Exploring issues in agile requirements engineering in the South African industry

by

Yanda Sebega

Submitted in accordance with the requirements
for the degree of

Master of Science

In the subject

Computing

at the

UNIVERSITY OF SOUTH AFRICA

Supervisor: **Prof Ernest Mnkandla**

January 2017

ABSTRACT

The agile manifesto has certainly changed the way software is produced in the Information Communications Technology (ICT) industry. However, many persistent challenges cripple agile software development. One challenge is that the constant change in technology makes the requirements hard to implement. Another is that issues of the agile requirements engineering (ARE) process are abundant and pervasive throughout software projects. The aim of this study is to determine common issues in agile requirements engineering in the South African software industry and identify tools and frameworks to mitigate risks emanating from such problems. This includes finding out how much value software practitioners put in the agile principles. This study was essentially quantitative, based on a cross-sectional survey. Self-administered questionnaires were used to collect required data which was then subjected to exploratory data analysis using SPSS (Statistical Package for the Social Sciences), a tool for statistical analysis. The results show that software practitioners have a strong penchant for principles of the Agile Manifesto. Major issues in agile requirements engineering include lack of proper validation tools and techniques, scope problems, lack of proper documentation, issues of prioritisation, as well as unavailability of customer representative. A detailed baseline of issues in agile requirements engineering was created along with a set of recommended tools and techniques used in the software industry. As for the recommendation, it is suggested that companies invest more on validation tools and techniques and consider non-functional requirements integration during software development.

Keywords: Agile requirements engineering; agile tools and techniques; issues of agile requirements engineering; customer collaboration; non-functional requirements

Dedication

To my wife Chantal, my son Melvin and my daughter Murielle

DECLARATION

Name: Yanda Sebega

Student number: 44934203

Degree: MSc in Computing

Exact wording of the title of the dissertation as appearing on the electronic copy submitted for examination:

Exploring issues in agile requirements engineering in the South African industry

---

I declare that the above dissertation is my own work and that all the sources that I have used or quoted have been indicated and acknowledged by means of complete references.

I further declare that I submitted the dissertation to originality checking software and that it falls within the accepted requirements for originality.

I further declare that I have not previously submitted this work, or part of it, for examination at Unisa for another qualification or at any other higher education institution.

(The dissertation will not be examined unless this statement has been submitted.)

__ _____ __12/04/2017____

SIGNATURE DATE

ACKNOWLEDGMENTS

First of all, I have my outstanding supervisor, Prof Ernest Mnkandla, to thank for this dissertation. His availability, guidance, patience, made this journey possible. Tirelessly, he provided me with valuable comments and suggestions to complete this study.

For giving me the opportunity to conduct this research, I thank the University of South Africa. I am grateful for the financial support I received from the university. It made things easier to conduct this study.

I am also especially appreciative of the companies in the software industry that took time to participate in the survey questionnaire. The data collection section was entirely achievable because of their participation and willingness to provide valuable information.

My thanks go as well to my colleagues for their support and help. To those who used their free time to proofread and give their input and others who directly contacted the companies they knew would help, I am sincerely thankful.

Finally, but not the least, I owe a special thank you to my family for their patience and support during this study. I particularly thank my wife for her understanding, support, and tolerance.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND ACRONYMS

AM	Agile Manifesto
APM	Agile Project Management
ARE	Agile Requirements Engineering
ASD	Agile Software Development
DMS	Defect Management System
DSDM	Dynamic System Development Method
EDA	Exploratory data analysis
FDD	Feature-Driven Development
GORE	Goal-oriented Requirements Engineering
ICT	Information Communications Technology
JAD	Joint Application Development
JIT	Just-in-time
NFR	Non-functional requirements
QC	Quality Centre
RE	Requirements Engineering
RAOD	Right amount of documentation
RR	Return rate
SPSS	Statistical Package for the Social Sciences
XP	Extreme Programming (XP)

CHAPTER 1: INTRODUCTION

1.1 Background information

Murphy et al (2013) empirically state that, over time, the tendency towards the adoption of agile methods has increased. According to Janes and Succi (2012:313), being agile is not a solution for everything, but agile methods are enormously popular and are on the increase. Despite this growth of popularity of agile methods, software project managers continue to face all sorts of challenges, for example, (a) software invisibility; (b) the "methodology jungle - difficulty of selecting the appropriate methodology for a given project" (Mnkandla 2008:3); (c) scarcity of scientific studies on agile methods (Laanti, Salo & Abrahamsson 2011:276); (d) software complexity (Yonghee & Laurie 2008:47) and requirements volatility (Ferreira et al 2009:1568), etc. This study focuses on issues related to Agile Requirements Engineering (ARE) in order to raise awareness in the software industry and intends to determine tools and frameworks present in the software industry to deal with these issues.

Lucia and Qusef (2010:214) suggest that 30% of the problems that take place in the development of challenging systems relate to the requirements phases. The 2009 Circa Report states that requirements changes are among the major cost drivers for software applications (Jones 2009:2). Requirements engineering (RE) establishes a "solid base for design and construction" and without it, the resultant software has a high probability of uncertain outcomes (Pressman 2009:120). Furthermore, RE is one of the most critical aspects in software development (Sillitti & Succi 2005:309) and perhaps the more complex activity in agile software project management. Similarly, the idea of Silliti and Succi (2005:309) applies to RE in the agile environment for the simple reason that it is merely RE performed iteratively. Collecting, understanding, and managing requirements (Silliti & Succi 2005:309) are not easy tasks especially when requirements become more and more dynamic. Thus, dynamic requirements are indeed endemic to the software industry (Jones 2009:438).

Today's global marketplace is dynamic and extremely competitive (Bopp, Bing & Forte-Trammell 2009:xvii), and therefore project managers may need to adopt new ways of thinking (Cobb 2011:107) to remain relevant to stakeholders (Bopp et al 2009:xvii). The

agile idea of remaining relevant to the needs of customers is constant accommodation of requirements through collaboration and negotiation. Agile software development (ASD) is a collaborative effort from both the customer and developer. This can be very challenging. Ambiguities and complexities of natural languages (Kamalrudin, Grundy & Hosking 2010:255) are typical challenges that software practitioners encounter in requirements elicitation. Another challenge is the undesirability of some requirements, that is, some requirements that tend to evolve quickly and become obsolete even before project completion (Cao & Ramesh 2008:60). The complexity of information technology infrastructure, the dynamics of market-driven needs (Stober & Hansmann 2010:5), or the lack of experience of stakeholders in agile development, are detrimental factors to successful RE. Accordingly, questions related to issues observed in the ARE arise.

Change in the world of information technology is coming more quickly than ever before (Cobb 2011:63) and RE is a fairly new subset of software engineering (Jones 2009:461). Little is known about how real agile projects conduct RE in practice (Cao & Ramesh 2008:61; Shen & Zhang 2011:1). For the context of this topic, more research is needed on ARE in South Africa, hence, this study is intended to shed light on the issues related to ARE. The next paragraph clarifies the problem that this study seeks to address.

1.2 Problem statement

Software companies in South Africa face issues related to dynamic requirements and the influence of the current market change, as well as the non-functional requirements integration in software production. These issues are current and pervasive throughout ASD. The aim of this research is therefore to determine common issues related to ARE, identify the tools and frameworks to help mitigate risks emanating from such problems, evaluate the impact that these problems have on project outcomes and finally explore the extent to which requirements engineering is adopted in the agile environment context. The next paragraphs outline the research questions which are aligned with this problem statement. This helped in proposing solutions to the research problem.

1.3 Research questions

The main question for this research is defined as follows:

> **How do issues in agile requirements engineering impact the outcome of projects in the software industry in South Africa considering today's constantly evolving marketplace?**

The sub-questions to be addressed will be:

1. What are the common issues in agile requirements engineering in the South African software development industry?
2. What are the tools and techniques that help in dealing with such issues?
3. How do software practitioners value agile principles that relate mainly to requirements engineering?
4. How collaborative are customers and software practitioners in terms of requirements engineering?
5. How do agile requirements engineering issues impact project outcomes?

The objectives of this research conformed to the above questions are now explored in the next section.

1.4 Research objectives

This dissertation will:

a) delimit the scope of 'agile requirements engineering' in terms of best practices;
b) define and determine a baseline for agile requirements engineering problems;
c) determine project managers receptivity vis-à-vis requirements from customers during software development at any point in time (final stage included);
d) compile a set of recommended tools and frameworks that to help deal with these problems;
e) get the usability of agile principles such as customer satisfaction, simplicity, communication, collaboration, or good design; and
f) Evaluates from a project manager's perspective the degree of interaction that exists between agile software practitioners and the stakeholders.

1.5 Research methodology

The objectives mentioned in the research objectives section are achieved through a research methodology, that is, the overall scientific approach to the research process (Oates 2006:35) to solve the research problem(s). This research was essentially quantitative, based on surveys. Self-administered survey questionnaires were used to gather data to get the required data for exploratory data analysis (EDA) through SPSS, a tool for statistical analysis.

To justify the choice of an empirical approach for this study, the answers to the questions outlined above as the research questions have quantitative measurable outcomes which are in line with the research objectives. Quantitative methods such as surveys are widely accepted and used in the field of information technology (Oates 2006:93). Research based on surveys "provides a quantitative or numeric description of trends, attitudes, or opinions of a population by studying a sample of that population" (Creswell 2009:146). This research also aimed to determine requirements issues experienced in the field of agile software development (ASD) by surveying the companies in South Africa. In addition, collecting large data from participants using standardized instruments such as questionnaires proved to be relatively easy (Oates 2006:93).

1.6 Research ethics

It is important to state the adequacy of a survey questionnaire for this research, but issues related to research ethics always emerge from the methodology adopted. Drew, Hardman and Hosp (2007:56) suggest that research ethics become the "cornerstone for conducting effective and meaningful research". The quantitative nature of this research requires direct interaction with participants and this may raise ethical concerns. Research ethics in issues such as to obtain informed consent, protect from harm, and ensure privacy (Drew et al 2007:57), and the anonymity, or confidentiality vis-a-vis participants were primarily responsibilities of the researcher. A letter obtained from Unisa to inform participants of the nature of the research was of great help. This helped

to avoid ethics breaches such as, raising false expectations, dishonest means of persuasion or unrealistic promises (Walliman 2010:47).

1.7 Rationale for the study

Researchers argue that agile methods have gained popularity but no clear trends in practise adoption are to be found (Cao & Ramesh 2008:61, Shen & Zhang 2011:1, Murphy et al 2013). In addition, over one third of the problems that occur in the development of challenging systems are attributed to the requirements phases. Nowadays, people rely more and more on ICT solutions. The consequences of systems failure caused by lack of proper RE can be a devastating disappointment for companies, not to mention the financial and economic implications. The rationale behind the choice of this topic comes from the penchant of the researcher for agile methodologies and the aspiration to take on challenges relating to modern software development, considering the failure rate of projects initiated every year.

In terms of the significance of this research, the contribution of this study would be of interest to software practitioners, more especially to project managers in charge of software projects as this study explores issues in ARE. The research objectives helped to determine the following: (a) a baseline for issues experienced in ARE (i.e., so that software practitioners are aware of the problems, and as a result will be more careful); (b) a repository of tools and frameworks that contribute in mitigating risks in ARE.

1.8 Limitations and delimitations

Like any other study, this research has limitations. These limitations are mostly encountered in the methodology used, that is, (a) limitations related to data collection, for example, limited depth in answers of the survey questionnaire, time constraints, or the lack of flexibility in response in particular (Walliman 2010:99); (b) the limitations attributed to lack of resources about agile methodologies in South Africa (Noruwana & Tanner 2012:41) despite the popularity that these methodologies have gained; or (c) limitations in the findings (anticipated uncertainty about generalizations).

The aim of this study is to determine the most common issues in the ARE process. These issues are limited to the RE aspects in an agile environment, more especially, issues emanating from functional and non-functional requirements in ASD. Tools and frameworks of the ARE, which are determined through a survey questionnaire, are also part of this study. Thus, the population of this study (survey participants) is limited to agile software practitioners, that is, individuals familiar with the concept of agile development. In terms of the geographical scope, this research is exclusively limited to companies in South Africa.

1.9 Definitions of key terminology

The meanings of the following key terms used in this study may be different when used in different contexts.

Agile environment. This refers to the dynamic settings in which software is developed.

Agile Manifesto. This is also referred to as 'The Manifesto for Agile Software Development', which is "a formal proclamation of four key values and 12 principles to guide an iterative and people-centric approach to software development" (WhatIs.com 2012).

Agile requirements engineering. This refers to requirement engineering that is done iteratively (in an agile environment).

Agile software development. This is the process of developing software that is entirely based on key values of the Agile Manifesto and its principles.

Change is defined by the Oxford dictionary as an act or process through which something becomes different.

Exploratory data analysis. This is an approach to analysing data sets through graphical methods.

Requirement. This is something that is needed or wanted, i.e., key features of software applications.

Requirements engineering. This is “the process of studying user needs to arrive at a definition of system, hardware, or software requirements” (IEEE Standards definition).

Software practitioners. This is to designate any person that is actively involved in the software development process.

Stakeholders. This is “any person whose opinions, needs, or preferences are likely to be relevant to the success of the project” (Berenbach, Paulish, Kazmeier, & Rudorferet 2009:140).

1.10 Dissertation outline

This section briefly outlines the remaining chapters of the dissertation. The second chapter (“Literature Review”) explores different aspects of agile requirements engineering, issues related to activities of requirements engineering. Chapter three (“Research Methodology") refers to the overall approaches and perspectives to the research process with the following main sections: research design, target population, sampling frame and instrumentation, data analysis, and ethical considerations. Chapter four (“Data Presentation, Analysis and Interpretation”) presents and discusses the data collected. Chapter five (“Conclusion and Recommendations”) recapitulates the findings, discusses the theoretical implications of this study, as well as its limitations. This is followed by references (a full repository of resources cited in this dissertation) which is also followed by appendices (supporting documents).

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

In the ICT industry, change is pervasive and current (Stober & Hansmann 2010:75). Responses to changes brought forth by different factors in the software industry led to the creation of the Agile Manifesto (Williams 2012:72) over a decade ago. The global tendency is the adoption of concepts such as agility to reinvent customer relationships (Highsmith 2013:3). Thus, agility has become a business imperative (Highsmith 2013:3). Agility generates 30% higher profits (Highsmith 2013:4) and is a key to global success; this includes economic success (Stober & Hansmann 2010:75).

Yet despite a noticeable growth in the adoption of agile methods, developing modern software faces many challenges. There is always the “unknown” phenomena caused by the complexity of the information technology used, the pressure of the accelerating time-to-market (Stober & Hansmann 2010:xi); the intensity of the global recession (Jones 2010:47); turbulence in business environments (Highsmith 2013:4); the inherent invisibility software (Grand 2016:122; Brooks Jr 1987:11); or by the market dynamism (Cao & Ramesh 2007:42). Agile software development (ASD) faces all these challenges. Requirements engineering is a fairly new subset of software engineering according to Jones (2009:461). Agile requirements engineering (ARE) is also fairly new.

Cao and Ramesh (2008:67) suggest that ARE is an iterative discovery approach. The field of ARE is a large domain (Tripathi & Goyal 2014:215) which calls for more research. Challenges in the ARE process are real and current for reasons already mentioned. It is judicious to mention that no study has proven, to date, that the adoption of ARE practices have successfully solved the problem of dynamic requirements which remain a perpetual challenge in traditional RE (Inayat, Salim, Marczak, Daneva & Shamshirband 2014:12). Classical phases of the traditional requirements engineering, that is, elicitation, analysis, documentation, validation, and management, are subject to many problems. Brooks (1987:11) enumerates complexity, conformity, changeability, and invisibility as examples of these problems.

Indeed, these same problems are profuse and persistent in the ARE process. The following four examples are a few ARE issues experienced in the software industry:

a) ambiguity in human languages, that is, lexical, syntactic, or semantic (Rojas & Sliesarieva 2010:102);
b) requirements creep;
c) changeability of requirements; and
d) lack of trust between developers and customers (Cao & Ramesh 2008:63) etc.

Another persistent issue that draws less attention comes from deficiency in handling properly non-functional requirements (NFR) during software development. Simplistically, NFR refer to all the requirements related to the quality of the software being developed, in other words "non-behavioural requirements" as Stellman and Greene (2005:113) suggest. Quality is "the extent to which the product satisfies its specifications" (Schach 2011:156). And what makes quality difficult or NFR integration into software complex is dynamic requirements.

Lastly, the purpose of this review is essentially to explore agile software project management; and outline issues related to the different functions of ARE in today's dynamic and constantly evolving marketplace with special focus on the South African software industry. Furthermore, this review explores tools and frameworks used by software practitioners to tackle problems in ARE. Lastly, a review of the current state of ASD in South Africa will also follow to reveal the problems experienced in the software industry. That is in essence the structure of this chapter.

2.2 Theoretical background

2.2.1 Agile project management

Agile software project management is a modern technique that defines the way software is produced. Augustine (2005:37) has comprehensively defined Agile Project Management or APM as:

> [T]he work of energizing, empowering, and enabling project teams to rapidly and reliably deliver business value by engaging customers and continuously learning and adapting to their changing needs and environments.

The above definition is significant and applicable in the field of ASD. Agile software project management is set of agile project management practices applicable to software. Customer implication, continuous adaptability to change, and rapid business value delivery are key instances of agile principles. Agile Project Management encompasses agile methods such as Scrum, Extreme Programming (XP), Dynamic System Development Method (DSDM) or Feature-Driven Development (FDD) which are well-known for their particular focus in rapidly responding to change (major goal), although, they differ in their specifics (Coram & Bohner 2005). And this goal is seamlessly substantiated in the Agile Manifesto (see Appendix 1 and 2) which is defined in terms of values, twelve original principles and practices (Williams 2012:72).

Coram and Bohner (2005) have suggested that software is inherently challenging because of its constant change. A couple years later, Cao & Ramesh (2007:42) put forward that change varies with market dynamism and this comes with issues such as velocity in requirements, requirements changeability and obsolescence which impede the ARE process. In order to understand challenges of the ARE process, it is essential to give a brief definition of ARE. That will be covered in the next paragraph before dealing with issues related to this practice.

2.2.2 Agile requirements engineering

This section defines ARE, outlines agile principles related to ARE and finally explores the activities of the ARE process as well as issues related to those activities.

Definition

Simplistically, ARE refers to RE in an agile environment. Cao and Ramesh (2008:67), for instance, define ARE as an iterative discovery approach; as it is more dynamic and adaptive. In addition, Pressman (2009:120) suggests that RE is a major software engineering action that is performed during the communication and modelling activities. He adds that it is “the broad spectrum of tasks and techniques that lead to an understanding of requirements”.

The latter definitions could not apply more in today's trends in the software industry, and adequately merge with an agile environment, more especially when building computer software is challenging (Pressman 2009:120). Agile requirements engineering makes the RE process more flexible and consequently quicker (Batool et al 2013:1006) and it encompasses many activities which are performed iteratively. Furthermore, agility qualifiers are flexible and quicker. The next paragraph defines agility in the context of software development.

Agility in software development

Agility is a term found in many disciplines and is not a concept unique to software development (Cao & Ramesh 2007:42). But, according to Dr Rico ([sa]), in the context of software development, agility is:

- the ability to create and respond to change in order to profit in a turbulent global business environment;
- the ability to quickly reprioritise use of resources when requirements, technology, and knowledge shift;
- a very fast response to sudden market changes and emerging threats, by intensive customer interaction;
- the use of evolutionary, incremental, and iterative delivery to converge on an optimal customer solution;
- maximising the business value with right-sized, just enough, and just-in-time processes and documentation.

The Figure 1 depicts the general idea behind agility in ASD. One team, through constant iterations, bound by the same common values (adaptability, transparency, simplicity, and unity), accelerates a release of working software, by keeping in mind the concept of visibility (velocity and tests). Thus, during iterations, requirements are subject to change, in accordance with many factors such as cost, budget, or velocity.

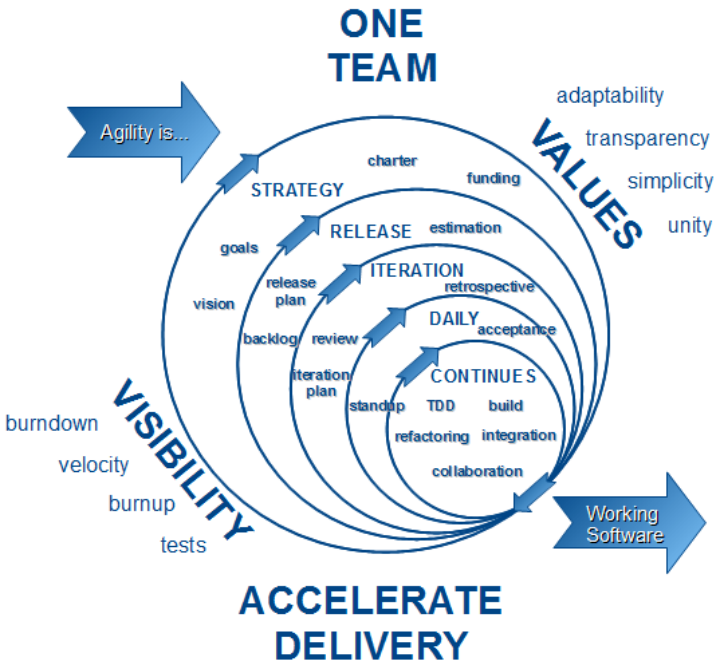


Figure 1: Agility by Scrumhint (2015)

Undeniably, what define agility in the context of software development are the values and principles outlined in the Agile Manifesto.

Agile principles related to agile requirements engineering

The Agile Manifesto encompasses four core values for enabling high performing teams and these values are supported by twelve key principles (Sutherland 2010). Some principles such as:

- customer satisfaction
- collaboration
- face-to-face communication
- simplicity

are indispensable to ARE; in other words, these principles are core to ARE (with customer satisfaction of paramount importance). Thus, the customer has that role to actively define and manage the project requirements, and he should feel just as responsible for the project outcome as the development team (Cobb 2011:114). For a country like South Africa, with advanced technology in some areas and extreme

disparities in others, what do agile principles of the ARE mean to software practitioners? One of the objectives of this study is indeed to seek an answer to such question.

Activities of agile requirements engineering

According to Zhu (2009:24), RE phases or activities are unclear in the agile environment; they are rather compounded and repeated through iterations. To reiterate, ARE is an iterative discovery approach (Cao & Ramesh, 2008:67). Every iteration (in ARE) depends on the nature of the project and encompasses the following classical activities (Lucia & Qusef 2010:216; Zhu 2009:24). These activities are:

- requirements elicitation
- requirements analysis
- requirements documentation
- requirements validation
- requirements management

Conversely, Pressman (2009:121) suggests that RE encompasses the following seven distinct requirements engineering functions: inception, elicitation, elaboration, negotiation, specification, validation, and management. In addition, Pressman (2009:121) adds that certain functions are inclusive of others. The requirements elicitation, for example, combines elaboration, negotiation, and specification (Pressman 2009:128). To avoid confusion, and considering that ARE is more dynamic and adaptive, the five classic activities will give a framework to this review. In addition to the five activities mentioned above, other activities of the ARE process include requirements modelling and requirements analysis, and negotiation. However, it is imperative to point out that there is no linearity whatsoever in performing these activities in an agile environment. These activities can follow a linear order if one decides to do so, although it is not really against adaptability, one of the principles of agile methodology. In addition, in comparison to traditional or open-source RE, ARE activities differ. What make these activities of the ARE different from the other methodologies are parameters such as change, customer satisfaction, and minimal reliance on documentation. Table 1 depicts these differences.

Table 1: Comparative analysis between agile and traditional requirements engineering

RE tasks	Traditional methods	Open-source methods	Agile methods
Requirements Elicitation	Determine all the requirements upfront prior to developing the system	Determine requirements iteratively though discourse analysis (chat rooms, forums, bulletins, etc.), introspection, focus group, questionnaire, open-ended Interviews	Determine the requirements iteratively and incrementally throughout the development process
Requirements Analysis & Negotiation	Check the requirements' feasibility, consistency, and completeness; in addition to prioritising them	Check requirements through requirements reading, logic, accountability	Refine, change, and prioritise the requirements iteratively
Requirements documentation	Methodical way of documentation	Informal documentation (chat rooms, forums, emails, etc are forms of documentation)	Minimal documentation
Requirements modelling	Provide a form of visual representation to the entire system	Provide a continually emerging webs of software discourse (emails, system vision statements, prior domain-specific knowledge)	Communicate understanding of the minor part of the system to be developed
Requirements validation	Ensure the consistency and the completeness of the requirements document	Validate requirements. This phase is Co-mingled with design, implementation, and testing descriptions and software artefacts, as well as with user manuals and usage artefacts	Ensure that the current software release reflects the current needs of the customer
Requirements management	Track changes in requirements, design, or documentation to understand why any changes purposes, through keeping intensive documentation of the system	Requirements are largely emergent; rapid change, commonly owned, continually evolving - "never" finalised; so forever informal management of requirements	Track changes with minimal documentation; user stories are written electronically, and are maintained in the product backlog list

Source: Elshandidy & Mazen 2013:478; Henderson 2000:28-30; Scacchi 2002

The APM, ARE, and agility are key concepts that help to put this study in perspective. Definitions about these concepts have been previously outlined. Activities of ARE have been identified and agile principles of the agile manifesto overviewed. The Agile Manifesto supports efficient requirements engineering (Lucia & Qusef 2010:219), but "extreme terminologies" such as agility, flexibility, dynamism, speed and adaptability are characteristics of ASD and are liable to issues in the ARE process. Focus is now directed to issues of ARE.

Issues related to agile requirements engineering

One of the many difficulties that a software engineer faces is to understand the requirements of a problem (Pressman 2009:119). Irrespective of the agile method used, issues about ARE emerge. Those are issues related to main activities of ARE such as requirements elicitation, analysis, documentation, validation, management, as well as NFR issues. Non-functional requirements in particular constitute a major concern (Cao & Ramesh 2008:64) in ARE, but this topic will be discussed in detail in the NFR issues section. The particularity of the ARE process (Figure 2) is that many parameters such as 'rapidly changing technology', 'evolving requirements' and 'time constraints' impact the ARE practices which always result in neglected non-functionals, inadequate architecture, lack of requirements verification, or prioritization in single dimension. This is what distances the ARE process from traditional methodologies.

Rather than following a formula to produce a complete documentation of the system, ARE is more dynamic and adaptive (Cao & Ramesh 2008:67). It is dynamic for requirements keep changing, and adaptive because of the inherent nature of agile principles. Issues related to main activities of ARE processes are now discussed in succession below.

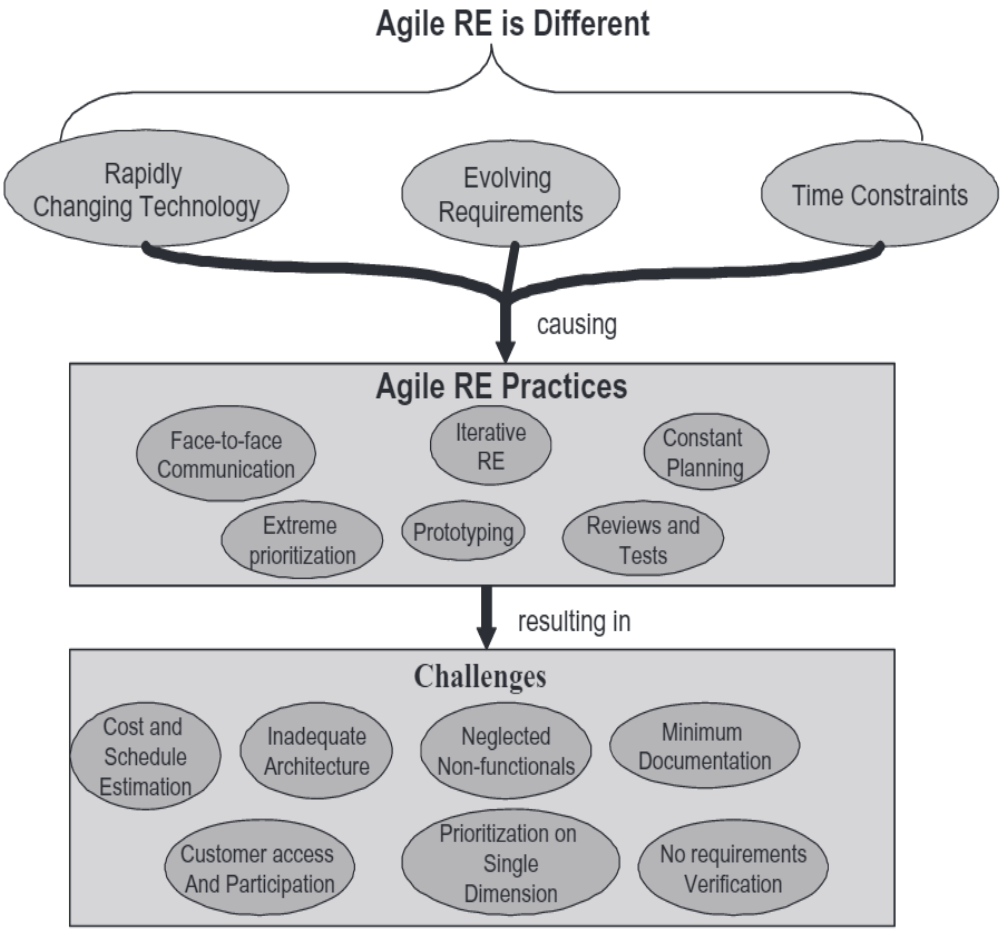


Figure 2: Issues in ARE (Source: Ramesh, Cao & Baskerville 2010:456)

a. Requirements analysis issues

The requirements elicitation comprises elements of problem solving, elaboration, negotiation, and specification (Pressman 2009:128). One main problem in ARE is that the requirements elicited during that process have the tendency to become obsolete (Cao & Ramesh 2008:60; Helmy, Kamel & Hegazy 2012:293). Issues of the requirements elicitation can be categorised into three types: problems of scope, problems of understanding, and problems of volatility (Elshandidy & Mazen 2013:473; Pressman 2009:121-122). These three types cover, to a certain extent, a wide variety of issues. Functions of requirements elicitation include interviews and questionnaires, prototyping (Eberlein & Leite 2002; Paetsch et al 2003), brainstorming, focus groups, use of case/scenarios (Paetsch et al 2003); questionnaires, requirements recovery, discourse analysis, ethnography and re-use (Eberlein & Leite 2002). All these tasks,

when executed, do come with different issues that need to be handled. To provide a better understanding, the outlines on how these functions are performed in practice follow, along with issues sourced in them and their relevance to this study follow.

- *Interviews and questionnaires.* These are achieved through interaction between stakeholders and software practitioners. Interviews require direct communication and the presence of both parties (stakeholders and software practitioners) is very important. However, issues such time, cost, lack of experience in either party, misunderstanding, ambiguity in language used can cripple this technique. Instead, questionnaires are easy to put in place, cost effective, and can use a synchronous and an asynchronous type of communication. Despite these advantages, questionnaires are limited only to the view of developers. In other words, questionnaires will yield only requirements that software practitioners intend to collect and sometimes, this practice leads to ill-defined requirements. Whether one uses interviews or questionnaires for requirements elicitation, the outcome is that there are always issues involved in both techniques, not to mention cost and time constraints.

- *Brainstorming.* This is a popular technique for generating creative ideas (Dugosh & Paulus 2005:313) or creative solutions for given problems (Paetsch et al 2003). This technique is used across most disciplines. However, according to Brown and Paulus (2002:208), research has shown that group brainstorming is less effective than individual brainstorming. For the context of this study, brainstorming is used to determine and define requirements. Unfortunately, the "human cognitive model cripples the requirements elicitation process" for the simple reason that "humans use distortion, deletion and generalization to express their thoughts", which can lead to ambiguous requirements (Ktata & Lévesque 2009:59). An ambiguous requirement admits more than one possible interpretation and it is notably incompatible with the goal of producing deterministic software (Rojas & Sliesarieva 2010:102). Basically, ambiguous or ill-defined requirements are issues that can possibly come out of this task.

- *Prototyping.* This is a well-known strategy or mechanism for identifying and determining requirements from the perspective of the customer (Pressman 2009:44) with the help of developers. The experience of the customer in

defining his software needs is capital in this situation. Nevertheless, prototyping causes issues such as scalability, security, and robustness, as well as maintenance problems (Ramesh, Cao & Baskerville 2010:461). In addition, some customers might get carried away or become overwhelmed by the prototype. This produces potential hazards in prototyping (ur Rehman, Khan & Riaz 2013:43).

- The *focus group* is a technique used to elicit requirements by inviting stakeholders from diverse backgrounds with different sets of skills to a group meeting (Paetsch et al 2003; ur Rehman et al 2013:45). Nonetheless, considering the diversity of the group, one major issue is that focus group requires lot of energy to conduct such meeting (ur Rehman et al 2013:43). For the purpose of agility in software development, this technique might not be ideal, for the simple reason that teams might spend much more time on their divergence only to come to a consensus.

- *Software reuse.* This makes ARE tasks more prescriptive and systematic (Elshandidy & Mazen 2013:479). Reuse refers to using existing software artefacts to help develop a different product (Schach 2011:310) but requires other adjacent activities such as testing which entails cost. With regard to the issues in this task, software reuse could be chaotic if the source of the reused requirements is uncertain. Thus leading to cost and time constraints, because testers will have to spend more time fixing the requirements before reuse. In addition, Elshandidy and Mazen (2013:479) suggest that there is lack of tools to effectively and efficiently manage and document variability. And it is common knowledge that change is pervasive throughout ASD. Minimal documentation is one of the twelve principles of the Agile Manifesto. As a result, change and minimal documentation can both impede requirements reuse.

Interviews and questionnaires, brainstorming, prototyping, focus group and software reuse are only five of the many techniques used in the software industry for eliciting requirements. Every technique has its own advantages and its problems. Despite the efforts to satisfy the customers as discussed above, further obstacles to requirements elicitation are: *requirements variability*, *wastages* (waste always generates further waste), *lack of clarity*, *list size*, and *requirements confusion or amalgamation* (Silliti &

Succi 2005:320). *Software complexity* and *lack of trust* within the group (Cao and Ramesh 2008:63), *lack of flexibility* and *objectivity* (Schach 2011:355) are also challenging issues in eliciting requirements; not to mention *unrealistic expectations*, *inconsistent information, vague customer needs,* and *scope problems*. Requirements elicitation is considered to be the basis for successful software projects. All the issues discussed here, if not handled properly would contribute to cripple the elicitation process particularly, and in more general terms, impede the ARE process. Even though there is no linearity in ASD, the requirements analysis would suffer a great deal if requirements are not appropriately elicited.

b. Requirements analysis issues

The term requirements analysis refers to the process of refining and extending the initial set of requirements that has been drawn up during requirements capture (Schach 2011:315). Agile practices of requirements analysis comprise *joint application development (JAD) sessions*, *prioritisation*, and *modelling* (Jones 2010:118; Lucia & Qusef 2010:216; Paetsch et al 2003). These tools and techniques help in checking the requirements for necessity, consistency, completeness, and feasibility (Paetsch et al 2003) as well as minimising downstream changes (Jones 2010:118).

- The purpose of the *JAD sessions* is to define a special project by giving meticulous details and help in further requirements elicitation (Paetsch et al 2003). The JAD participants are diverse and include JAD project and top management, managers, systems analysts and other IT staff members, as well as recorders (Shelly & Rosenblatt 2009:142). But, considering the diversity of the team of JAD sessions, on the one hand discussions can be very productive, and on the other hand chaotic for there is a risk of participants “running out of course” (Paetsch et al 2003). In addition, the diversity of the group can be the source of a clash of ideas during brainstorming as already indicated.

- *Prioritisation* is essentially a decision-making process (Daneva et al 2013:1334). Both developers and customers have to collaboratively provide their input when prioritising requirements (Paetsch et al 2003). However, this can be subject to a clash of ideas, if not handled with care and thorough negotiation. Some participants observed that performing re-prioritisation continuously, without

caution, leads to instability (Paetsch et al 2003). In addition, using business as a "primary criterion for requirements prioritisation" can lead to major scalability problems (Cao & Ramesh 2008:65). This last argument from Cao and Ramesh seems to raise a contradiction in ASD. The primary goal of the agile manifesto is to develop for the business. So throughout the entire process, either during requirements elicitation, or analysis through prioritisation, everything is done for the business.

In addition to handling issues of the elicitation process, issues in the analysis process require proper management for the sake of the ARE process. What would be the point of refining requirements if there were flaws during the initial stage? These are all complications of the ARE process which make this topic more interesting and worth exploring. Scalability issues, issues of prioritisation, and issues of requirements negotiation are basically what impair most the ARE process through the analysis phase.

c. Requirements documentation issues

One of the four principles of the Agile Manifesto values (see Appendix 1) is 'working software over comprehensive documentation'. All agile methods focus less on documentation. Heavy documentation is replaced by informal and frequent communication and collaboration (Cao & Ramesh 2007:42). Conversely, Highsmith (2003:4) suggests that documentation in moderation aids communication and preserves historical information. Most of the time, requirements documentation in agile methods revolves around *user stories*, *product backlogs*, *index cards, burn down charts,* etc. (Inayat et al. 2014:10).

- *User stories* are extremely short descriptions of the feature of the system that the development team implement (Sillitti & Succi 2005:322; Tripathi & Goyal 2014:215). They are brief and often transient (Abdullah, Honiden, Sharp, Nuseibeh & Notkin 2011). It is a fact of life that some people are concise when communicating, others are grandiloquent. Some things can be said in few words, while others require many words to be expressed effectively. Unfortunately, in this case, the risk of having multiple interpretations for these short stories arises. Again, considering the iterative aspect of agile methods, any early user stories could become superfluous or irrelevant because the

adaptability of the agile principles with regard to requirements (Lucia & Qusef 2010:219).

- *Product backlog.* This a technique mostly encountered in Scrum for requirements. It is simply an ordered list of eventual 'things' that are needed in the product and at the same time a single source of requirements for any change in the requirements (Sutherland & Schwaber 2011:12). However, these listed items are subject to constant change because of the nature of agile methods. Manageability is the main challenge in product backlog.

- The *index cards* are where one can find the requirements documentation or specification in the agile software management. The ARE process does not waste time in building huge and complex documentation; instead it rather develops only the concise documentation for future (Batool et al 2013:1009). Minimal documentation as suggested by the Agile Manifesto can be a real challenge in ARE. As an illustration, putting merely the essential thoughts on index cards can be an obstacle to recollection or even interpretation and elaboration considering the complexity of software development. Besides, managing pieces of paper can also be cumbersome.

- In addition to user stories, product backlog, or index cards, Batool et al (2013:1011) propose the term "right amount of documentation" (RAOD) to qualify technique for requirements documentation in agile development. RAOD, as the name indicates, lacks detailed requirements. Furthermore, Cao and Ramesh (2008:64) suggest, minimal documentation can be challenging and might cause a variety of problems such as communication breakdown. Coram and Bohner (2005), for instance, suggest that without formal documentation, there is high probability that turnover leads to loss of critical knowledge. Moreover, Paetsch et al (2003) have proposed that, in ASD, consistency and completeness in documentation is considered as unfeasible or, at least, not cost effective.

Lastly, recent studies have acknowledged numerous difficulties resulting from the lack of details in requirements documentation (Cao & Ramesh 2008:64). Admittedly, staff turnover, dynamic requirements, unavailability of customer representatives, and minimal

documentation can cause serious problems in ARE. All these issues of the agile manifesto are fortunately remediable through enough documentation to preserve valuable information. However, there is serious lack of empirical studies on issues regarding requirements documentation. Hence, this is more reason to scrutinise the documentation process in ARE.

d. Requirements validation issues

The intention behind requirements validation is to reveal critical stakeholders needs, expectations, interfaces, as well as constraints (Chrissis, Konrand & Shrum 2011:56) by examining requirements for inconsistency, omissions, and ambiguity (Pressman 2009:145). Techniques for requirements validation encompass the following:

a) evolutionary prototyping (Chrissis et al 2011:472; Lucia & Qusef 2010:219);
b) requirements reviews, unit testing (Lucia & Qusef 2010:217);
c) acceptance testing (Cao & Ramesh 2008:66; Lucia & Qusef 2010:214);
d) frequent review meetings and face-to-face communication (Cao & Ramesh 2008:62; Ramesh et al 2010:455);
e) simulations and demonstrations (Chrissis et al 2011:472); and
f) requirements testing (Paetsch et al 2003).

Like any of the tasks discussed in previous phases, these tasks are always filled with issues. The next paragraphs outline what these issues entail.

- The purpose of a *prototype* is to exhibit key functionalities of the software when requirements are vague (Schach 2011:348). And once stakeholders have a clear understanding of the final product, the prototype is thrown away (Brooks 1995:180). In the case where the prototype is really thrown away, issues related to this task disappear in the process. Issues emerge usually when stakeholders decide to keep prototype for whatever reasons. As an example, stakeholders become too overwhelmed, due to a working prototype that displays what they believe to be the exact representation of their needs. So they ask that the intended software is based on it (evolutionary prototyping) or even worse ask to use the prototype in a production environment. Assuming, that the prototype is full of defects, evolutionary prototyping could be a serious concern since the

use of such system will cause nothing but agony to customers (Brooks 1995:241).

- *Review meetings* provide a “formal channel to validate requirements through informal communication” between stakeholders and developers (Ramesh et al 2010:468). Informality is common in agile practice, but informal communication is subject to errors, mistakes, and misunderstanding. In some instances, people get carried away and review meetings become very difficult to control, thus, leading to division and chaos.

Cao and Ramesh (2008:66) suggest that since there is no formal modelling, agile methods do not properly address requirements validation. In contrast, Inayat et al (2014:12) suggest that agile methods work well for requirements validation with the support of feedback from stakeholders. However, considering the adaptability of agile methods, it is up to software practitioners to decide how to properly perform requirements validation. The reality is that addressing requirements validation is more advantageous than no validation at all. Discussions on validation tasks such as prototyping, review meetings have been done. This is enough to provide a baseline for issues in validation which contribute to better understand the validation phase in the ARE process.

e. Requirements management issues

The purpose of requirements management is to maintain the requirements and to ensure that relevant plans and data are kept current (Chrissis et al 2011:473; Paetsch et al 2003). It includes activities concerned with change and version control, requirements traceability and requirements status tracking (Chrissis et al 2011:474). These activities are subsequently explored along with the issues in order to find out the relevance to this study.

- *Change and version control* includes version control of all artefacts, that is, memos, pictures, blueprints, meeting minutes, stakeholder requests, and so on. It is an important part of the ARE process for the simple reason that some of the artefacts managed in this phase are elicited requirements, documentation such as user stories, and validated requirements. Again change is pervasive

throughout the ARE process. Thus, a tool that keeps track of changes in the artefacts is capital. There is a large number of version control or configuration management tools available that can be used to control requirements artefacts: however, one major issue remains; that of choosing appropriate tools for a given activity (Mnkandla & Dwolatzky 2007).

- *Requirements traceability* is another activity of requirements management. Berenbach et al. (2009:13) suggest that a requirement is traceable "if and only if the origin of each of its component requirements is clear; and if there is a mechanism that makes it feasible to refer to that requirement in future development efforts". However, Jones (2010:463) suggests that effective traceability remains troublesome and imperfect in spite of hundreds of tools that are meant for requirement traceability. Admittedly, the life of a requirement in an agile environment is a turbulent one. Dealing with constant change and the indecisiveness of choosing the proper tools make the ARE even more complicated. Not to mention dimensions of agility such as flexibility, leanness, speed, and responsiveness which give the impression of chaos in ARE. Considering these terminologies, one can determine that agile requirements traceability is an activity that needs special attention, although, Paetsch et al (2003) suggest that agile methods provide a good base for requirements management.

Finally, requirements management is a dynamic and often recursive sequence of events (Chrissis et al 2011:66). Despite a good base for requirements provided by agile methods as Paetsch et al (2003) suggest, a special focus should be put on requirements management during the ARE process for adequacy and completeness of requirements. The ARE is exclusively about requirements. Properly describing the life of requirements from the initial phase until delivery is of paramount importance.

Change in requirements is inevitable in agile software development considering that it is primarily the central focus of agile methods. Essentially, lack of requirements traceability, lack of proper management tools can be chaotic to the survival of the ARE process. Iterations after iterations, these issues remain and cripple the entire system in construction. Requirements elicitation in detail is a good base for requirements analysis, which in turn, is a good base for modelling, validation and management. This is reason

well enough to seek more answers to pervasive problems of the ARE outlined through research questions defined in the introduction chapter. Besides issues discussed in sections above, other issues, with equal consideration, seriously impede the ARE process.

f. Other issues of the ARE process

The ARE process is not limited to just the five classic functions mentioned above. Other activities such as *requirements modelling*, *requirements negotiation*, and *NFR* are important aspects of ARE.

- *Requirements modelling issues.* The fundamental idea about this task is that one does just barely enough modelling at the beginning of the project to understand the requirements which will then be detailed on a just-in-time (JIT) basis (Ambler 2001). Requirements modelling activity in agile development encompasses initial requirements modelling, iteration modelling, model storming, and acceptance test-driven development. It comes in any of three forms: a) usage model; b) initial model domain; and c) user interface model. Goal-sketching is another technique used in modelling agile requirements (Inayat et al 2014:8). It intends to provide intuitive and easy-to-read goal graphs to developers (Boness, Harrison & Liu 2007:3). Yet again, change is the constant factor that modelling has to deal with. Unavailability of project stakeholders, complexity of the proposed technology solutions, customers' narrowness, rigidity and ignorance to modelling artefacts, or conflicting priorities are common challenges of agile requirements modelling (Ambler 2001). In addition, requirements modelling in ARE develop models that are mostly throw-away models for the simple reason that these models are drawn on the whiteboard and erased after fulfilling their purpose (Paetsch et al 2003). As a result, this can be a serious challenge for requirements documentation or even for recollection.

- *Requirements negotiation issues.* Negotiation is an essential part of agile software development. It is not a contest or a game (Pressman 2009:102). Software developers and stakeholders are constantly in negotiation to come to a common agreement on some aspect of the software to be developed.

According to Schach (2011:354), the ability of both parties to negotiate with either developers or stakeholders is a constant challenge. He suggests that it is often essential to scale down what the clients want but that is not easy especially when one has to deal with determined customers who consider that what they need is critical to the final product. Similarly, Pressman (2009:121) suggests that it is not unusual for different customers to come up with conflicting requirements, firmly believing in the authenticity of their requirements. The resolution of such crisis needs proper negotiation skills. He (2009:102) adds that it works best when both parties win. Not surprisingly, almost every stakeholder would love to have a product that includes everything that he conceivably needs (Schach 2011:354). However, this may cause scope problems. In addition, in a case where the requirements brought forth are low-level requirements, re-negotiation (Hull, Jackson and Dick 2005:150) in this situation may cause a shift in the schedule, a budget amendment, or even code alteration.

- *NFR issues*. The NFR specifies properties of the target product itself. According to Zhu (2009:24), NFR are poorly defined in agile approaches. Singh and Saxenna (2014:547) suggest that sometimes software systems fail due to lack of consideration of NFR. The NFR cover “portability, maintainability, scalability, safety, or performance” (Cao & Ramesh 2008:64). The list from Stellman and Greene (2005:114) is even longer. In addition to Cao and Ramesh (2008:64), Stellman and Green (2005:114) enumerated the following as features of NFR: a) availability, b) efficiency, c) flexibility, d) portability, e) integrity, f) performance, g) reliability, h) reusability, i) robustness, j) scalability, k) and usability. NFR is essentially about quality requirements.

Although these are very important features to have, the obvious fact is that no application can encapsulate all these characteristics and even their implementation will certainly raise some issues. Potential issues in NFR encompass a) hidden or inappropriate functionality, b) misleading cue, c) inadequate IT solutions, d) conflicting priorities, and even more general issues like e) views inconsistencies, f) customer rigidity, g) vague requirements, h) problem domain complexity, or i) budget problems. Ignoring these issues compromises the quality of software. According to Humphrey (2005:19), software marketplace focuses more and more on cost, schedule and

function. Little attention is given to quality (in his words, quality “is lost in the noise”). Unfortunately, the root cause of most problems related to software cost and schedule is poor quality performance.

Consequently, common issues of the ARE process are countless and diverse. All the issues discussed above are not an exhaustive list of enumerated issues in ARE. But the sum of all the issues previously elaborated in the five phases and the other issues discussed afterwards is enough to provide good foundation to the research problems.

To conclude this section, in the ARE process, the main driving factor of issues in NFR, modelling or negotiation is change. The same applies to issues in requirements elicitation, analysis, documentation, validation and management. The pervasiveness of change (Pressman & Lowe 2009:13) in agile methods make the ARE process a more complicated task, partly due to the inherent complexity of software itself. In addition, software invisibility is a challenge through software engineering. Requirements are a means to make software visible. Accordingly particular attention should be given to RE. This surely will increase the results for positive software projects outcome. It is now time to explore the ARE tools, techniques and frameworks that are used by software practitioners in the industry which is one of the objectives of this study.

2.2.3 Agile requirements engineering tools and techniques

Sillitti and Succi (2005:322), simplistically suggest that, in RE, and more especially in several agile methods, paper, pencil, and pin board prove to be the most popular tools. The existing literature mentions a plethora of distinct techniques and tools for ARE. As already mentioned briefly above, many issues cripple activities of the ARE process. A few more examples of these issues are: a) requirements variability, b) ambiguity, c) minimal documentation, d) lack of trust between customer and developers, e) wastes, or f) the human cognitive model that cripples requirements elicitation. Techniques and tools exist to help solve, if not mitigate risks related to these issues during ASD. Techniques for ARE have been discussed along with the problems experienced in that field in paragraphs above. The following sections are about tools (software program and frameworks) developed by leading companies for RE in agile environments. This also includes handling NFR during development.

Zhu (2009:36), in his thesis, investigated how RE is conducted under an agile environment. He proposed three tools relevant to requirements used by agile teams: Scrumworks (to store backlogs), Quality Center (or QC to log use cases) and Defect Management System (or DMS to log defects). Carrillo de Gea et al (2011:7) have put forward that the most capable tools for both requirements elicitation and requirements validation are Cognition Cockpit, Cradle, QPack, and Reqtify. More tools explicitly designed for requirements elicitation include MKS Integrity, Polarion Requirements, or JAD. The Method for Elicitation, Documentation and Validation of Software User Requirements (MEDoV) guide stakeholders during requirements elicitation (Dragičević, Čelar & Novak 2014:66). A simple application like Microsoft Excel is a very good tool for product backlog.

Agile methods focus less on documentation. The little documentation that one can have can be done using simple tools such Microsoft Word or Excel, Notepad, Open Office, vi Editor, etc. Microsoft Visio or Caliber by Borland can be used for minimal modelling. MEDoV, already mentioned above, can also be usable for documentation.

Requirements validation is performed with the help of tools such as Aligned Elements, Case Spec, GMARC, IRqA, PACE, ReqMan, and TraceCloud (Carrillo de Gea et al. 2011:7). Other tools for requirements validation include QPack and Reqtify which are also used for requirements gathering.

The secret of the success of agile requirements engineering is (a) customer collaboration; (b) good agile developers; and (c) experienced project managers (Lucia & Qusef 2010:219), but it is not always easy to have all three sets of skills in order to build a good agile team. Assuming that a team is constructed around the three parameters above, other issues such as a) requirements inconsistencies, b) redundancy, c) incompleteness and omissions resulting from the complexities and ambiguities of natural language would emerge (Kamalrudin et al 2010:255). It is in that perspective that Rojas and Sliesarieva (2010:106) developed a tool for lexical analysis using Perl's engine for regular expressions. This technique is common and simple and allows identification of key words or phrase structures that reveal specific weaknesses in requirements. More specifically for ambiguities, it allowed for the location of adverbs and non-deterministic constructs.

Tools for requirements management include Jira Agile, Rally, HP Agile Manager, Mingle, VersionOne etc., Agilo, Jazz, or even the generic Microsoft Project can all help in managing requirements. Another example is RE-KOMBINE, a tool that can provide answers regarding which requirements to work on or manage evolving requirements (Ersnt, Borgida, Mylopoulos & Jureta 2012).

As a final point, Highsmith (2013:15) suggest that 'agilists' focus more on the velocity in requirements and forget quality. NFR is everything about the quality of software. It is the forgotten aspect of RE or rather the aspect that developers pay least attention to. There are tools for NFR. For instance, FURPS (Functionality Usability Reliability Performance Supportability) developed by Hewlett-Packard, is a model for classifying software quality attributes and NORMAP, which stands for Non-functional Requirements Modeling for Agile Processes, is a conceptual framework to integrate NFR into agile processes (Farid 2012:323).

This section is a sufficient but not an exhaustive coverage of tools and techniques to handle issues in ARE. However, this is enough background to help achieve the objectives of this study which is to determine a set of set of recommended tools and techniques in the software industry. For extensive study on tools, the internet is a great resource where one can find tools of all genres for ARE. Atlantic Systems Guild Ltd has done an excellent job by compiling requirements used in the software industry both traditional and agile environment. They can be found on the link http://www.volere.co.uk/tools.htm. The following is an excerpt of what can be found on the website:

- Borland CaliberRM 2005
- ARCWAY Cockpit or Accompa (tools for managing requirements)
- Balsamiq Mockups (a useful tool for sketching)
- CaseComplete (helps with use cases and Axure with requirements specification)
- Gatherspace.com and Innoslate (powerful tools that provide features collaborative requirements engineering activities).

Despite such a collection of tools and techniques to handle ARE issues, there is lack of tailored tools that are adaptable to all projects. Every project is inherently specific and requires unique solution. In this perspective, it is a business imperative to invest in tools

in order to achieve good software quality (Highsmith 2013:8) and software costs reduction.

Since the geographical scope of this study is limited to South Africa attention is now turned to grasp the current state of ASD in this country. This, indeed, includes issues related to ARE experienced by software companies in the industry.

2.2.4 Agile requirements engineering in South Africa

Current state of software in South Africa

South Africa is a country that has characteristics of both an advanced and a developing economy (Gillwald, Moyo & Stork 2012). The advanced economy in South Africa has good infrastructure in big cities. The developing economy (reflected in the social lives in townships) is not any different from some other places in developing countries. The education system is in crisis and in constant need of restructuring because of the shift from the old regime to a more democratic system. The consequences of severe inequalities in the education system in South Africa can be attributed to correlated dimensions such as wealth, school location, language and province (Spaull 2013:7). The South African economy is certainly in a constant state of emergence, and so is the technology.

Although technology is changing faster than we can master it (Highsmith 2003:3), its pervasiveness impacts the economy and the education system, both pillars for countries growth. As Siriram (2011:13) has suggested, technology is a “catalyst for competitive advantage”. Since the technology integration within the community goes through the education system, the demand for software projects will certainly grow with the needs from the community. However, in their recent empirical study, De Wet and Visser (2013:14) indicate that the average success rate of software projects in South Africa is very low.

Marketline (formerly Datamonitor) is an international company providing a variety of data analyses. In their empirical reports, they have reported increases in the software market in South Africa from 2009 to 2016 (including software market forecast). The

table below (Table 2) give a summary of four years of annual growth, with a slight decline in the fourth year. While Figure 3 denotes a market value forecast (2011 – 2016).

Table 2: Annual software market growth in SA

	2009	2010	2011	2013
Growth	4.2%	9.0%	11.9%	7.0%

Source: Marketline 2009-2014

The growth forecast from 2011 until 2016 denotes an average constancy (see Figure 3).

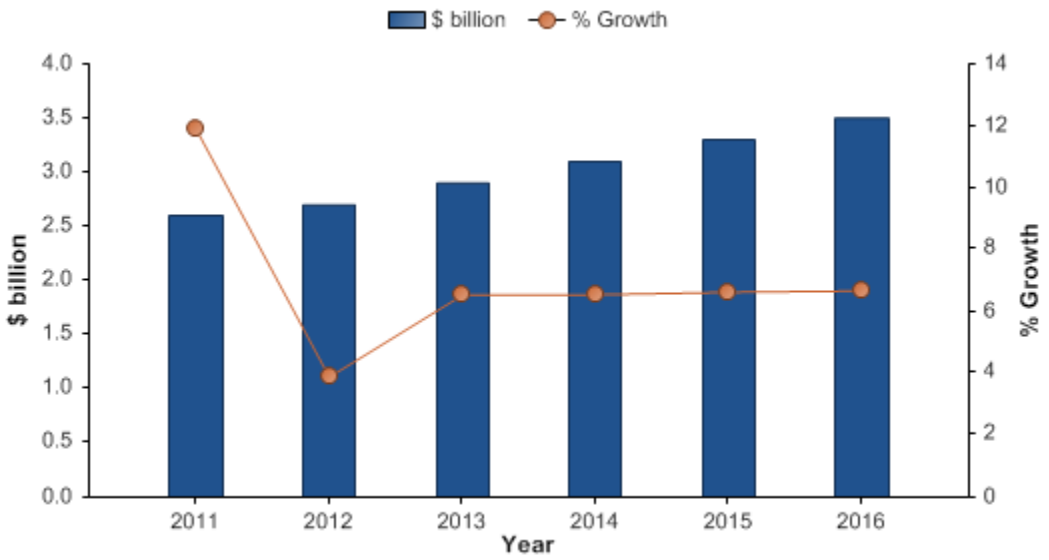


Figure 3: South African market value forecast: $billion, 2011-2016
(Source: MarketLine 2012:9)

Clearly, the future lies with software (Hislop 2011:2) considering the ubiquity of technology. Nonetheless, despite the noticeable growth in the software market in South Africa, 27% still represents a significant waste of financial resources, time and effort (Marnewick & Labuschagne 2009:81). Indeed, the success rate of completing software projects to specifications in South Africa is low (De Wet & Visser 2013:10) and needs to be addressed. That is where South Africa stands in term of software in general. Considering all of the above, where does South Africa currently stand in terms of ASD? A tentative answer to this question follows next.

Agile software development in South Africa

A decade ago, Mnkandla & Dwolatzky (2004:236) noted that, “in South Africa, the majority of the software development organizations are small to medium sized and most of the applications are developed without following any development methodologies and sometimes without following any project management methodology”. A few years later, in an empirical study, Ferreira and Cohen (2008:51) suggested that there is a large room for improvement for South African software development projects. In another more recent case study, Noruwana and Tanner (2012:41) uncovered empirical evidence on the adoption of agile methods by South African companies as well as disparities between agile prescriptions and practices. They have indicated that the popularity of agile methods with regard to software development in South Africa continues to rise; however, little has been said about the approach and its adoption challenges in the South African context. In addition, De Wet and Visser (2013:15) suggest that the adequacy and suitability of agile methods is questionable, arguing that these methods originated from Western or European countries, and were designed to address risks in their respective settings. However, one could also argue that most of the computer principles that currently direct our lives are adopted from Western or European countries. The unfortunate truth is that the literature, with regard to the adoption of agile methods in the South African context, is limited (Noruwana & Tanner 2012:42). That constitutes a real challenge in ASD and indeed, that inevitably includes ARE challenges. Tentatively, that is where South Africa stands in terms of ASD in general. But the central focus of this study is rather more narrowed to ARE issues.

2.2.5 Issues of agile requirements engineering in South Africa

South Africa encounters many problems in the software industry, especially when it comes to ASD. Friedrich and Van Der Poll (2007:189) identified a serious communication gap between IT specialists (i.e. project manager, systems architects, business analysts, system analysts) and clients (users) leading to incomplete or incorrect requirements. A lack of understanding of business requirements is another challenge experienced by software development companies (Noruwana & Tanner 2012:53).

Sonnekus and Labuschagne (2003:9) and Labuschagne, Jakovljevic, and Marnewick (2008:18) consistently reported a lack of communication between stakeholders and developers, lack of strategy to handle change, lack of user involvement and executive support as major factors for failed projects. Communication, welcoming change and user involvement are principles of the agile manifesto, per se, functions of ARE. Thus, companies in the software industry that are agile ought to improve on these areas that are fundamental to project success.

Software is currently the dominant force of change of new products (Hull et al 2005:1). To reiterate, software complexity is exacerbated by problems such as rapid changes, technology changes, time-to-market pressures and rapid changes in competitive threats (Cao & Ramesh 2008:60). As stated above, Noruwana and Tanner (2012:41) mentioned a growth in popularity on the adoption of agile methodologies by software practitioners in South Africa. Unfortunately, there is a lack of documentation that properly lists the common problems experienced in ARE in the software industry in South Africa. Hence, the questions outlines in this study with regards to exploring the issues in the software industry.

This study contributes in raising more awareness on the common issues experienced in ARE. As suggested by Ferreira and Cohen (2008:53), future research into agile methods practice would also "benefit South African developers in understanding the contexts in which agile development is most appropriate, as well as long-term implications for system quality and maintainability". Therefore, there are positive implications on the software industry.

2.3 Summary

The future of the world lies in software. The pervasiveness of technology is global and the impacts are social and economic. As time passes, the prediction on how a computer-based system will evolve is practically impossible. Pressman (2009:66) suggests that, "in the modern economy, market conditions change rapidly, end-user needs evolve, and new competitive threats emerge without warning". The ability of organisations to adapt to rapid change is the most significant challenge of the 21st

century (Ktata & Levesque 2009:59). The fact is that ARE is not an easy process as it is filled with challenges and problems. More generally, reports have stated that South African projects failure is similar to those of the world. And it is argued that over 90% of contributive factors in project failure emanate from requirements. Researchers have argued that there is lack of resources in the field of software engineering in South Africa. Some question the adequacy of these agile procedures by arguing that agile methods and frameworks were created in the Western or European context which is not necessarily a match for the reality of the software industry in South Africa. ASD is a vast area. So is the area regarding issues of the ARE process. Focus must therefore be put on requirements as the market is so demanding and software must conform to requirements. This chapter overviewed concepts such as the agile manifesto and agile principles, and covered classic issues related to ARE and finally reviewed tools and techniques to help mitigating requirements problems experienced by agile practitioners. In addition, few paragraphs highlighted the current state of ASD in South Africa, as well as issues related to ARE in the country. In the light of all the above-mentioned, and considering the pace at which the technologies evolve, there is a pressing need to do more research on ARE in order to remain relevant to current technology trends. Thus, exploring common issues of the ARE in South Africa is an achievable goal.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

Prior to the objectives section in chapter 1, a series of questions were asked in the research questions section. To seek answers to these questions, a survey was used as the main “strategy of inquiry” (Williams 2011:18). Documents, interviews, observations and questionnaires are the four distinct types of data generation techniques used in survey research. A questionnaire is adequate to collect data for this study. Succinctly, the *Oxford English Dictionary* (OED) defines a questionnaire as “a formulated series of questions by which information is sought from a selected group, usually for statistical analysis”. Data collected were subjected to statistical analysis.

Oates (2006:93) suggests that surveys are quantitative and are mostly associated with the philosophical paradigm of positivism which underlies what is called “the scientific method”. Exploratory data analysis (EDA) was selected as the main method for statistical data processing and analysis. According to Jackson (2013:50) despite the efforts of the quantitative researcher to seek scientific truth, in some cases, choices of the positivist are oriented by his subjectivity. Unfortunately, that is a panoply of impediments that positivists have to deal with in order to remain objective and scientific in their studies. This chapter encompasses several sub-sections.

3.2 Research design

Creswell (2009:3) distinguishes three types of research design: quantitative, qualitative and mixed methods. The present study is conducted within the quantitative paradigm. A quantitative study is “one in which the data we collect and analyse involves the accurate measurement of phenomena and, often, the application of statistical analysis” (Murray & Hughes 2008:200). This study used cross-sectional surveys as a strategy of inquiry. Surveys are non-experimental designs for collecting data through questionnaires. A survey questionnaire designed for the circumstance is discussed in the instrumentation section below along with the variables that meet the research questions and the objectives outlined in the initial phase of this study. Questionnaires and survey research

are interrelated, in most cases, responses from questionnaires are analysed through means of statistical techniques (Berndtsson, Hansson & Olsson 2007:63). Before subjecting the data collected to statistical analysis, the survey questionnaire was piloted using a small group of people. The underlying philosophical paradigm is positivism as surveys are strongly associated with such a paradigm (Oates 2006:93).

Regarded as set of scientific procedures for data collection, surveys have pros and cons. The advantages of surveys are as follows: (a) cost (surveys are relatively inexpensive); (b) flexibility (surveys can be administered in many modes); and (c) dependability (anonymity) (Oates 2006:104). Survey research produces empirical data on real-world observation (Kelley, Clark, Brown & Sitzia 2003:262) and provides a numeric representation of trends, attitudes, or opinions of a population being studied (Creswell 2009:145). In addition, from Sincero's (2012) perspective, surveys provide participants with a standardised stimulus, there is little or no subjectivity from the researcher, and they yield statistically significant and precise results.

Notwithstanding the popularity of surveys in the field of computer science, many critics consider it as inefficient. Kelley et al (2003:262) and Oates (2006:105) suggest that data produced using surveys are likely to lack depth. Oates (2006:299), for instance, suggests that surveys are weaker than experiments, arguing that surveys can only confirm an association (with reference to interpretations of patterns observed in the responses of the participants). He adds that surveys do not establish cause and effect. Accordingly, Kelley et al (2003:266) suggest that sometimes survey research is regarded as an easy design method for the simple reason that, in some cases, surveys do not stand up to academic scrutiny and have less value in terms of a contribution to knowledge. Another negative point about questionnaires is from Gillham (2008:1) who suggests that no single method has been so much abused. The limitations of the respondents vis-à-vis closed questions are also much criticised because of predetermined answers.

Finally, attention is now directed to the reasons that led to a choice of such research strategy. Survey research may be considered a "quick fix" for research methodology (Gillham 2008:1), nonetheless, there are three main reasons to justify the choice of an empirical approach for this study:

- It is suggested that surveys are widely accepted and used in the field of information systems and computer science and it is relatively easy to collect large data from participants through standardised instruments such as questionnaires (Oates 2006:93), which are entirely adequate to bring answers to the research questions underlined in this study. In addition, there is the versatility and efficiency of surveys (Check & Schutt 2011:183) that justify the choice of such research paradigm.
- The main objectives of this research encompass the following three key points: a) determine issues related to agile requirements engineering; b) develop a set of recommended tools to deal with these issues; and c) determine the extent to which key principles of the Agile Manifesto are perceived by software practitioners. Only agile software practitioners are holders of such knowledge (their problems). All these objectives are achievable by surveying directly software industries.
- In addition to the fact that objectives in this study are quantifiable, cost and time are other reasons that led to an empirical study. Relative to some other strategies, surveys can produce a lot of data in a short time at a reasonably low cost (Oates 2006:104). Judiciously, cost and time can also be predicted in advance, which considerably helps in terms of planning and managing the project.

Quantitative methods are adequate for deductive approaches (Borrego, Douglas & Amelink 2009:54). Some researchers, have manifestly criticised the use of surveys as research methodology as too simplistic or may have less contribution to knowledge but despite all these criticisms and after proper scrutiny of the objectives of this study, surveys are a good fit for this research. The pros of surveys certainly overcome the cons from the perspective of this study. To reiterate, overall, this study is descriptive, quantitative in nature, based on cross-sectional surveys, with questionnaires as primary source of data. The next section justifies the objectives in the instrumentation section and the choice of an empirical approach for this study.

3.3 Methodology

The term ‘methodology’ is etymologically from an old Greek word (Berndtsson et al 2007:13). It is the amalgamation of research strategies and data generation techniques used during the entire research process (Oates 2006:112) and is concerned with main issues such as bringing answers to the research questions (Jackson 2013:57). This section encompasses the research instrument, the target population and sample, as well as the data processing piece.

3.3.1 Research instruments

Instrumentation

The instrumentation section of a methodology chapter describes the particular variables used to give answers to the research questions and hypotheses (Rudestam & Newton 2014:88). As previously mentioned, this study is based on survey-based research. The core of a survey is its questionnaire (Krosnick & Presser 2010:263). The research instrument was designed to achieve the research objectives outlined in the introduction chapter. The designed research questionnaire is divided into seven sections.

The first section (“General Information”) is for demographic purposes, in other words, it helps to collect information about the identity of the respondents. Information such as the company name, the role of the respondent in the company, and his/her contact details are collected in this section. Although this section is of less importance to the study it gives structure to the questionnaire and it would be odd to have a survey questionnaire without demographic details section.

Section two (named “Agile Processes”) includes variables to collect data about agile methods used in order to get a sense of agile framework, method, or methodology used in the software industry. Respondents were asked to select from the list of the ten most popular agile methods listed, in other words, they were given explicit response categories (Check & Schutt 2011:168). They also had the ability to list theirs in the ‘other’ field. This helps to point out the most common agile method in the South African software industry.

The third section (called “Agile Requirements Engineering”) has two questions. The first question is a Likert-type question varying from 1 to 5. This section collects data regarding the extent to which agile practices are adopted. The agile practices listed were the ones related to ARE practices such a) face-to-face communication; b) iterative requirements engineering; c) requirements prioritisation, etc. The second question intends to determine activities of the ARE performed during iterations. Since the ARE process is performed in iterations (Cao & Ramesh 2008:67), main traditional activities of the RE were listed to select from along with a field to extend the list. Answers from both questions helped to delimit the scope of ARE in terms of best practices. That is, the first objective of the study.

Section four (or “Issues related to Agile Requirements Engineering”) collects data about different issues during activities of the ARE process. This question relates to the second objective which seeks to define and determine a baseline for ARE problems. It is divided into six sub-sections. Issues in elicitation, analysis, documentation, validation, management are explored to allow respondents to select all listed issues during those respective phases. The same applies to issues resulting from negotiation, modelling, or evolution in the ARE process which are consolidated in the ‘other issues’ section.

Section five (titled “Tools for Agile Requirements Engineering”) includes variables that help to collect information to determine set of recommended tools used to tackle issues related to ARE. That is the fourth objective of this study. This section is essentially defined by questions of type list.

Customer interaction and collaboration is the sixth section (named “Stakeholders Interaction & Collaboration”) of the survey questionnaire. In this section, variables measure a) customer experience on ASD; b) gather information about the type of software companies usually develop the most; and c) give the ratings of ARE principles. The ratings of the ARE principles (Likert-type), for instance, are to d) get the degree of comprehension of these principles from e) the software practitioners’ perspective in order to achieve the fifth objective. Then f) to evaluate the degree of interaction between software practitioners and customers constitutes the sixth objective. Another sub-question in this section is to find out the amount of requirements implemented or thrown-away. In this sense, a Likert-scale question was created to find out whether or not developers consider dynamic requirements during software development. This also

determines how interactive agile teams are. Further, this question also helps to achieve the third objective which relates to the receptivity of software practitioners regarding dynamic requirements.

Finally, section seven (“Non-Functional Requirements”) is about the NFR aspect of software development, which is usually forgotten or given less attention. The NFR constitute a major concern for software development (Cao & Ramesh 2008:64). This section gathers data about the types of NFR considered during ASD, and the constraints that NFR put on projects. This contributes to the achievement of the second objective which seeks to determine common problems of the ARE. The formats of the questions in this section are in the Likert-type format and lists type.

The questionnaire was developed in accordance with objectives stated above. Variables encompassing the instrument were consistently defined to collect all information necessary for data analysis. Before commencing data collection, the instrument needed piloting.

Pilot test

Researchers usually carry out a preliminary test of the questionnaire before it is used extensively (Walliman 2010:98) and this is termed the ‘pilot study’. It is usually performed on a small group of people (Oates 2006:226) representative of the target population with the intention to refine and improve the data instrument, explicitly, in this case the survey questionnaire. Therefore, the questionnaire was piloted before commencing data collection from the participants.

Data collection

Data was gathered using a questionnaire attached to the emails (see Appendix 3). An online version, designed using Google forms was also used. Indeed, the Internet has become a very promising medium to collect information (Benfield 2006). The participants were from LinkedIn, from BizCommunity or Top500, both online repositories of companies in South Africa. Some of the participants who were very enthusiastic about this study volunteered to help by inviting other participants via email by stipulating the importance of this research. This technique, termed the ‘snowball technique’, helped

in getting more participants and positive responses came out of it. The participants listed online as software developers were contacted by telephone. Those who agreed to participate in the survey questionnaire were sent three files which included the survey information letter, the survey informed consent letter and the questionnaire.

The LinkedIn participants were selectively contacted. The criterion for selection was based on the job title of the participants (project manager, java developer, agile developers, etc.). A short email was sent introducing the motif of the research with a request to be added to the researcher's network. Those who agreed to participate in the survey did accept the LinkedIn invite and then, were sent an email with the survey questionnaire. A few LinkedIn participants who were interested in the research survey did invite other friends to also participate (another good advantage of the snowball technique). Follow-up emails were sent almost every week as a reminder. During the last month of data collection, telephone calls were made to respondents every week to use the advantage of synchronous communication. All responses (electronic versions and online versions) were consolidated in Google forms.

Finally, the philosophical paradigm of positivism is the central focus of surveys. This is based on the notion that scientific methods yields results that are valid, reliable and replicable and that the researcher is independent (Pather & Remenyi 2005:142). In addition, the quality of a positivist research is assessed using criteria such as, reliability, validity or objectivity (Oates 2006:287).

Reliability

The accuracy of the research method (meaning), and the accuracy of the implementation method (the questionnaire) in measuring an honest representation of findings define reliability (Berndtsson et al 2007:56, Jackson 2013:57). Ensuring the simplicity of survey questionnaire, the clarity and understanding (non-ambiguity) of these questions is the responsibility of the researcher.

To determine the degree of correlation between items in a questionnaire, the researcher made use of reliability analysis which is the "overall index of the repeatability or internal consistency" (SPSS IBM 2011:460), and Cronbach's alpha (α) was used. Briefly, Cronbach's alpha (also called the coefficient alpha) is a regularly used technique to

determine an internal reliability of an instrument. Cronbach (1951:302) suggests that a coefficient alpha is the “mean of all possible split-half coefficients”. Its value varies from 0 to 1. When alpha is less than 0.6, it indicates that the consistency reliability is unsatisfactory, but it is interpreted otherwise if the value of alpha is over 0.7 for basic research (Panayides 2013). For this study, the Cronbach’s alpha coefficient is 0.906 and the coefficient based on standardised items is 0.869.

Validity

Validity indicates the extent to which a test in fact measures what it purports to measure (Rudestam & Newton 2014:96; Thanasegaran 2009:37). Berndtsson et al (2007:56) suggest that validity is “the relationship between what you intend to examine and what you actually examine”. Validity involves accurate questioning, collected data, and interpretation in relation to the research question (Jackson 2013:57). The literature distinguishes different types of validity: a) construct validity, b) content validity, and c) factorial validity. For the context of this study, only construct validity and content validity vis-à-vis questionnaires will be discussed.

- C*onstruct validity* is concerned with the structural aspects of the questions (Oates 2006:227). Hence, the instrumentation section consistently discussed the objectives of this research by matching them with the sections of the survey question.
- *Content validity* is a non-statistical type of validity. It is simply concerned with the intentions behind the questions asked (Oates 2006:227).

Only items related to the objective of this study were selected (sampling adequacy) to include in the survey questionnaire. And to achieve construct validity, the survey questionnaire was piloted. This technique achieves construct validity.

Objectivity

Zikmund et al (2012:8) suggest that applied research ensures objectivity through scientific methods by gathering evidence and testing new ideas. More exhaustively, Reiss and Sprenger (2013) suggest that objectivity “expresses the idea that the claims, methods and results of science should not be influenced by particular perspectives, value commitments, community bias or personal interests, to name a few relevant

factors". Thus, giving full opportunity to participants to complete the survey questionnaire (no invasion of any kind), and using statistical tests to uncover values of the data collected (Stanford Encyclopedia of Philosophy 2014; Williams 2011:66) was one way to ensure objectivity.

3.3.2 Target population and sample

Target population

The target population are all members of the entire agile team involved in software production. Comprehensively, what constitute the target population are the following agile team members:

- project managers
- software architects
- engineer managers
- product developers
- software quality assurance engineers
- software testers
- software requirements engineers.

A sample came from the target population or rather the accessible and available population.

Sample

Selecting a sampling frame was a challenge since there was only a vague idea of the target population (non-existence of list of all agile companies in South Africa). In addition, uncertainty about the representativeness of the sample (Oates 2006:97) hung over this study. For obvious reasons, the researcher relied on a published list from Top500, that is, a repository of best managed companies in South Africa. This list was used in conjunction with another list in platforms such as Google or BizCommunity (another South African website repository for software development companies). Owing to the extent of the target population, determining a sampling frame in this context was a cumbersome and superfluous task. Considering the uncertainty of the

representativeness of the sample (Oates 2006:97), convenience sampling, a non-probabilistic sampling technique, was used to determine a tentative sample. As this research progressed, participants were added based on the criteria for inclusion (which was the ability to develop software in the agile environment) and on their accessibility and availability. Participants who were suggested by other participants were included in the study, provided that they agreed to participate (this is termed the snowball technique as suggested by Oates (2006:98)). Thus, the final sample was defined using a convenience sampling technique alongside a snowball technique. In terms of the size limit of the sample, the expectation was to have conveniently 200 returned questionnaires. All data collected from the respondents was collated for eventual data analysis.

3.3.3 Data analysis

Statistical analysis commenced once data collection was over. This task was to subject the information collected from participants to statistical analysis. Oates (2006:245) suggests that the idea of quantitative data analysis is to look for patterns in the data and draw conclusions. Seventy-five percent of the responses were done online. The other 25% were captured onto Google forms. Thus, all responses were exported to a file and then imported into SPSS. The IBM SPSS Grad Pack Premium v23 was used in conjunction with Microsoft Excel v10 for quantitative data analysis. Greasley (2007:7) suggests interval or ratio, ordinal, and categorical or nominal as three distinct types of data for statistical analysis. During the import process 157 variables of types nominal, ordinal and interval were created in SPSS to accommodate the entire questionnaire. It is important to mention that the number of variables created in SPSS is high because of the number of multiple responses question. Every item of multiple responses set is treated as a separate variable in SPSS. On a different note, one particular aspect of this research is the moderate response rate or return rate (RR).

With regard to the statistical analysis, keeping in mind that this study is descriptive, exploratory data analysis (EDA) was used. Simple analysis or visual aids (Oates 2006:249) and simple descriptive statistical techniques are examples of EDA practices. The survey questionnaire has two scales of measurement: multiple response questions or dichotomies (categorical), and ratings questions (Likert-scale). For multiple response questions, a variables set was computed in order to group variables by question and

then frequencies were calculated. It was then much easier to get visual aids (such as bar charts, histograms, or pie charts) once the frequencies were computed. For the Likert-scale type questions, cumulative frequencies on those variables were computed. Bar charts usefully displayed the results and tables based on the cumulative frequencies were drawn.

3.4 Ethical considerations

Ethical questions are apparent today in issues such as: a) personal disclosure; b) authenticity and credibility of the research report; c) the role of researchers in cross-cultural contexts; d) and issues of personal privacy through forms of internet data collection (Creswell 2009:87). Another apparent area in research ethics that needed attention was missing data. Missing data constitutes an impediment to ethically sound research. Briefly, missing data is information that variables fail to capture (Enders & Gottschall 2011:358). To reiterate, the philosophical paradigm for this study was positivism which was materialised by data collection from participants through questionnaires. Pursuing this objective while considering the participants rights, presents complex issues (Drew et al 2007:66). Main ethical issues concerned data collection and issues related to data processing and analysis.

3.4.1 Ethical issues in data collection

Ethical issues are always raised in any research when working with participants, especially humans (Walliman 2010:42). Participants have basic rights that are inherently fundamental to their lives such as:

- the right not to participate
- the right to withdraw
- the right to give informed consent
- the right to anonymity
- the right to confidentiality.

Participants were treated with due ethical consideration, meaning that everyone involved in this research was treated fairly and with honesty.

The Unisa guidelines mention that before commencing any data collection for research purposes, an ethical clearance must be obtained. The Ethical Clearance Application form was fully completed and submitted to the School of Computing for approval. Only after receiving the ethics clearance certificate (Appendix 4), were participants contacted and sent the instruments for data collection. Ethical issues in data collection are of paramount importance for research in any academic study, more especially in computer science. Equally important are ethical issues in data processing and analysis.

3.4.2 Ethical issues in data processing and analysis

Ethical issues arise in discussions about codes of professional conduct for researchers and in commentaries about ethical dilemmas and their potential solutions (Creswell 2009:87). As already mentioned, information collected was subjected to statistical analysis, essentially EDA. The importance of EDA in this study was to help understand raw data. Ethical issues in data collection have been discussed in the previous section. Data processing and analysis is heavily dependent on data collection procedures.

Jerry, Jack and Stephen (2015:79-83) have suggested the following “seven areas of scientific dishonesty” which are believed to have serious impact on data processing and analysis. These areas are:

- plagiarism
- fabrication and falsification
- non-publication of data
- faulty data-gathering procedures
- poor data storage and retention
- misleading authorship
- unacceptable publication practices.

Fabrication and falsification, faulty data-gathering techniques, plagiarism or devious publication practices are self-explanatory and were avoided as any of these four areas can seriously influence the outcome of a research and tarnish the image of the researcher. Poor data storage and retention can be a potential risk to the privacy of the participants. The researcher is responsible to find solutions to all seven areas noted above if these problems threaten the research. One set of solutions to these problem areas was to follow principles of an ethical researcher such as “no unnecessary

intrusion", "behave with integrity" or "follow appropriate professional codes of conduct" (Oates 2006:60). Honesty is one of the main criteria for ethically sound research so any type of dishonesty and use of furtive methods (Walliman 2010:49) such as being selective in the data used or data distortion were avoided. Such potential 'lapses of honesty' were handled with care in order to maintain scientific objectivity in this study.

3.5 Summary

This section has discussed the appropriate scientific method used to answer the research problem. To reiterate, this study's aim is to list issues in the ARE process as 37% of problems experienced by software practitioners are in the requirements phases. Another important aspect of this study is to determine a set of recommended tools and frameworks used to tackle issues mentioned above. The dynamism of the global market place and customer needs are contributing factors to software complexity. Questions in this study that relate to:

a) values of agile principles;
b) collaboration and interaction between stakeholders and developers; or the
c) the importance of NFR in ASD

require quantitative data.

Thus, this study is quantitative in nature. Within the quantitative paradigm, surveys were adequately selected as a research strategy. Two non-probability sampling techniques were used. Convenience sampling technique and snowball technique were used to get a representative sample for this research. Questionnaires were used to collect quantitative data through surveys (research strategy). Issues of validity and reliability have been discussed as well as ethical issues related to the research.

This research design will bring some answers to issues related to the ARE process in ASD. According to some researchers, these problems are endemic to the software industry. In addition, these are constant issues of modern software production. The next chapter shows what the data really represented.

CHAPTER 4: DATA PRESENTATION, ANALYSIS AND INTERPRETATION

4.1 Introduction

Behavioural science uses questionnaires as powerful tools to acquire information (Mangal & Mangal 2013:339). The objectives of this research required data collected from the survey questionnaires. The intention was to collect data from software companies in South Africa to get the sense of ARE issues in the industry. Self-administered survey questionnaires were conducted as outlined in the previous chapter, i.e., the methodology chapter. Different sets of answers were received from different participants. Answers from respondents varied from the unavailability of time to take the survey questionnaire, to the irrelevancy or inadequacy of the survey in their field of work, not to mention their strong beliefs in keeping private the company data even though this data collection does not cover sensitive data aspect.

This chapter encompasses the results presentation section, the response rate section, as well as the discussion about the findings. The results presentation is outlined through tables, graphs and histograms and narrative preceding each of them. The discussion section focuses on interpreting and explaining some of the findings in this study. And as with any human deeds, there is a section allocated to limitations. Possible future research section and a conclusion follow on to close this chapter.

4.2 Response rate

A total of 227 participants were contacted. Out of 227, only 107 agreed to participate in the survey. About 20% of participants were contacted via the professional social network LinkedIn. Approximately 10% of all the participants were either not interested in participating in the survey, or found this study irrelevant to their field of work. Some did not have time altogether. Basically, 10% of the companies contacted develop software in South Africa but have not yet adopted the agile principles or are completely inexperienced with the concept of ASD. Other participants agreed to participate in the

survey but distance themselves after receiving the survey questionnaire. It became important to keep reminding them of the importance of their participation. To remain objective, and do ethical and sound research with a low response rate is very challenging. Variation in willingness and ability to respond (Frary 2003:169) to these questionnaires remain an impediment to “behavioural science” in general. Overall, 25 participants responded, which makes about a 23% response rate (or return rate) for this study from the 107 participants who agreed to participate.

4.3 Background of the participants

This study involved participants from diverse backgrounds. Those who participated are in positions such as senior developer, software engineer, solution architect, business analyst, project manager, solutions engineer, and IT manager.

4.4 Results presentation and interpretation

The returned questionnaires were captured into SPSS (Grad Pack Premium v23). Collected data are systematically analysed and presented through tables, bar charts and pie charts in subsequent paragraphs. The objectives are tied to the five research sub-questions. Discussions about the steps performed to answer every research question and achieve every research objective in this study follow every question. This is exclusively based on the primary source of data generated.

4.4.1 What are the common issues in agile requirements engineering in the South African software development industry?

Issues of the ARE process are diverse and are encountered throughout software development. As already mentioned, traditional phases have been used to better segregate and group issues related to this process in general. Whether issues of elicitation, documentation, analysis, validation, management, or other issues such as negotiation, modelling, or evolution, the same statistical analysis techniques have been performed. Variables sets were defined respectively for elicitation, analysis, documentation, validation, management and other issues and then frequencies were

computed to get the total number of occurrences and percentages of each issue. The output is a table (see Table 3) listing all issues grouped by phase.

Table 3: Issues related to agile requirements engineering

		Responses		Percentage of cases
		N	Percent	
Issues of Elicitations	Scope problems	17	13.5%	**68.0%**
	Lack of clarity	13	10.3%	52.0%
	Wastages	8	6.3%	32.0%
	Lack of trust	8	6.3%	32.0%
	Incompleteness	8	6.3%	32.0%
	Misinterpretations	11	8.7%	44.0%
	Omissions	12	9.5%	48.0%
	Unrealistic expectations	11	8.7%	44.0%
	Vague customer needs	12	9.5%	48.0%
	Inconsistent information	9	7.1%	36.0%
	Scalability issues	3	2.4%	12.0%
	Issues of prioritisation	14	11.1%	**56.0%**
Issues of Analysis	Scalability issues	3	12.0%	13.0%
	Issues of prioritisation	14	56.0%	**60.9%**
	Issues of requirements negotiation	8	32.0%	34.8%
Issues of documentation	Lack of proper documentation	16	29.1%	**64.0%**
	Staff turnover	8	14.5%	32.0%
	Minimal documentation	16	29.1%	**64.0%**
	Unavailability of customer representative	15	27.3%	**60.0%**
Issues of validation	Evolutionary prototyping issues	10	37.0%	43.5%
	Lack of proper validation tools & techniques	17	63.0%	**73.9%**

		Responses		Percent-age of cases
		N	Percent	
Issues of management	Lack of proper management tools	12	28.6%	**50.0%**
	Lack of Requirements Traceability	12	28.6%	**50.0%**
	Requirements changeability	14	33.3%	**58.3%**
	Problem with version control	4	9.5%	16.7%
Issues of Others	Customer rigidity	6	5.3%	25.0%
	Limited access to project stakeholders	9	8.0%	37.5%
	Indecisive project stakeholders	12	10.6%	**50.0%**
	Views inconsistencies	8	7.1%	33.3%
	Misleading cue	4	3.5%	16.7%
	Hidden functionality	7	6.2%	29.2%
	Missing functionality	7	6.2%	29.2%
	Vague requirements	12	10.6%	**50.0%**
	Conflicting priorities	12	10.6%	**50.0%**
	Overwhelming participation	2	1.8%	8.3%
	Inadequate IT solutions	6	5.3%	25.0%
	Problem domain complexity	3	2.7%	12.5%
	Ambiguous requirements	10	8.8%	41.7%
	Inappropriate functionality	5	4.4%	20.8%
	Budget problems	10	8.8%	41.7%

The first question intended to find out the common issues of the ARE process. The objective was to determine the issues that impact the most RE during ASD. Table 3 lists issues experienced in the software industry in general. Table 8 (which is an excerpt from Table 3) presents the top thirteen most common issues in the ARE process. Lack of proper validation tools & techniques (73.9%) is highly ranked as a major problem in the industry, followed respectively by scope problems (68%), lack of proper

documentation (64.0%), minimal documentation (64.0%), or issues of requirements prioritisation (60.9%). The top 13 issues extracted from Table 3 are encountered in all phases of software development process. This indicates that during agile development, focus should be put on every aspect of iterations, with special focus on the validation phase, an area where software practitioners need help the most.

The Agile Manifesto focuses on a minimum essential amount of documentation (Cobb 2011:7) in order to lessen the lack of information in case of staff turnover. This study shows that lack of proper documentation (Table 3) remains a major concern. One technique for requirements validation is face-to-face communication (Cao & Ramesh 2008:66, Ramesh at al 2010:469). However, this technique is not suitable for the validation phase since 73.9% still lack proper validation tools & techniques. To correct issues emanating from the documentation phase, there is a need to fix the ambiguity created by the choice of appropriate tools and techniques in the software industry which remains a major issue. This was exhaustively emphasised by Highsmith (2003:4), Mnkandla (2008:3), and Elshandidy and Mazen (2013:479).

Common issues of the ARE process also include NFR. Cao and Ramesh (2008:64) suggest that non-functional requirements constitute a major concern for software development in general. The constraints that NFR put on ASD sometimes depend on the nature of the software developed. To determine whether or not lack of consideration of NFR cripples software production, the pie chart below (Figure 4) portrays the varied opinions of the respondents.

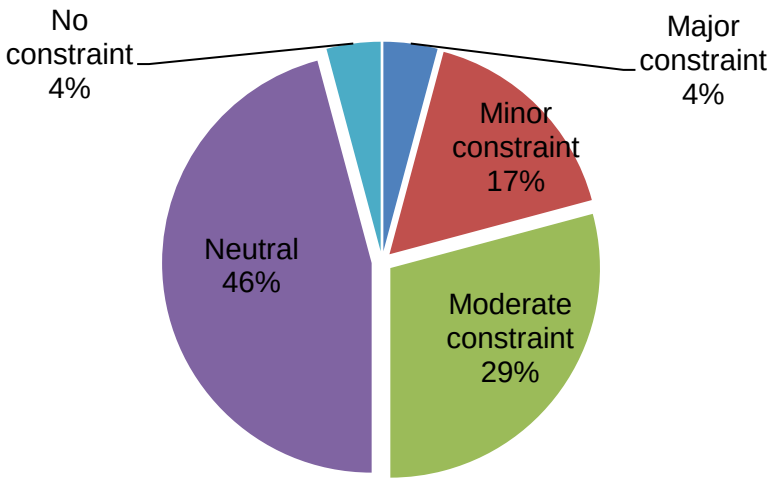


Figure 4: Amount of NFR constraints on ASD

With reference to the issues related to the process, NFR always presents problems in the software industry because it is given little or no attention. Sometimes, lack of consideration to NFR leads to software systems failures (Singh & Saxenna 2014:547). Only 4% believe that NFR put major constraints on ASD; while 4% think NFR does not influence ASD; yet a 46% are neutral about the topic; however, 29% opt for moderate constraints and 17% for minor constraints. These results show how divided software practitioners are on the subject. This is an indication that NFR are poorly defined in agile approaches (Zhu 2009:24). This requires more scrutiny. One consequence of poorly defined NFRs is the following illustration. Seventy-six percent of the applications that customers request are web-based applications. Web-based applications are accessed via the Internet. 'Cloud computing' is a new term that properly describe this trend. Web-based applications are always at risk because of public accessibility. The NFR on these applications must be properly defined in order to remain fully operational and safe. The confidentiality, integrity and availability (or CIA) of the data of the company depends intrinsically on how NFR are defined. Web-based applications security ought to be up to standard, because of the threats constantly encountered over the Internet. This can be done by putting more emphasis on NFR such as security, usability, performance, and scalability etc. For instance, Figure 15 shows that usability (60.9%) and performance (73.9%) are considerably important in this study. These are also main key features of web-based applications. Considering the divergent views of software practitioners on NFR considerations (Figure 5), NFR integration remains an issue of the ARE.

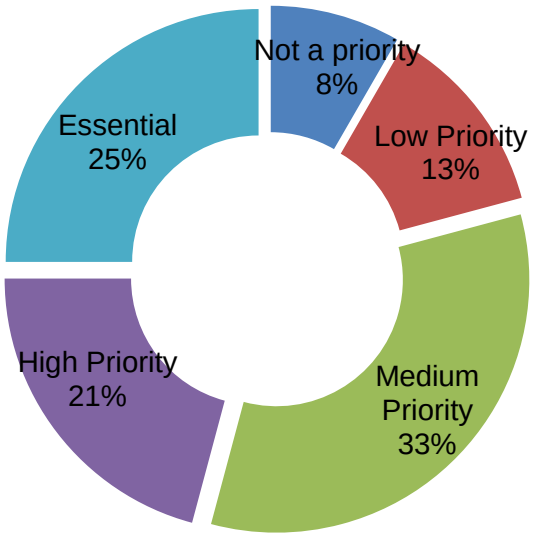


Figure 5: Proportion of NFR consideration as priority in ASD

The following table (Table 4) with cumulative counts was computed to give the overall idea about the activities of the ARE process that put too much constraint on the projects. In addition, a bar chart (see Figure 6) based on the above table is presented.

Table 4: Overall constraints from different activities of ARE

	Responses		**Percentage of cases**
	N	Percent	
Elicitation	7	12.3%	30.4%
Analysis	8	14.0%	34.8%
Documentation	9	15.8%	39.1%
Validation	9	15.8%	39.1%
Management	9	15.8%	39.1%
Modelling	3	5.3%	13.0%
Negotiation	6	10.5%	26.1%
Traceability	3	5.3%	13.0%
Evolution	3	5.3%	13.0%
Total	**57**	**100.0%**	**247.8%**

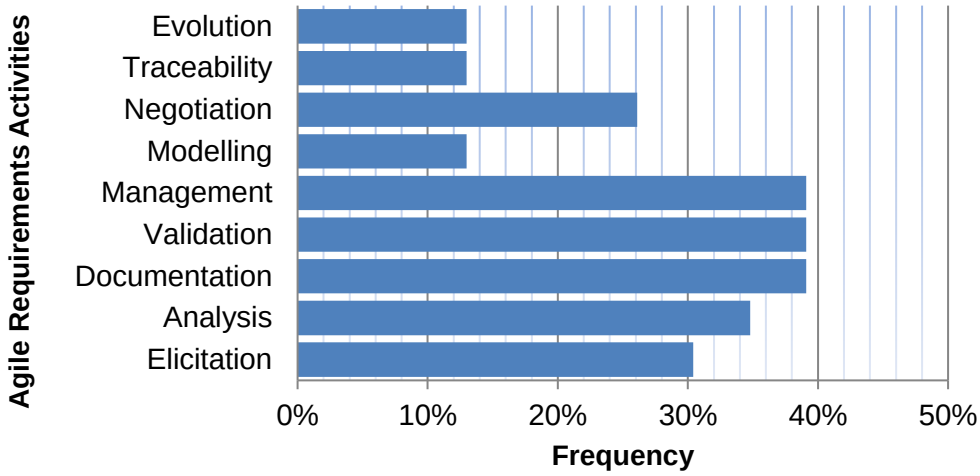


Figure 6: Frequency of constraints from different activities of ARE

Overall, requirements documentation, requirements validation and requirements management are the areas that need improvement in terms of RE. And as for NFR, it should be integrated during software development, not during post-delivery maintenance.

4.4.2 What are the tools and techniques that help in dealing with such issues?

The second question intended to find tools and techniques used in the software industry. The objective relating to this question sought to determine a set of recommended tools and techniques to deal with the above issues. Frequencies were run to get the total number of counts for each tool or technique along with percentages after defining a variables set to group all tools and techniques. The tools and techniques with higher percentages are represented in Figure 7 to show the popular tools and techniques used or adopted in the software industry. Tools and techniques for agile requirements engineering include tools for requirements prioritisation, JAD sessions, ScrumWorks, Documentation Management System and so on. Other tools and techniques mentioned by participants included Trello, Asana, or SQL (see Figure 7).

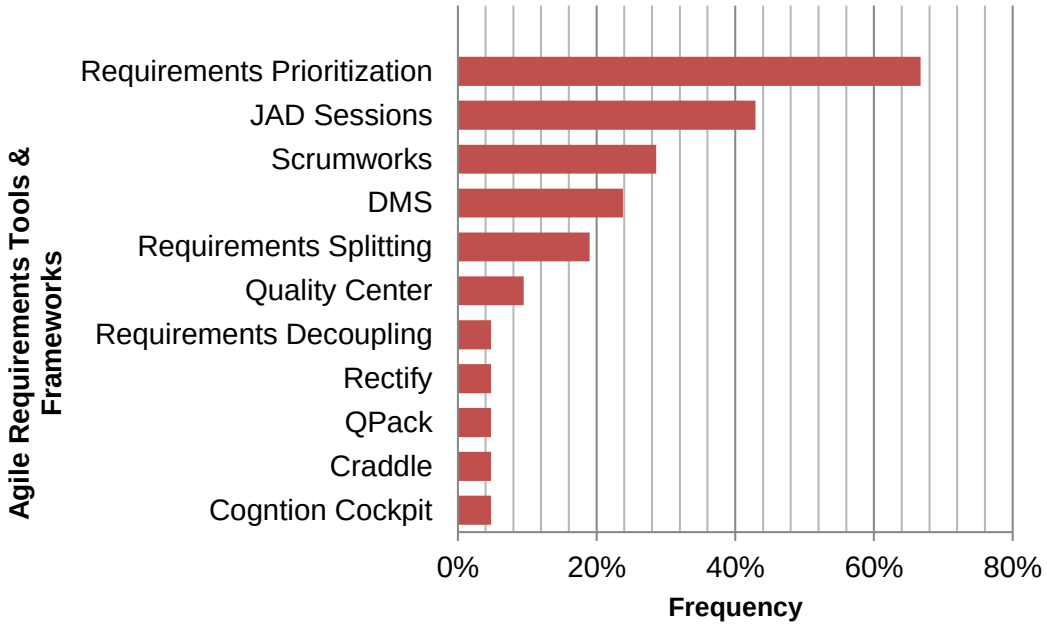


Figure 7: ARE Tools and techniques by order of preference

Figure 7 shows that requirements prioritisation is the most popular technique (66.7%) in ASD. According to Inayat et al (2014:8), requirements prioritisation is an intrinsic part of iterations during ASD. Tools or techniques such as JAD sessions (42.9%), ScrumWorks (28.6%), requirements splitting (19.0%), and documentation management system (23.8%) are relatively common tools used in the software industry, yet the issue of selecting appropriate tools and techniques for given problems remains (Mnkandla 2008:3). No study can stress this complexity enough. Nevertheless, to achieve the objective noted in this section, Figure 7 can be considered the set of recommended tools and techniques (this include commonly used tools such as SQL, Asana, or Trello listed by the participants).

4.4.3 How do software practitioners value agile principles that relate mainly to requirements engineering?

The objective (tied to this question) was to get a sense of agile principles such as customer satisfaction, simplicity, and communication and so on. The perception of the respondents was measured using five items on a 5-point Likert scale (where 1 is less important and 5 very important). Essentially, the Cronbach alpha for the reliability test for this subset is 0.819; thus satisfactorily valid for analysis (Tavakol & Dennick 2011:54).

The adequate statistical analysis for a ratio question is to run frequencies on every agile principle and then cross-tabulate the results (see Table 5). Afterwards, a graph based on the percentages calculated (Figure 8) is drawn to show the perception of the respondents on agile principles.

Table 5: Preferred agile practices in the industry

	Not Important	Somewhat important	Quite important	Very important	Extremely important
Face-to-face communication	0.0%	0.0%	16.7%	45.8%	37.5%
Iterative requirements engineering	0.0%	0.0%	41.7%	33.3%	25.0%
Requirements prioritisation	0.0%	0.0%	20.0%	40.0%	40.0%
Prototyping	0.0%	12.5%	29.2%	29.2%	29.2%
Test-driven development	0.0%	4.2%	37.5%	33.3%	25.0%
Review meetings and acceptance test	0.0%	4.2%	20.8%	33.3%	41.7%
Continuous validation	0.0%	4.3%	17.4%	39.1%	39.1%

The following figure is a graphical representation of the Table 5 above.

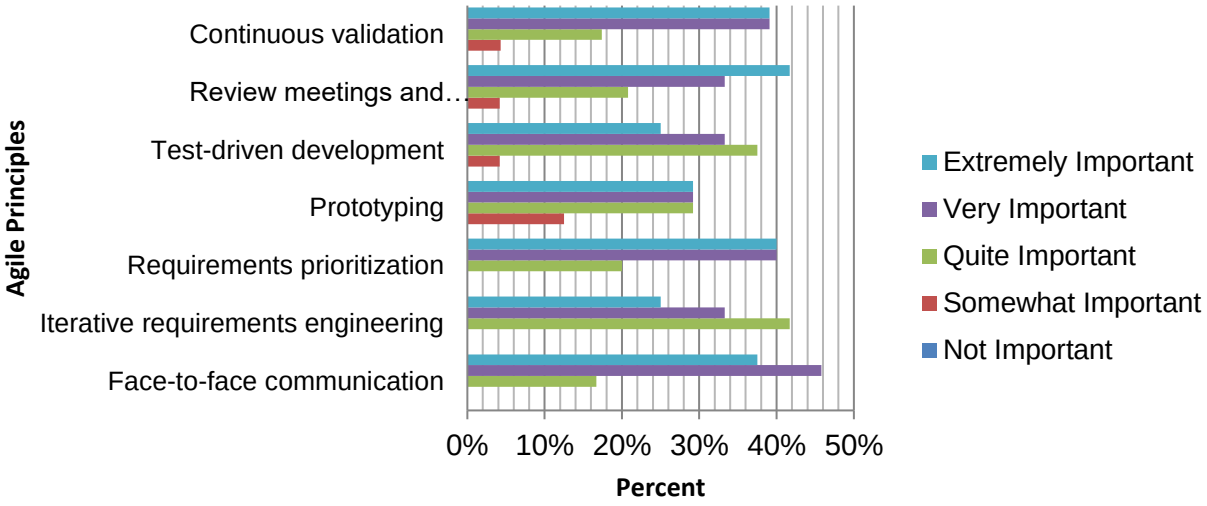


Figure 8: Preferred agile practices in the industry

The third question sought to determine how much value software practitioners put on agile principles that relate mainly the ARE process. In agile development, frequent review meetings, face-to-face communication, or prototyping are agile principles that define requirements engineering. Figure 8 shows that while all seven agile principles are important to a certain extent; two in particular are extremely popular than others: a) the review meetings and acceptance test principle, and b) the requirements prioritisation principle. Principles such as: a) face-to-face communication; b) requirements prioritisation; and c) continuous validation are very important to software practitioners. Another observation is that all these seven principles were rated by the majority as quite important. This confirms the philosophy behind the Agile Manifesto. It is good to value tools and processes, or comprehensive documentation, or contract negotiation, but it is more important to value the principles that relate to customer interaction.

4.4.4 How collaborative are customers and software practitioners in terms of requirements engineering?

Customer collaboration and face-to-face communication are principles of the ARE process that demonstrate a degree of interaction between customers and software practitioners. Two objectives relate to this question: a) determine the degree of interaction between customers and software practitioners; and b) determine the receptivity of software practitioners towards dynamic requirements. For the first objective, all ten principles of the agile manifesto are rated in a 5-point scale question

varying from 1 (less important) to 5 (very important). The Cronbach alpha coefficient (reliability test) for this subset is 0.886; thus adequately valid for analysis (Tavakol & Dennick 2011:54).

In order to know how collaborative customer and developers are, a cross-tabulation is computed (Table 6) to get cumulative frequencies. In addition a stacked bar chart (Figure 9) shows the valued principles of the AM.

Table 6: The perception of software practitioners towards agile principles

	Not important	Somewhat important	Quite important	Very important	Extremely important
Customer satisfaction	0.0%	4.3%	13.0%	17.4%	65.2%
Welcome changing requirements	8.3%	4.2%	16.7%	41.7%	29.2%
Frequent working short releases	0.0%	0.0%	20.8%	33.3%	45.8%
Collaboration (business people & developers)	0.0%	0.0%	20.8%	37.5%	41.7%
Face-to-Face communication	0.0%	0.0%	33.3%	25.0%	41.7%
Working software is measure to progress	0.0%	0.0%	41.7%	29.2%	29.2%
Continuous attention to technical excellence	0.0%	4.2%	20.8%	29.2%	45.8%
Simplicity (is essential)	0.0%	4.2%	20.8%	33.3%	41.7%
Team self-organisation	0.0%	4.2%	33.3%	20.8%	41.7%
Frequent refactoring	4.2%	4.2%	33.3%	37.5%	20.8%

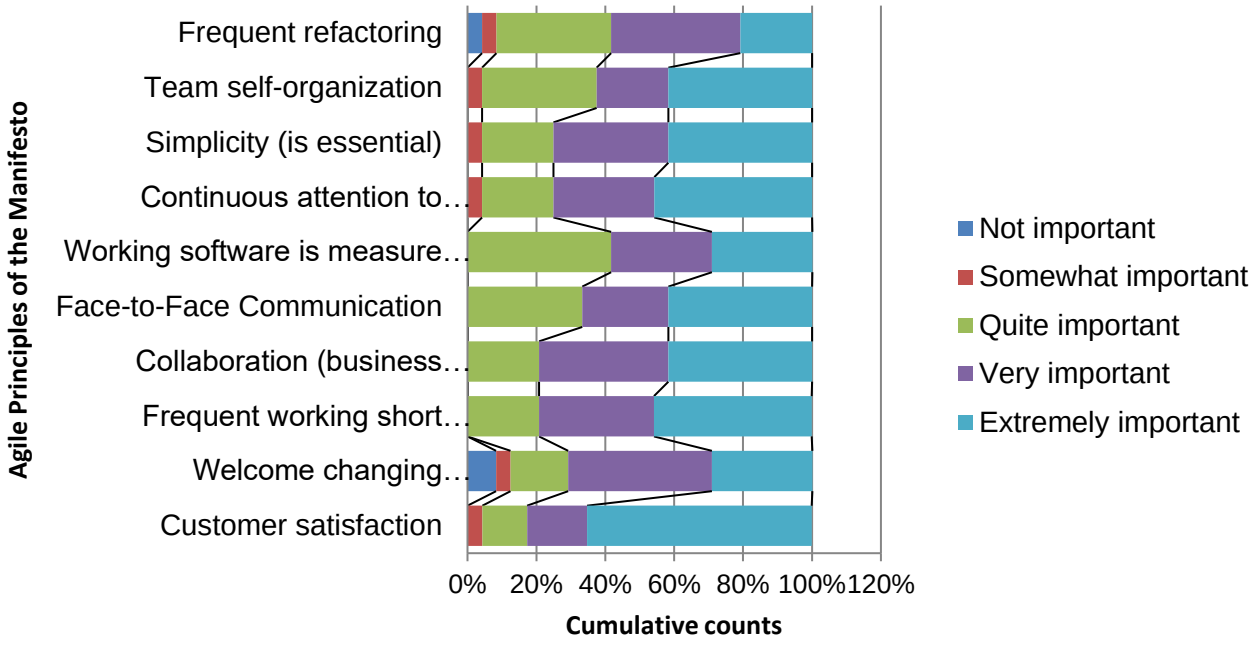


Figure 9: Proportion of valued principles of the Agile Manifesto

Welcoming change is the general idea behind the twelve principles espoused by the Agile Manifesto. These principles shape the philosophy of all of the agile methods to quickly deliver, or welcome dynamic requirements at any stage of the software production. Receptivity towards dynamic requirements defines a certain level of collaboration. For the second objective, to get the perception of software practitioners to welcome dynamic requirements, five items are listed in a 4-point scale question asking whether or not requirements are accepted or rejected at different stages of software development. To answer this question, a cross-table below (Table 7) shows cumulative percentages. Furthermore, another stacked bar chart (Figure 10) shows the same level of acceptance in the requirements.

Table 7: Level of acceptance of dynamic requirements

	Ignore	Somehow consider	Consider	Definitely consider
¼ of completion	0.0%	16.0%	28.0%	48.0%
½ of completion	0.0%	36.0%	24.0%	32.0%
¾ of completion	16.0%	20.0%	28.0%	28.0%
Final stage	24.0%	16.0%	28.0%	24.0%
Post-maintenance delivery	16.0%	8.0%	32.0%	36.0%

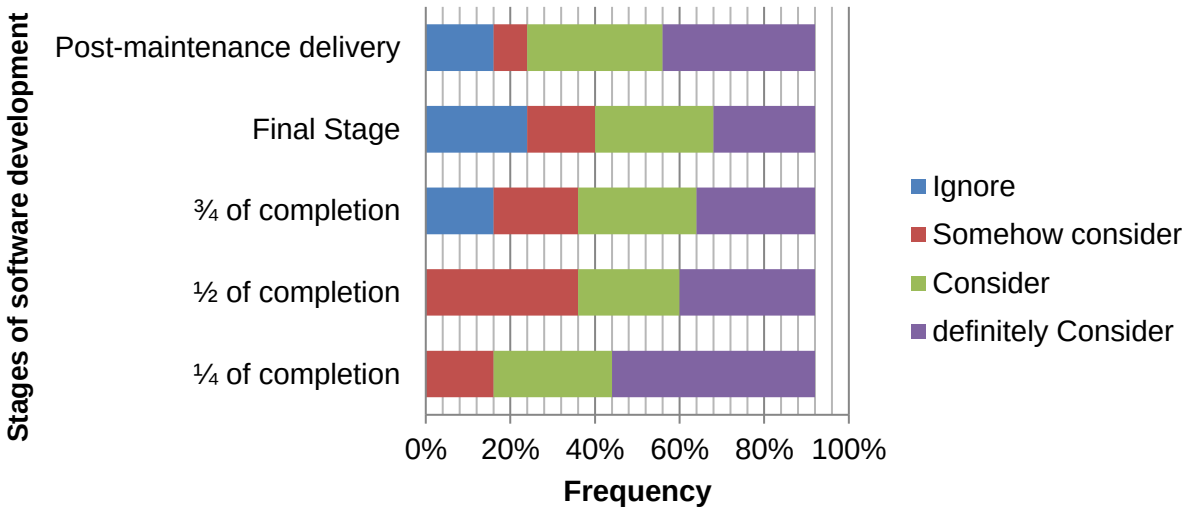


Figure 10: Level of acceptance of dynamic requirements

In addition, the following graph (Figure 11) shows the overall amount of requirements accepted and/or implemented by developers.

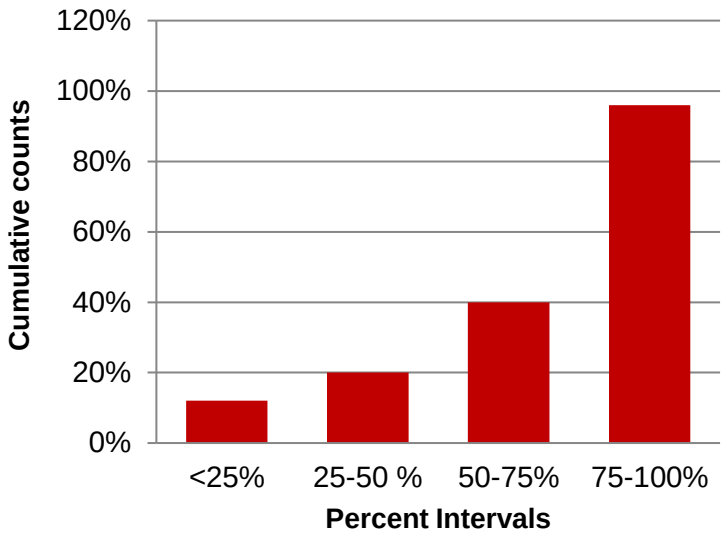


Figure 11: Overall amount of requirements implemented

One group of people in the centre of ASD are stakeholders or customers. The Agile Manifesto mentions customer involvement in every aspect of software development through customer collaboration (fourth question). Figure 9 shows that almost the entire stacked bar is composed with 3 of the 5 items of the Likert-scale. These results (Figure 9) show that agile principles are quite important for some, but considered very important and extremely important for others. From the three colours that compose the entire stacked bar, one can deduce that agile principles are really valued in the agile software industry. As six of these principles mostly define collaboration and interaction between customers and software practitioners on the graph (Figure 9), consequently two parties are really collaborative. With regard to the second objective related to this question - the degree of acceptance of dynamic requirements - it is not evenly spread throughout software project timeline. Not every requirement accepted (Table 7) is necessary implemented (Figure 11). But in order to reinvent customer relationships, the global tendency is the adoption of agile principles (Highsmith 2013:3) such as face-to-face communication, welcome changes, or even customer collaboration. Figure 10 shows that at any stage of software development, from the requirements brought forth by customers, a big proportion is considered. It also shows that half way through software completion, all requirements are considered. Software practitioners start rejecting requirements towards the end of software completion. During the final stage, for instance, 24% of requirements are ignored. Sixteen percent are also rejected during post-delivery maintenance. However, Figure 11 shows that, overall, 96% of

requirements from the customers are implemented. Thus, not much is rejected after all. This is again an indication of good collaboration between customers and software practitioners.

4.4.5 How do agile requirements engineering issues impact project outcomes?

This question is inclusive of some aspects of the questions already answered above. The impacts that issues of the ARE have on project outcomes depend exclusively on the common issues determined above. Common issues of the ARE process included issues of the phases of ARE, dynamic requirements and constraints from NFR. For obvious reasons, common issues negatively impact project outcomes. Table 8 below shows the top thirteen issues of the ARE that impede most software projects. Table 6 created as answer to the previous question, already determines the degree of receptivity towards changing requirements.

Table 8: Top issues experienced in agile requirements engineering

Rank	Issues	Percent
1	Lack of proper validation tools & techniques	73.9%
2	Scope problems	68.0%
3	Minimal documentation	64.0%
4	Lack of proper documentation	64.0%
5	Issues of prioritisation	60.9%
6	Unavailability of customer representative	60.0%
7	Requirements changeability	58.3%
8	Issues of prioritisation	56.0%
9	Lack of clarity	52.0%
10	Indecisive project stakeholders	50.0%
11	Lack of requirements traceability	50.0%
12	Lack of proper management tools	50.0%
13	Vague requirements	50.0%

In ASD, the contribution from the customers or stakeholders is critical (Buresh 2008) for the success of the software projects. Thus, the experience of the customers is fundamental in ensuring successful software deliveries. This is shown in Figure 12 below.

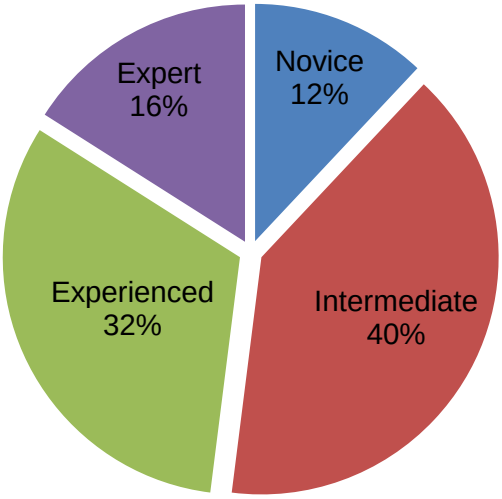


Figure 12: Customer experience in agile software development

In addition, Figure 13 shows the quality of requirements that customers come up with is essential in the software being developed.

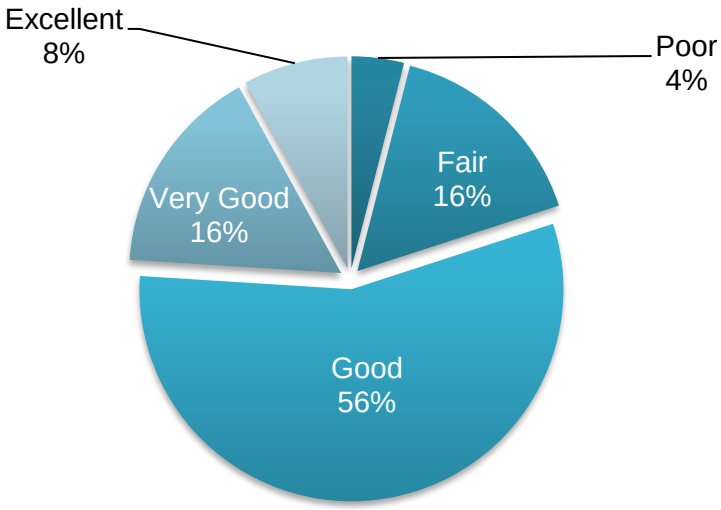


Figure 13: Quality of requirements brought forth by customers

Equally important is the customer awareness to software development. It is certain that a degree of knowledge on the topic impacts project outcomes. As Olsson, Bosch, and Alahyari (2013), customer involvement results in significantly faster deliveries of software increments. The experience and knowledge of the customer are vital in this context. This is revealed in Figure 14.

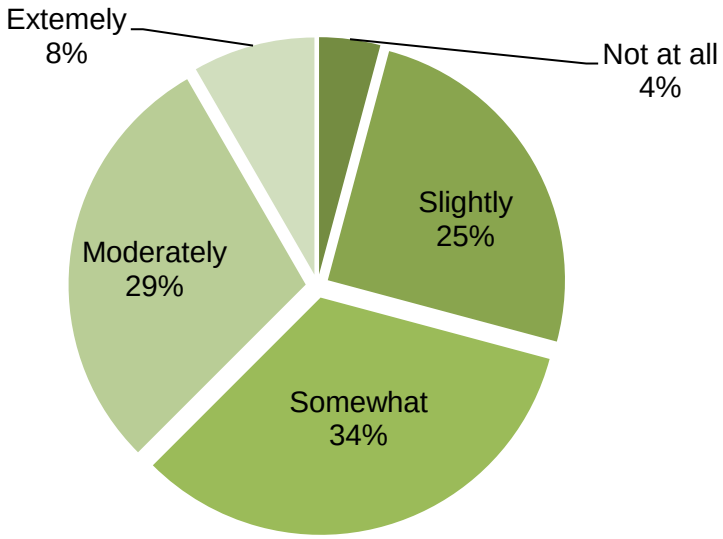


Figure 14: Awareness of customers towards the agile principles

Consideration towards NFR is also important in project outcomes. The proportion of every non-functional requirement is represented in Figure 15.

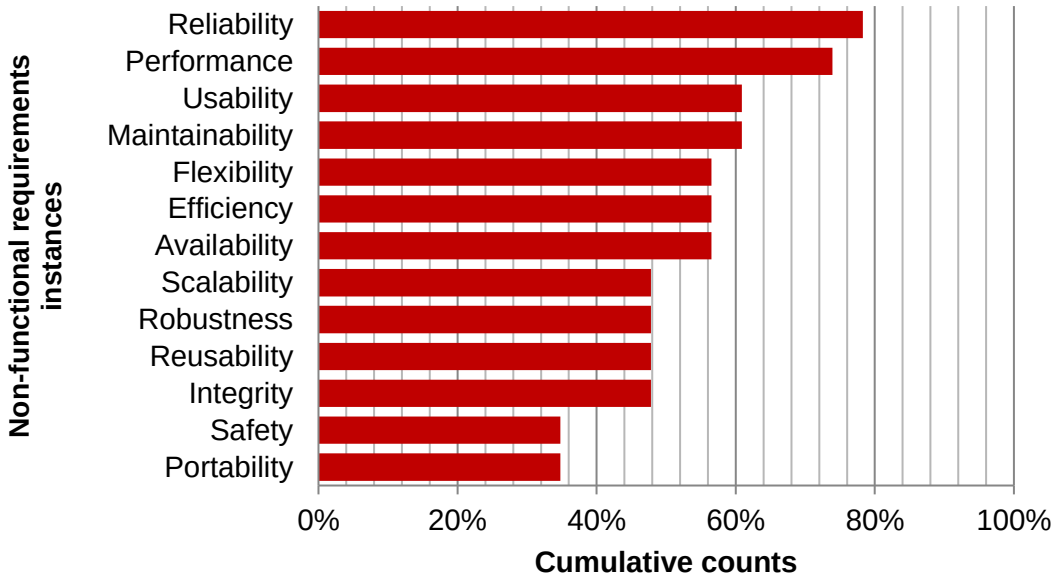


Figure 15: Non-functional requirements

Question five reads: how do issues of the ARE process impact projects outcomes? The Top 13 (Table 8) issues of the ARE show that during iterations, every classical phase is impacted. For instance, a) issues of prioritisation negatively impact both

elicitation and analysis phases, b) the documentation phase suffers from minimal documentation and lack of proper documentation; c) the validation phase is crippled by serious lack of proper validation tools and techniques; and d) the management phase suffers from requirements changeability. Basically, when the processes of the methodology are impacted, the whole methodology is impacted. Thus, the whole project is compromised. And that is the snowball effect.

Another issue that impact projects is the choice of proper methodology. Figure 16 shows Scrum as the most popular agile method used. Recent studies have suggested that Scrum has gained popularity and remains the most popular methodology (or framework) in the industry (Weinreich Neumann, Riedel & Müller 2015:566). The figures in this study certainly corroborate this fact.

Dynamic requirements have always been endemic (Jones 2009:438) to RE. Figure 10 shows that requirements are rejected towards the end of project completion. Reasons for rejecting requirements are diverse: poor quality, fear of missing target date, scope problems, etc. The dilemma here is that accepting requirements would mean facing all the reasons for requirements rejection, and rejecting requirements is against agile principles. This does not seem to cause problem in the software industry in South Africa as results show that only 4% of requirements from customers are rejected towards the end of software completion.

The consequence of ignoring non-functional requirements is a final product that is not in accordance with security standards. Figure 4 shows that 46% are neutral about the constraints put on software projects by NFR. This indicates that NFR integration is a serious problem and needs to be addressed.

4.4.6 How do issues in agile requirements engineering impact the outcome of projects in the software industry in South Africa considering today's constantly evolving marketplace?

This question is inclusive of all the other questions previously answered. The phrase 'today's constantly evolving marketplace' is the element that distances this question from sub-question 5. There is no need to repeat what has been said about the

negative impacts of the different issues on project outcomes. The world is driven by constant change which impacts today's requirements. Software practitioners welcome change half way through software completion by accepting all requirements from customers (Figure 10).

On a more general note, the South African software industry makes use of agile principles extensively (Figure 9). Although the participants converge in their views regarding their acceptance of change in requirements (Figure 10), they diverge in their consideration towards NFR as a priority (Figure 5). Web-based applications (Figure 17) are more and more requested these days certainly because of the cloud computing trends. Results (Figure 7) show diverse tools and techniques to choose from to help develop more efficiently under the agile methodology.

Despite all these challenges, the software industry is up to date in terms of the adoption of agile principles (Figure 9), or the consideration towards dynamic requirements. Issues of inadequate IT solutions (Table 3) are minor in the software industry. This also indicates that IT solutions in South Africa are moving at a reasonable pace. Although one can infer from all of the above that the software industry is not considerably impacted by the technological trends, the methods for software development need improvement because of the issues enumerated here.

Before ending this section on results presentation and interpretation, it is judicious to mention the difference that this study made to the body of knowledge. The main difference is the level of detail with regard to the issues in the ARE process in spite of lack of empirical evidence in the adoption of agile principles in South Africa. The issues enumerated in this study are known to software developers. To tackle general issues in areas where the sources of these issues are known is of great advantage. The set of recommended tools and techniques is an advantage in dealing with these issues.

The results also show that agile principles of the manifesto, more specially the ones related to the ARE process, are used in earnest. All pieces put together are instrumental to the awareness of software practitioners be there awareness in terms of:

- common issues of ARE;

- good practices such as values of agile principles; or
- the indifference of some participants towards NFR.

In addition, the results show that 68% have on-site customers. One positive impact on software projects is the availability of a customer representative. The presence of customers at site may give clear direction to the team members during change dependent on the customers understanding of the project (Inayat et al. 2014:9). Empirical studies show that the unavailability of a customers' representative constitutes the overall challenge in software development (Ramesh et al 2010:467) which always results in project failure (Labuschagne et al 2008:17; Sonnekus & Labuschagne 2003:9). This also raises awareness in terms of customer collaboration and interaction.

Finally, this study corroborated a popular fact: the popularity of Scrum as agile technique. South Africa is moving with the trends of technological innovations at a reasonable pace, that is, with cloud-based computing, as pointed out earlier. There is a moderate consideration towards NFR that needs to be noted. Since the study is entirely about requirements, only 4% of the suggested requirements are poor quality. These are all facts to know in ASD.

4.4.7 Other general data presented

Sometimes, common issues are consequences of choices of agile methods used. The graph (Figure 16) shows what these agile methods are and the proportion of the adoption in agile software projects in the South African software industry.

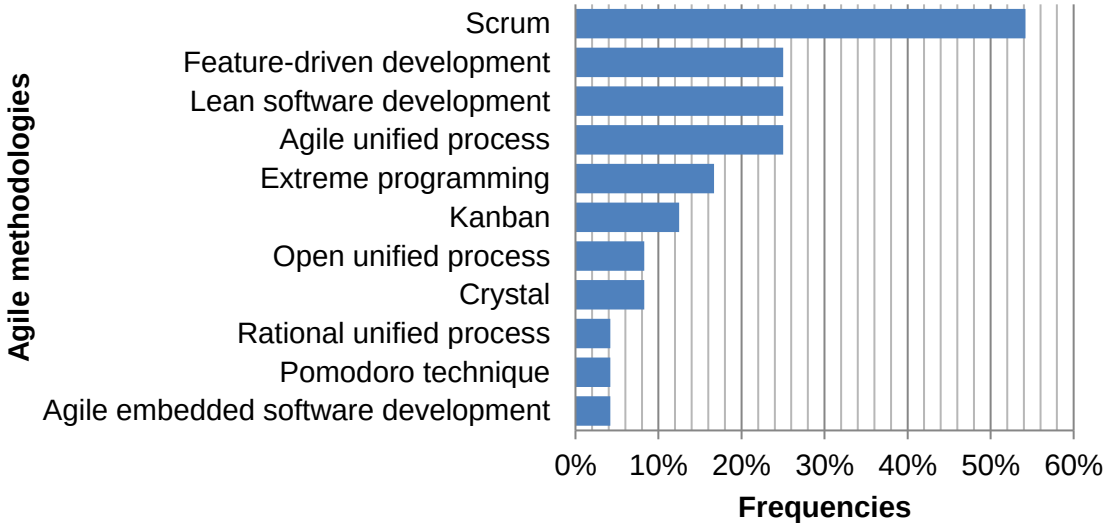


Figure 16: Agile methods used in the software industry in South Africa

More generally, stakeholders request software that falls under the following categories: a) web-based applications; b) business applications; c) systems or real-time applications. Figure 17 presents the proportion of these requests.

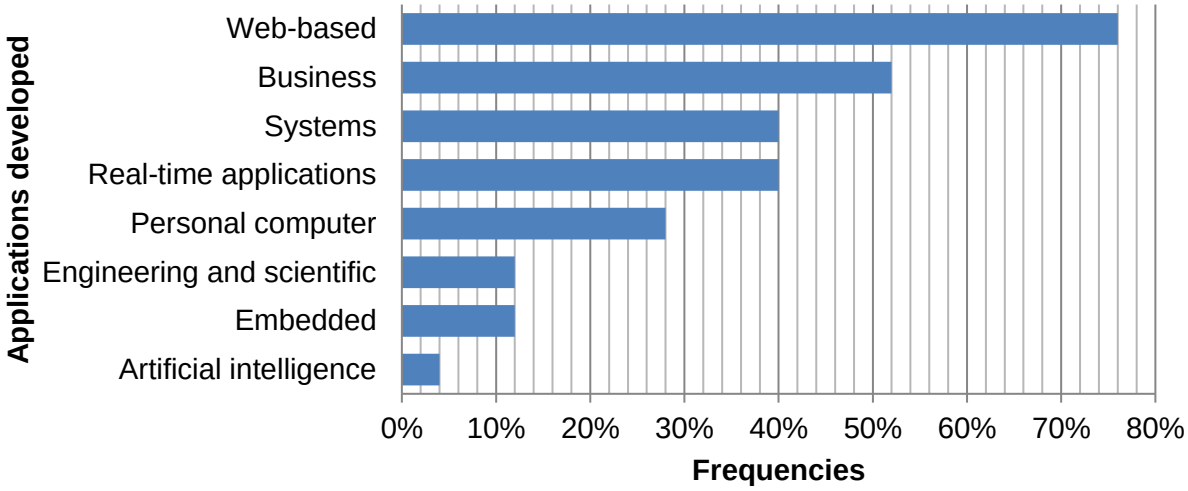


Figure 17: Genre of applications requested by stakeholders

The ARE process encompasses a set of activities that are performed in a non-linear order. That is because of the adaptability of agile methodologies (Lucia & Qusef 2010:213). Major activities are defined on the following graph (Figure 18).

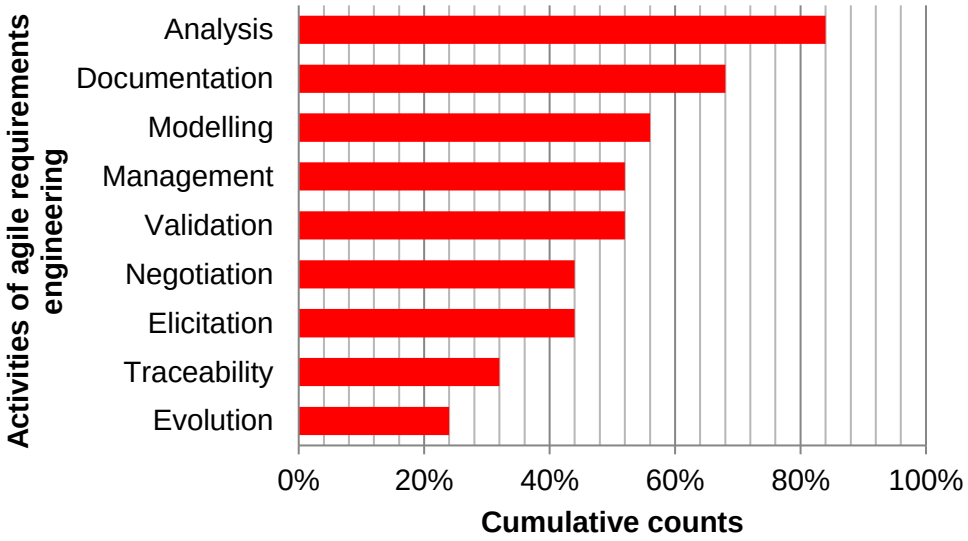


Figure 18: Activities used during agile requirements engineering

The availability of a customer representative is determined in Figure 19.

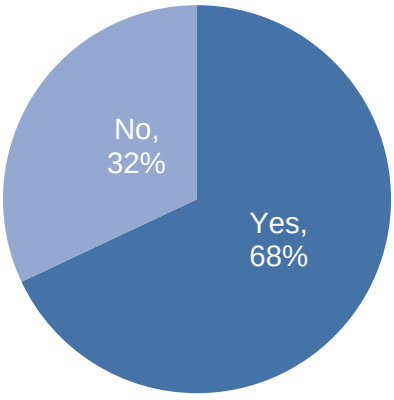


Figure 19: Proportion of on-site customer representative

4.5 Possible future research

People are moving gradually towards web-based solutions because of the ubiquity of IT solutions (Veersamany & Labuschagne 2014). Smartphones nowadays have all the functions of personal computers or laptops. Applications are cross-platform and work well on mobile devices without any conversion of any type. Software requests from customers are more orientated towards web-based applications. Could this be triggered by the fact that customers are embracing more cloud-based solutions? Or considering the pace at which the technology advances, are these solutions proposed by developers to remain competitive on the market?

Another observation in this study regarding NFR is the even and low consideration towards safety (34.8%) and portability (34.8%). It would be excessive and unsubstantiated to conclude that applications in South Africa, and more especially web-based applications, suffer serious security issues or vulnerabilities, and that these applications lack cross-platform functionalities. This topic also needs more academic scrutiny.

Agile principles such as customer satisfaction, collaboration, or face-to-face communication have been considered (very and extremely) important by developers (see Table 9 below).

Table 9: Customer satisfaction, collaboration and communication

	Not important	Somewhat important	Quite important	Very important	Extremely important
Customer satisfaction	0.0%	4.3%	13.0%	17.4%	65.2%
Collaboration (business people & developers)	0.0%	0.0%	20.8%	37.5%	41.7%
Face-to-face communication	0.0%	0.0%	33.3%	25.0%	41.7%

Getting the views of the stakeholders on these principles is altogether another topic of research and requires more intensive studies. As suggested by Inayat et al (2014:12), more empirical results are required to get the real impact of these ARE practices. While all of the above are avenues for future research, the issues of the ARE on their own are serious difficulties in ASD. These issues are pervasive throughout software development. In addition, the lack of prior research in the domain (Noruwana & Tanner 2012:56) makes it even harder.

4.6 Summary

A considerable number of questionnaires were distributed. The return rate (RR) was moderate but usable responses were acquired in the process. An electronic version (Word document) was used in conjunction with an online version (Google forms) to collect data from companies in the ASD field. Whittaker (2013:146) suggests that technology extends our ability to change the world. The findings from this study are intended to extend companies knowledge of the recurrent problems in the software industry, more especially in an agile setting. The aim is to outline the most common problems of ARE in the South African industry, and determine tools and frameworks used to tackle some of the issues. This also raises awareness in terms of problems, their sources and solutions.

Finally this study lays out avenues for future research. It is known that applications are more and more cloud-based; thus more research should focus on NFR such as usability, scalability, safety, or portability to ensure that applications that are developed adhere to basic security requirements. Thus, companies in the software industry that are agile ought to improve on these areas that are fundamental to project success. Considering the lack of documentation that properly lists the common problems (Noruwana & Tanner 2012:44), and taking into consideration the findings emanating from this research, the outcomes can be considered a repository of common problems experienced in ARE in South Africa.

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This study aimed to determine common problems experienced by software practitioners in the ARE process, determine a set of recommended tools and techniques adopted to deal with these issues, get a sense of how they value agile principles, comprehend the level of collaboration and interaction between customers and software practitioners.

Agile methods are certainly popular methods but there are no clear trends of their adoption in practice (Cao & Ramesh 2009:61, Shen & Zhang 2011:1; Murphy et al 2013). Requirements engineering are the foundation to software construction (Pressman 2009:120) and the most critical characteristic of software development (Sillitti & Succi 2005:309). Analogously, ARE processes, as a subset to ASD, are the foundation to agile methods. This area of ASD remains under-researched or lacks empirical studies and is filled with challenges. It was in this perspective that questions regarding ARE processes in the context of ASD in South Africa arose. A literature review was conducted to put this research in an academic perspective. This study was designed as essentially quantitative, conducted within the positivism paradigm. It was based on a survey questionnaire and SPSS was used as the tool for statistical analysis.

This chapter is linearly structured as follows: recapitulation, theoretical implications; recommendations for future research, limitations of this research and a conclusion.

5.2 Recapitulation

The instrument for data collection was designed, piloted and deemed valid to collect data. A total 107 questionnaires were distributed and only 25 responses were collected. The data subjected to data analysis indicated a reliability coefficient (Cronbach alpha) of 0.906 and the statistical procedures were (in general) limited to cumulative frequencies, counts, and cross-tabulation. The visual aids used were

limited to tables, plotting of histograms and pie charts. The previous chapter extensively presented the results from the data collected and discussed the findings that emerged from this study. A recapitulation of these findings follows.

The first question intended to determine common issues in agile requirements engineering in the software industry. Major issues in this study include: lack of proper validation tools and techniques (73.9%); scope problems (68%); lack of proper documentation (64%) issues of prioritisation (60.9%); unavailability of customer representative (60%); or requirements changeability (58.3%), etc. These are common in ARE and consistent with reasons for project failure such as handling change, customer involvement, and requirements definition defined by Sonnekus and Labuschagne (2003:10) and Labuschagne et al (2009:18). The validation process is another major challenge to ARE due to the lack of proper validation tools and techniques. Minimal documentation is another serious impediment to the ARE process. Lastly, there is a noticeable indifference of software practitioners towards NFR integration as 46% are neutral about the constraints that this has on software projects.

The second question sought to find tools and techniques to deal with issues elaborated above. The amount of tools and techniques in the software industry used for the ARE process is overwhelming. Among the panoply of agile tools and techniques listed by the existing literature, the top five tools or frameworks used by developers to speed up software development or mitigate risks emanating from issues during development are, in order of preference: requirements prioritisation; JAD sessions; ScrumWorks, Documentation Management System, and requirements splitting.

The third question attempted to find out how software practitioners value agile principles, more especially the ones related ARE. The results show, for instance, that: customer satisfaction (65.2%); face-to-face communication (41.7%); continuous attention to technical excellence (45.8%); and collaboration (41.7%) are among the principles that are valued the most by agile practitioners. Thus, one major finding in this sub-section is that software practitioners have a high opinion of agile principles in general.

The fourth question intended to determine the degree of collaboration that one can find between customers and software practitioners. The results reveal that 68% of companies have customers at site. As already noted in the previous paragraph, software practitioners have a strong penchant for agile principles. This is significant in promoting customer involvement through agile principles such as customer satisfaction, collaboration, and face-to-face communication. It is also an indication of collaborative effort from both sides. Basically, in terms of ARE, customers and software practitioners are very collaborative. This is essentially one of the reasons influencing project success in South Africa (Sonnekus & Labuschagne 2003:9, Labuschagne et al 2009:18).

Determining ARE issues that impact project outcomes is essentially the content of the fifth question. Empirical studies in South Africa suggest that strategy to handle change, user involvement, lack of communication, or executive support are parameters that influence the most project outcomes (Marnewick & Labuschagne 2009:84). Issues of the ARE process are abundant, and range from problems regarding customers to the ones related to software practitioners. For instance, one major finding from the results is about scope. Scope challenges are a source of budget problems, requirements changeability can change the project timeline. With regard to customer involvement, customer experience is critical in eliciting requirements. Cost, time, and quality therefore are attributes to successful projects.

The marketplace is driven by technology which, in turn, drives the request for software development. The results revealed that software practitioners are somewhat receptive vis-à-vis dynamic requirements at any stage during software development. This study also showed that agile principles that relate to the customer in particular gained high attention. The consequence of this high penchant to the agile principle is customer satisfaction. Constant changes in marketplace (Cao & Ramesh 2007:42) do put constraints on software projects. Software practitioners attributed 58.3% to requirements changeability issues. Results show that more than 96% of the requirements brought forth by customers are implemented. Basically, from these figures, issues of the ARE do put constraints on software projects but not to the extent to cripple the entire process.

5.3 Theoretical implications

It was established in the previous chapter that all these findings are consistent with previous empirical studies. For instance, a more recent study shows that Scrum remains the most popular of the agile methods (Machado, Pinheiro & Tamanini 2015:1) which is also confirmed in this study. Another example relates to the tools and techniques. Software practitioners tend to use popular tools and techniques. Over 60% lack customer representatives. Some agile practices such as requirements prioritisation need revision. Results show that 83.8% preferred validation through face-to-face communication. However, Friedrich and Van Der Poll (2007:189) identified a serious communication gap between IT Specialists and stakeholders. And since 73.9% lack proper validation tools and techniques, this contradiction needs further scrutiny.

Implications related to the findings are diverse. Firstly, this study gives a better understanding of the ARE process in terms of the adoption agile principles and best practices. Secondly, it raises awareness in terms of common issues experienced in ARE. Thirdly, in practice, this study shows that software practitioners value principles of the Agile Manifesto. Fourthly, it was also established that developers build more web-based applications than any other type of software. This either implies a “new ubiquitous computing and new communication era” (ITU 2005:2) or customers just trying to “capitalize on new technology” (Johnson, Becker, Estrada, & Freeman 2014:26). Finally, it is argued that a proper definition of requirements plays a considerable role in project success (Marnewick & Labuschagne 2009:18). This implies that tackling issues of scope, prioritisation, minimal documentation, or increasing the use of proper tools and techniques inevitably reduce risks in software project outcomes.

5.4 Recommendations

Agile software development has always been a passionate topic. At the same time topics related to RE in the agile environment are also an obsessive and vast area of interest. However, one can be easily overwhelmed by the depth of agile methods, and more especially the ARE process. Issues in ASD are also abundant. Marnewick

and Labuschagne (2009:17) group all the main reasons for project failure under two categories: people and processes (this includes tools and techniques).

This study shows fair consideration to people through agile principles such as face-to-face collaboration, customer satisfaction, and communication. Focus should be put more on the processes for the mere reason that it is an area filled complexities. Major issues pointed out in this study include lack of adequate tools and techniques for validation, scarcity of studies in the ARE process, and lack of NFR integration.

Companies should invest more on tools for requirements validation, documentation and management as these areas are not properly handled during ASD. This study showed that companies lack adequate requirements validation tools and techniques. Tools for requirements validation such as Cognition Cockpit, QPack, or Reqtify can be of great help. The Method for Elicitation, Documentation and Validation of Software User Requirements (MEDoV) is adequate not only for validation but also guide stakeholders during elicitation (Dragičević, Čelar & Novak 2014:66). MedoV can also be the solution to the requirements documentation problems experienced by software practitioners. As for requirements managements, tools such Jira Agile, HP Agile Manager, or Microsoft Project are deemed user-friendly and thus adequate for requirements traceability and volatility. Taking into consideration the budget, the OPEN Process Framework Repository Organization has a repository of open-sources and free applications for requirements.

As change is due quicker than ever before (Cobb 2011:63), it is apparent that applications are becoming increasingly cloud-based. This study shows that 76% of the applications requested are cloud-based. As already mentioned, to ensure confidentiality, integrity and the availability of the data of the companies, NFR integration should be seriously taken into consideration as 46% of the companies remain neutral about this topic. Tools for NFR integration include FURPS (Functionality Usability Reliability Performance Supportability) developed by Hewlett-Packard and NORMAP (Non-functional Requirements Modeling for Agile Processes). The use of these tools can improve the quality of cloud-based applications in terms of usability, scalability, safety, and portability to ensure that these applications adhere to basic security requirements. Another recommendation

is that NFR should be integrated during software development, not during post-delivery maintenance. This is vital for software project success.

Exploring these options as future research strategies can help software practitioners adopt a new mindset and be more competitive in a modern and constantly changing world driven by technology. Further, empirical research into agile methods is sparse (Ferreira & Cohen 2008:48), and little is said about agile methods (Noruwana & Tanner 2012:41) in South Africa, or about how it is implemented in practice (Cao & Ramesh 2008:61, Shen & Zhang 2011:1) in general. Thus, additional studies in the ARE field are expected to raise awareness in particular areas such as requirements documentation, validation, management, and NFR integration.

If the agile approach is considered to be more an attitude than process, or more environment than methodology (Highsmith 2003:3), then, there is a need to fully embrace an agile attitude in order to create an agile environment that suits everyone (software practitioners and stakeholders). This goal is achievable through further research. This will help to identify “best practices and generate insights valuable to managing software development projects in the future” (Ferreira et al 2009).

5.5 Limitations of this research

Limitations, also known as potential weaknesses (Simon 2011) or restrictions in the study (Rudestam & Newton 2014:105) were encountered in three specific areas which are the review of the literature, the data collection and the data analysis.

Firstly, with regard to the limitations in the review of the literature, empirical research into agile methods in South Africa is sparse (Ferreira & Cohen 2008:48) and prior research in ARE is limited (Noruwana & Tanner 2012:42). Some articles that relate to ARE in the context of South Africa are over a decade old. This impediment also goes against the contemporary nature of this research.

Secondly, the survey questionnaire used as the data collection strategy has limitations in that: a) the survey questionnaire lacks flexibility in the responses but are moderately easy to organise (Walliman 2010:99); and b) sometimes the honesty

of the response of the participants is questionable. Other limitations encountered in this study included: time constraints during data collection; a non-exhaustive list of software development companies (which needs a regular update); unavailability of participants due to strict policies regarding their participation in research surveys of any kind which create a feeling of rejection.

Thirdly, keeping in mind the Cronbach alpha coefficient (α=0.906 mentioned above) which demonstrates the reliability of this study, one would easily assert that this study is reliable and therefore valid for generalisation. However, statistical inference requires much more data and the sampling size and the RR (of 23%) reflect otherwise.

As a final point, the scarcity of the literature on agile processes, the impediments encountered during data collection and lack of generalizability of this study are the limitations encountered during this research. Despite the best efforts to construct a well-designed study by taking into consideration all possible confounding variables, it is almost impossible to cover all bases (Murray & Hughes 2008:168).

5.6 Conclusion

In spite of the challenges encountered in the theoretical review due to lack of resources, or in the data collection and analysis, this study has been very productive. The major findings that emerge from this study are: 1) a baseline for issues of ARE is defined (Table 3); 2) a repository of recommended tools and frameworks (Figure 7) is determined; 3) more importantly, in theory, software practitioners do value principles such as customer satisfaction, collaboration, face-to-face communication, and continuous attention to technical excellence. These are key principles in eliciting requirements; 4) although neglected, software practitioners in South Africa do consider NFR, but it has been suggested that more emphasis should be put on the topic as the integration of NFR to ASD can improve software production in terms of software quality.

References

Abdullah, NNB, Honiden, S, Sharp, H, Nuseibeh, B & Notkin, D. 2011. Communication patterns of agile requirements engineering. Paper presented at the 1st workshop on agile requirements engineering ACM. UK, Lancaster, July1.

Ambler, SW. 2001. Agile requirements modeling. *The Official Agile Modeling (AM) Site*. Available at http://agilemodeling.com/essays/agileRequirements.htm (accessed on 15/01/2016).

Augustine, S. 2005. *Managing agile projects*. Virginia, Annadale: Prentice Hall PTR.

Batool, A, Motla, YH, Hamid, B, Asghar, S, Riaz, M, Mukhtar, M & Ahmed, M. 2013. Comparative study of traditional requirement engineering and agile requirement engineering. Paper presented at the IEEE's 15th International Conference on Advanced Communication Technology (ICACT). Pakistan,Rawalpindi:1006.

Benfield, J. A., & Szlemko, W. J. (2006). Internet-based data collection: Promises and realities. *Journal of Research Practice, 2*(2), Article D1. Retrieved [date of access] from, http://jrp.icaap.org/index.php/jrp/article/view/30/51

Berenbach, B, Paulish, D, Kazmeier, J & Rudorfer, A. 2009. *Software & systems requirements engineering: in practice.* McGraw-Hill, Inc.

Berndtsson, M, Hansson, J & Olsson, B. 2007. *Thesis projects: a guide for students in computer science and information systems*. London: Springer Science & Business Media.

Boness, K, Harrison, R & Liu, K. 2009. Goal sketching: an agile approach to clarifying requirements. *International Journal on Advances in Software* 1(1):1-13.

Bopp, MA, Bing, DA & Forte-Trammell, S. 2009. *Agile career development: lessons and approaches from IBM*. Indiana, Crawfordsville: Pearson Education.

Borrego, M, Douglas, EP & Amelink, CT. 2009. Quantitative, qualitative, and mixed research methods in engineering education. *Journal of Engineering Education* 98(1):53-66.

Brooks Jr, FP. 1987. No silver bullet-essence and accidents of software engineering. *IEEE Computer* 20(4):10-19.

Brooks Jr, FP. 1995. *The mythical man-month. Essays on software engineering*. Anniversary edition. USA: Pearson Education.

Brown, VR & Paulus, PB. 2002. Making group brainstorming more effective: recommendations from an associative memory perspective. *Current Directions in Psychological Science* 11(6): 208-212.

Buresh, D. 2008. Customer Satisfaction and Agile Methods. *IEEE Reliability Society*.

Cao, L & Ramesh, B. 2007. Agile software development: ad hoc practices or sound principles? *IT professional* 9(2):41-47.

Cao, L & Ramesh, B. 2008. Agile requirements engineering practices: an empirical study. *Software, IEEE* 25(1):60-67.

Carrillo de Gea, Juan M, Nicolás, J, Alemán, JLF, Toval, A, Ebert, C & Vizcaíno, A. 2011. Requirements engineering tools. *Software, IEEE* 28(4):86-91.

Check, J & Schutt, RK. 2011. *Research methods in education.* USA, California: Sage Publications.

Chrissis, MB, Konrad, M & Shrum, S. 2011. *CMMI for development: guidelines for process integration and product improvement.* USA, Boston: Pearson Education.

Cobb, CG. 2011. *Making sense of agile project management: balancing control and agility*. USA, New Jersey, Hokoben: John Wiley & Sons.

Coram, M & Bohner, S. 2005. The impact of agile methods on software project management. Paper presented at the 12th IEEE International Conference and Workshops on Engineering of Computer-Based Systems. 2005 (ECBS'05), April:363-370.

Creswell, JW. 2009. *Research design: Qualitative, quantitative, and mixed methods approaches.* USA, California: Sage.

Chrissis, MB, Konrad, M & Shrum, S. 2011. *CMMI for development: guidelines for process integration and product improvement, 3rd edition.* USA, Massachusetts: Pearson Education.

Cronbach, LJ. 1951. Coefficient alpha and the internal structure of tests. *Psychometrika* 16(3):297-334.

Daneva, M, Van Der Veen, E, Amrit, C, Ghaisas, S, Sikkel, K, Kumar, R, Ajmeri, N, Ramteerthkar, U & Wieringa, R. 2013. Agile requirements prioritization in large-scale outsourced system projects: An empirical study. *Journal of Systems and Software* 86(5):1333-1353.

De Wet, B & Visser, J. 2013. An evaluation of software project risk management in South Africa. *South African Journal of Industrial Engineering* 24(1):14-29.

Dragičević, S, Čelar, S & Novak, L. 2014. Use of method for elicitation, documentation and validation of software user requirements (MEDoV) in agile software development projects. Paper presented at the Sixth International Conference on Computational Intelligence, Communication Systems and Networks (CICSyn2014). Macedonia, Tetova.

Drew, CJ, Hardman, ML & Hosp, JL. 2007. *Designing and conducting research in education.* USA, California: Sage Publications.

Dugosh, KL & Paulus, PB. 2005. Cognitive and social comparison processes in brainstorming. *Journal of Experimental Social psychology* 41(3):313-320.

Eberlein, A & Leite, J. 2002. Agile requirements definition: a view from requirements engineering. Paper presented at the International Workshop on Time-Constrained Requirements Engineering (TCRE'02). Canada, Calgary.

Elshandidy, H & Mazen, S. 2013. *Agile and traditional requirements engineering: a survey. International Journal of Scientific & Engineering Research 4(9)*

Enders, CK & Gottschall, AC. 2011. *The impact of missing data on the ethical quality of a research study.* USA, New York: Routledge.

Ernst, NA, Borgida, A, Mylopoulos, J & Jureta, IJ. 2012, Agile requirements evolution via paraconsistent reasoning. *Advanced Information Systems Engineering.* Berlin:Springer:382-397.

Farid, WM. 2012. The NORMAP methodology: lightweight engineering of non-functional requirements for agile processes. Paper presented at the 19th IEEE Asia-Pacific Software Engineering Conference (APSEC), Hong Kong: December: 322-325.

Ferreira, C & Cohen, J. 2008. Agile systems development and stakeholder satisfaction: a South African empirical study. Paper presented at the 2008 annual research conference of the South African Institute of Computer Scientists and Information Technologists on IT research in developing. Practices to help solve them. *Journal of Object Technology* 6(1):17-33.

Ferreira, S, Collofello, J, Shunk, D & Mackulak, G. 2009. Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation. *Journal of Systems and Software* 82(10):1568-1577.

Frary, RB. 2003. A brief guide to questionnaire development. *Virginia Polytechnic Institute & State University* 9*,* October:168-180.

Friedrich, WR & Van Der Poll, JA. 2007. Towards a methodology to elicit tacit domain knowledge from users. *Interdisciplinary Journal of Information, Knowledge, and Management* 2(1):179-193.

Gillham, B. 2008. *Developing a questionnaire.* UK: A & C Black.

Gillwald, A, Moyo, M & Stork, C. 2012. *Understanding What is Happening in ICT in South Africa.* South Africa, Cape Town*.*

Grand, S. 2016. *Routines, strategies and management: engaging for recurrent creation 'at the edge'.* Edward Elgar Publishing.

Greasley, P. 2007. *Quantitative data analysis using SPSS: an introduction for health & social science.* UK: McGraw-Hill Education.

Helmy, W, Kamel, A & Hegazy, O. 2012. Requirements engineering methodology in agile environments. *International Journal of Computer Science Issues* 9(5). Egypt, Giza: 293-300.

Henderson, LG. 2000. Requirements elicitation in open-source programs. *CrossTalk-The Journal of Defense Software Engineering* 13(7):28-30.

Highsmith, J. 2003. Agile project management: principles and tools. *Cutter Consortium* 4:1-37.

Highsmith, J. 2013. *Adaptive leadership: accelerating enterprise agility.* USA*,* Chicago*:* Addison-Wesley.

Hislop, J. 2011. *Response: towards a competitive South African software industry.* South Africa, Cape Town.

Hull, E, Jackson, K & Dick, J. 2005. *Requirements engineering.* London: Springer-Verlag.

Humphrey, WS. 2005. Acquiring quality software. *CROSSTALK The Journal of Defense Software Engineering,* December:19-23.

Inayat, I, Salim, SS, Marczak, S, Daneva, M & Shamshirband, S. 2014. A systematic literature review on agile requirements engineering practices and challenges. *Computers in Human Behavior 55*:915-929.

ITU Internet Reports 2005. The internet of things. *Geneva: International Telecommunication Union (ITU).*

Jackson, E. 2013. Choosing a methodology: philosophical underpinning. *Practitioner Research in Higher Education* 7(1):49-62.

Janes, AA & Succi, G. 2012. The dark side of agile software development. Paper presented at the ACM international symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, ACM. USA, New York, NY, October:215-228.

Jerry, RT, Jack, K & Stephen, JS. 2015. Research methods in physical activity. *Human Kinetics* 3:14-15.

Jones, C. 2009. *Software engineering best practices.* USA, New York: McGraw-Hill, Inc.

Johnson, L, Becker, S, Estrada, V & Freeman, A. 2014. Horizon Report: 2014 Higher Education. Austin, Texas: The New Media Consortium.

Kamalrudin, M, Grundy, J & Hosking, J. 2010. Tool support for essential use cases to better capture software requirements. Paper presented at the IEEE/ACM Iternational Conference on Automated Software Engineering ACM. USA, New York, NY, September:255-264.

Kelley, K, Clark, B, Brown, V & Sitzia, J. 2003. Good practice in the conduct and reporting of survey research. *International Journal for Quality in Health Care: Journal of the International Society for Quality in Health Care / ISQua* 15(3):261-266.

Krosnick, JA & Presser, S. 2010. Question and questionnaire design in *Handbook of survey research* 2nd edition. UK:Emerald Group Publishing Limited:263-314.

Ktata, O & Lévesque, G. 2009. Agile development: issues and avenues requiring a substantial enhancement of the business perspective in large projects. Paper presented at the 2nd Canadian Conference on Computer Science and Software Engineering ACM, May:59-66.

Laanti, M, Salo, O & Abrahamsson, P. 2011. Agile methods rapidly replacing traditional methods at Nokia: a survey of opinions on agile transformation. *Information and Software Technology* 53(3):276-290.

Labuschagne, L, Jakovljevic, M & Marnewick, C. 2008. *The Prosperus Report 2008*. South Africa: PMSA.

Labuschagne, L, Marnewick, C & Jakovljevic, M. 2008. *IT project management maturity: a South African perspective.* South Africa, Midrand: PMSA.

Lucia, AD & Qusef, A. 2010. Requirements engineering in agile software development. *Journal of Emerging Technologies in Web Intelligence* 2(3):212-220.

Machado, TCS, Pinheiro, PR & Tamanini, I. 2015. Project management aided by verbal decision analysis approaches: a case study for the selection of the best SCRUM practices. *International Transactions in Operational Research* 22(2):287-312.

Mangal, S & Mangal, S. 2013. *Research methodology in behavioural sciences.* PHI Learning Pvt. Ltd.

MarketLine 2012. Software in South Africa. Available at: www.marketline.com (accessed on 25/04/2015).

Marnewick, C & Labuschagne, L. 2009. Factors that influence the outcome of information technology projects in South Africa: an empirical investigation. *Acta Commercii* 9:78-89.

Mnkandla, E & Dwolatzky, B. 2004. A survey of agile methodologies. *The Transactions of the SA Institute of Electrical Engineers,* December:236-247.

Mnkandla, E. 2008. *A selection framework for agile methodology practices: a family of methodologies approach*. PhD Thesis, University of Witwatersrand, Johannesburg, South Africa.

Murphy, B, Bird, C, Zimmermann, T, Williams, L, Nagappan, N & Begel, A. 2013. Have agile techniques been the silver bullet for software development at Microsoft? Empirical software engineering and measurement. Paper presented at the ACM/IEEE International Symposium on IEEE, October:75-84.

Murray, N. & Hughes, G. 2008. *Writing up your university assignments and research projects: a practical handbook*. Maidenhead: Open University Press.

Noruwana, N & Tanner, M. 2012. Understanding the structured processes followed by organisations prior to engaging in agile processes: a South African perspective. *South African Computer Journal* 48:41-58.

Oates, BJ. 2006. *Researching information systems and computing.* London: Sage Publications Limited.

Olsson, HH, Bosch, J & Alahyari, H. 2013. Customer-specific teams for agile evolution of large-Scale embedded systems. *2013 39th Euromicro Conference on Software Engineering and Advanced Applications* IEEE: 82.

Paetsch, F, Eberlein, A & Maurer, F. 2003. Requirements engineering and agile software development. Paper presented at the IEEE's 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises IEEE Computer Society*,* June, 2012 :308-313.

Panayides, P. 2013. Coefficient alpha: interpret with caution, *Europe's Journal of Psychology* 9(4):687–696.

Pather, S & Remenyi, D. 2005. Some of the philosophical issues underpinning research in information systems-from positivism to critical realism: reviewed article. *South African Computer Journal (*35):76-83.

Pressman, R. 2009. *Software engineering: a practitioner's approach*. 7th edition. USA, New York, NY: McGraw-Hill.

Pressman, RS & Lowe, DB. 2009. *Web engineering: a practitioner's approach.* McGraw-Hill Higher Education.

Ramesh, B, Cao, L & Baskerville, R. 2010. Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal* 20(5):449-480.

Reiss, J. & Sprenger, J. 2014. Scientific objectivity. *Stanford Encyclopedia of Philosophy*.

Rico, D [sa]. What is agility? Available at: http://davidfrico.com/agile-definition.pdf (accessed on 05/04/2015).

Rojas, AB & Sliesarieva, GB. 2010. Automated detection of language issues affecting accuracy, ambiguity and verifiability in software requirements written in natural language. Paper presented at the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas Association for Computational Linguistics, June:100.

Rudestam, KE & Newton, RR. 2014. *Surviving your dissertation: a comprehensive guide to content and process.* USA, California: Sage Publications.

Scacchi, W. 2002. Understanding the requirements for developing open source software systems. Paper presented at the conference on Software, IEE -IET, February:24-39.

Schach, SR 2011. *Object-Oriented & Classical Software Engineering*. Global Edition. 8th edition. USA, New York, NY: McGraw-Hill.

Scrumhint. 2015. Snapshot of agile software development. Available at: http://www.scrumhint.com/snapshot-of-agile-software-development/ (accessed 20/01/2016).

Shelly, G & Rosenblatt, HJ. 2009. *Systems analysis and design.* USA, Boston: Cengage Learning.

Shen, HL & Zhang, Y. 2011. *An exploratory study of organisational adaptation to agile project management: an investigation of IT industry in China*. Master thesis, Umea School of business.

Sillitti, A & Succi, G. 2005. Requirements engineering for agile methods in engineering and managing software requirements. *Springer* :309-326.

Simon, M. 2011. Assumptions, limitations and delimitations, Dissertation and scholarly research: recipes for success. Seattle, WA: Dissertation Success, LLC. Available at: www.dissertationrecipes.com (accessed December 2015) .

Sincero, SM. 2015.Advantages and disadvantages of surveys. Available at: https://explorable.com/advantages-and-disadvantages-of-surveys (accessed on 30/06/2015).

Singh, M & Saxena, R. 2014. Agile approach to requirement engineering: how agile processes can help in time-constrained requirements engineering. *International Conference on Multidisciplinary Research & Practice* 1(8):544-547.

Siriram, R. 2011. Convergence of technologies. *South African Journal of Industrial Engineering* 22(1):13-27.

Sonnekus, R & Labuschagne, L. 2003. *The Prosperus Report 2003: ICT project management maturity versus project success in South Africa*. Johannesburg: Rand Afrikaans University.

Spaull, N. 2013. *South Africa's education crisis: the quality of education in South Africa 1994-2011.* Centre for Development and Enterprise, October:1-65.

*Stanford Encyclopedia of Philosophy. Winter 2014 Edition. "*Scientific Objectivity". Available at: http://stanford.library.usyd.edu.au/archives/win2014/entries/scientific-objectivity/ (accessed on 24/05/2015).

Stellman, A & Greene, J. 2005. *Applied software project management.* USA, California, Sebastopol: O'Reilly Media.

Stober, T & Hansmann, U. 2010. *Agile software development.* Berlin Heidenberg: Springer.

Sutherland, J. 2010. Agile principles and values. Available at: http://msdn.microsoft.com/en-us/library/dd997578.aspx (accessed on 03/01/2016).

Sutherland, J & Schwaber, K. 2011. The scrum guide. The definitive guide to scrum: the rules of the game. Available at: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf (accessed on 15/04/2015).

Thanasegaran, G. 2009. Reliability and validity issues in research. *Integration & Dissemination* 4:35-40.

Thomas, JR, Nelson, JK, Silverman, SJ. 2011. *Research methods in physical activity*. USA: Human Kinetics.

Tripathi, V & Goyal, AK. 2014. Agile requirement engineer: roles and responsibilities. *International Journal of Innovative Science, Engineering & Technology* 1(3):213-219.

ur Rehman, T, Khan, MNA & Riaz, N. 2013. Analysis of requirement engineering processes, tools/techniques and methodologies. *International Journal of Information Technology and Computer Science (IJITCS)* 5(3):40.

Veerasamy, N & Labuschagne, WA. 2014. Determination of meme proliferation factors. Paper presented at the 13th European Conference on Cyber Warfare and Security ECCWS-2014 The University of Piraeus Piraeus, Greece, July:188-197.

Walliman, N. 2010. *Research methods: the basics.* USA, New York: Routledge.

Weinreich, R, Neumann, N, Riedel, R & Müller, E. 2015. Scrum as method for agile project management outside of the product development area, in Umeda, S, Nakano, M, Mizuyama, H, Hibino, H, Kiritsis, D, von Cieminski, G (Eds.), *Advances in Production Management Systems: Innovative Production Management Towards Sustainable Growth. Springer*, September:565-572.

WhatIs.com. 2012. Available at: http://whatis.techtarget.com/definition/Agile-glossary-Words-2-Go (accessed on 02/13/2016).

Williams, C. 2011. Research methods. *Journal of Business & Economics Research (JBER)* 5(3).

Williams, L. 2012. What agile teams think of agile principles? *Communications of the ACM* 55(4):71-76.

Whittaker, DJ. 2013. *The Impact and Legacy of Educational Sloyd: Head and Hands in Harness,* Routledge.

Zhu, Y. 2009. *Requirements engineering in an agile environment.* Master Thesis, Uppsala University, Sweden.

Zikmund, W, Babin, B, Carr, J. & Griffin, M. 2012. *Business research methods.* USA, Mason: Cengage Learning.

Appendices

Appendix 1: The Agile Manifesto

The Manifesto for Agile Software Development

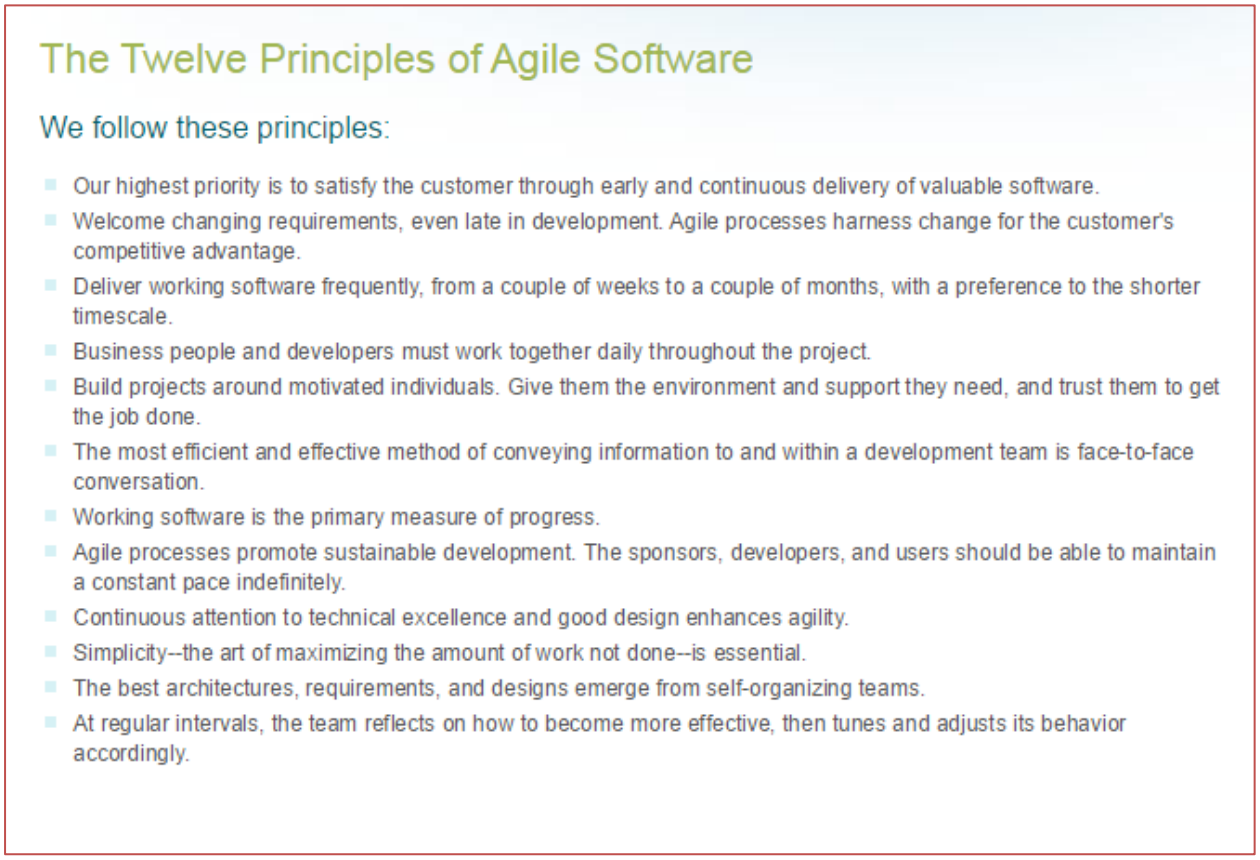
We are uncovering better ways of developing software by doing it and helping others do it.
Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Appendix 2: The principles of the Agile Manifesto



Appendix 3: Agile Requirement Engineering - Survey Questionnaire

1 - GENERAL INFORMATION (Respondent Info)

Company Name: Click here to enter text. Phone: Click here to enter text.

Respondent Title: Click here to enter text. Email: Click here to enter text.

2 - AGILE PROCESSES

2.1 Please select agile method(s) your company uses:

☐ Rational Unified Process
☐ Feature-Driven Development
☐ Lean Software Development
☐ Open Unified Process
☐ Dynamic Systems Development Method
☐ Crystal
☐ Scrum
☐ Agile Unified Process
☐ Kanban
☐ Extreme Programming

Other(s) please list:Click here to enter text.

3 - AGILE REQUIREMENTS ENGINEERING (ARE)

3.1 To what extent do you value the following agile practices? (Rate from 1-5; 1=not important to 5=very important)

	1	2	3	4	5
Face-to-face communication	☐	☐	☐	☐	☐
Iterative requirements engineering	☐	☐	☐	☐	☐
Requirements Prioritisation	☐	☐	☐	☐	☐
Prototyping	☐	☐	☐	☐	☐
Test-driven development	☐	☐	☐	☐	☐
Review meetings and acceptance test	☐	☐	☐	☐	☐
Continuous Validation	☐	☐	☐	☐	☐

3.2 What are the activities performed during iteration(s) in Agile Requirements Engineering?

☐Elicitation
☐Analysis
☐ Documentation
☐ Validation
☐ Management
☐ Modelling
☐ Negotiation
☐ Traceability
☐ Evolution

Other(s) please list:Click here to enter text.

4 - ISSUES RELATED TO AGILE REQUIREMENTS ENGINEERING

4.1 What are issues in requirements elicitation you usually experience during iteration(s)?

☐ Ambiguous requirements
☐ Requirements Volatility
☐ Scope problems
☐ Lack of clarity
☐ Wastages
☐ Lack of trust
☐ Incompleteness
☐ Misinterpretations
☐ Omissions
☐ Unrealistic expectations
☐ Vague customer needs
☐ Inconsistent Information

Other(s) please list:Click here to enter text.

4.2 What are issues in requirements analysis you usually experience during iteration(s)?

☐ Scalability Issues
☐ Issues of Prioritisation
☐ Issues of requirements negotiation

Other(s) please list:Click here to enter text.

4.3 What are issues in requirements documentation you usually experience during iteration(s)?

☐ Lack of proper documentation ☐Minimal documentation
☐ Staff turnover ☐Unavailability of customer representative
Other(s) please list:Click here to enter text.

4.4 What are the issues in requirements validation you usually experience during iteration(s)?
☐Evolutionary prototyping issues ☐Lack of proper validation tools & techniques
Other(s) please list:Click here to enter text.

4.5 What are the issues in requirements management you usually experience during iteration(s)?
☐Lack of proper management tools ☐Requirements Changeability
☐ Lack of Requirements Traceability ☐Problem with version control
Other(s) please list:Click here to enter text.

4.6 Please select other issues in requirements you usually experience during iteration(s)? (Others include issues related to requirements negotiation, modelling, validation, or evolution).
☐ Customer rigidity ☐ Conflicting priorities
☐ Limited access to project stakeholders ☐ Overwhelming participation
☐ Indecisive project stakeholders ☐ Inadequate IT Solutions
☐ Views Inconsistencies ☐ Problem domain complexity
☐ Misleading cue ☐ Ambiguous requirements
☐ Hidden Functionality ☐ Inappropriate Functionality
☐ Missing Functionality ☐ Budget problems
☐ Vague Requirements
Other(s) please list:Click here to enter text.

4.7 Overall, which activities put more constraints on your projects?
☐Elicitation ☐ Management ☐ Evolution
☐Analysis ☐ Modelling Other(s) please list:Click here to enter text.
☐ Documentation ☐ Negotiation
☐ Validation ☐ Traceability

5 - TOOLS FOR AGILE REQUIREMENTS ENGINEERING

5.1 What are the agile tools & techniques your company habitually uses?
☐ JAD Sessions ☐ QPack ☐ Scrumworks
☐ GORE ☐ Rectify ☐ DMS (Documentation Management System)
☐ Cognition Cockpit ☐ Requirements Splitting
☐ Craddle ☐ Requirements prioritisation Other(s) please specify:
☐ Quality Center ☐ Requirements Decoupling Click here to enter text.

6 - STAKEHOLDERS INTERACTION & COLLABORATION

6.1 Please select the type of software you usually develop.
☐ Artificial intelligence ☐ Personal computer Other(s) please list:Click here to enter text.
☐ Embedded ☐ Real-time applications
☐ Engineering and scientific ☐ System
☐ Web-based ☐ Business

6.2 Customer(s) Experience in Agile Software Development (ASD)& Quality of Requirements

Onsite customer(s): ☐ YES ☐ NO
Customer(s) Experience: ☐ Novice ☐ Intermediate ☐ Experienced ☐ Expert
Quality of requirements in general: ☐ Poor ☐ Fair ☐ Good ☐ Very good ☐ Excellent
Awareness to ASD: ☐ Not at all ☐ Slightly ☐ Somewhat ☐ Moderately ☐ Extremely

6.3 Do you consider requirements changes when the project is at _?

	Ignore	Consider	Somehow consider	definitely Consider
¼ of completion	☐	☐	☐	☐
½ of completion	☐	☐	☐	☐
¾ of completion	☐	☐	☐	☐
Final Stage	☐	☐	☐	☐
Post-maintenance delivery	☐	☐	☐	☐

6.4 Overall, what is the amount of requirements implemented (from customers)?
☐<25% ☐ 25-50 % ☐ 50-75% ☐ 75-100% ☐ NA

6.5 How do you value the following agile principles that relate to RE (rate from 1 being the less important to 5 the very important)

	1	2	3	4	5
Customer satisfaction	☐	☐	☐	☐	☐
Welcome changing requirements	☐	☐	☐	☐	☐
Frequent working short releases	☐	☐	☐	☐	☐
Collaboration (Business people & developers)	☐	☐	☐	☐	☐
Face-to-Face Communication	☐	☐	☐	☐	☐
Working software is measure to progress	☐	☐	☐	☐	☐
Continuous attention to technical excellence	☐	☐	☐	☐	☐
Simplicity (is essential)	☐	☐	☐	☐	☐
Team self-organization	☐	☐	☐	☐	☐
Frequent Refactoring	☐	☐	☐	☐	☐

7 - NON-FUNCTIONAL REQUIREMENTS (NFR)

7.1 Please select the NFR that you consider when developing software.

☐ Availability ☐ Maintainability ☐ Portability ☐ Robustness
☐ Efficiency ☐ Integrity ☐ Reliability ☐ Safety
☐ Flexibility ☐ Performance ☐ Reusability ☐ Scalability
☐ Usability

7.2 NFR Issues

	No constraint	Minor constraint	Neutral	Moderate constraint	Major constraint
How much of a constraint the selected NFR are on software?	☐	☐	☐	☐	☐

7.3 Are NFR a priority to your projects?
☐ Not a priority☐ Low priority☐ Medium priority☐ High priority☐ Essential

Appendix 4: Ethics clearance certificate

UNISA | college of science, engineering and technology

Dear Mr. Yanda Sebega (44934203)

Date: 2015-08-07

Application number:
080/YS/2015/CSET_SOC

REQUEST FOR ETHICAL CLEARANCE: (Exploring issues in agile requirements engineering in the South African Industry)

The College of Science, Engineering and Technology's (CSET) Research and Ethics Committee has considered the relevant parts of the studies relating to the abovementioned research project and research methodology and is pleased to inform you that ethical clearance is granted for your research study as set out in your proposal and application for ethical clearance.

Therefore, involved parties may also consider ethics approval as granted. However, the permission granted must not be misconstrued as constituting an instruction from the CSET Executive or the CSET CRIC that sampled interviewees (if applicable) are compelled to take part in the research project. All interviewees retain their individual right to decide whether to participate or not.

We trust that the research will be undertaken in a manner that is respectful of the rights and integrity of those who volunteer to participate, as stipulated in the UNISA Research Ethics policy. The policy can be found at the following URL:
http://cm.unisa.ac.za/contents/departments/res_policies/docs/ResearchEthicsPolicy_apprvCounc_21Sept07.pdf

Please note that the ethical clearance is granted for the duration of this project and if you subsequently do a follow-up study that requires the use of a different research instrument, you will have to submit an addendum to this application, explaining the purpose of the follow-up study and attach the new instrument along with a comprehensive information document and consent form.

Yours sincerely

Prof Ernest Mnkandla
Chair: College of Science, Engineering and Technology Ethics Sub-Committee

Prof IOG Moche
Executive Dean: College of Science, Engineering and Technology

RECEIVED
2015 -08- 07
OFFICE OF THE EXECUTIVE DEAN
College of Science, Engineering and Technology

University of South Africa
College of Science, Engineering and Technology
The Science Campus
C/o Christiaan de Wet Road and Pioneer Avenue,
Florida Park, Roodepoort
Private Bag X6, Florida, 1710
www.unisa.ac.za/cset

UNISA | college of science, engineering and technology

INVOICE

To: **Yanda Sebega**
From: Kathy Wood
Freelance copy editor
Date: **10th February 2016**

Task: **Exploring issues in agile requirements engineering in the South African software industry**

Booking Fee	**R 500**
Upfront payment	**R1 725**
Final payment	**R1 225 (this invoice) Payable within 48 hrs of delivery of an acceptable quality edit**
Total cost	**R3 450**
Method of payment	Cash or EFT

Banking details

Name	K M Wood
Bank	First National
Branch	Carlswald, Midrand, Gauteng
Account type	Current
Account no	506 300 29722
Branch Code	250117