used, as thirty (30) respondents ticked this option. Twenty four (24) respondents ticked Object Oriented Analysis, which makes it quite popular as well. Other popular techniques include structured analysis which twenty (22) respondents ticked, system development language which twenty two (22) respondents and goal oriented analysis which twenty one (21) respondents ticked. The least used technique is AHP as only nine (9) respondents ticked it as a popular technique.

**Table 3.9: Popular Techniques used in RE Analysis and Negotiation**

| Respondents per tool | Actual respondents |
|---|---|
| AHP | 9 |
| Card Sorting | 16 |
| Fault Tree Analysis | 12 |
| Goal Oriented Analysis | 21 |
| Object Oriented Analysis | 24 |
| Scenarios | 30 |
| Structured Analysis | 22 |
| System Development Language | 22 |
| View point Oriented Analysis | 17 |

### 3.3.3.6 Popular techniques used in RE Verification and Validation (VV)

The participants were asked to choose the most popular requirement engineering verification and validation techniques listed in the questionnaire. All forty five (45) respondents answered this question. Forty one (41) respondents ticked RI as the most popular technique, thirty seven (37) respondents ticked CO, thirty three (33) respondents ticked SDL, and thirty (30) respondents ticked RT while only twenty one (21) respondents ticked VPBV. This is shown in table 3.10

Table 3.10: Popular Techniques used in Verification and Validation

| Respondents per tool | Actual respondents |
|---|---|
| CO | 37 |
| RI | 41 |
| RT | 30 |
| SDL | 33 |
| VPBV | 21 |

### 3.3.3.7 Popular techniques used in RE Documentation

When asked to choose the most popular technique used during documentation, only forty one (41) respondents answered to this question. All forty one (41) respondents ticked UML from the list of techniques, thirty seven (37) respondents ticked SNLS, thirty (30) respondents ticked SDL, thirty three (33) respondents ticked state diagram, while twenty one (21) respondents ticked User Story Card. This shows that on the average, there is significant use of documentation process during software development

Table 3.11: Popular Documentation Techniques

| Respondents per tool | Actual respondents |
|---|---|
| SNLS | 37 |
| UML | 41 |
| SDL | 30 |
| State Diagram | 33 |
| User Story Card | 21 |

## 3.4 Requirement Engineering Process Framework (REPF)

This framework provides the mechanism for the construction of RE processes. It serves as a template which is a standard pattern that is applicable to most types of software development projects and can be seen as the superset of any RE process models. This is illustrated in figure 3.1

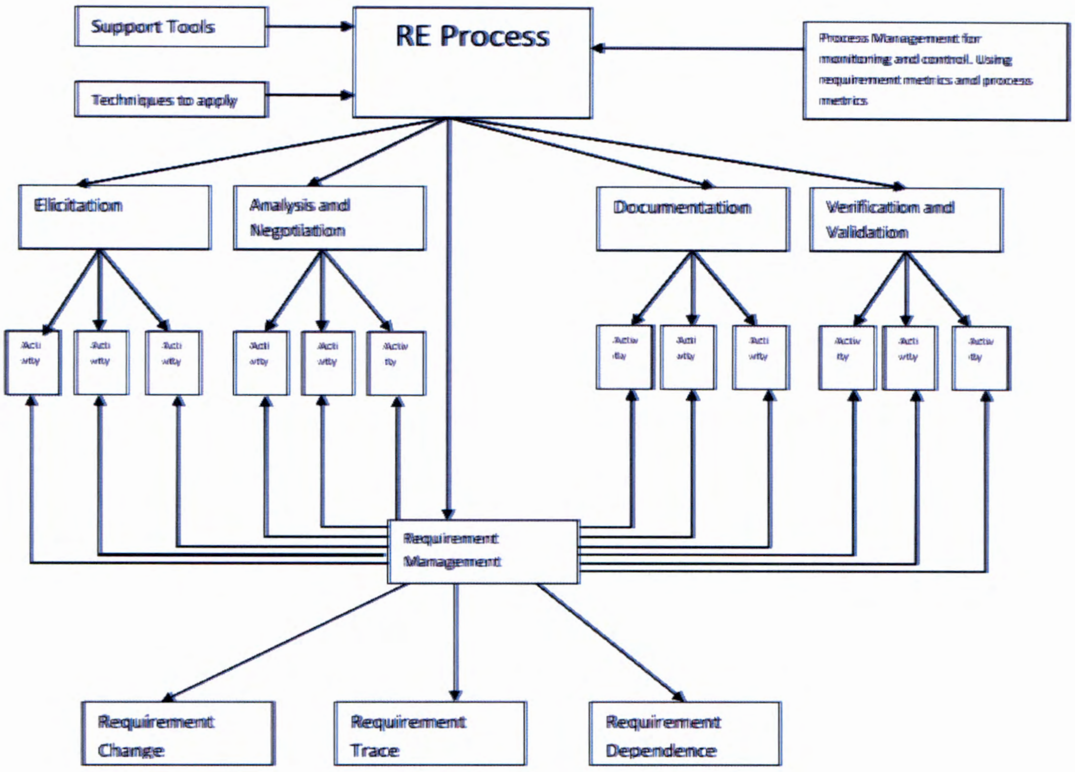


**Figure 3.1: RE Framework**

All the essential phases for a RE process is included and we present a graphical view in figure 3. 1. The framework consists of five distinct phases, which serve as the building blocks of the RE process. These phases includes; requirements elicitation, requirements analysis and negotiation, requirements documentation, requirements verification and validation, and requirements management. Each phase can be seen as a process and is composed of a number of activities. The five phases represent the five categories of RE activities. As illustrated in figure 3.1, requirements management is carried out in parallel with the other four phases. Requirements management considers planning, monitoring, and controlling the changes of requirements in the RE process.

### 3. 4.1 Framework Analysis

The framework analysis consists of five building blocks which include elicitation, analysis, and negotiation, verification and validation, documentation and management.

### 3.4.1.1 Requirement Elicitation Building Blocks

Elicitation building block is one of the segments of our Framework, it comprises of several activities that enable the understanding of the goals, objectives, and motives for building a proposed software system. Also, the building block, involves identifying the requirements that the resulting system must satisfy. In order to achieve these goals, it show cases a set of activities in a multi focused way, such that it encompasses all activities that can generically elicitate requirements within various software development projects. Table 3.12 shows a list of the activities that can be tailored to meet specific project

**Table 3.12: Activities in Elicitation Building Block**

| No | Activities |
|---|---|
| 1 | Defining the system's operating environment: |
| 2 | Use business concerns to drive requirements |
| 3 | Look for domain constraints |
| 4 | Record requirements rationale |
| 5 | Collect requirements form multiple viewpoints |
| 6 | Prototype poorly understood requirements |
| 7 | Use scenarios to elicit requirements |
| 8 | Define operational processes |
| 9 | Reuse requirements |
| 10 | Elicit non-functional requirements and system constraints |
| 11 | Elicit functional requirements |
| 12 | Elicit domain specific requirements |
| 13 | Identify initial goal of system |
| 14 | Formalize goals and identify the objects related to the goals |
| 15 | Elicit new goals through WHY questions. |
| 16 | Eliciting new goals through HOW questions. |

### 3.4.1.2 Analysis and Negotiation Building Block

Analysis and Negotiation building block is another major phase of the RE process. There are a lot of different views about the activities that should be carried out at this phase. Typically, there are two types of activities related to this phase. These activities include analysis for understanding, and analysis for design. In our framework, requirements analysis is defined as analysing requirements, and modelling requirements for understanding the system. Therefore, the activities related to the design or requirements verification are not included in the requirements analysis and negotiation phase. Currently, there are 30 activities included in the

Analysis and Negotiation building block. These activities are good practices in the requirements analysis and negotiation process as shown in table 3.13

**Table 3.13: Analysis and Negotiation Building Block**

| No | Activities |
|---|---|
| 1 | Define system boundaries |
| 2 | Provide software to support negotiations |
| 3 | Prioritize requirements |
| 4 | Develop complementary system models |
| 5 | Model the system's environment |
| 6 | Model the system architecture |
| 7 | Create safety requirements checklist |
| 8 | Classify requirements using a multi-dimensional approach |
| 9 | Use interaction matrices to find conflicts and overlaps |
| 10 | Use structured methods for system modeling |
| 11 | Use a data dictionary |
| 12 | Document the links between stakeholder requirements and system models |
| 13 | Identify and analyze hazards |
| 14 | Derive safety requirements from hazard analysis |
| 15 | Cross-check operational and functional requirements against safety requirements |
| 16 | Assess requirements risks |
| 17 | Model and understand the functional requirements |
| 18 | Negotiate with stakeholder to resolve the conflicts in the requirements |
| 19 | Understand non-functional requirements |
| 20 | Understand and model the relationship of requirements |
| 21 | Develop the test cases for the requirements with high business value. |
| 22 | Use formal specification to define requirements for safety-critical system |
| 23 | Use UML to model the requirements for the system. |
| 24 | Identify potential responsibility of each agent in the system |
| 25 | Derive agent interfaces for the system |
| 26 | Operationalize the goals |
| 27 | Clarifying and restating the requirements |
| 28 | Identify the minimum requirements that meet real needs |
| 29 | Use checklists for requirements analysis |
| 30 | Plan for conflicts and conflict resolution |

### 3.4.1.3 Documentation Building Block

Requirement documentation building block is an essential and integral part of this framework. The purpose of requirements documentation is to communicate requirements effectively. The requirements documentation phase includes all the activities related to the definition of the requirements, which includes the structure of the document, the notations used, and others. The requirements can be documented with informal, semi-formal or formal notations. The requirement documentation building block currently has 19 activities as shown in table 3.14

**Table 3.14: Activities in Documentation Building Block**

| | |
|---|---|
| 1 | Define a standard document |
| 2 | Explain how to use the document |
| 3 | Include a summary of the requirements |
| 4 | Make a business case for the system |
| 5 | Define specialized terms for documentation |
| 6 | Lay out the document for readability |
| 7 | Help readers find information |
| 8 | Make the document easy to change |
| 9 | Define standard templates for describing requirements |
| 10 | Use language simply, consistently and concisely |
| 11 | Use diagrams appropriately |
| 12 | Supplement natural language with other descriptions of requirements |
| 13 | Specify requirements quantitatively |
| 14 | Specify system using formal specification |
| 15 | Document the functional requirements |
| 16 | Document the non-functional requirements |
| 17 | Document the relationship among requirements |
| 18 | Document the requirements test cases |
| 19 | Use user story card to document requirements |

### 3.4.1.4 Verification and Validation Building Block

We extract the definition of Requirements Verification and Validation (VandV) from Kotonya [Kotonya, 1998] who sees verification and validation as inseparable, defines it as a phase for examining, verifying and validating requirements to determine that the software requirements are specified correctly and completely. The primary objectives of this phase are to: verify that requirements follow a defined standard Requirement, requirements are complete, consistent, and are clearly defined without ambiguity, To validate requirements with stakeholders is to ensure that a set of requirements are the real needs of stakeholders without having "not sure" requirements left in the requirements document.

Currently, there are 16 activities included in the Verification and Validation building block as shown in table 3.15

**Table 3.15: Activities in Verification and Validation Building Block**

| 1 | Check that the requirements document meets yours standards |
|---|---|
| 2 | Organize formal requirements inspections |
| 3 | Use multi-disciplinary teams to review requirements |
| 4 | Define verification checklists |
| 5 | Involve external reviewer in the validation process. |
| 6 | Use prototyping to animate requirements |
| 7 | Write a draft user manual |
| 8 | Propose requirements test cases |
| 9 | Paraphrase system models |
| 10 | Check the correctness and preciseness of requirements |
| 11 | Check the completeness of requirements |
| 12 | Check the unambiguity of requirements |
| 13 | Check the achievability and implementability |
| 14 | Check requirements interaction |
| 15 | Check the understandability of the requirements |
| 16 | Check requirements redundancy |

### 3.4.1.5 Requirements Management Building Block

Requirements Management building block serves as a key to good project management that captures, stores, disseminates, and manages information. Requirements management is a process which is carried out in all four phases discussed so far and it is mainly related to the activities of requirements change and management. It also includes version control, requirements tracing, and requirements status tracking. Impact analysis caused by requirements changes is also included in this phase. Thus, tool support is very important for effective requirements management. There are 19 activities in the Requirements Management building block as shown in table 3.4.5

**Table 3.16: Activities in RE Management Building Block**

| Number | Activity |
|---|---|
| 1 | Uniquely identify each requirement |
| 2 | Define policies for Requirements management |
| 3 | Define traceability policies |
| 4 | Maintain a traceability manual |
| 5 | Use a database to manage requirements |
| 6 | Define change management policies |
| 7 | Identify global system requirements |
| 8 | Identify volatile requirements |
| 9 | Record rejected requirements |
| 10 | Collect incident experience |
| 11 | Learn from incident experience |
| 12 | Establish an organizational safety culture |
| 13 | Manage the relationships between requirements |
| 14 | Manage dependencies between the requirements document and related documents |
| 15 | Identify the major metrics of RE process |
| 16 | Measuring the RE process based on the measurement definition |
| 17 | Monitoring the RE process based on the measurement |
| 18 | Define quality attributes of requirements |
| 19 | Manage requirements risk |

## 3.5 Chapter Summary

This chapter presents the methodology used in investigating the various variables in this research. We examined the literature review of RE techniques. We used experimental survey as a primary tool, in gathering information from the industry. In addition, we presented the analysis of the qualitative method which involves the use of questionnaires during experimental survey. The questionnaires sooths both time and cost requirements which was a primary factor during data collection. Furthermore, we designed and explained our RE framework, and also explained the set of activities that should be followed when implementing the framework

# Chapter 4

## RE Discussion of Findings

### 4.1 Chapter Overview

In this chapter, we discuss the findings obtained from the experimental survey through the use of questionnaires. Furthermore, for the purpose of explicitness and clarity, we represent the results obtained from the experimental survey using pie and bar chats, followed by a brief discussion about the findings.

### 4.2 Educational background

Figure 4.1 illustrates that the educational background of RE team, does not only consist of members with degrees in computer science and information technology. The RE team, also consist of experts with related experience, whose field of expertise constitute other professions. This means that, there are other professionals within the companies who are part of the RE team, and who are not computer science or information technology professionals.

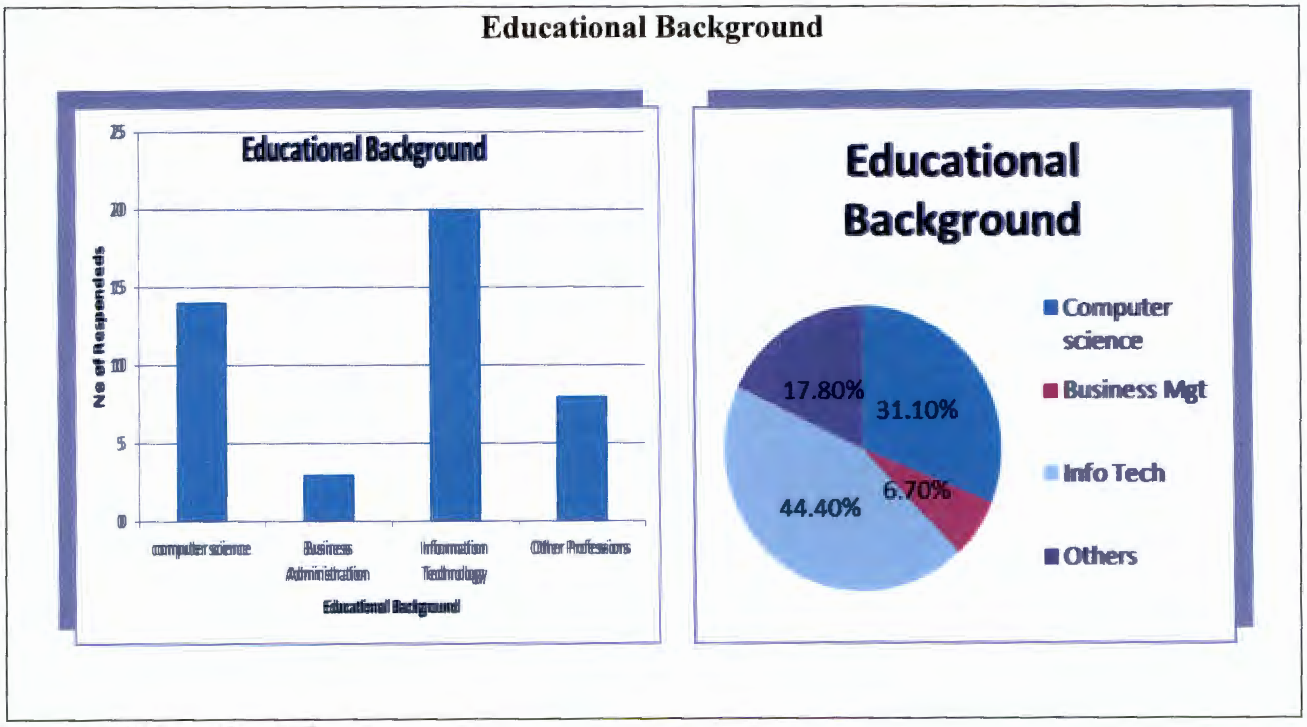


Figure 4.1: **Educational Background**

## 4.3 Years of Experience

In most software development firms, RE team members have two types of years of experience. This is a situation whereby some team members have worked in firms that are not related to software development and did not do any software development work. We refer to this group as general work experience group. While the team members with software development years of experience, we refer to them as information technology experience group.

**Respondents according to years of Experience**

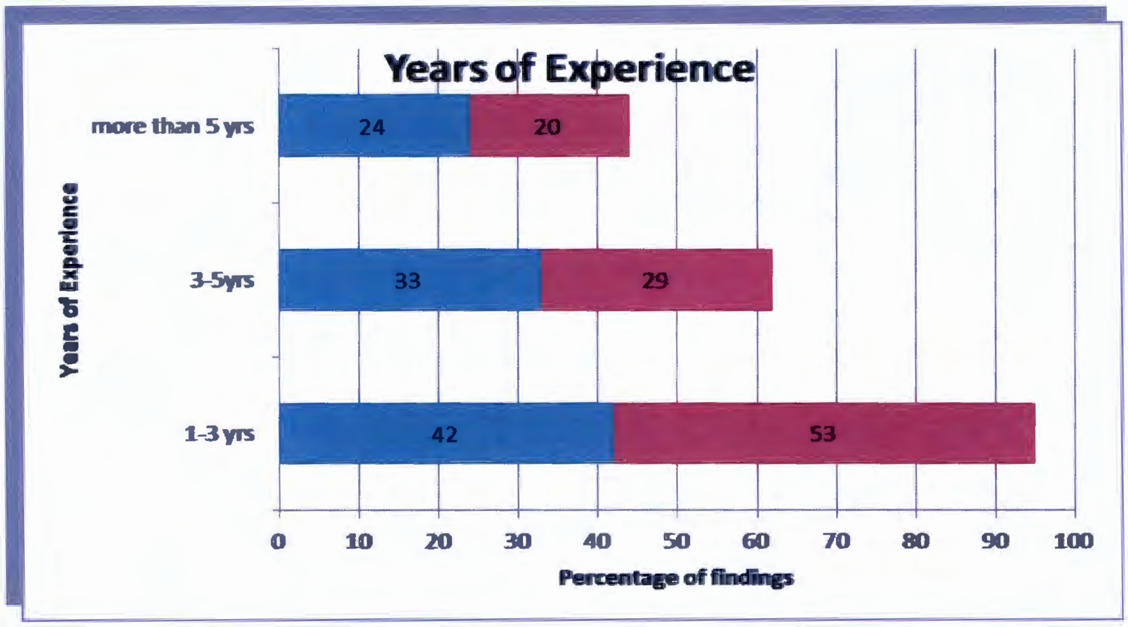


**Figure 4.2: Years of Experience**

The graph above in figure 4.2 shows that there are many professionals who have less than 3 years of experience in the software development field. Figure 4.2 further shows that 53% of those with IT experience have less than 3years of experience, 29% fall between 3-5years experience, while 20% of respondents, are team members with more than 5years experience. This is an indication that experience in software development is still at as low level and this inversely implies that software development in South Africa is at its developmental stage.

## 4.4 Importance of Requirements Engineering to software development

Knowing the importance of RE in software development, it increases the rate at which software developers, adequately implements RE processes.

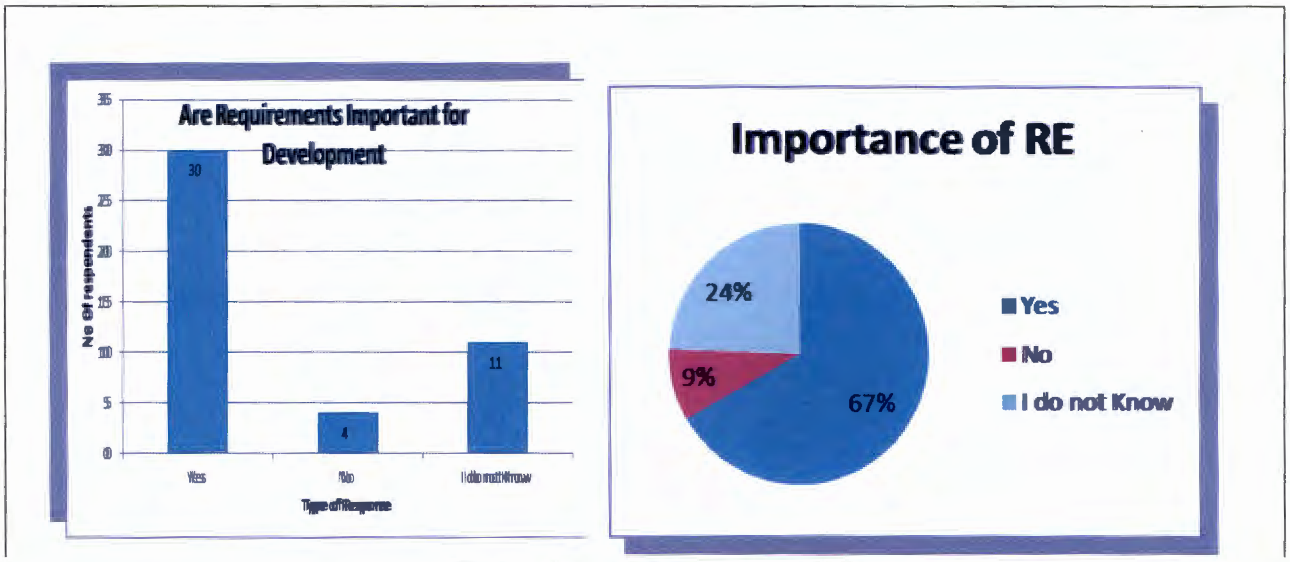


**Figure 4.3: Importance of RE**

Figure 4.3 shows that 67% of respondents indicated that RE is important during software development, 24% does not know, if RE is important or not important, while 9% indicated that RE is not important during software development. We could deduce from figure 4.3 that there is still high percentage of professional who does not know the importance of RE. We could further deduce that the level of RE awareness needs to increase. Furthermore, Figure 4.3 shows that most team members only follow instructions from senior professionals when implementing RE and are not really aware of the importance of RE process or why they are using it.

## 4.5 Comparison of Years of Experience with Importance of RE

Figure 4.4 shows the relationship between years of experience and the importance of requirement engineering during software development projects. Figure 4.4 further shows that all participants above 5 years of experience who understand the importance of RE in every software development project. We could also see that 73% of those that indicated "I do not know" fall within the group of "3-5 years" of experience, while 27% of those that said "I do not know" fall in the category of "less than 3 years" of experience. It is very important to note that, only those with less than 3 years of experience indicated "No" to importance of years of experience question. This analysis confirms that, the level of experience in software development and particularly, in RE, affects the perception on the importance of RE in software development. Therefore, we deduce that experience generally plays a major role in improving software development process, and system development managers have immerse responsibility to increase the awareness on the importance of RE within their organizations.

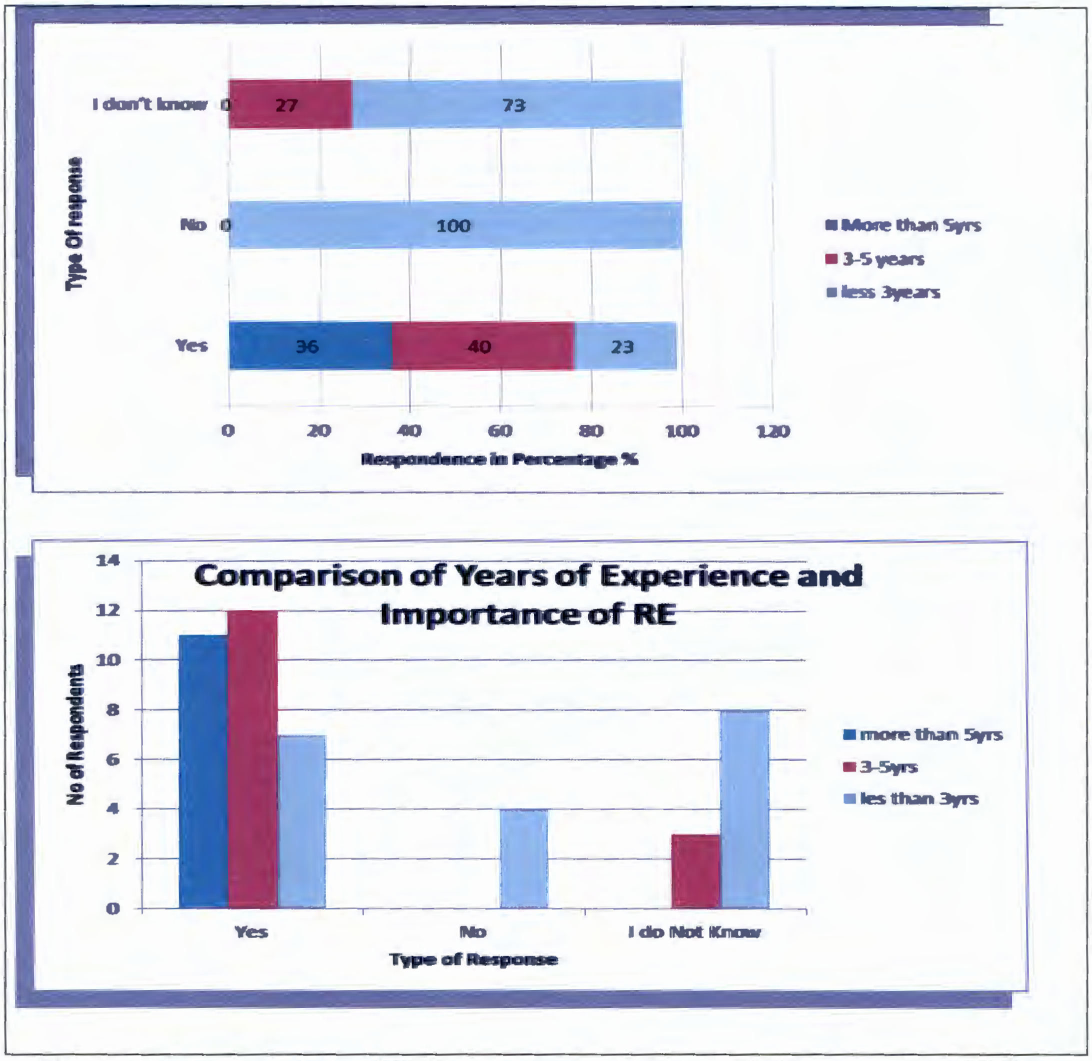


**Figure 4.4: Comparison OF Years of Experience with Importance of RE**

## 4.6 Use OF RE Process

Figure 4.5 illustrates the various requirement engineering processes implemented in organizations. It also shows that requirement elicitation process is extensively used by most firms in South Africa. Furthermore, we could deduce from figure 4.5, that software engineering firms implement one form of requirement elicitation technique during RE process. Analysis and negotiations is also considerably used but still needs more awareness. Verification and Validation, as well as Documentation are the least used process. We could therefore say that, years of experience and perception of importance also plays a role in limiting the use of other processes of RE. Therefore, more level of awareness need to be implemented in various organizations.

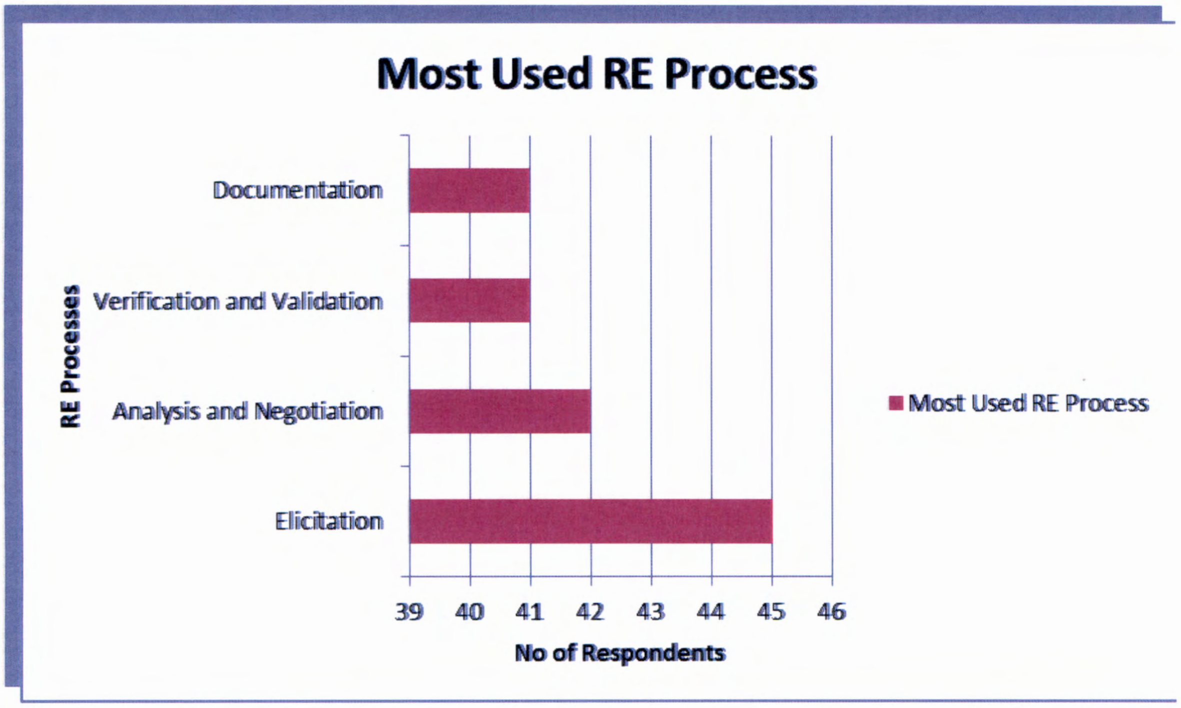


**Figure 4.5: RE Engineering Process**

## 4.7 Most Effective RE Tool

The respondents were asked to choose the most effective requirement elicitation technique from a list of elicitation techniques that are used within their organizations. From Figure 4.6, we could deduce that even though there are a lot of effective techniques, the use of questionnaire is considered the most effective with 91%. Figure 4.6 further shows that interviews and reuse requirements are also highly effective techniques with 80% each respectively. Other effective techniques are prototyping (73%), social analysis (65%), and brain storming (64%). This also shows that these techniques are also very effective because more than 50% respondents indicated that they are effective. The least effective techniques include scenarios (45%), JAD (44%), and UCD (35%). UCD is identified as the least effective technique.

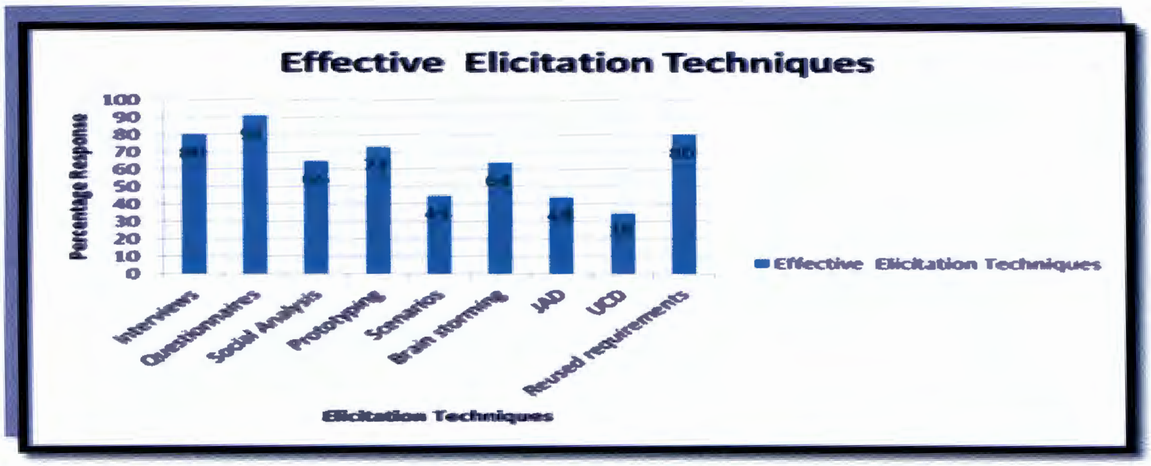


**Figure 4.6: Most Effective Elicitation Techniques**

Figure 4.7 presents a feedback on the most popularly used requirement elicitation techniques shows that elicitation techniques are used often within participating organizations but some techniques are more preferred than the other techniques. As illustrated in Figure 4.7, we could deduce that questionnaire with 91% is the most used elicitation technique within South African software development organizations. It is also important to note that a number of elicitation techniques are frequently used during software development as more than seven (7) types of elicitation techniques have a popularity percentage of above 50. The less popular ones fall between 47% and 22%. UCD is the least popular technique with 22%.

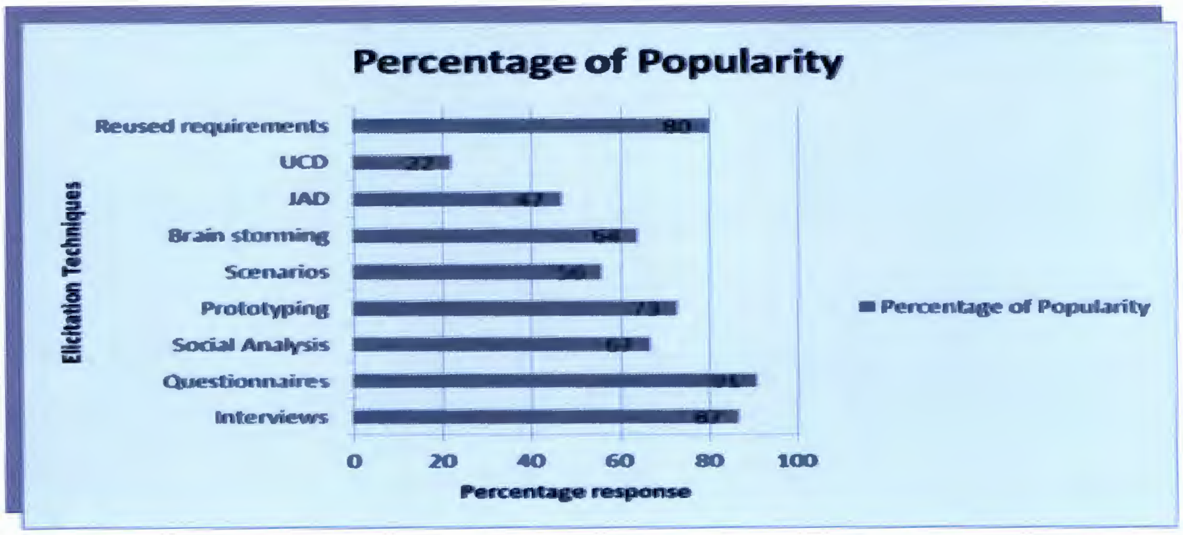


**Figure 4.7: Popular Elicitation Techniques**

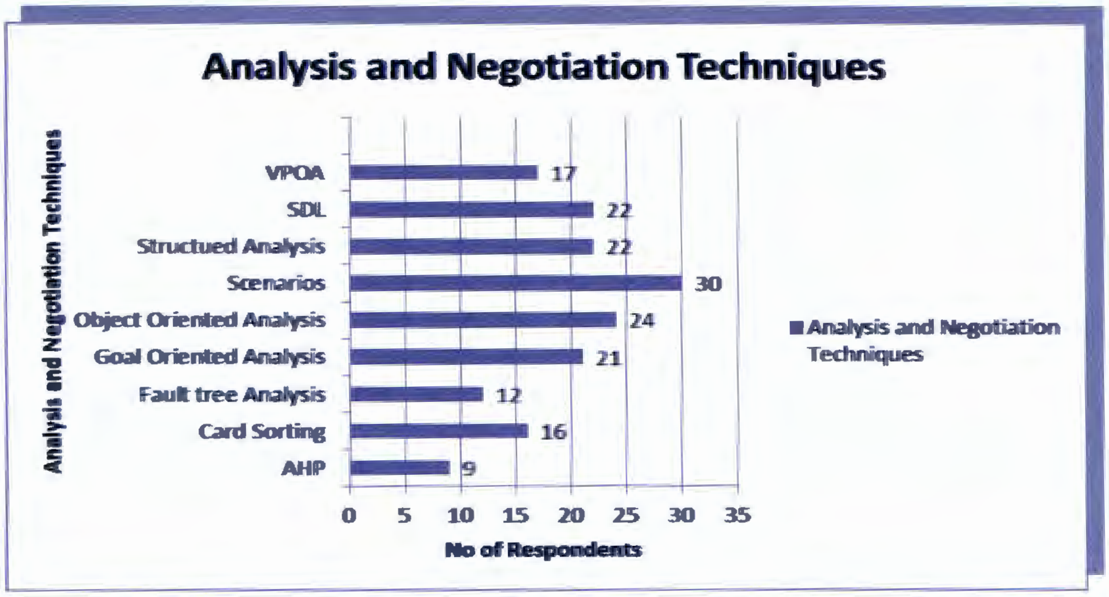


**Figure 4.8: Analysis and Negotiation Techniques**

Figure 4.8 shows the graph on the requirement analysis and negotiation techniques used during software requirement engineering process. It further shows that scenario is the most used requirement analysis and negotiation technique. Other popularly used techniques include; object oriented analysis, SDL, structured analysis and goal oriented analysis. VPOA and card sorting are averagely used while AHP is the least used requirement analysis and negotiation techniques.

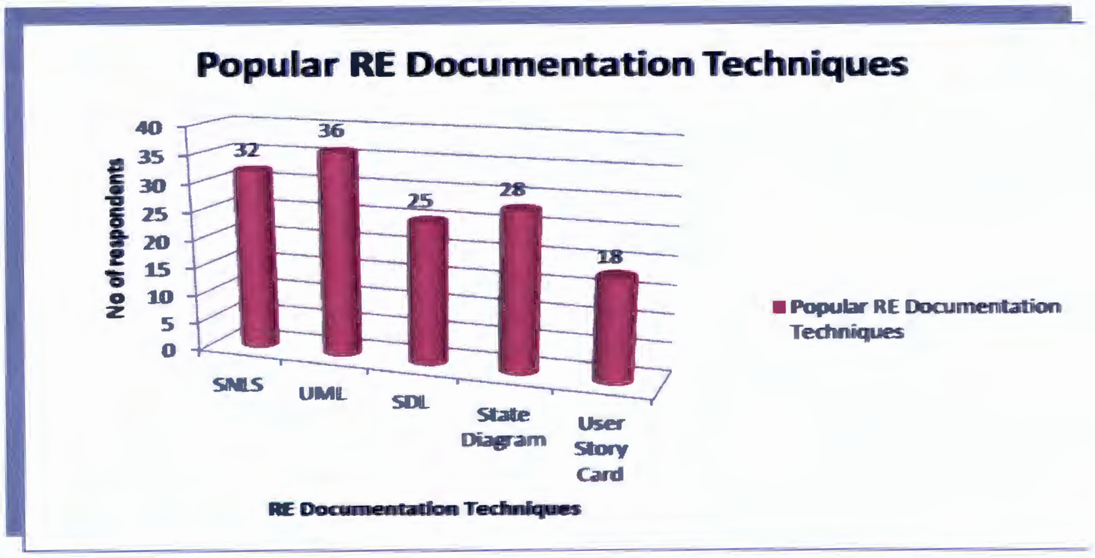


**Figure 4.9: Popular RE Documentation Techniques**

From figure 4.9, we could deduce that UML and SNL are the most popularly used requirement documentation technique, while user story card is the least used technique. Furthermore, other techniques that are being used but are not as popularly used as UML and SNL are state diagram and SDL.

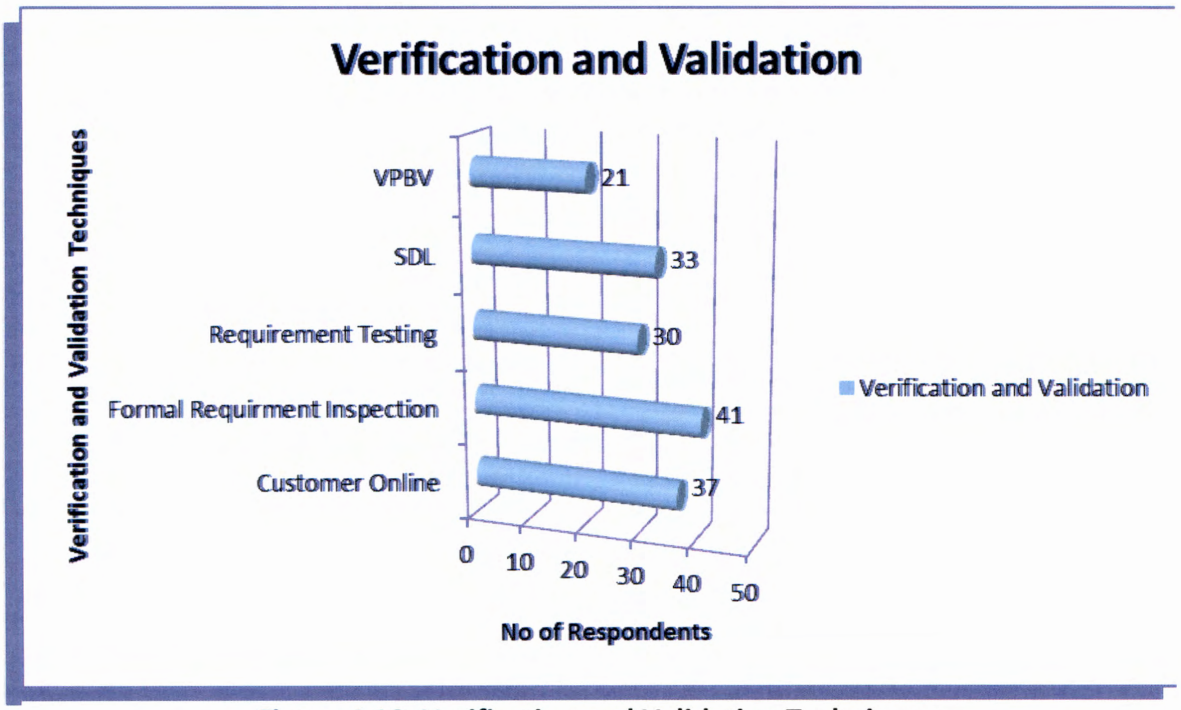


Figure 4.10: Verification and Validation Techniques

Figure 4.10 presents, formal requirement inspection (FRI) as the most used requirement verification and validation technique. Other popular techniques include; customer online, requirement testing and SDL while VPBV is the least used technique.

## 4.9 Result Analysis

From the survey conducted, we were able to deduce some very interesting results. These results include;

a) Experience Is Important

If you ask an inexperienced developer he would probably say that an application can be developed just with the knowledge of the developer, where as a seasoned programmer would be more concerned about the various parameters that affects the quality of the software. This was shown in Figure 4.1 and Figure 4.3. People with higher experience believed that requirements engineering process is an important task in software development lifecycle.

b) Popularity of the Requirement Engineering in Industries

   The industries were more concerned with the cost effectiveness of various techniques. Requirement elicitation process is the most implemented RE process across software development companies in South Africa. Requirement Documentation and requirement verification and validation are the least implemented processes. This low level of implementing all stages of RE processes also contributes to the high rate of scope creep, budget over run and time delays in a number of software application implemented in South Africa

## 4.10 Chapter Summary

This chapter discusses the results obtained from the questionnaires. The results obtained are centered on educational background, years of experience, effectiveness of RE techniques and popularity of RE processes in industries. These results were further analyzed and presented using bar and pie charts. These results shows that RE is not widely used in industries, experience are necessary in determining the importance of RE during software development processes, and that more awareness is necessary to increase the use of RE process in industries.

# Chapter 5

# Summary, Conclusion and Future Work

## 5.1 Summary

Software requirements engineering is achieved with the help of the standard technologies, processes and methodologies. Requirements engineering is essentially the initial stage of any software development activity in which the requirements from the customer are elicited and documented. This activity is absolutely important for the success of the project because all other project activities depend upon RE such as designing, implementation, testing, operation and maintenance. But RE is an iterative process which continues iteratively until the project is complete. The RE process models are the set of activities used to define the life cycle model for RE.

The name “requirements engineering” may seem a little awkward. Both words carry some unfortunate connotations: Requirements’ suggests that there is someone out there doing the ‘requiring’ – a specific customer who knows what she wants. In some projects, requirements are understood to be the list of features (or functions, properties, constraints, etc.) demanded by the customer. In practice, there is rarely a single customer, but rather a diverse set of people who will be affected in one way or another by the system. These people may have varied and conflicting goals. Their goals may not be explicit, or may be hard to articulate. They may not know what they want or what is possible. Under these circumstances, asking them what they ‘require’ is not likely to be fruitful. While engineering’ suggests that RE is an engineering discipline in its own right, whereas it is really a fragment of a larger process of engineering software-intensive systems. The term ‘engineering’ also suggests that the outputs of an RE process need to be carefully engineered, where those ‘outputs’ are usually understood to be detailed specifications. It is true that in some projects, a great deal of care is warranted when writing specifications, especially if misunderstandings could lead to safety or security problems. However, in other projects it may be reasonable not to write detailed specifications at all. In many RE processes, it is the understanding that is gained from applying systematic analysis techniques that is important, rather than the documented specifications.

There are many requirements engineering process models such as linear sequential model, linear iterative processes model, iterative process model and spiral model. These models have certain advantages and disadvantages hence there is no ideal RE process model but the problems with RE process models can be minimized by making the active involvement of the stakeholders whose concerns need to be addressed. The other main element in RE is the utilization of tools for requirements elicitation. The variety of tools exists such as interview, survey, questionnaires, task analysis, group-work, card sorting, observations, and prototyping and repository grids. The whole study has been grouped into requirements engineering processes, tools, technologies and methodologies along with their negative and positive aspects and also our suggested approaches. We also suggested the use of new technologies to automate the requirements engineering process.

## 5.2 Conclusion

We would like to conclude that RE is the process of establishing the services that the customer requires from a system and the constraints under which it operates and is developed. The requirement themselves are the descriptions of the system services and constraints that are generated during the requirements engineering process. The processes used for RE vary widely depending on the application domain, the people involved and the organization developing the requirements. RE needs many capabilities such as interviewing and listening skills, facilitation and interpersonal skills, writing and modeling skills and organizational ability. RE is more than just modeling, it is a social activity. To investigate, assess or evaluate the whole RE process usually involve using the method of either interviews or questionnaires or a combination of both methods. For this research, we used the questionnaire's approach to assess RE process for some selected South Africa software development firms. The main intention for this assessment is to know the state-of-the-art and suggest software process improvement strategies that would eventually reduce cost and schedule overruns in a software development firm.

Seemingly, in chapter three (3) of this research we conducted analysis of finding in RE and in chapter four (4) we discussed the findings obtained from the assessment.

## 5.3 Future Works

This thesis work can be further extended by using different methods other than questionnaire. Also, the number of respondents can be increased to reduce the margin of error and to increase the quality of the result.

## Reference

[1] N Maiden, "User requirements and system requirements", City university of London, IEEE publications, vol 25, issue 2, April 2008, pages 90-91.

[2] Y. Xiaoguang, HR Giangyi, "Research on Organizational-Level Software Process Improvement Model and Its Implementation," *Computer Science and Computational Technology, Int. Symp.* Vol. 2, 2008, pp. 285-289.

[3] H. Alzena, "Understanding the Requirement Engineering for Organization: The Challenges," *8th Int. Conf. Computing Technology and Information Management (ICCM)*, Vol. 2, Seoul, South Korea, 2012, pp.556- 567.

[4] K.E Wiegers, "When Telepathy Won't Do: Requirement Engineering Key Practices" *Cuter IT Journal*, Vol.13, no 5, 2000, pp. 9-15.

[5] B. A. Nuseibeh and S. M. Easterbrook, "Requirements Engineering: A Roadmap to the Future of Software Engineering," *In Proc. 22nd Int. Conf. on Software Engineering, IEEE Computer Society Press*, Vol 15, 2000, pp. 53-78.

[6] Christ of Ebert: "Practical Requirements engineering solutions", University of twente, IEEE publications, volume 21, issue 2, April 2004, pages: 16-18

[7] B. H. C. Cheng and J. M. Atlee, "Research Directions in Requirements Engineering: The Future of Software Engineering," *29th Int. Conf. Software Engineering*, Minneapolis, MN, USA, 2007, pp. 285-303.

[8] V. Sadique,"MEDICOM health information system," Limpopo Department of Health and Social Development Annual Performance Plan, 2008/2009, 2011, pp. 236-312.

[9] K. El Emam and N. H. Madhavji, "A Field Study of Requirements Engineering Practices in Information Systems Development" *In Proc. 2nd IEEE International Symposium on Requirements Engineering,* York, England, 1995, pp. 196-211.

[10] A.M Hickey and A. M. Davis, "Requirements Elicitation and Elicitation Technique Selection: A Model for Two Knowledge-Intensive Software Development Processes," *in Proc Hawaii Int. Conf. Systems Sciences*, Hawaii, USA, 2003, p.10

[11] M.k Zarinah and S.S. Siti, “Requirements Engineering for Survivable Systems” *Software Engineering Institute,* Technical Note CMU/SEI-2003-TN-013, USA, Sep. 2003.

[12] M.J. Simonette et al., “Soft Systems Engineering Tools in Requirements Elicitation” International Journal of Systems Applications, Engineering & Development, Issue 3, Volume 4, 2010, pp. 34-41.

[13] A. Ullah, and R. Lai, “A requirements engineering approach to improving IT-Business alignment,” *in Proc. ISD, 19th Int. Conf. Information Systems Development, Prague, Czech Republic,* 2010, pp. 1-9.

[14] K. Petters, “Extending Critical Success Factors Methodology to Facilitate Broadly Participative Information System Planning,” *A Journal of Management Information Systems*, Vol. 20, 2003, PP. 51-85

[15] J Herbsleb, D Zubrow, D Goldenson, W Hayes, M Paulk “Software Quality and the Capability Maturity Model”.Association for Computing Machinery. Communications of the ACM, Vol 6, 1997 pp30-40.

[16] JD Herbsleb, DR Goldenson “A Systematic Survey of CMM Experience and Results”. In Proceedings of the 18th International Conference on Software Engineering, IEEE, Los Alamitos CA,1996 pp. 323-330

[17] M Glinz: “Improving the Quality of Requirements with Scenarios”. Preceedings of Second World Congress for Software Quality,sept 2000

[18] D Pandey, U Suman, A.K Ramani. “An Effective Requirement Engineering Process Model for Software Development and Requirements Management. International Conference on Advances in Recent Technologies in Communication and Computing, ,2010 pp.287-291.

[19] CMMI-PDT Capability Maturity Model Integration (CMMI), Version 1.1. CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing Version 1.1 (CMMISE/SW/IPPD/SS, V1.1). 14. (2004)

[20] A.S., Khan. S.A.K Gahyyur. "Requirement Engineering Processes, Tools/Technologies, & Methodologies", International Journal of Reviews in Computing (IJRIC), ISSN: 2076-3328, 2009-2010. Vol.2

[21] Abras, C., Maloney-Krichmar, D., Preece, J.: "User-Centered Design". In Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications 2004.

[22] Y Asnar, P Giorgini, J Mylopoulos. "Goal-driven risk assessment in requirements engineering" 2011

[23] JP Kuilboer, N Ashrafi "Software Process and Product Improvement": An Empirical Assessment. Information and Software Technology Vol1, 2000 pp27-34.

[24] G Elahi, E S. K. Yu, N Zannone . "A vulnerability-centric requirements engineering framework: analyzing security attacks, countermeasures, and requirements based on vulnerabilities" , Journalof Requirements Engineering - RE , vol. 15, no. 1, 2010,pp. 41-62.

[25] E Felix, O Delande, F Massacci, F Paci. " Managing changes with legacy security engineering processes", Conference on Intelligence and Security Informatics , 2011

[26] D H. Rhodes, A M. Ross. "Five aspects of engineering complex systems emerging constructs and methods", Annual IEEE Systems Conference - SysCon , 2010

[27] Yihwa Irene Liou: "Integrating Group Support Systems, Joint Application Development, and Computer-Aided Software Engineering for Requirements Specification ".IEEE, 1993, vol 3, pages 4-12

[28] I. Sommerville, "Integrated requirement engineering: a tutorial," IEEE software, Vol. 22, no 1, 2005, pp. 16-23.

[29] Grzogorz Loniewski, Ausias Armesto, Emilio. "An architecture-oriented model-driven requirements engineering approach," InsfranModel-Driven Engineering Workshop - MoDRE , 2011

[30] T Yue, L. C. Briand, Y Labiche. "A systematic review of transformation approaches between user requirements and analysis models" Requirements Engineering - RE , vol. 16, no. 2, 2011, pp. 75-99.

[31] A J. Nolan, S Abrahao, P Clements, A Pickard. "Managing requirements uncertainty in engine control systems development", Requirements Engineering, IEEE International Conference - RE , 2011, pp. 259-264.

[32] S Hallerstede, M Jastram, L Ladenberger , "A Method and Tool for Tracing Requirements into Specifications ",Special:Publication/HalJasLad 2013

[33] M Kost, J-C Freytag, F Kargl, A Kung. "Privacy Verification Using Ontologies", Microprocessors and Microsystems, 2011

[34] J. Michael Moore, Frank M. Shipman: "A Comparison of Questionnaire-Based and GUI-Based Requirements Gathering". IEEE publications, 2000, pages 35-43.

[35] Hove, S.E.; And, B: "Experiences from conducting semi-structured interviews in empirical software engineering research". Software Metrics, 11th IEEE International Symposium, Volume, Issue, 19-22, Sept 2005, Page(s):10pp

[36] C F. Manski1 and F Molinari: "Skip Sequencing:A Decision Problem In Questionnaire Design". North-western University and Cornell University, 2008, vol 2, pages 264-285

**APENDIX A**

Requirement Engineering
Questionnaire

Version 1.0

By

Uchechukwu Desmond Anokwuru Prof O.O Ekabua

Department of Computer Science
Faculty of Agriculture, Science and Technology
Northwest University
Mafikeng Campus
North West
South Africa

Department of Computer Science
Faculty of Agriculture, Science and Technology
Northwest University
Mafikeng Campus
North West
South Africa
Email: uched@africamail.com
Cell: +27 82 680 2794

## Preface

### How to fill in the questionnaire

Answer all the questions according to your knowledge, skills and position within your organisation. The questionnaire aims to check up the requirements engineering practice within your organisation pointing out different viewpoints.
Each question has a multiple-choices answer (tick one of the choices). The lecture-key of the answers is as follows:
**N/A**: Not Applicable, if the question does not fit your organisation
**UN**: Unknown, if you cannot answer the question according to your knowledge, skills or position within your organisation
The other answers consist of three different levels, namely; **No, I Do Not Know** and **Yes** If you think that the question fits your knowledge, skills or organisation answer the relative question with one of the above levels selecting that one, which represents most your knowledge, skills or organisation.

**Remark** *This questionnaire is NOT to assess people and their work or knowledge. The questionnaire aims only to assess the requirements engineering practice within organisations.*

General Information
Please fill in the following with the relevant information.

| | |
|---|---|
| Name (optional): | ______________________________ |
| E- Mail: | ______________________________ |
| Age: | ______________________________ |
| Organization: | ______________________________ |
| Education: | ______________________________ |
| Designation | ______________________________ |
| Date: | ______________________________ |

Survey on Requirements Engineering Techniques [1-15]

| | |
|---|---|
| 1 | How long have you been working (Years of working Experience) : less than 3yrs ( ), 3 – 5yrs( ), more than 5yrs ( ) |
| 2 | How long have you been working in a software development organization: less than 3yrs(), 3 – 5yrs ( ), more than 5yrs ( ) |
| 3 | Do you use any process to obtain requirements in a project Yes ( ) No ( ) I do not Know ( ) |
| 4 | Do you like to carry out feasibility study before starting a new project? Yes ( ) No ( ) I do not Know ( ) |
| 5 | Do you like to carry requirements elicitation before going to start a project? Yes ( ) No ( )I do not Know ( ) |
| 6 | Have you ever used any requirement engineering methods in |

| | |
|---|---|
| | any form during your work activity? |
| 7 | From your perspective, is the requirements engineering important in any software development project?<br>Yes ( ) No ( )I do not Know ( ) |
| 8 | Tick the methods which are effective for Requirement engineering (your personal opinion, multiple selections allowed. Use '*' to indicate your selection)?<br>v. Requirement elicitation ( )<br>vi. Requirement analysis and negotiation ( )<br>vii. Requirement verification and validation ( )<br>viii. Requirement Documentation management ( ) |
| 9 | Tick the methods which are effective for Requirement eElicitation (your personal opinion, multiple selections allowed. Use '*' to indicate your selection)?<br>x. Interviews ( )<br>xi. Questionnaires ( )<br>xii. Observations ( )<br>xiii. Social analysis ( )<br>xiv. Prototype ( )<br>xv. Scenario ( )<br>xvi. Brain Storming ( )<br>xvii. JAD ( )<br>xviii. User centered design ( ) |
| 10 | Tick the popular methods which are used in your companies as a requirements analysis and negotiation tool (multiple selections allowed)?<br>xi. AHP ( )<br>xii. Card Sorting ( )<br>xiii. Designer Apprentice( )<br>xiv. Goal Oriented Analysis ( )<br>xv. Scenarios ( )<br>xvi. Structured Analysis Development ( )<br>xvii. System development Language ( )<br>xviii. Contextual Enquiry ( )<br>xix. Fault Tree Analysis( )<br>xx. Entity Relationship Diagram |
| 11 | Tick the popular methods which are used in Requirement Documentation your companies as a tool (multiple selections allowed)?<br>x. Structured Natural Language Specification( ) |

| | |
|---|---|
| | xi. Unified Modeling Language ( )<br>xii. System Development Language ( )<br>xiii. User Story Card ( )<br>xiv. View  Point Documentation ( )<br>xv. LOTOS ( )<br>xvi. Service Definition Template ( )<br>xvii. Document mining ( )<br>xviii. State diagram ( ) |
| 12 | Tick the popular methods which are used in your companies as a requirements Verification and validation (multiple selections allowed)?<br>ix. Request Inspection ( )<br>x. Request Testing ( )<br>xi. Request Checklist ( )<br>xii. System Development Language ( )<br>xiii. View Point Base verification( )<br>xiv. CO ( )<br>xv. Ethnography ( )<br>xvi. Utility testing ( ) |

13 Can you provide any general information regarding the different methods of requirements? (I mean which method do you think will give the good result depends on the project and the situation).

______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
______________________________________________________________________
__________________________________________________