UNIVERSITY OF KWAZULU-NATAL

The Adoption of Agile Software Development Methodologies by Organisations in South Africa

By

Cassim Vanker

A dissertation submitted in fulfilment of the Requirements for the degree of Master of Commerce in Information Systems & Technology

School of Management, Information Technology & Governance

College of Law and Management Studies

Supervisor: Prof Rembrandt Klopper

Co-Supervisor: Mr Karunagaran Naidoo

2015

DECLARATION

I __Cassim Vanker________________declare that

(i) The research reported in this dissertation/thesis, except where otherwise indicated, is my original research.

(ii) This dissertation/thesis has not been submitted for any degree or examination at any other university.

(iii) This dissertation/thesis does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

(iv) This dissertation/thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

a) their words have been re-written but the general information attributed to them has been referenced:

b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

(v) This dissertation/thesis does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation/thesis and in the References sections.

Vanker

____ 02 December 2015

C VANKER, 951011260 Date

ACKNOWLEDGMENTS

It is with a sense of thankfulness, appreciation and gratitude that I acknowledge the invaluable suggestions, encouragement, assistance and constructive inputs that I received from the following:

- My supervisor Prof Rembrandt Klopper for your patience, motivation, guidance, support and immense knowledge.
- My co-supervisor Karunagaran Naidoo for your insightful comments, suggestions and encouragement throughout my research.
- The statistician Gill Hendry for meticulously computing and assisting with all the statistics involved in this study.
- Prismic Reflections for their assistance with the art work for the graphical report.
- My family for the constant encouragement, support and the numerous sacrifices that they had to make during my research.
- My brother Faisal Vanker for your unfailing and willing support and assistance in helping me secure respondents.
- Sameer Sabjee for assisting with the administrative work associated with the compilation of this research.
- The proof readers for your diligence in reading and correcting errors throughout the research process.

ABSTRACT

The software development life cycle (SDLC) is considered to be the oldest software development methodology and is often described as a deliberate, methodical and structured approach that is used by development teams to develop information systems. There are two SDLC methodologies, which are currently being utilized by programming practitioners, namely traditional development and Agile development.

The traditional view to software development assumes that the customer does not have a full understanding of their requirements and would thus need a detailed specification designed before development begins. Unlike the traditional view, the agilest views software as unpredictable and rely on people and their creativity rather than on processes.

Anecdotal evidence suggests various benefits associated with Agile methodologies, these include increased collaboration, the quality of the end product, transparency and productivity. An international Survey conducted, amongst software development companies, shows that 80% of respondents indicated that they had practiced Agile.

While international research shows the popularity of Agile methodologies very little is known about Agile in South Africa. This research provides insight into the adoption of Agile, by South African organisations. The results of the study indicate that Agile methods are being used by organisations in South Africa. However, there is no evidence to suggest its being used overwhelming. Furthermore, this study uses the Gartner hype cycle as the underpinning theoretical model and places Agile methods on the Gartner hype cycle as being in the trough of disillusionment.

Keywords: *Agile, Software, Development, Processes, Traditional, Requirements, Methodologies, Projects, People, Organisations*

TABLE OF CONTENTS

TABLE OF FIGURES

TERMS AND ABBREVIATIONS

Acronym	**Definition**
ANOVA	Analysis of Variance is a method in statistics that is used to test differences between two or more means.
DOI	Diffusion of innovations is an information systems theory that seeks to explain how new innovations and ideas are spread.
ERP	Enterprise resource planning refers to software that allows organisations to manage the organisation and automate back office functions.
FDD	Feature-driven development refers to an incremental and iterative software development process.
GHC	The Gartner hype cycle is a graphical presentation of the adoption, maturity and social application of specific innovations.
IBM	International Business Machines Corporation is a multinational technology and consulting corporation based in the USA.
KANBAN	Kanban is a technique for managing software development in an efficient way.
PMBOK	Project Management Body of Knowledge refers to a book that contains a set of guidelines and terminology for project management.
SCRUM	Scrum is an agile software development framework.
SDLC	The systems development life cycle is a term used to describe the processes behind the planning, creating, testing, and deployment of an information system.
SEDA	The Small Enterprise Development Agency is an organisation that provides development and support services for small enterprises in South Africa.
SIC	Means "intentionally so written" and is generally included after a quote. Sic indicates that the quote has been transcribed exactly as found.
SPSS	Statistical Package for the Social Sciences is a software package used for statistical analysis.
STREET	Scope, Track, Rank, Evaluate, Evangelize and Transfer refers to a set of processes defined in the Gartner hype cycle that helps organisations make sound adoption decisions.
SWOT	Strengths, weaknesses, opportunities and threats is a planning method used to strategically evaluate a project or in a business venture.

Acronym	**Definition**
TAM	The Technology Acceptance Model is an information systems theory that is used to determine how users accept and use a technology.
TBP	The theory of planned behaviour is an information systems theory that is used to link belief and behaviour when adopting technology.
TOE	The technology-organisation-environment framework describes process by, which organisations adopt and implement innovations.
UTAUT	The unified theory of acceptance and use of technology is a technology acceptance model that aims to link user intentions and subsequent usage behaviour.

Chapter 1

PROBLEM STATEMENT AND RESEARCH DESIGN

1.1 Problem Statement

Agile software development methodologies are gaining traction internationally. However, in South Africa little is known about the state of Agile or the adoption barriers and the impact of Agile methodologies on software being developed.

1.1.1 Sub-Problems

The following sub-problems were identified:

- The state of Agile software development in South Africa is not known.
- The barriers preventing the adoption of Agile by South African organisations have not been identified.
- It has not yet been determined how Agile software development methodologies have impacted the development of software in South Africa.

1.1.2 Aim

The Aim of this study is to determine the state of Agile software development in South Africa.

1.1.3 Research Objectives

The following research goals and sub-goals have been identified from the problems:

- To determine the state of Agile software development in South Africa.
- To identify barriers preventing the adoption of Agile by South African organisations.

- To determine the impact of Agile methodologies on the development of software in South Africa.

1.1.4 Research Questions

After extensively reviewing the problem statement and sub-problems and after consultations with practitioners within the field the following research questions were identified:

- What is the state of Agile software development in South Africa?
- What are the barriers preventing the adoption of Agile by South African organisations?
- What is the impact of Agile methodologies on the development of software in South Africa?

1.1.5 Learning Objectives

The researcher intends to use the research process as a means to test the relationships between the theoretical framework and the empirical results that will be ultimately derived from this study. This will be achieved through the following objectives:

- To use the theoretical framework to determine the state of Agile software development in South Africa.
- To use the theoretical framework to identify barriers preventing the adoption of Agile by South African organisations.
- To use the theoretical framework to determine the impact of Agile methodologies on the development of software in South Africa.

In the final chapter of this study, the researcher will conduct a self-assessment of this research project. This assessment will indicate the strengths, weaknesses, opportunities and strengths that have been identified for this study. In addition, the researcher will indicate to what extent he will be able to do empirical analysis using the framework. These factors will assist improving future research.

1.1.6 Introduction

Software is developed as a project, which involves people guided by strategies and goals that produce documents and code through the use of tools. As such one of the key ingredients to managing software development projects is the use of a software development methodology (Burback, 1998). A software development methodology is a framework that is used to plan structure and control the software development process (Centres Medical and Medicaid Services, 2008).

The software development life cycle (SDLC) is considered to be the oldest software development methodology and is often described as a deliberate, methodical and structured approach that is used by development teams to develop information systems (Naderuzzaman, Rabbi & Beg, 2011). There are two SDLC methodologies, which are currently being utilized by programming practitioners, namely traditional development and Agile development (Leau, Loo, Tham & Tan, 2012).

1.1.7 Agile Software Development

The traditional approach to software development assumes that software requirements are in the minds of the customer and one just needs to work hard enough to get to them (Janes & Succi, 2012). Traditionalists emphasize that variations are identifiable and by measuring and refining processes they then can be eliminated (Nerur, Mahapatra & Mangalaraj, 2005). Furthermore, the traditional approach is often characterised as process-and document-centric (Kalermo & Rissanen, 2002). As a result some programming practitioners have found this process-and document-centric view to software development frustrating (Awad, 2005).

In February 2001, a group of 17 programming practitioners, called the Agile Alliance, formulated the four core values and principles of Agile in an Agile Manifesto, which is stated below (Beck, Beedle, Bennekum, Cockburn, Cunningham, Fowler, Grenning, Highsmith, Hunt, Jeffries, Kern, Marick, Martin, Mellor, Schwaber, Sutherland & Thomas, 2001a):

"We are uncovering better ways of developing software by doing it and helping others to 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." (Beck *et al.*, 2001a: 1).

1.1.8 Traditional Software Development vs. Agile

The advent of the Agile paradigm has divided the software development communities into two camps namely, traditionalist and agilists, each claiming superiority. Unlike the traditional view, the agilist views software as unpredictable and rely on "people and their creativity rather than on processes" (Highsmith & Cockburn, 2001; Rao, Naidu & Chakka, 2011). Furthermore, Fowler and Davenport describe three key differences between Agile and traditional methods, the first difference relates to "adaptability vs. predictability" (Davenport, 2005; Fowler, 2005).

Traditional methods plan out large parts of the software development process and are thus change resistant, while Agile methods on the other hand welcome change (Davenport, 2005; Fowler, 2005). According to Schwaber & Sutherland typical software development processes have more than 35% of their requirements changed (Schwaber & Sutherland, 2012). These changes can be attributed to the dynamic market place, business struggling to define their requirement or the difficulty associated with defining a system before its completion (Schwaber & Sutherland, 2012). A study by the Standish Group highlight three major factors that challenge traditional projects, these factors are the "Lack of user support", "Incomplete requirements & Specifications" and "Changing requirements and specifications" (Standish Group, 1995; Attarzadeh & Ow, 2008). Although the Standish

Group reports are widely quoted, the group's policy of keeping their research data private has raised questions regarding the validity of their research (Eveleens & Verhoef, 2010).

The second difference relates to "people-orientation vs. process-orientation". Traditional methods assert that processes work well irrespective of the people using those processes, while Agile methods assert that processes cannot make up the skills of the team (Fowler, 2005; Gerber, 2010). Traditional methodologies group people into different roles, for example "Testers", "Developers" and "Analysts", placing emphasis on roles and not people (Fowler, 2005). This notion of roles raises the question "Are people replaceable parts?" (Ambler, 2003; Fowler, 2005). The argument is further expounded by Fowler, Schwaber & Sutherland, Bernstein and Ambler who agree that people are not predictable components and have different skillsets, attitudes and intelligence (Ambler, 2003; Fowler, 2005; Bernstein, 2010; Schwaber & Sutherland, 2012).

Cockburn known for his people-centric views makes a compelling argument that people do similar things, but never the same thing, because their reactions vary from day to day (Bernstein, 2010). Furthermore, according to Boehm & Papaccio, Sudhakar, Farooq and Patnaik studies have shown that the largest variation in software quality can be attributed to difference in the people (Boehm & Papaccio, 1988; Sudhakar, Farooq & Patnaik, 2012). Agilists have strong beliefs that people are central to the success of a project (Awad, 2005). According to Highsmith & Cockburn people transfer ideas faster face to face than by reading and writing (Highsmith & Cockburn, 2001). Talking to customers allows developers to iron out difficult issues and examine alternate ways forward.

The third difference refers to the "Sequential vs. Iterative" nature of the traditional and Agile approaches to software development. Traditional methodologies tend to leave the testing and feedback to the last stages of the lifecycle whereas Agile methodologies are distinguished by short iterative cycles with periods of reflection and introspection, collaboration and continuous integration of code changes (Awad, 2005; Nerur *et al.*, 2005). According to Fowler the key to iterative development is to produce frequent working versions of the final system with a subset of the features (Fowler, 2005). The ultimate goal of Agile is to help organisations maintain and develop applications for dynamic markets (Cao, Mohan, Xu & Ramesh, 2009).

1.1.9 The Popularity of Agile Software Development

According to the 2011 State of Agile Survey conducted internationally, amongst software development companies, 80% of respondents indicated that they had practiced Agile. The survey also shows that 60% of respondents indicated that over half of their projects were Agile based. The most common Agile methodology, with 52% of the respondents practicing, was Scrum (VersionOne, 2011). Scrum, a term derived from rugby and football, can be described as a radical approach to planning and managing software development, by bringing decision making authority to the level of operations (Schwaber, 2004; Lacey, 2012). A similar survey of organisations in India shows that 88% of all respondents had reasonable experience with Agile (Thoughtworks.com, 2011).

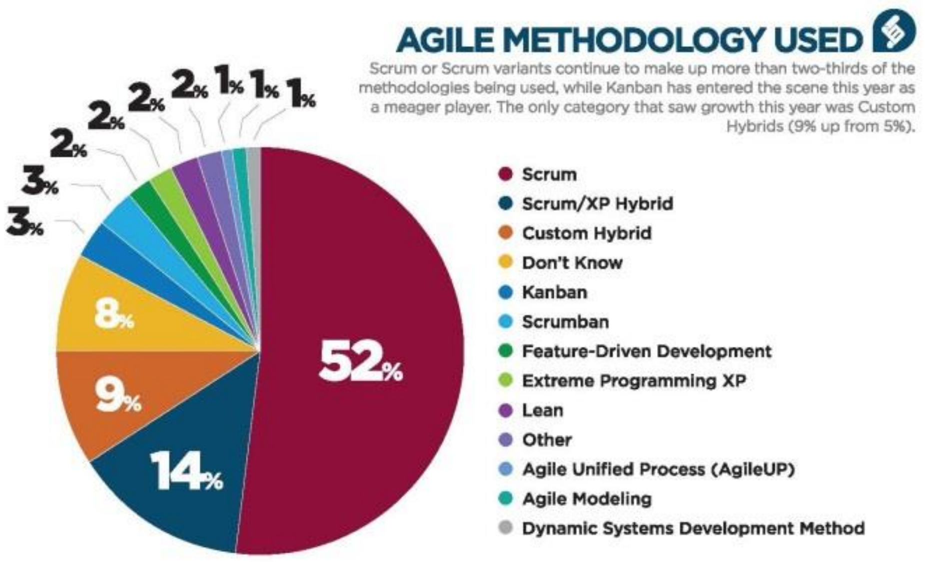


Figure 1-1: Pie chart showing the state of Agile survey.
Adopted from VersionOne (VersionOne, 2011).

A study in 2010 on "The level of adoption and effectiveness of software development methodologies in the software development industry of South Africa" shows that 38.5% of organisations had adopted Agile. The results of this study were based on responses from only 32 team managers (Ramnath, 2011). This raises the question of the sample being representative of the population.

1.1.10 Agile Benefits and Challenges

Anecdotal evidence suggests various benefits associated with Agile methodologies, these include increased collaboration, the quality of the end product, transparency and productivity (Laanti, Salo & Abrahamsson, 2011). In addition, Lycett, Macredie, Patel, Paul, Schwaber, Leganza & D'Silva, Laanti *et al.*, and Vijayasarathy & Turk have identified several other benefits and these include the ability to respond to dynamic market changes, reduced waste of code, more reusable code, better predictability, planning and managing requirements in a more flexible way, improved morale within the organisation and better meeting customer requirements (Lycett, Macredie, Patel & Paul, 2003; Schwaber, Leganza & D'Silva, 2007; Laanti *et al.*, 2011; Vijayasarathy & Turk, 2012). Recent studies by the Standish group show that software developed using Agile processes are three times more likely to succeed than traditional waterfall based processes. In addition these projects have a much lower percentage of cost and time overruns (Cohn, 2012; Schwaber & Sutherland, 2012).

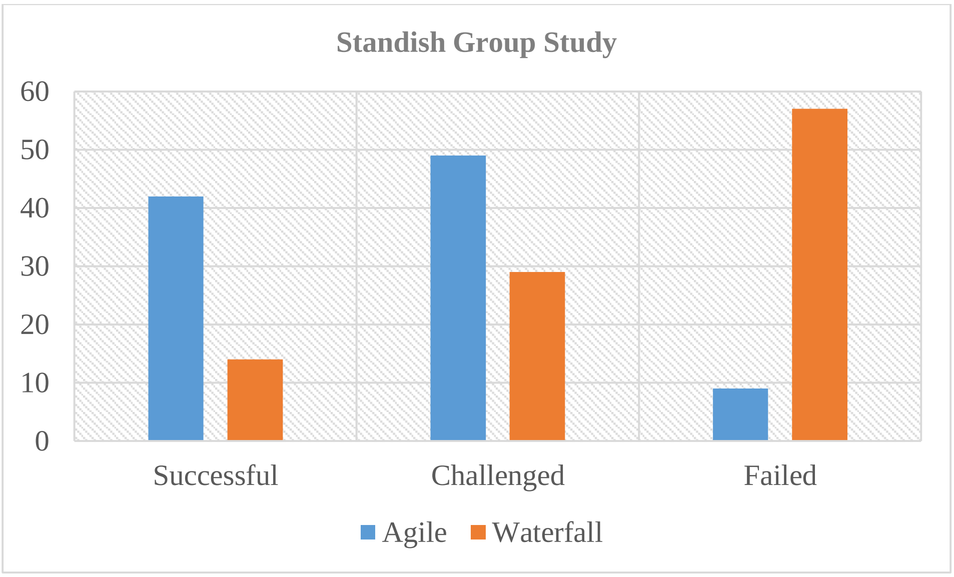


Figure 1-2: Bar graph showing the results from the Standish Group Study. Adapted from Schwaber & Sutherland (Schwaber & Sutherland, 2012).

The Stacey Graph is a matrix proposed by Ralph Stacey that helps management decide on, which method to use when considering an array of complex approaches. It helps by guiding management to an appropriate method when considering complex and adaptive systems based on the degree of certainty and the level of agreement about the issue in question.

Ralph Stacey initially proposed this matrix to help management identify decisions based on two dimensions: the degree of certainty and the level of agreement (GP-Training.Net, 2013). The graph was later adapted by Ken Schwaber to take into account the complexities involved in software development process. The graph shown below is a convergence of three factors: User Requirements, Technology and People (Duka, 2011; Schwaber & Sutherland, 2012).

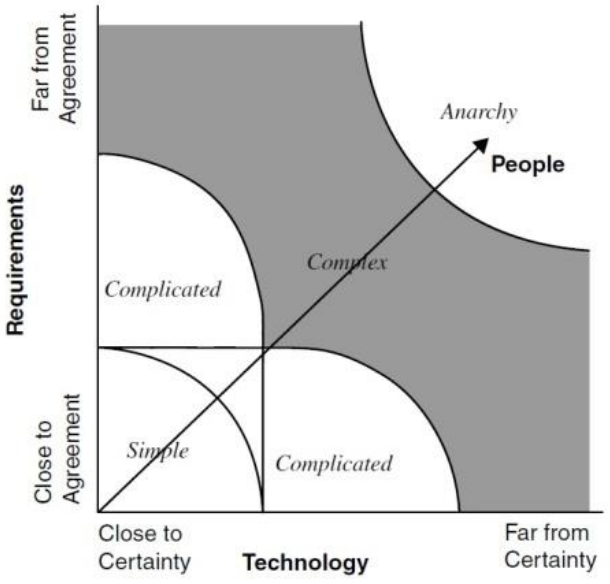


Figure 1-3: Graphic representing the Stacy Graph. Adopted from Schwaber & Sutherland (Schwaber & Sutherland, 2012).

Using the "Stacy Graph" we can plot software projects as follows (Duka, 2011):

- *Simple:* Refers to projects that have agreement on the requirements and certainty on the technology.

- *Complicated:* Refer to projects that have near agreement on requirements or certainty on the technology that will be used.
- *Complex:* Refer to projects that do not have a clear agreement on the requirements and do not have certainty on the technology that will be used.
- *Chaotic:* Refer to projects where technology and requirements are uncertain.

The three constrains are summarized by Schwaber & Sutherland as follows (Schwaber & Sutherland, 2012):

- *Requirements:* "Close to certain" with no risk of change to "far from certain" with vague descriptions and expectations.
- *Technology:* "Well-known and understood" to "far from certainty".
- *People:* "Known and constant in small teams" to "constantly changing with hundreds of people in larger teams".

The Stacy Graph shows that software projects are sometimes chaotic and complex. Furthermore, according to Schwaber & Sutherland traditional methodologies are applicable to simple repeatable work (Schwaber & Sutherland, 2012). They suggest that the right software methodology can be determined by looking at the yield rate or degree of success. The Standish Group case study shows that the predictive life cycle has a yield rate of 14%, when compared to the 42% offered by adaptive life cycle projects (Schwaber & Sutherland, 2012).

Past research has also shown that software methodology process changes are complex organisational wide changes that cannot be accomplished by replacing technologies and tools. Such changes could impact organisational culture, structures and management practices; therefore organisations considering implementing Agile have to understand organisational wide ramifications (Nerur *et al.*, 2005). As such there are several case studies that report impediments to the adoption of Agile processes. These case studies have concluded that organisations are generally unaware of what Agile adoption really means (Laanti *et al.*, 2011).

The customer centric nature of Agile allows for quick feedback loops; however, this can pose a significant risk in projects where customers have insufficient knowledge, especially in larger complex systems (Cao *et al.*, 2009). According to a study by Dybå & Dingsøyr, Agile is more suitable for smaller teams than for larger teams (Dybå & Dingsøyr, 2008). The study also reported a case that depicted a lack of attention to design and architectural issues amongst the limitations. Further studies by Gartner 2008, Laanti *et al.*, and Vijayasarathy & Turk citing reasons for organisations failing to adopt Agile lists human and social factors as well as the polarization of individuals, organisations failure to plan systematically for Agile adoption and Agile requiring a fundamental change in philosophy as key issues to be considered when adopting Agile (Gartner, 2008a; Laanti *et al.*, 2011; Vijayasarathy & Turk, 2012).

While international research and surveys continue to prove the gaining popularity of Agile methodologies, very little is known about the state of Agile software development in South Africa.

This research thus aims to gain insight into the adoption of Agile, by studying organisations that have implemented Agile methods.

1.2 Research Design

Research design is a plan of how the work will be undertaken (Maxwell, 2012). Good design leads to components working harmoniously together and flawed design leads to poor operation (Maxwell, 2012). Listed below is the research design description of this study.

1.2.1 Qualitative Research

This study will be conducted using qualitative methods through the use of questionnaires. Quantitative research is based on the presumption that knowledge is only acquired through the eyes of the researcher (Jonker & Pennink, 2009). Furthermore, quantitative research involves the application of systematic or scientific steps to measures the relationships between variables (Edmonds & Kennedy, 2012). Finally, through the use of statistical inference a researcher can make generalisations about a target population.

1.2.2 Research Onion

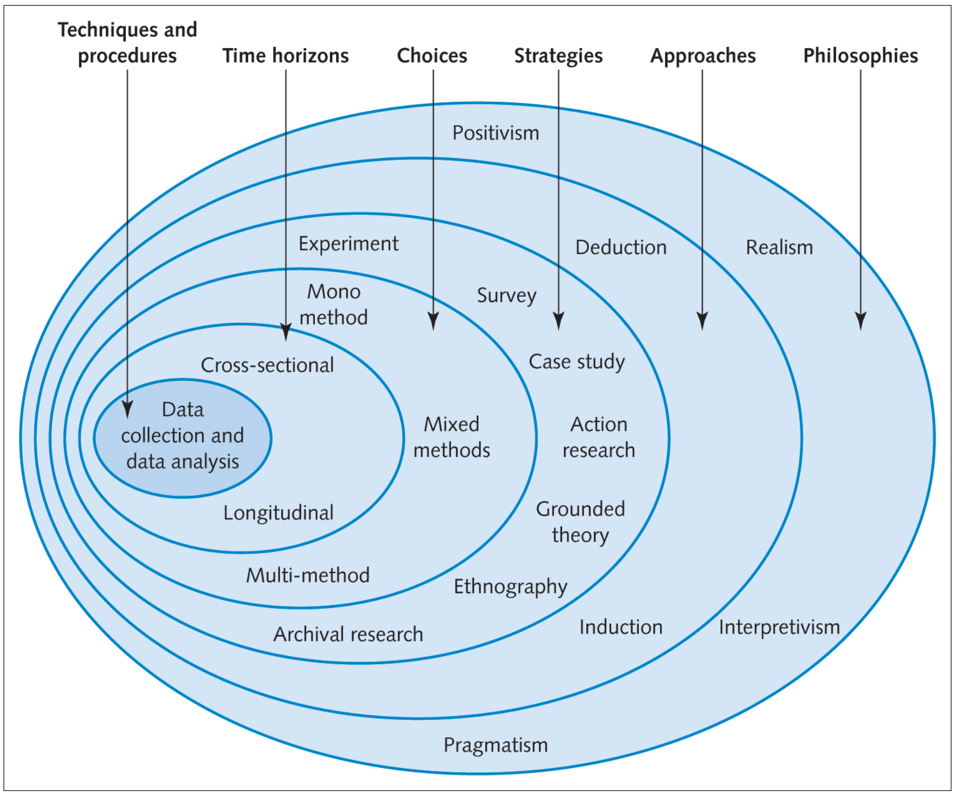


Figure 1-4: Graphical representation of the research onion. Adopted from Saunders, Lewis & Thornhill (Saunders, Lewis & Thornhill, 2012).

Saunders et al., describes the research approach as a six layered ‘onion’. Firstly the researcher has to establish the research philosophy(Saunders *et al.*, 2012). The next step is to determine the appropriate research approach. Following on from the previous two steps the third step deals with the research strategy (Saunders *et al.*, 2012). The fourth and fifth steps respectively deal with the choices and time horizon for the study (Saunders *et al.*, 2012). Finally, the sixth step identifies the data collection methodology. Each of these layers is described in the sections below (Saunders *et al.*, 2012).

1.2.2.1 Research Philosophy

The outer layer of the research onion is called the philosophy layer. The philosophy layer describes how the researcher views the world and his or her personal views of what constitutes acceptable knowledge (Saunders *et al.*, 2012).

A researcher viewing the world through observation and cause and effect reflects the philosophy of positivism. Positivism assumes that reality is external and objective and that subjects should be studies through objective methods rather than subjective methods (Kulatunga, Amaratunga & Haigh, 2007).

Another scientific based enquiry philosophy is realism. Researchers who take a direct realist position often argue that our senses provide us with direct awareness of the world whereas researchers reflecting the critical realist approach argue that awareness must be processed subjectively by the mind (Saunders & Tosey, 2013).

Interpretivism is an approach where the researcher is concerned with the gathering of rich insights and providing law-like generalisations into subjective matter. Interpretivism is researched on people rather than objects and is often conducted with an empathetic stance in order to gain an understanding of the social world (Saunders & Tosey, 2013).

Finally, pragmatism considers that no single viewpoint can give a clear understanding of the problem as there are multiple realities. As such the researcher must ensure the reliability, credibility and relevance of the data to be collected (Saunders & Tosey, 2013).

This study will use the positivism research philosophy. Positivists assume that meaning exists apart from consciousness and that meaning is inherent in the object being examined (Crotty, 1998). A study by Orlikowski & Baroudi classifies information system research as positivist if there is evidence of quantifiable measures of variables, formal propositions, hypothesis testing and the drawing of inferences (Orlikowski & Baroudi, 1991). Furthermore, a positive philosophy leads to a systematic scientific approach to research and therefore lends itself to the use of quantitative methods (Mukherji & Albon, 2009).

1.2.2.2 Research Approaches

This layer of the research onion deals with two broad groups of research approaches known as deductive and inductive approaches. The deductive approach is aimed at testing a theory from the top down "a theory to hypotheses to data that adds to or contradicts the theory" (Soiferman, 2010: 3). On the other hand, the inductive approach is used to generate a new theory from the bottom up by generating themes and interconnected themes from the data (Soiferman, 2010). The difference between inductive and the deductive approaches are highlighted in the figure below:

Deductive	**Inductive**
Moving from theory to data	Moving from data to theory
This approach is commonly used in natural sciences	This approach is commonly used in social sciences
A highly structured approach	Flexible structure to permit changes

Deductive	**Inductive**
Explain causal relationships between variables	Understanding the meanings that humans attach to events
The selected sample should be of sufficient size in order to make generalised conclusions	There is less concern with generalisation.

Figure 1-5: A table showing the major differences between the deductive and inductive approaches to research. Adopted from Pathirage, Amaratunga & Haigh (Pathirage, Amaratunga & Haigh, 2008).

This study will use the deductive research approached as this approach is highly structured and will be used to establish casual relationships between variables. Furthermore, the deductive approach is linked to a positivism paradigm while an inductive approach is linked with the interpretivism paradigm (Crowther & Lancaster, 2008).

1.2.2.3 Research Strategies

Moving from the research approaches layer of the research onion to the next layer reveals the research strategies layer. This layer deals with how the researcher plans to carry out his or her work. A researcher may adopt one or more strategies in his or her research design as he or she plans to go about answering the research questions (Saunders & Tosey, 2013). These strategies include questionnaires, case studies, action research and other related approaches described in the Research Onion, contained in figure 1-4.

This research will be conducted through the use of questionnaires. Questionnaires have advantages over interviews because of the low cost associated with their collection and processing (Jones, Murphy, Edwards & James, 2008). Furthermore, they have the ability of reaching a larger audience and this makes them ideal for this study (Jones *et al.*, 2008).

1.2.2.4 Research Choices

The choices layer described in Saunders & Tosey research onion includes the mixed method, the mono method and the multi-method (Saunders & Tosey, 2013). In the mixed-methods the researcher uses two or more methods of research (Saunders & Tosey, 2013). As such mixed-methods often use both quantitative and qualitative methodologies. On the other hand the mono-method involves the use of only one research approach for the study (Saunders & Tosey, 2013).

Finally, in the multi-method approach research is divided into segments and each segment produces a specific dataset. Each dataset is analysed using techniques derived from quantitative or qualitative methodologies (UK Essays, 2013). With the above in mind it must be noted that this research will be conducted using a single source of data based on quantitative techniques and as such will use a mono method.

1.2.2.5 Research Time Horizon

This layer highlights the time over, which the researcher plans to undertake the study. Research undertaken to address a problem or answer a question at a particular time is described as cross-sectional (Saunders & Tosey, 2013).

On the other hand when data is being collected over an extended period of time it is described as longitudinal (Saunders & Tosey, 2013). This study can be described as cross-sectional as it will be undertaken at a particular point in time.

1.2.3 Ethical Clearance

This research will conform to the ethical requirements of the University of KwaZulu-Natal. These requirements include:

- **Voluntary and informed consent**: A consent document will be sent to gatekeepers within organisations and will be written in lay language excluding technical terms. Participants would be enlightened about the study before they are required to give consent to participate in the study. Furthermore, participants would be able to withdraw from the research at any time. Anonymity and privacy when requested would be observed.

- **Beneficence or convey**: This study will support the best interests of society and its participants with fairness and justice of outcomes and procedures.

1.2.4 Design of The Questionnaire

Each sub-problem of the research domain will be used as a sub section in the questionnaire and questions will be formulated in those respective areas. A statistician will be used to identify the appropriate statistical test for each question and to ensure that the questions are aligned appropriately. Feedback from the statistician will be used to rework the questions.

This study will use Survey Monkey to administer the questionnaire. Survey Monkey is an online survey tool that allows questionnaires to be administered electronically (Inc.com, 2010). A new survey will be created on Survey Monkey and the questions will be uploaded accordingly. The informed consent will also be added into Survey Monkey. Page logic will be used in Survey Monkey to terminate the survey in the following instances:

- If respondents do not provide consent to participate in the survey.
- If respondents are not from South Africa
- If respondents do not have Agile experience.

Radio buttons will be used in Survey Monkey to limit the user to a single response when a single selection is required and in the case of checklist questions, checkboxes will be used to allow the user to select multiple responses.

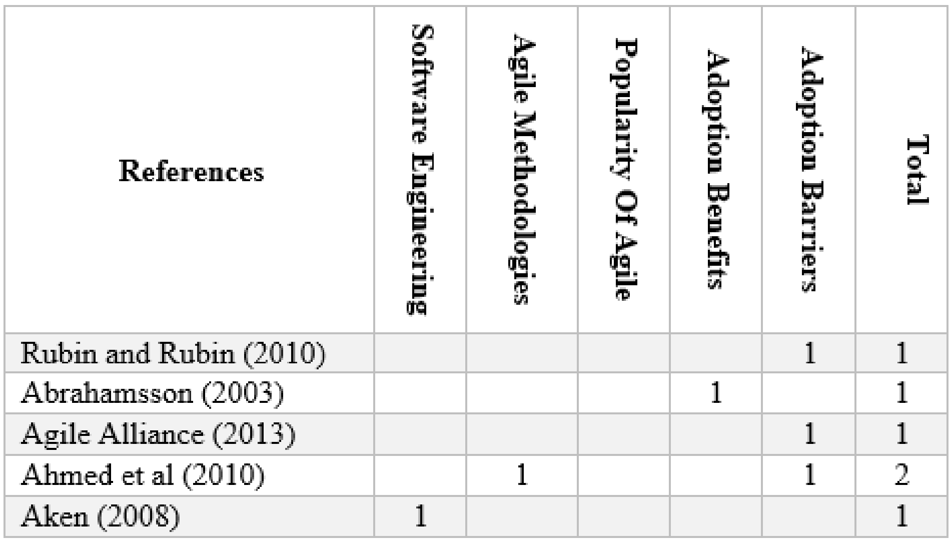
In the case of questions that require ranking and percentage responses, numerical textboxes will be used with data validation to ensure that only numerical data can be captured. In order to ensure that data will be captured accurately and consistently by all respondents the above validations and restrictions will be implemented.

1.2.5 Concept Matrix

According to Klopper & Lubbe the concept matrix is a powerful tool that can be used to provide traction, cohesion and closure in research projects (Klopper & Lubbe, 2012). Furthermore, the concept matrix is used to organise the literature review and allows the researcher to conduct a critical comparative review of all literature under each concept (Klopper & Lubbe, 2012). This significantly strengthens the relevance of the research (Klopper & Lubbe, 2012).

The concept matrix consists of columns that represent concepts. Each concept is extracted from the problem statement (Klopper & Lubbe, 2012). The rows in the matrix represent references from the literature. References are shown using Harvard style abbreviations (Klopper & Lubbe, 2012).

If a particular reference discusses a concept then the number one is placed in the appropriate column (Klopper & Lubbe, 2012). The concept matrix is used in this study to ensure that the literature review is aligned to the study. A completed concept matrix for this study can be found in addendum 13. However, for illustrative purposes an example of the concept matrix is presented in figure 1-6 below.



References	Software Engineering	Agile Methodologies	Popularity Of Agile	Adoption Benefits	Adoption Barriers	Total
Rubin and Rubin (2010)					1	1
Abrahamsson (2003)				1		1
Agile Alliance (2013)					1	1
Ahmed et al (2010)		1			1	2
Aken (2008)	1					1

Figure 1-6: Graphical representation of the concept matrix. Adopted from Klopper & Lubbe (Klopper & Lubbe, 2012).

1.2.6 Theoretical Framework

The ethical clearance process at the University requires that the researcher choose an appropriate framework before applying for ethical clearance. This process puts the cart before the horse as the selection of a theoretical framework requires a thoughtful and deep understanding of the problem, its purpose, significance, and research questions (Grant & Osanloo, 2014). Furthermore, Grant & Osanloo use the analogy of an electrical system in a house to describe the use of a theoretical framework suggesting that a successful implementation of a theoretical framework should connect all element of a dissertation together (Grant & Osanloo, 2014). This becomes difficult to achieve without a comprehensive literature review. Furthermore, after the researcher has been granted ethical clearance he or she becomes bound to the chosen framework as any changes to the framework would need to be resubmitted for ethical clearance.

1.3 IMPORTANCE/SIGNIFICANCE OF THE STUDY

According to research by Gartner, Agile practices will be utilized in 80% of software development projects by 2012. Organisations not utilizing key Agile practices or not investing in support tools and training will find themselves shifting towards pseudo Agile. While potentially delivering short-term productivity boasts, these organisations in the long-term will experience declines in productivity and quality (Proulx, 2010).

This research will give a general overview of the current state of Agile processes among organisation in South Africa, providing valuable information to those organisations intending to make the process shift. Organisations such as the Institute of Information Technology Professionals South Africa and the Agile user group can use this research as a basis for further studies within the field.

1.4 CHAPTER SUMMARY

This chapter lays the foundation for the thesis. The chapter is divided into two sections. The problem statement section of this chapter introduces the research background and outlines the principles behind the two most common software development methodologies. Namely traditional software development and the Agile software development. The differences between these methodologies and a brief popularity survey are also presented. The chapter also includes a synopsis of the challenges and benefits associated with Agile adoption.

The second section of this chapter describes the research design of the study. This section outlines a plan for the execution of this study. Finally, a synopsis of the chapters that will follow is provided below:

- **Chapter Two**: Examines applicable theoretical frameworks for this study. It also summarizes the frameworks and finally suggests an appropriate framework for this study.

- **Chapter Three**: Analyses and examines the existing literature on the topic.
- **Chapter Four**: Provides an in-depth discussion about the selected theoretical framework.
- **Chapter Five**: Defines the research methodology that was used to analyse the data required to answer the research questions.
- **Chapter Six**: This chapter presents the results from the sample by examining the data in terms of the research questions.
- **Chapter Seven**: Discusses the results in terms of the theoretical framework, literature review and research topic.
- **Chapter Eight:** This chapter is the concluding chapter and highlights the findings, limitations and finally provides recommendation for future research.

Chapter 2

THEORETICAL FRAMEWORK

2.1 Introduction

The theoretical framework provides the structure for this study and is often used to explain phenomena that we experience in the world (Moore, 2006). As a starting point this chapter identifies suitable information systems frameworks, describes those frameworks, provides a comparison between frameworks and finally prescribes a framework, which will be used in this study.

2.2 Suitable Information Systems Theoretical Frameworks

Several theories are used in information systems research. However, after examining the literature and studies by Oliveira & Martins and Janes & Succi the following technology adoption based theories were shortlisted for this study (Oliveira & Martins, 2011; Janes & Succi, 2012):

- Technology acceptance model (TAM)
- The theory of planned behaviour (TPB)
- The unified theory of acceptance and use of technology (UTAUT)
- The diffusion of innovation (DOI) theory
- The technology, organisation and environment framework (TOE)
- The Gartner hype cycle (GHC)

2.3 Technology Acceptance model (TAM)

An influential and commonly employed model that is related to technology adoption and use is TAM (Benbasat & Barki, 2007). TAM addresses the issues related to the acceptance or rejection of technology amongst users (Davis, 1989). According to TAM the determinants of individual acceptance of technology can be used to explain and predict an individual behaviour across a broad range of technologies (Suryaningrum, 2012).

TAM measures behaviour relevant components through the use of two variables:

- The first variable is the "*perceived usefulness*", which refers to users who use technology because they believe it will help them perform their job better (Davis, 1989).
- Secondly even though users believe that a technology is useful they may also believe that the effort required outweighs the benefits. This is referred to as "*perceived ease of use*" (Davis, 1989).

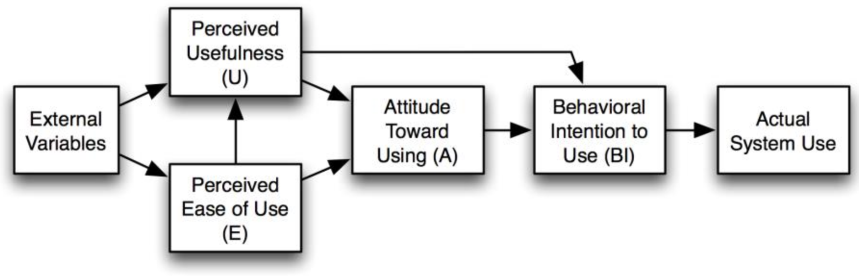

External Variables
Perceived Usefulness (U)
Perceived Ease of Use (E)
Attitude Toward Using (A)
Behavioral Intention to Use (BI)
Actual System Use

Figure 2-1: Graphical representation of the Technology Adoption Model (TAM). Adopted from Davis (Davis, 1989).

The model also shows that behavioural intention and attitude towards technology are driven by the "*perceived usefulness*" and "*perceived ease of use*" of that technology (Masrom, 2007). Furthermore, one's behaviour and the intent to behave are related to one's attitude (Masrom, 2007). As such a user's attitude refers to the user's appraisement towards a particular technology whereas behavioural intention refers to the likelihood that a person will use that technology (Masrom, 2007; Abu-Dalbouh, 2013). The value of TAM lies in its parsimony and is often the model of choice when research costs and outcomes must be considered (Porter & Donthu, 2006; Lin, 2013).

Critics of the TAM suggest that the narrow focus of adoption concepts hinders the researcher from identifying other drivers of adoption (Luo, Warkentin & Li, 2013). Furthermore, Bagozzi criticise TAM for its deterministic presupposes that once there is '*perceived usefulness'* and *'perceived ease of use'* that intension will be formed leading to adoption (Bagozzi, 2007; Tobbin, 2012). These presuppose neglect social, group and cultural decision making aspects (Tobbin, 2012). As a result, a number of additions have been made to the TAM model over the years. Critics argue that these additions have acted as barriers and created the illusion of knowledge accumulation and prevented the fruitful extension of the model backwards towards information technology implementations (Benbasat & Barki, 2007).

Fred Davis developed the standardized questionnaire that is used to measure technology acceptance. This questionnaire primarily consists of 2 parts with 10 items to measure usefulness and 10 items to measure ease of use (Davis, 1989). These items are shown in the table below (Davis, 1989):

Usefulness	**Perceived ease of use**
Effectiveness	Controllable
Job performance	Cumbersome
Quality of work	Frustrating
Increase productivity	Understandable
Critical to my job	Mental effort
Accomplish more work	Rigid and Inflexible
Work more quickly	Ease of remembering
Make job easier	Ease of learning
Control over work	Effort to be skilful
Useful	Easy to use

Figure 2-2: Table showing the items of usefulness and ease of use. Adopted from Davis (Davis, 1989).

2.4 The Theory of Planned Behaviour (TPB)

Explaining human behaviour is a difficult complex task. Various theoretical frameworks have been proposed to deal with the psychological process involved (Ajzen, 1991). TPB is an extension of the theory of reasoned action and was proposed to deal with the limitations of the model (Ajzen, 1991).

A central factor in the TBP is an individual's intention to perform a given behaviour (Ajzen, 1991). TBP proposes that intentions are motivational factors that influence behaviour. According to Ajzen the TPB is represented by the following variables (Ajzen, 1991):

- **Subjective norms**: Refers to the social pressure that encourages an individual to engage in a particular type of behaviour.
- **Attitudes**: Refers to the degree to which an individual has a positive or negative appraisal of a behaviour.
- **Perceived behavioural control**: Refers to the extent to which an individual feels able to enact the behaviour of interest. Perceived behavioural control varies across situations.

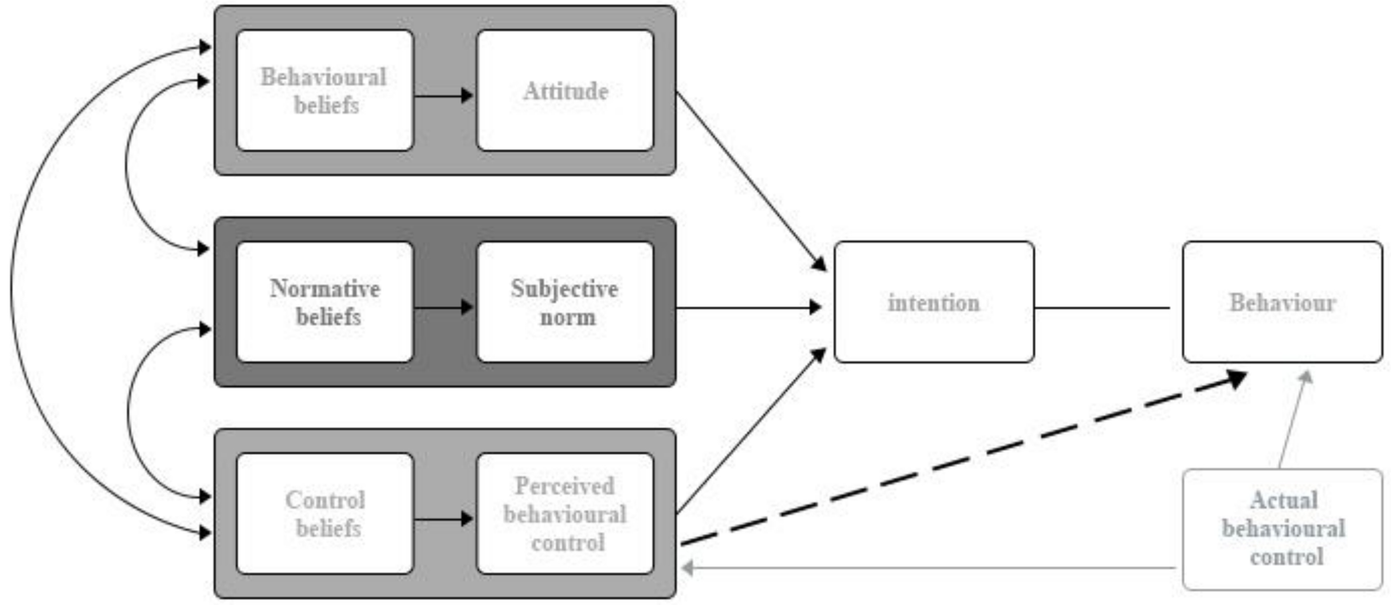


Figure 2-3: Graphical representation of the Theory of Planned Behaviour. Adopted from Ajzen (Ajzen, 1991).

The relationship between subjective norm, attitude and perceived behavioural control varies from situation to situation. However, as a general rule a favourable attitude to a specific behaviour together with a positive subjective norm causes greater behavioural control and a stronger intention to perform the behaviour under consideration (Ajzen, 1991).

Critics of the TPB model point out that the relationship between attitude, subjective norm and perceived behaviour control and the belief structure are not essentially well understood (Suryaningrum, 2012). Furthermore, there are no absolutes in trying to predict social behaviour and the premise that behavioural intention leads to actual behaviour is not necessarily true (Armitage & Conner, 2001; Suryaningrum, 2012).

To design a questionnaire using the TPB one would need to evaluate each of the theory's major constructs. This can typically be done using a seven-polar bipolar adjective scale to measure (Ajzen, 2006):

- Attitude,
- Perceived norm,
- Perceived behavioural control and
- Intention.

2.5 The Unified Theory of Acceptance and Use of Technology (UTAUT)

The UTAUT is the newest information system theory that proposes to give explanations about end users acceptance behaviour (Taiwo & Downe, 2013). Furthermore, it is believed to be more robust in predicting and evaluating technology acceptance.

UTAUT was developed as a combination of eight prominent information technology acceptance models. The UTAUT uses four core determinants, which are believed to play a significant role in user acceptance and behaviour (Venkatesh, Morris, Davis & Davis, 2003; Taiwo & Downe, 2013):

- **Performance expectancy**: Refers to the degree to which technology will provide benefits to an individual (Venkatesh, Thong & Xu, 2012).
- **Effort expectancy**: Refers to the degree of ease of use of a technology (Venkatesh *et al.*, 2012).
- **Social influence**: Refers to the social pressures that individuals consider as important influences to the adoption of an innovation (Venkatesh *et al.*, 2012).
- **Facilitating conditions**: Refers to the degree to which an individual believes that the technology infrastructure in an organisation exists to support the use of the system (Venkatesh *et al.*, 2012).

Furthermore, each of the core determinants is theorised to be moderated by the following four variables. These variables are factors that influence “use behaviour” or “behaviour intention” and are listed below (Venkatesh *et al.*, 2003; Taiwo & Downe, 2013):

- Gender
- Age
- Experience
- Voluntariness

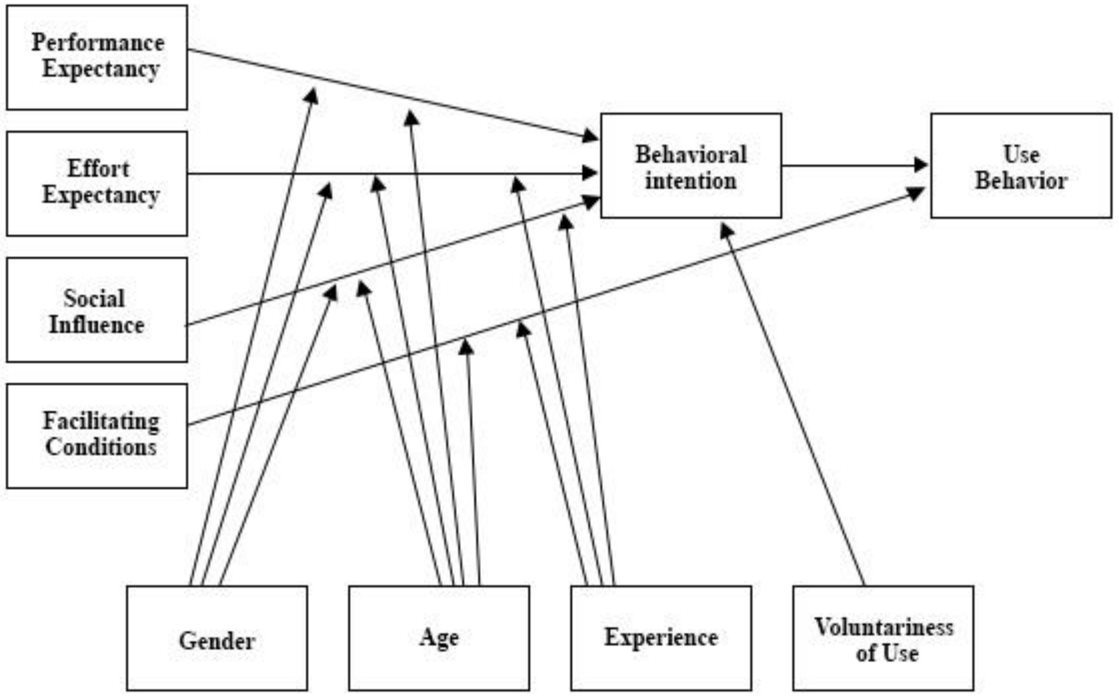


Figure 2-4: Graphical representation of the Unified theory of Acceptance and use of Technology. Adopted from Venkatesh *et al.*, (Venkatesh *et al.*, 2003).

According to Bagozzi UTAUT presents a model that has led to a state of chaos because of the number of independent variables (Bagozzi, 2007). Although UTAUT has been cited by a large number of studies very few have implement the full model probing all of its constructs (Akbar, 2013). As such additional testing of the model is needed to further support its validity (Akbar, 2013).

Questionnaires for UTAUT can be designed by evaluating the major constructs of the theory (Venkatesh *et al.*, 2003). Probing questions could be used to evaluate the following areas:

- Performance Expectancy
- Effort Expectancy
- Social Influence
- Facilitating Conditions

2.6 Diffusion of Innovation (DOI)

DOI is another one of the most commonly used theories that attempt to understanding how and at what rate innovations are adopted in different sectors (Archibald & Clark, 2014). Everett Rogers popularized the theory in his book published in 1962. Everett wanted to determine why certain innovations in the farming industry were adopted and others were not. Everett theorized that other factors and possibly economic conditions were at work (Rogers, 2003).

According to Al- Jabri & Sohail the DOI refers to the adoption of innovation over time into the social system (Al-Jabri & Sohail, 2012). When new ideas are invented they are diffused and can either be accepted or rejected, which inevitably leads to social change (Rogers, 2003).
As such the diffusion of innovation theory has four characteristics (Rogers, 2003; Al-Jabri & Sohail, 2012):

- **Innovation**: Refers to an idea or object that is perceived as new.

- **Communication**: Refers to the process by which individuals or organisations share information about an innovation.
- **Time**: Time is involved in the DIT in three ways:
    - **Innovation-decision**: Refers to the duration between the first knowledge of an innovation to its adoption or rejection. This consist of five steps:
        - Knowledge
        - Persuasion
        - Decision
        - Implementation
        - Confirmation
    - **Innovativeness**: The degree to which an organisation is innovative in its adoption process. Everett defines five categories:
        - Innovators
        - Early Adopters
        - Early Majority
        - Late Majority
        - Laggards
- **Rate of adoption**: Refers to the rate at which an innovation is adopted. The following characteristics determines an innovations rate of adoption:
    - Relative advantage
    - Compatibility
    - Complexity
    - Trialability
    - Observability
- **Social System**: Ultimately diffusion of innovation occurs within a social system. As such a social system refers to a set of individuals, organisations or groups that are involved in joint problem solving

or reaching a common goal. The social system consists of change agents who are either responsible or influence innovative change. Finally, innovations in social systems can be adopted or rejected by individuals or the entire social system. The decision to adopt or reject innovations fall into three categories:

- Option innovative decisions
- Collective innovative-decisions
- Authority innovative-decisions

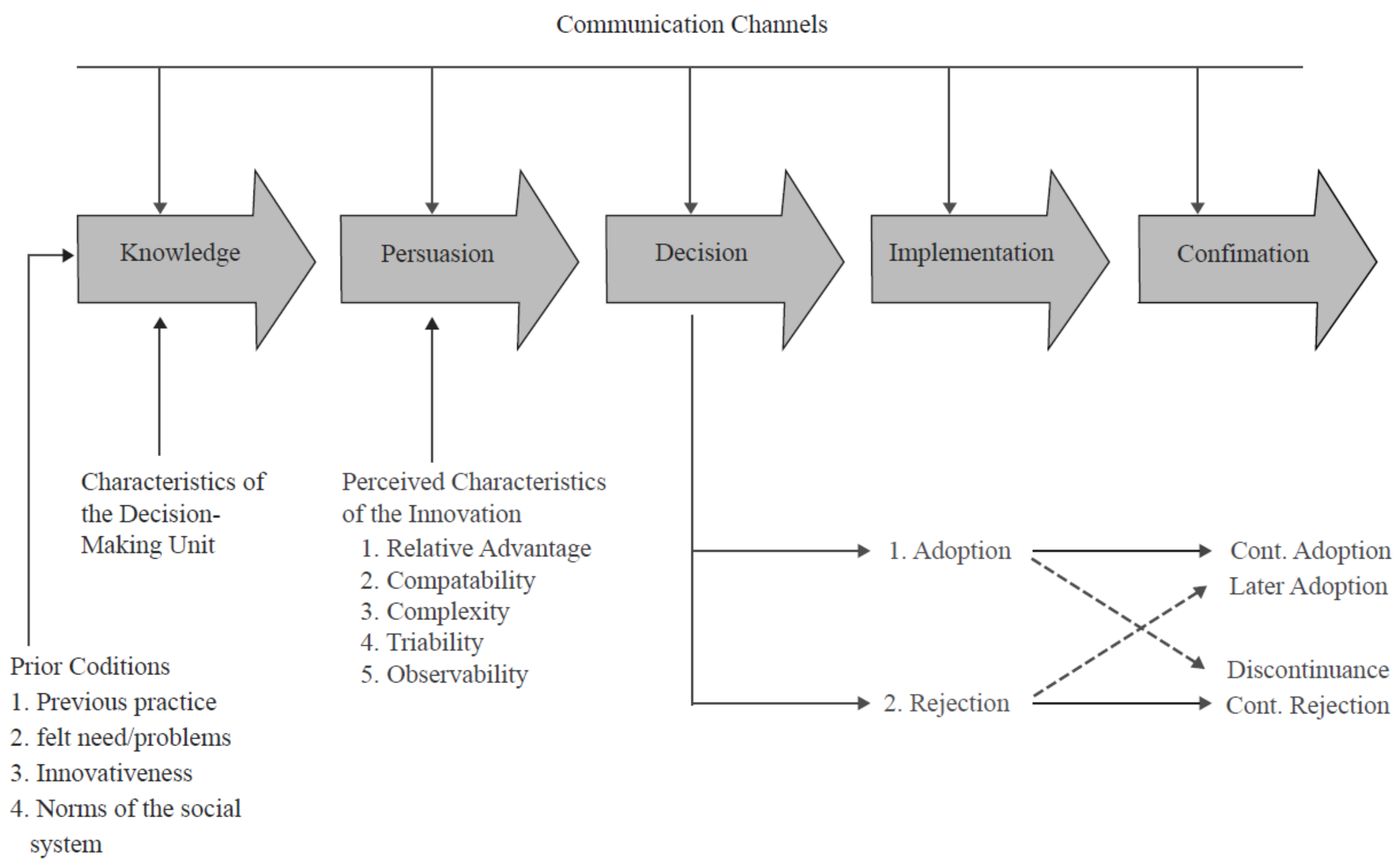


Figure 2-5: Graphical representation of the Diffusion of Innovation model. Adopted from Rogers (Rogers, 2003).

The DOI is not without criticism, as this is evident in the study by Lambkin and Day who suggest that the DOI model is aligned with demand side economics cycle (Lambkin & Day, 1989). Arguing that Rodgers model ignores the dynamics of competition, competitive advantage, resource allocation and the speed of diffusion in line with the product life cycle (Lambkin & Day, 1989). According to Gross the application of DOI in developing countries has undesirable consequences (Stephenson, 2003):

- The DOI proposes that after adoption the benefits spread and become homogeneous. However, experience in Latin America showed the gap in inequities widening.
- Non-adopters are also affected by DOI as bigger farmers increase production as a result of adopting an innovation. This resulted in a decrease in price amongst all farmers.

Questionnaires for DOI can be designed by evaluating the major constructs of the theory (Sonnenwald, Maglaughlin & Whitton, 2001). This is typically done by asking questions in the following areas:

- Relative advantage,
- Compatibility,
- Trialability and
- Observability

2.7 Gartner Hype Cycle (GHC)

The Gartner hype cycle is used to characterise the over-enthusiasm and subsequent disappointment that typically happens with the introduction of new technologies (Fenn & Raskino, 2008). The cycle also shows how technologies move beyond the hype, by offering benefits and becoming widely adopted (Fenn & Raskino, 2008). The diagram below shows the “hype” phases associated with the theory:

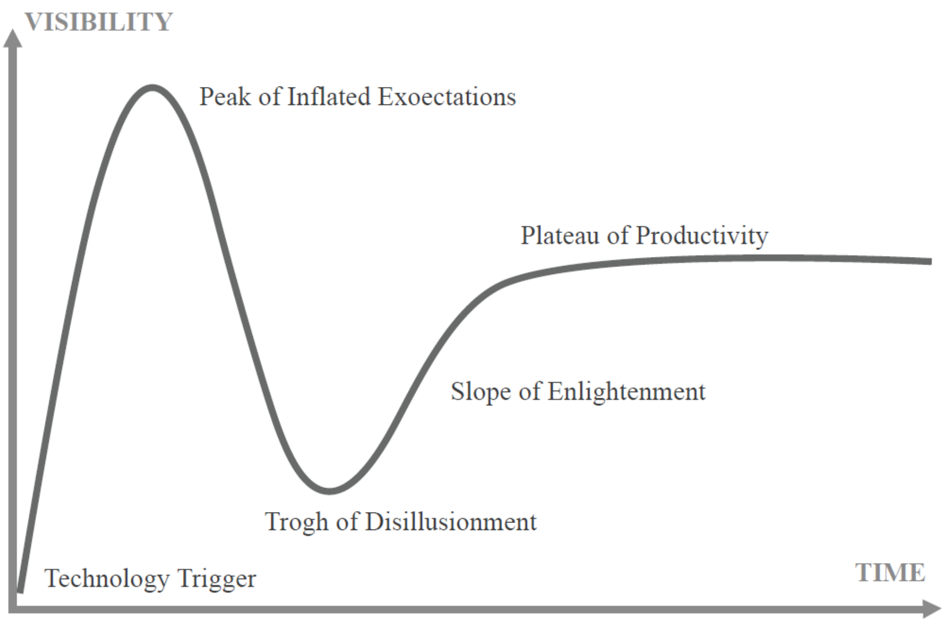


Figure 2-6: Graphical representation of the Gartner hype cycle. Adopted from Fenn & Raskino (Fenn & Raskino, 2008).

The s-curve of hype is based on a sudden irrational reaction to the introduction of new technology and is characterised by five stages (Fenn & Raskino, 2008; Steinert & Leifer, 2010)**:**

- **Technology trigger**: The launching of an innovation signals the trigger.
- **Peak of inflated expectations**: The point at which an innovation is receiving attention by early adopters and the media.
- **Trough of disillusionment**: Unfavourable stories about the use of an innovation emerge.
- **Slope of enlightenment**: Early adopters find new ways to use the innovation.
- **Plateau of productivity**: Benefits of an innovation are accepted.

Steinert & Leifer suggest that there is a need to setup the hype cycle model in a mathematical quantitative way as the theoretical foundation is empirically flawed (Steinert & Leifer, 2010). This theory is further supported by the fact that Gartner has not clearly defined the framework and its measuring variables (Steinert & Leifer, 2010). Furthermore, the framework is relatively new with a consultative background and as such the model has not been picked up systematically by academics (Steinert & Leifer, 2010).

Data for the GHC is primarily derived from real world statistics essentially by looking at demands and trends related to a specific technology (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

2.8 TECHNOLOGY, ORGANISATION AND ENVIRONMENT FRAMEWORK (TOE)

The technology, organisation and environment (TOE) framework provides a useful framework to study the adoption of different types of information technology innovation (Oliveira & Martins, 2011). The framework has a strong theoretical basis and is described in the figure below (Oliveira & Martins, 2011).

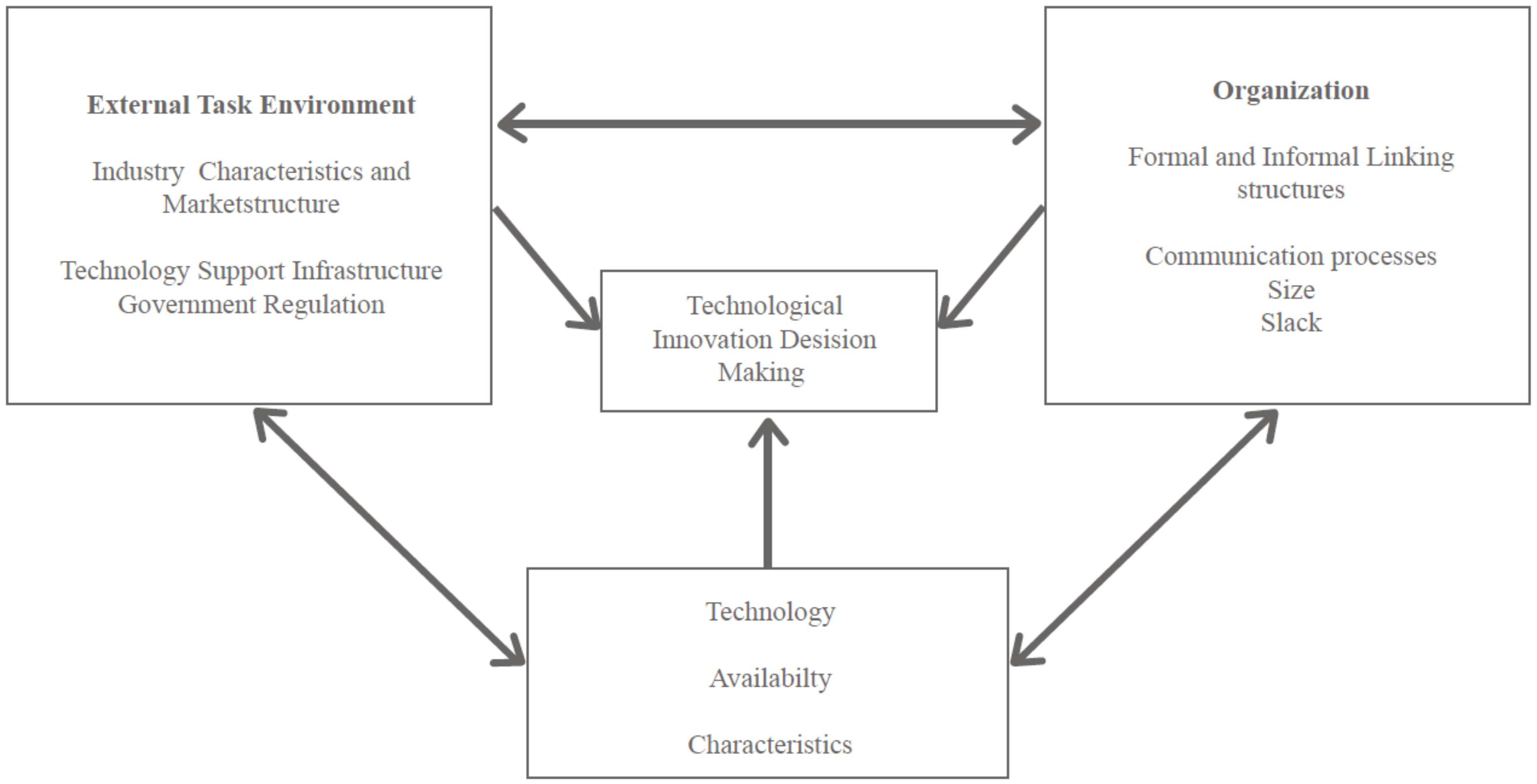

External Task Environment
Industry Characteristics and Marketstructure
Technology Support Infrastructure
Government Regulation
Organization
Formal and Informal Linking structures
Communication processes
Size
Slack
Technological Innovation Desision Making
Technology
Availabilty
Characteristics

Figure 2-7: Graphical representation the Technology, organisation and environment framework. Adopted from Tornatzky & Fleischer and Oliveira & Martins (Tornatzky & Fleischer, 1990; Oliveira & Martins, 2011).

According to Tornatzky & Fleischer three elements influence the decision within an organisation to adopt technology (Tornatzky & Fleischer, 1990; Oliveira & Martins, 2011; Angeles, 2014):

- **Environmental Context**: Deals with the arena of stakeholders that surround a firm and their influence on how a firm interprets the need for innovation.
- **Organisational Context**: A range of characteristics define the organisational context. These include the complexity of its managerial structure, its size, the amount of resources it has and a company's formal and informal linkages.
- **Technological Context:** Deals with both the internal and external technologies that might be useful in improving productivity to a firm.

The TOE framework proposes the influences of the above three contexts on the decision to adopt (Nkhoma & Dang, 2013) and as such offers a holistic approach by looking at the various contexts within an organisation (Nkhoma & Dang, 2013). While this holistic approach is a major benefit of this model it is also a point of criticism as critics have raised the issue that the contexts of the model are characteristic of large organisations (Awa, Ojiabo & Emecheta, 2015).

Questionnaires for TOE can be designed by evaluating the major constructs of the theory (Angeles, 2014). These are typically done by asking probing questions in the following areas:

- Environmental Context,
- Organisational Context and
- Technological Context

2.9 Comparison

In this chapter, the researcher discussed the various theoretical frameworks that are considered suitable for this study. This section provides a matrix to evaluate and compare these theoretical frameworks. The columns of the matrix represent the criteria used to evaluate the frameworks while the rows of the matrix represent the respective frameworks.

Model	**Suitable for adoption studies**	**Applies to companies**	**Applies to individuals**	**Used to determine when to adopt**	**Examine Environment**	**Examine Behaviour**	**Adoption Benefits**	**Adoption Risks**	**Social Influence**	**Used in research on Software**	**Suitability to Questionnaire**
TAM	X		X			X					
TPB	X		X			X			X		
UTAUT	X		X			X			X		
DOI	X	X		X					X	X	
TOE	X	X		X	X				X		
GHC	X	X		X			X	X	X	X	X

Figure 2-8: A table showing the comparative differences between the different theoretical frameworks.

Taken at face value the theoretical models reviewed in this chapter are suitable for adoption based research studies. However, the TAM, TPB and UTAUT models apply to individuals adopting technology whereas the DOI, TOE and GHC apply to organisations intending to adopt technology (Oliveira & Martins, 2011).

Furthermore, DOI, TOE and GHC are used by organisations to determine when to adopt a particular innovation whereas the TAM, TPB and UTAUT examine an individual's behaviour that ultimately leads to adoption. The TOE is the only model that examines environmental factors that affect adoption. Similarly, the GHC is the only model that examines the benefits and risk associated with the adoption of a technology. With the exception of the TAM model all of the models reviewed in this chapter consider social influences.

After evaluating the questionnaire for this study and matching the questions to the respective theoretical models it was found that the GHC is the only model that is suitably aligned. The other models require specific constructs and probing questions to be included in the questionnaire.

With the above in mind three models are suitable for this study. These models are the DOI, TOE and GHC as all three of these models deal with adoption at organisation level. The DOI and GHC models have also been successfully used by researchers in previous studies on Agile software development (Janes & Succi, 2012; Overhage & Schlauderer, 2012).
The inclusion of adoption benefits, compliance to the questionnaire as well as risk addressed by the GHC is of particular importance to this study. Furthermore, the GHC is popular amongst large firms when considering adoption strategies (Steinert & Leifer, 2010). This is in alignment with the outcomes of this research to assist organisation planning to make adoption decisions about Agile methods. Although the GHC has its fair share of criti-

cism. The strong characteristic and alignment of the model make it appropriate for this study and as such has been chosen as the underpinning theoretical framework.

2.10 Chapter Summary

This chapter discusses the theoretical framework that underpins this research. The chapter begins by identifying six adoption related information system theoretical frameworks:

- Technology acceptance model (TAM)
- The theory of planned behaviour (TPB)
- The unified theory of acceptance and use of technology (UTAUT)
- The diffusion of innovation (DOI) theory
- The technology, organisation and environment framework (TOE)
- The Gartner hype cycle (GHC)

Each of the above theoretical frameworks are summarised in this chapter. Finally, the chapter concludes with a comparative study and the selection of the underpinning framework for this study.

Chapter 3

LITERATURE REVIEW

3.1 INTRODUCTION

As discussed in chapter 1 of this study a concept matrix is a powerful tool that enables the researcher to align the literature to the research topic (Klopper & Lubbe, 2012). After thoroughly investigating the literature a concept matrix was designed for this study and provides the underpinning structure for this chapter (Refer to addendum 13 for the complete concept matrix).

3.2 SOFTWARE ENGINEERING

Post the 2000 era, the context in which software is conceived, created and maintained has evolved rapidly and significantly. Furthermore, software has grown in complexity and size while developers and users have placed more emphasis on speed, quality, ease of use and time to market (Clutterbuck, Rowlands & Seamons, 2009; Stavru, 2014). Software development companies have begun to release products more frequently, placing significant emphasis on flexible development methods (Clutterbuck *et al.*, 2009; Murphy, Bird, Zimmermann, Williams, Nagappan & Begel, 2013).

Numerous software development methodologies have emerged over the last decade and this had led to fierce debates amongst software developers who classify these methodologies into two groups (Jiang & Eberlein, 2009; Chuang, Luor & Lu, 2014).

1) **Traditional Methodologies:** Often referred to as heavy weight methodologies and are based upon the original engineering paradigm, which assumes that software design follows sequential thinking (Jiang & Eberlein, 2009; Mnkandla, 2009). According to Fowler

and Williams, the inspiration for traditional methodologies comes from disciplines such as mechanical and civil engineering (Fowler, 2005; Williams, 2007a). These disciplines place emphasis on planning and design before building (Fowler, 2005).

2) **Agile Methodologies:** Often referred to as light weight methodologies and are in contrast to traditional methodologies. Agile methodologies address the unpredictability of the world by relying on "people and their creativity rather than process" (Beck *et al.*, 2001a; Dybå & Dingsøyr, 2008; Jiang & Eberlein, 2009).

According to Vijayasarathy & Turk traditional development emphasizes extensive analysis, creation and the maintenance of models and long durations between milestones with small amounts of user involvement (Vijayasarathy & Turk, 2012).

Literature on the difference between Agile and traditional methodologies point to the fact that traditional methodologies assume that customers know their requirements. While Agile methodologies assume that the customers do not have a full understanding of their requirements (Ionel, 2009).

Furthermore, one of the biggest challenges associated with traditional methodologies is that software development is not a defined process as numerous changes occur over time. Many software development projects experience changes in requirements, scope and technology that are beyond the control of the development team (Highsmith & Cockburn, 2001; Williams, 2007a; Aken, 2008). As such it is highly unlikely that a plan with a set of predefined steps would lead to a desirable outcome because of the unpredictable nature of software development (Highsmith & Cockburn, 2001; Williams & Cockburn, 2003; Fowler, 2005; Cao & Ramesh, 2008; Leau *et al.*, 2012).

Traditional methodologies are widely used and typically perform well when technology is feasible, requirements are predictable and plans are irrevocable. However, such characteristics in software development projects are rare (Vinekar, Slinkman & Nerur, 2006; Chuang *et al.*, 2014).

Another problem with the traditional approach is that the cost associated with software rework can significantly increase by factors of 50 to 200 during the various phases of the software development lifecycle (Boehm & Papaccio, 1988; Highsmith & Cockburn, 2001).

Traditional methodologies try to reduce software rework through process refinement, continuous monitoring and error correction. Advocates believe that if a team tries hard enough they can anticipate a complete set of requirements early during the software development life cycle (Highsmith & Cockburn, 2001). The differences between Agile and traditional methods are highlighted in the figure below:

Criteria	**Agile**	**Traditional**
User requirements	Iterative acquisition	Detailed user requirements are well defined before coding
Rework cost	Low	High
Development Direction	Readily changeable	Fixed
Testing	On every iteration	After coding phase
Customer Involvement	High	Low
Extra quality required for developers	Interpersonal skills and basic business knowledge	Nothing in particular

Criteria	Agile	Traditional
Suitable project scale	Low to medium-scaled	Large-scale

Figure 3-1: A table providing an overview of Agile and traditional methods. Adopted from Leau *et al.*, (Leau *et al.*, 2012).

3.3 AGILE METHODOLOGIES

In the 1990's software development practitioners found that the rate of change in software requirements was beyond the capabilities of traditional methodologies (Williams, 2007a). To address this problem a group of 17 programmers, called the Agile Alliance, got together in 2001 and developed the outlining principles of an alternate software development methodology in what is called the Agile Manifesto (Beck *et al.*, 2001a).

"We are uncovering better ways of developing software by doing it and helping others to 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; and 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" (Beck *et al.*, 2001a: 1).

Supplementing the Manifesto, the Agile Alliance detailed 12 principles of Agile Software development. These principles are listed below (Beck, Beedle, Bennekum, Cockburn, Cunningham, Fowler, Grenning, Highsmith, Hunt, Jeffries, Kern, Marick, Martin, Mellor, Schwaber, Sutherland & Thomas, 2001b: 1):

- "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."

- “Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.”
- “Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.”
- “Business people and developers must work together daily throughout the project.”
- “Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.”
- “The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.”
- “Working software is the primary measure of progress.”
- “Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.”
- “Continuous attention to technical excellence and good design enhances agility.”
- “Simplicity--the art of maximizing the amount of work not done--is essential.”
- “The best architectures, requirements, and designs emerge from self-organizing teams.” (sic)
- “At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.”

The roots of Agile can be tracked back to 1991 when the term “lean development” was coined by the manufacturing industry. The aim was to deliver as fast as possible while amplifying learning and eliminating waste (Dybå & Dingsøyr, 2008; Jiang & Eberlein, 2009). Later on Youssef conceived the term “Agile manufacturing”. It is thus paradoxical that Agile has its roots in engineering discipline (Youssef, 1992).

The literature shows that software development principles and practices are still maturing and a clear definition for Agile methodologies is yet to emerge:

Duka describes Agile software development as an evolutionary approach that is disciplined, highly collaborative and quality-focused, whereby working software is produced at regular intervals (Duka, 2012).

Mirnalini & Venkata defines Agile as "The use of continuous stakeholder feedback to produce high quality consumable code through use cases and a series of short time-boxed iterations" (Mirnalini & Raya, 2010: VI–242).

Ahmed, Ahmad, Ehsan, Mirza & Sarwar state that Agile practices promote self-organizing teams, quality and customer collaboration while reduce documentation and time to market (Ahmed, Ahmad, Ehsan, Mirza & Sarwar, 2010).

Vijayasarathy & Turk suggest that Agile development focuses on early and fast working code, small frequent incremental changes, short iterations, pair programming with rapid and continual user interaction (Vijayasarathy & Turk, 2012).

Erickson, Lyytinen & Siau describe agility as a means to strip away the heaviness associated with traditional methodologies and to promote quick responses to changing environments, user requirements and accelerated deadlines (Erickson, Lyytinen & Siau, 2005).

3.4 Popularity of Agile Methods

Many industry surveys have been administered to study the popularity of Agile methods. These surveys generally try to evaluate Agile:

1) In terms of the number of organisations or professionals adopting or planning to adopt Agile.
2) In terms of the number of organisations or professionals using specific Agile methods.

3) In terms of empirical studies on the use of Agile methodologies.

According to Stavru majority of the surveys are being administered by vendors, consultants and independent research organisations and this raises questions about the scientific methods followed when administering these surveys (Stavru, 2014).

In 2001 the “Clutter Consortium” conducted a survey of nearly 200 people across America and Europe. The survey found that many organisations were using at least one Agile methodology. Furthermore, they also found that Agile projects showed slightly better performance in terms of delivery when compared to traditional projects with increased employee morale (Cockburn & Highsmith, 2001).

A survey conducted by Rumpe & Schröder on 45 software development projects that were using Extreme Programming (XP) showed the following (Rumpe & Schröder, 2002):

- All respondents said that they would like to use XP again
- Almost all of the projects were successful
- The absence of the customer was the highest risk identified

In 2005 a survey in US and Europe revealed that 14% of companies were using Agile methods and 49% were aware of Agile (Dybâ & Dingsøyr, 2009). A survey by Dr Dobb’s Journal showed that in 2009 76% of the respondents had reportedly adopted Agile and a similar survey in 2010 by Agile Journal showed that 88% of companies had adopted Agile (Duka, 2012).

In 2010 Forester announced that Agile software development had become mainstream (Bustard, Wilkie & Greer, 2013). The “State of Agile” survey that is run annually by VersionOne showed that in 2011 80% of respondents were familiar with Agile. This figure increased in 2013 to 88%. Furthermore, 39% of respondents reported that the number of Agile projects developed by their respective companies were between 50% and 100%. In 2013 this figure increased to 52% (VersionOne, 2011, 2013).

A study by Stavru in 2014 which compared 3 case studies between 2011 and 2012 on the most common Agile method found that between 42% to 83.1% of the respondents were using SCRUM (Stavru, 2014). These figures are in line with the study by VersionOne as well as Paasivaara and Lassenius, which also identifies SCRUM as the most widely used Agile practice (Paasivaara & Lassenius, 2003; VersionOne, 2011). The VersionOne study shows that 52% of respondents used the framework in 2011 and 54% in 2013 respectively (VersionOne, 2011, 2013).

Empirical studies published by Dingsøyr & Dybå and Ionel call for both an increase in the quality and number of studies on Agile (Dybå & Dingsøyr, 2008; Ionel, 2009). Their review also found that most studies focused on Extreme Programming rather than Scrum. However, evidence has shown that Scrum has become popular.

3.5 Benefits Associated with Agile Adoption

Studies by Poole, Murphy, Huisman & Higgins, Maurer & Martel, Abrahamsson, Layman, Williams & Cunningham, Maurer, Melis & Marchesi, Sharp & Robinson and Dingsøyr & Dybå, show an increase in productivity when using Agile or XP practices (Poole, Murphy, Huisman & Higgins, 2001; Maurer & Martel, 2002; Abrahamsson, 2003; Layman, Williams & Cunningham, 2004; Mannaro, Melis & Marchesi, 2004; Sharp & Robinson, 2004; Dybå & Dingsøyr, 2008). Furthermore the productivity increase in Poole et al., was also attributed to an increase in moral resulting from paired programming (Poole *et al.*, 2001).

A case study at IBM spanning a period of 1 year on a small team (7- 11 team members) showed an improvement in productivity and a 40% reduction in pre-release defect density when compared to the same metrics from an earlier release (Layman *et al.*, 2004).

Bedoll studied a case of infinite productivity involving a team of 20 developers employing Boeings standard development methodology (Bedoll, 2003). The team's initial release was scrapped after two months of trials. However, a second attempt by a two-person team that employed practices similar to Extreme Programming was able to deliver a working product in only six weeks.

Most studies report predominately positive evidence that Agile practices have a number of adoption benefits. These benefits include improved software quality, improved quality of the development process, reduction in defects, reduce time to market and documentation, better customer collaboration, shared learning, improved communication and productivity, better predictability and increased transparency (Abrahamsson, 2003; Nerur *et al.*, 2005; Vinekar *et al.*, 2006; Schwaber *et al.*, 2007; Dybå & Dingsøyr, 2008; Laanti *et al.*, 2011; Senapathi & Srinivasan, 2011; Sriram & Mathew, 2012; Murphy *et al.*, 2013). The benefits of Agile methods are described in the figure below:

Feature	**Benefits**
Continuous requirements gathering	Provides flexibility by allowing customers to delay crucial decisions.
Frequent face-to-face interactions	Building trust and overcoming misunderstanding amongst team members
Pair programming	Improves code ownership and promotes teamwork.

Feature	**Benefits**
Refactoring	Progressive improvement of code without creating shock waves.
Continuous release and integration	Supports early detection and correction of bugs. Resulting in higher quality software.
Early expert customer feedback	Reduces costly code overhauls in the end. Also lowers the cost of development.
Reduced documentation	Lowers the cost of documentation. Resulting in shorter development time.

Figure 3-2: A table showing the key features/Benefits of Agile Methods. Adopted from Blose (Blose, 2008).

Schatz & Abdelshafi reports on a case study at Primavera Systems. The study found that their software development team was able to increase quality by 30% in the first nine months of Agile adoption (Schatz & Abdelshafi, 2005).

Another study by Auvinen, Back, Heidenberg, Hirkman & Milovanov of a pilot project at Ericsson, the mobile phone supplier, shows a 5.5% decrease in defects (Auvinen, Back, Heidenberg, Hirkman & Milovanov, 2005). The study measured code defects through the use of formal code reviews. Furthermore, a study by Korhonen reports an increase in early detection of defects by 57.5% during the first six months and 65% in 12 months (Korhonen, 2013). The study was conducted on 150 globally distributed experts.

A study of 40 software development companies in Northern Ireland in 2012 concluded that Agile methods were replacing waterfall methods. Furthermore, the study found that there were perceived benefits in software quality as well as the processes used to create software. These benefits include developer productivity, testing, coding, maintenance, detailed design and architectural design process enhancements (Bustard *et al.*, 2013).

3.6 AGILE ADOPTION BARRIERS

While most studies report that Agile practices are easy to adopt and work well, there are also compelling evidence on the challenges associated with Agile adoption. The challenges include the lack of attention in design and architecture issue, the perceived inability of Agile to scale, low levels of test coverage, failure to understand Agile practices and the lack of documentation (Cao *et al.*, 2009; Mazni, Syed-Abdullah & Yasin, 2011; Murphy *et al.*, 2013).

The Agile wave cannot be ignored by organisation; however, for those organisations that are steeped in tradition software development methods, the adoption of Agile will likely pose several challenges (Nerur *et al.*, 2005; Vinekar *et al.*, 2006). Furthermore, software development changes has complex organisational wide ramifications and can't be accomplished by merely replacing one technology with another (Nerur *et al.*, 2005; Vinekar *et al.*, 2006). The differences in project management components between Agile and traditional methods are described in the figure below:

Project Component	**Traditional**	**Agile**
Control	Process centric	People Centric
Management Style	Explicit	Tacit
Role Assignment	Individual – favours specialisation.	Self-organizing teams – encourages role interchangeability.
Communication	Formal and only when necessary.	Informal and continuous.

Project Component	**Traditional**	**Agile**
Project Cycle	Guided by task or activities.	Guided by product features.
Development Model	Life cycle model (Waterfall, Spiral or some variation).	The evolutionary-delivery model.
Desired Organisational Formal/Structure	Mechanistic (bureaucratic with high formalisation)	Organic (flexible and participative encouraging corporative social action)
Technology	No restriction	Favours object-oriented technology
Team Location	Predominately distributed	Predominately collocated
Team Size	Often greater than 10	Usually less than 10
Continuous Learning	Not frequently encouraged	Embraced
Management Culture	Command and Control	Responsive
Team Participation	Not compulsory	Necessary
Project Planning	Up-front	Continuous
Feedback Mechanisms	Not easily obtainable	Usually numerous available
Documentation	Substantial	Minimal

Figure 3-3: A table showing the difference between traditional and Agile methods. Adopted from Rehan, Assudani, Shrivastav & Deshmukh (Rehan, Assudani, Shrivastav & Deshmukh, 2014).

With the above in mind it must be noted that change in software development methodologies will impact different aspects of an organisation including its culture and management practices (Vinekar *et al.*, 2006). Furthermore, research shows that Agile adoption can have an impact on (Vinekar *et al.*, 2006):

1) Organisation Culture
2) Management Style
3) Organisation Knowledge
4) Development Team
5) Large Organisations

3.6.1 Organisation Culture

Changes in organisation culture are often regarded as extremely difficult to achieve (Nerur *et al.*, 2005). The culture of an organisation exerts considerable influence on the problem-solving, decision making, innovation as well as relationships within that organisation (Nerur *et al.*, 2005; Chandra Misra, Kumar & Kumar, 2010). The adoption of Agile may require radical modification in order to fit into this context (Laanti *et al.*, 2011). It has also been reported that organisations are not aware of what Agile adoption really means or organisations do not understand the ramifications of Agile adoption (Chandra Misra *et al.*, 2010).

Organisations that have centralized and deep hierarchical decision-making structures may be at conflict with Agile methods and this may in turn cause conflict between management and team members (Boehm & Turner, 2003). Furthermore, Agile methods seem to polarize stakeholders into groups of proponents and opponents each having different standpoints regarding the usefulness of Agile methods (Laanti *et al.*, 2011).To overcome the polarization barrier a change of philosophy and behaviour is required across the organisation. Furthermore, studies have shown that the appreciation of Agile methods increase after adoption (Laanti *et al.*, 2011).

Organisations that are non-customer centric will need to move to a customer centric role. Unlike in the traditional approach where the customer involvement is usually greatest at the beginning and at the end of the project, Agile methods instead require more frequent customer involvement (Coram & Bohner, 2005). Although this practice can significantly reduce the risk associated with software development, customers with large complex systems and customers that are unavailable or unable to commit to a project can pose challenges (Cao *et al.*, 2009).

Studies have also shown that training and coaching is a positive motivation factor towards Agile adoption (Schatz & Abdelshafi, 2005; Atlas, 2009; Vijayasarathy & Turk, 2012). A study by Benefield at Yahoo concedes that Yahoo would have faced less resistance had they educated management on the values of Agile methods early in the adoption process (Benefield, 2008). While Atlas suggests that 95% of the teams that asked for help at Amazon had not been coached (Atlas, 2009).

The success of many Agile activities is dependent on the attitudes of people in the team towards one another (Chandra Misra *et al.*, 2010). Practices like shared decision making, pair programming and shared learning can be influenced by team attitude (Chandra Misra *et al.*, 2010). In traditional organisations developers often practice solitary programming activities (Chandra Misra *et al.*, 2010). These habits have been inculcated over many years and can be restrictive to the adoption of Agile methods (Chandra Misra *et al.*, 2010).

Organisation culture and momentum play an important role in Agile adoption. According to Vijayasarathy & Turk, "unless people see a reason to do something other than what they are familiar with and experienced at, they will probably continue to do those same things" (Vijayasarathy & Turk, 2012: 145). As such management needs to publicize the benefits of Agile adoption in order to overcome the status quo (Vijayasarathy & Turk, 2012).

Furthermore, it has also been found that it is harder to facilitate Agile adoption in larger organisations than in smaller organisations. Studies have shown that the more experienced the developer is, the more resistant he/she may be towards Agile adoption. Smaller organisations are also readily able to apply new technologies and ideas because they are not constrained by organisation culture and history or momentum (Vijayasarathy & Turk, 2012).

3.6.2 Management Style

Traditional organisations planning to adopt Agile methods require a change in management style from *"command and control"* to *"leadership and collaboration"* (Pichler, 2006). The traditional role of the project manager is well formed, well defined and document centric (Chandra Misra *et al.*, 2010). However, these roles must be altered to that of a facilitator who co-ordinates and directs the efforts of the development team (Nerur *et al.*, 2005).

Schedules and plans are less important in Agile methods as the emphasis is more on responding to change rather than following a plan (Coram & Bohner, 2005). Managers need to recognise the challenges associated with adoption of Agile methods and tailor specific Agile tenets to the organisation culture. Agile methods may need to be balanced between the needs of top management, organisation culture and the culture of the development team (Cao *et al.*, 2009).

Decision making in an Agile environment can be difficult when compared to a traditional environment. It may take an organisation time, effort and patience in order to build a culture of trust amongst its employees (Nerur *et al.*, 2005).

Studies also suggest that Agile methods are growing from the bottom up and therefore require executive management support (Schatz & Abdelshafi, 2005; Livermore, 2007; Atlas, 2009). Such was the case at Amazon, where an Agile team was unable to raise the issue of organisational change above their level because of the lack of executive management support (Atlas, 2009).

Furthermore, executive management is often described as being risk adverse and opportunity focused (Coram & Bohner, 2005). In order to justify expenditure, they require committed delivery dates, progress on tasks and detailed schedule plans (Coram & Bohner, 2005). As such Agile methods represent a cultural change for them (Coram & Bohner, 2005). It is therefore incumbent that project managers convince management that Agile methods will deliver a better quality product in shorter time (Coram & Bohner, 2005).

A study of 17 organisations by Conboy, Coyle, Wang & Pikkarainen revealed that performance evaluation activities in many organisations neglected to take into consideration Agile methods (Conboy, Coyle, Wang & Pikkarainen, 2011). In five of the seventeen cases, it was found that performance evaluation criteria focused primarily on the candidate's ability to follow directions and their technical skills and it failed to distinguish Agile specific functions like self-organisation, creative thinking and social skills (Conboy *et al.*, 2011). In other cases, developers were evaluated according to traditional criteria. As a result the team's performance did not reflect their true abilities (Conboy *et al.*, 2011).

Successful adoption of Agile methods requires top management support and management needs to recognise the important role that they play during the adoption process (Cao *et al.*, 2009).

3.6.3 Organisation Knowledge

One of the statements of the Agile manifesto is to prefer "working software over comprehensive documentation" (Beck *et al.*, 2001a). In Agile methods the primary focus is not on documentation but on delivering the final product (Fowler & Highsmith, 2001). Each team has to determine for themselves what documentation is necessary (Fowler & Highsmith, 2001).

As such Agile methods promote a paradigm shift in knowledge management strategies (Chandra Misra *et al.*, 2010). From heavy document centric to that of tactile knowledge that resides within the development team (Chandra Misra *et al.*, 2010).

While opponents of Agile methods argue that knowledge is transferred from people to people by writing it on paper, proponents on the other hand argue that "Tacit knowledge cannot be transferred by getting it out of people's heads and onto paper" (Fowler & Highsmith, 2001: 1). As tacit knowledge does not only represent facts but it also represents the relationships between those facts (Fowler & Highsmith, 2001).

Critics on the other hand have further argued that the emphasis on tacit knowledge has made Agile methods dependent on experts (Turk, France & Rumpe, 2014). Adding to this is the real possibility of corporate memory loss that could inhibit an organisations ability to learn from its collective experience (Rubin & Rubin, 2011; Turk *et al.*, 2014).

Countering the argument of memory loss Nerur & Balijepally uses the metaphor of a holograph to raise an important point (Nerur & Balijepally, 2007). A holograph allows the image representing the whole film to be constructed from a fragment or piece of broken holographic film. His argument is taken further by the scientific research on the way the brain stores its memory, intelligence and functionality across its parts. This is done so that damage to one part cannot cause a complete loss of functionality (Nerur & Balijepally, 2007).

Nerur & Balijepally contends that Agile organisation create a culture that promotes the interchangeability of jobs and rolls. This interchangeability causes knowledge overlapping and creates redundant skills that enable the organisation to function even when members are missing (Nerur & Balijepally, 2007).

The traditionalist approach which emphasises document centeredness is not without its own challenges. Jansen et al., identified a set of challenges affecting architectural documentation (Jansen, Avgeriou & van der Ven, 2009):

1. Architectural documentation can be difficult to understand in large and complex systems.
2. A stakeholders understanding of the language and their background may hinder their ability to understand documentation.
3. It may be difficult to locate both formal and informal documentation on large projects.
4. It may be difficult to locate a specific set of knowledge in an architectural documentation.
5. Documentation needs to be updated otherwise stakeholders will lose confidence in its credibility.

The graph below shows the documentation through the SDLC:

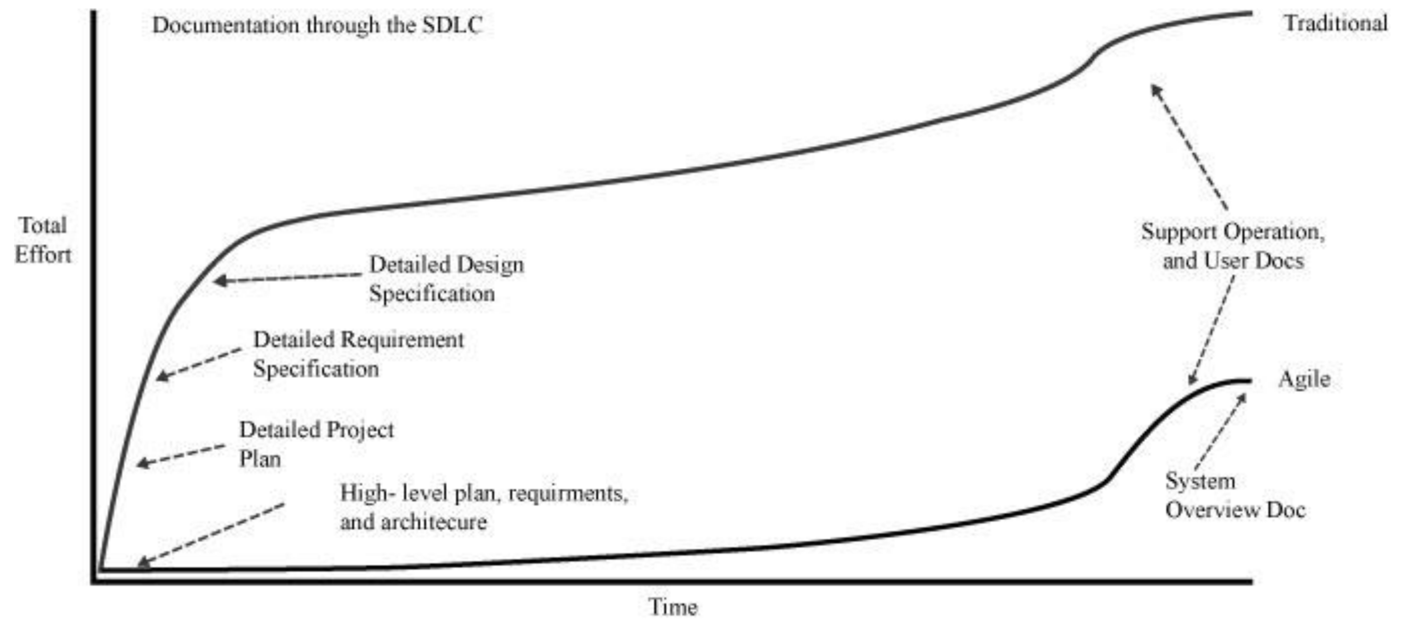


Figure 3-4: A graph showing the documentation through the SDLC. Adopted from Ambler (Ambler, 2012).

According to Ambler developers need to recognise that effective documentation is a balancing act and in order to get this balance right one must consider the following issues (Ambler, 2012):

1) ***Software developers are not technical writers***: Few software developers have good writing skills as a result they might not know where to get started when writing documentation. Studies have shown that poor documentation is often the reason for degradation in quality and confusion in maintenance (Ambler, 2012).

2) ***Waiting until information is stabilized***: Early in the project requirements and designs are very unstable. Often developers will wait until those requirements become stable before they complete the associated documentation. This can result in documentation that is behind (Ambler, 2012).

3) ***High level documentation vs. Low level documentation***: High level documentation provides an overview of the processes with links to the actual source code while low level documentation provides details of the exact process (Ambler, 2012). Ambler highlights the trade-off between larger and smaller documents (Ambler, 2012). He suggests that it might be perceived that larger documents have more information than smaller documentation. However, that perception could be wrong if the document contains irrelevant information (Ambler, 2012).

4) ***Document in/or outside code****:* Ambler raises the issues around code documentation (Ambler, 2012). His argument is that the development team needs to decide whether documentation should be inside the code or in a separate document or if the code should be designed in a way that it is self-documented (Ambler, 2012).

Architecture documentation in Agile methods is highly abstract and tends to strip away any unnecessary technical details (Hadar, Sherman, Hadar &

Harrison, 2013). Furthermore, organisations should not see the distinction between traditional and Agile methods as one of *no documentation* vs. *extensive documentation* but they should see it as a blend of documentation and conversations that fosters an environment and promote the sharing of both tacit and externalized knowledge (Fowler & Highsmith, 2001; Turk *et al.*, 2014).

3.6.4 Development Team

Learning to adopt Agile is seen as a cognitive process that requires the software developer to change their behaviour, attitude and opinions during software development activities (Mazni *et al.*, 2011).

The preference of face to face communication over formal communication and need for heightened interaction in the form or stand-up meetings, pair programming and retrospectives can be challenging for team members who have inherently weak communication skills (Dybå & Dingsøyr, 2008; Conboy *et al.*, 2011). Furthermore, developers who enjoy working on solitary activities may find the idea of pair programming, shared learning and collaborative decision making overwhelming (Nerur *et al.*, 2005).

While stand-up meetings are very effective in mass communications between the stakeholders and the team it was noted in a study by Rehan *et al.*, that the transparency of skill deficiency in some teams promoted developer fear (Rasmusson, 2003; Rehan *et al.*, 2014). During stand-up meetings developers are required to update their peers on their progress (Lacey, 2012; Rehan *et al.*, 2014). However, a lack of progress on the part of the developer can highlight their deficiency in communicational or technical skills (Rehan *et al.*, 2014). Exposing weaknesses of developers can create unhealthy environments in organisations and this could inevitability lead to counterproductively (Rehan *et al.*, 2014).

Agile methods value people and their interactions over processes. Furthermore, the eleventh principles behind the Agile manifesto specifically talks about self-organized teams "The best architectures, requirements, and designs emerge from self-organizing teams." (Beck *et al.*, 2001b: 1). As an important Agile principles self-organisation allows teams to manage and shift work amongst themselves while participating in team decision making (Hoda, Noble & Marshall, 2010).

In self-organized teams leadership is done in '*collaboration'* rather than '*command and control'* manner (Coram & Bohner, 2005). Leadership is seen as a light weight process that is adaptive and a lot like sport teams (Hoda *et al.,* 2010). Agile methods place emphasis on team work as opposed to individual role assignments that have characterised traditional methods (Moe, Dingsøyr & Dybå, 2008).

Self-organizing teams are also better at utilizing team talents because more minds are involved in the decision making processes (Ahmed *et al.*, 2010). Furthermore, adaptability can be achieved by making incremental decisions at operational level and delaying strategic decisions to allow for further input (Moe *et al.*, 2008).

Self-organised teams allow team members to develop by placing responsibilities on the shoulders of individuals working within those teams (Ahmed *et al.*, 2010). Further self-management directly improves speed and efficiency of a team as it facilitates decision making authority at the level of the operational problem (Moe *et al.*, 2008).

According to Takeuchi & Nonaka self-organized teams should not be uncontrolled. Management must establish checkpoints, that are not rigid and do not impair spontaneity and creativity, to prevent tension, ambiguity and chaos (Takeuchi & Nonaka, 1986).

Studies have shown that self-organised teams cannot be created by tearing down organisational hierarchies. They have to be fostered and supported through the creation of five general conditions: (Moe *et al.*, 2008):

- Clear engaging direction

- Enabling performing unit structure
- Supporting organisational context
- Coaching
- Adequate resources

Furthermore, in a study by Hoda *et al.*, it was found that team members adopt six roles to facilitate their team's self-organisation. These roles are described below (Hoda *et al.*, 2010):

Role	**Definition**	**Played by**
Mentor	Guide, support and help team members become more confident in their use of Agile practices.	Agile Coach
Coordinator	Acts on behalf of the self-organizing team to coordinate and communicate change requests.	Developer, Business Analyst
Translator	Understand, communicate and translate requirements between the business and customers.	Business Analyst
Champion	Champions the cause of Agile within the organisation in order to gain support for the self-organizing team.	Agile Coach
Promoter	Promotes the involvement and collaboration between the customer and the self-organizing team in order to support the team.	Agile Coach
Terminator	Identifies members that are threats to the functioning and productivity of the team and engages managements support in removing such members from the team.	Agile Coach

Figure 3-5: A table showing the roles facilitating self-organizing Agile teams. Adopted from Hoda *et al.*, (Hoda *et al.*, 2010).

From the above discussion one can deduce that self-organizing teams operate differently from traditional teams. As such organisations must be aware that the management of these teams need to be radically different in order to reap the rewards of Agile methods (Turk *et al.*, 2014).

Another problem that affects teams using Agile methods is the competence level of team members. Agile methods require developers that can work as a team and are able to solve problems and handle constant change (Coram & Bohner, 2005).

The reduction of waste processes in Agile methods gives software developers more time to develop software as a result the differences in productivity between best and worst programmers can be higher in Agile projects (Cohn & Ford, 2003). Having slow developers can signify one of two things for the development team (Cohn & Ford, 2003):

1. The entire team can be slowed down
2. The slower member can be left behind by his/her faster colleagues

In a study by Rehan *et al.*, it was found that developer roles were opaque on some Agile projects and that it was important for developers to be competent in a range of skills rather than being experts in a field (Rehan *et al.*, 2014). Thus it is increasingly difficult for project managers to find developers that exhibit all the necessary skills. Furthermore, Coram & Bohner suggest that it may be challenging to implement Agile methods in traditional organisations because highly skilled staff are always in demand (Coram & Bohner, 2005).

In a study by Lindvall, Basili, Boehm, Costa, Dangle, Shull, Tesoriero, Williams & Zelkowitz it was found that experience in building actual systems was more important than experience with Agile projects (Lindvall, Basili, Boehm, Costa, Dangle, Shull, Tesoriero, Williams & Zelkowitz, 2002). Furthermore, it was estimated that between 25% to 33% of project personnel need to be "competent and experienced"; however, that figure could be reduced to 10% if the team engages in the pair programming practice (Lindvall *et al.*, 2002). While there are studies to support the above claims it must be noted that the literature does not provide much empirical evidence on the impact of developer skills on the success of Agile projects and as such there is still work to be done in this area.

So as the debate on whether Agile processes require “good people” to be successful continues. Some agilest suggest that successful Agile projects also require the correct mentality and attitude. Individuals within the team need to be motivated by the “All for one and one for all” mentality rather than “This is not my problem” mentality (Beck, 1999; DSDM Consortium, 2013). This idea suggests that the entire team should pool together rather than putting the responsibility on an individual within the team.

Principle number eight in the Agile manifesto introduces the concept of sustainable pace. “Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely” (Beck *et al.*, 2001b: 1). The concept of sustainable pace is a term that was coined by guru Kent Beck in the first edition of his book “Extreme Programming Explained” published in 1999 (Beck, 1999). Kent suggests that when a team works at a reasonable pace they can sustain that pace indefinitely (Beck, 1999).

In 1914 Hendry Ford started a revolution and proved that productivity does not increase by working overtime as he made the largest profits from workers working a 40 hour week (Ford, 1914). Further studies on productivity have also shown that even though it is possible to perform some limited overtime work in the first few weeks with no productivity loss the likelihood of doing so is small (Thomas & Raynar, 1997).

Alistair Cockburn who describes people as non-linear systems takes this discussion further. “Expecting a linear relationship between input (more hours) and output (quicker accomplishment) is misguided” (Hazrati, 2008: 1). Perhaps Fred Brooks in his essay the “Mythical Man-Moth” summed up the discussion well when he said that the bearing of a child takes nine months no matter how many women are assigned (Brooks, 1995).

Aside from productivity concerns agilest suggest that sustainable pace is a social responsibility (Moreira, 2013). Working a 40 hour week promotes better team spirit and improves team morale (Moreira, 2013). Maintaining a sustainable pace also helps establish trust between management and the team (Moreira, 2013).

As such sustainable pace is about expending a team's energy levels vigorously and then regaining one's strength by resting. Teams need to ensure that they invest their energy wisely and set their priorities accordingly (Baudson & Beck, 2012).

Another important aspect affecting developers is the ownership of source code and other artefacts. Developers have to take responsibility for the quality, appearance, improvement, documentation, source code and test plans associated with Agile projects (Nordberg, 2003).

Generally there are five models for assigning ownership in this regard(Nordberg, 2003):

- **Product specialist**: An individual manages the source codes.
- **Subsystem ownership**: The source code in each subsystem has an owner.
- **Chief architect**: The lead programmer has ownership of the source code.
- **Collective ownership**: All source code is collectively owned by the team.
- **Non-ownership**: Occurs when a system or subsystem has no owner. Furthermore, developers have minimum accountability for communication and quality.

Collective ownership is one of the principles of XP and is popular amongst Agile methods (Beck, 1999). Collective ownership encourages team members to take responsibility for all aspects of the code (Beck, 1999). It is significantly different from non-ownership as team members are responsible for the integrity of the code when making modifications (Nordberg, 2003).

In collective code ownership no team member is responsible for a specific unit of code. Any developer can make changes by adding functionality, fixing errors or improve code via refactoring (Maruping, Zhang & Venkatesh, 2009). Once the code changes have been tested they can be committed to the repository (Maruping *et al.*, 2009).

Collective code ownership is not feasible without several other XP practices. According to Nordberg M.E. and Maruping *et al.*, these practices are as follows (Nordberg, 2003; Maruping *et al.*, 2009):

- **Unit Testing**: Refers to the process in which units of code are checked for correctness. Unit tests are generally automated but can also be done manually (Osherove, 2013; Saleh, 2013). Furthermore, unit testing requires that the smallest separate module in the system be tested (Runeson, 2006). In practice unit tests are performed by the developer. The use of coverage tools enables the developer to get insight into, which part of the code is being exercised by tests and also helps the developer reduce the number of code defects (Williams, Kudrjavets & Nagappan, 2009).

- **Strong Coding Guidelines**: Just like an artist, programmers have their own styles. However, when working in a team an individual’s style can get in the way of collective work (Shore & Warden, 2007). As such the development team has to agree on the coding style before they begin development. Having rules that govern the format and style of code facilitates effective communication between developers while maintaining a common base that can be used to understand the various units of code (Maruping *et al.*, 2009). Furthermore, coding guidelines also facilitates pair programming, refactoring, testing and collective code ownership by providing a common understanding amongst the development team (Nagler, 2012). Some of the standards to consider when developing coding guidelines are as follows:

Standard	**Description**
Formatting	Formatting refers to the way in which a document is prepared and includes the use of indentation and white space.
Code structure	The code structure refers to the layout of classes, files, resources and other source file types.
Naming conventions	Naming conventions is usually the first standard that is adopted. Several considerations must be made when naming objects. These considerations include the use of capital or small letters, symbols and the use of nouns or plurals when naming methods, classes, variables, events, and parameters (James, 2008).
Error handling	Error handling refers to the way in which objects handle error reports and log errors.
Comments	Comments refer to the description placed in the code that explains the logic of the code. Commenting improves the understanding and the maintainability of code.

Figure 3-6: A table showing the types of coding standards. Adopted from (Baird, 2002).

- **Continuous Integration**: Refers to the practice in which members of the development team integrate their code frequently or at least daily. Furthermore, an automated build that includes unit tests are used to verify code integrations (Melymuka, 2012). Continuous integration allows teams to reduce integration errors while providing support for a more cohesive and quicker software development environment (Melymuka, 2012).
- **Pair Programming**: Refers to a collaborative software development approach in, which developers work in pairs rather than individually (Dybå, Arisholm, Sjøberg, Hannay & Shull, 2007). Studies have shown that programmers working in tandem are more efficient and produce better quality software (Williams, Kessler, Cunningham & Jeffries, 2000). Furthermore, paired programming improves teamwork, increases knowledge about the code and facilities knowledge transfer (Williams *et al.*, 2000; Dybå *et al.*, 2007). When a team practices pair programming one from amongst the pair is referred to as the "Driver". The "Driver" is responsible for writing code while the other partner actively observes the "Driver's" work. "Drivers" periodically switch roles to give both parties equal and shared participation (Wray, 2010).

Collective code ownership also provides several advantages to development teams. These advantages are as follows:

- Reduces the dependency on individual developers and collectively makes all developers responsible for the codebase (Agile Alliance, 2013)
- Enforces a consistent philosophy and style across the system (Nordberg, 2003).
- Encourages developers to take responsibility for the overall quality of the system (Agile Alliance, 2013).

Teams should also be aware of the potential costs associated with collective code ownership. These costs are as follows:

- Having all team members responsible for quality is a state that is sometimes akin to having “no one” responsible for quality (Agile Alliance, 2013).
- It may become difficult to identify individual responsible for task or problems (Nordberg, 2003).

3.6.5 Large Organisations

Some development teams have reported difficulties in using Agile approaches in large organisations, while others have reported successes (Mnkandla & Dwolatzky, 2004; Vinekar *et al.*, 2006; Ahmed *et al.*, 2010; Korhonen, 2013; Stoica, Mircea & Ghilic-Micu, 2013). However, it must be noted that software development itself has problems scaling. According to Charette large scale projects fail three to five times more than smaller projects because of the complexities associated in handling the larger scope and teams (Charette, 2005).

For larger teams with thousands of developers the standard method of dividing work is used to distribute the workload amongst the team (Abrahamsson, 2003; Murphy *et al.*, 2013). However, trying to maintain a uniform practice and process is difficult. Furthermore, challenges associated with distributed teams can also discourage developers from practicing Agile methods as some Agile practices can be difficult to perform if teams are not co-located (Abrahamsson, 2003; Murphy *et al.*, 2013).

Another apparent limitation according to Mnkandla is the lack of up front design in Agile methodologies (Mnkandla, 2009). However, he concedes that this risk can be mitigated through the use of domain modelling and code refactoring (Mnkandla, 2009).

Although Agile implementations in large projects are challenging, it is encouraging to note that there are larger organisations that are transforming to

Agile. According to Korhonen the transformation requires special attention and should be initiated within small collocated teams first (Korhonen, 2013).

A study by Daneva, van der Veen, Amrit, Ghaisas, Sikkel, Kumar, Ajmeri, Ramteerthkar & Wieringa suggests that Agile methods require suitable adaptation in order for them to be effectively used in a distributed software environment (Daneva, van der Veen, Amrit, Ghaisas, Sikkel, Kumar, Ajmeri, Ramteerthkar & Wieringa, 2013). The study also pointed out that the maturity level of an organisation can also be a contributing factor to the successful implementation of Agile. Furthermore, Benefield at Yahoo and Atlas at Amazon.com suggest that Agile coaches can be beneficial as consultants especially if the organisation does not have any previous Agile experience (Benefield, 2008; Atlas, 2009).

While critics argue that Agile methodologies are not suitable for larger projects, proponents of Agile on the other hand argue that Agile methods are more suitable for larger projects than traditional methods (Abrahamsson, 2003; Ambler, 2006; Nord & Tomayko, 2006). According to Ambler Agile projects not only reduce risk by engaging the customer regularly but improve quality by introducing unit tests and also provide effective feedback via paired programming (Ambler, 2006). He suggests that the following strategy be used when dealing with large Agile projects:

1) Organise the project into smaller sub-teams and integrate the work on a regular basis (Ambler, 2006; Vinekar *et al.*, 2006; VersionOne, 2010).
2) Model the requirements and architecture early and at a high level (Ambler, 2006).
3) Deliver working software at regular intervals (Ambler, 2006).
4) Coordinate and communicate regularly (Ambler, 2006). Daily meetings enhance communication, coordination and improve project management activities (Paasivaara & Lassenius, 2003).

5) Adopt common philosophies and hire skilled people (Ambler, 2006).

Organisations such as ABB, Daimler Chrysler, Motorola, Yahoo, Microsoft and Amazon have all adopted and recorded positive experiences in using Agile methods (Benefield, 2008; Blose, 2008; Murphy *et al.*, 2013).

One of the larger Agile projects discussed by Ambler is the Eclipse team project. Ambler describes the Eclipse team project as a program comprising of 10 projects, 23 subprojects and 262 committees working in 15 different companies in 12 countries with 7 million lines of code. According to Ambler the Eclipse team successfully delivered six major release on time, every time using Agile methods (Ambler, 2006).

The Agile implementation at Yahoo is also worth noting. Yahoo kicked off its Agile program in 2005 when four teams volunteered to try Scrum and share their experiences. In a survey conducted by Yahoo 74% of respondents reported that Scrum had improved their productivity and 81% of the respondents wanted to continue using Scrum (Benefield, 2008).

Microsoft reported in 2012 that 57% of their developers were using Agile methods. Team members responsible for testing on a project reported a 75.7% positive response regarding Agile methods. While developers on average reported a 75.2% positive response (Murphy *et al.*, 2013).

Another example of a large project that has successfully implement Agile is the United States based insurance and financial services company Nationwide. The company has 26 development teams and use Agile methods to partner with 7000 information technology staffers distributed throughout its 23 business units (Babcock, 2011).

Nationwide's development centre has been able to produce 100 applications with 70% of them been defect-free. Furthermore, through the success of Agile they plan to increase the number of Agile teams from 25 to 60 by 2014 (Babcock, 2011).

3.7 Agile Software Development Methodologies

There are a number of Agile methods available to software developers (Stoica *et al.*, 2013). Each of these methods share the values described in the Agile manifesto (Stoica *et al.*, 2013). This section provides an introduction to three common Agile methods. For each method a brief overview will be provided.

3.7.1 Extreme Programming (XP)

Extreme Programming(XP), developed by programmer Kent Beck, can be described as a light weight software development technique that is built around rapid iterations and places emphasis on code writing and customer interaction (Copeland, 2001; Mnkandla & Dwolatzky, 2004). According to Kent Beck XP reduces project risk, improves productivity and responsiveness to changing business requirements throughout the life of a system (Noll, 2007).

Instead of large functional specifications software developed using XP starts by creating stories that describe the functionality of the system to be developed (Copeland, 2001; Mnkandla & Dwolatzky, 2004). Developers estimate the value and cost of stories and the customer decides which stories to develop (Mnkandla & Dwolatzky, 2004).

After two weeks developers deliver working stories to the customer thereafter the customer chooses another two weeks of work and as such the system grows in functionality (Mnkandla & Dwolatzky, 2004).

XP promotes an informal design specification process where developers sketch out models to help them understand and communicate ideas during the software development process (Turk *et al.*, 2014).

XP consists of five core values and thirteen practices (Loftus & Ratcliffe, 2005). The core values are as follows:

Simplicity: XP requires that developers seek the simplest solutions that satisfy the current customer needs and developers are discouraged from pursuing solutions that solve future problems (Loftus & Ratcliffe, 2005; Williams, 2007a; Turk *et al.*, 2014).

Communication: Communication is an important medium for the exchange of rapid, continual feedback in XP teams. It also supports agility and the spread of tactile knowledge and allows the team to respond to changing requirements as the client develops a clearer understanding of the system (Loftus & Ratcliffe, 2005; Turk *et al.*, 2014). In XP team's face-to-face communication between team members, the team and the client is considered the richest form of communication (Loftus & Ratcliffe, 2005).

Feedback: The XP team should receive feedback at regular interval from the customer as this is critical to the delivery of working software (Turk *et al.*, 2014).

Respect: In XP team members should respect the expertise of each other and should strive to achieve high quality code and design (Williams, 2007a; Turk *et al.*, 2014).

Courage: The value of courage is required to enforce the other XP values (Williams, 2007a). In an environment where the team is not communicating, practicing simple design or generating feedback one needs to show courage to begin implementing these XP practices. According to Kent Beck a team needs courage when faced with the following situations (Agile In A Flash, 2009: 1):

- "To make architectural corrections"
- "To throw away tests and code"
- "To be transparent, whether favourable or not"
- "To deliver complete, quality work in the face of time pressure"
- "To never discard essential practices"

- “To simplify code at every turn”
- “To attack whatever code the team fears most” (sic)
- “To take credit only for complete work”

The five core principles of XP are supported by thirteen practices and they are as follows:

- **Pair programming**: Developers work in pairs at a workstation (Loftus & Ratcliffe, 2005).
- **Collective code ownership**: The team owns the code base (Loftus & Ratcliffe, 2005).
- **Continuous integration**: Code is integrated at least once a day (Loftus & Ratcliffe, 2005; Turk *et al.*, 2014).
- **Test-first Development**: The XP team develops the system by writing the test case first and then the implementation code (Loftus & Ratcliffe, 2005).
- **Sit Together**: The team works in an open space (Williams, 2007a).
- **Whole Team**: The XP team needs to be cross functional. This includes testers, developers, the client and quality assurance team members (Williams, 2007a).
- **Energized work**: XP teams should work a 40-hour week. Long periods of overtime are counterproductive (Williams, 2007a).
- **Stories**: The XP team must write short statements describing the functionality of the desired product (Williams, 2007a). The team must also estimate the size and prioritize the stories (Williams, 2007a).
- **Weekly cycle**: On a weekly basis a progress review meeting should be held to allow the customer the opportunity to pick a week’s worth

of stories to be implemented (Williams, 2007a).. The meeting is also used to break the stories down into tasks (Williams, 2007a).

- **Quarterly cycle**: During the quarterly cycle the team chooses larger themes or interrelated stories that will be developed over a quarter (Williams, 2007a; Münch, Armbrust, Kowalczyk & Soto, 2012). Themes allow teams to see the larger picture (Williams, 2007a).
- **Slack**: Low priority tasks that can be dropped if the team is behind (Williams, 2007a; Münch *et al.*, 2012).
- **Ten-minute build**: The whole system including the unit test must be built and run in 10 minutes (Williams, 2007a; Münch *et al.*, 2012).
- **Incremental design**: The team should invest in the design of the system on a daily basis rather than developing and anticipating future features (Williams, 2007a; Münch *et al.*, 2012).

3.7.2 Scrum

Scrum is a framework for managing and tracking software development (Clutterbuck *et al.*, 2009). Scrum used fixed iterative cycles called sprints to deliver working software (Lacey, 2012). The term Scrum is derived from Rugby where the strategy of the game is to use team work to bring a lost ball back into play (Stoica *et al.*, 2013). The graphic below describes the Scrum framework:

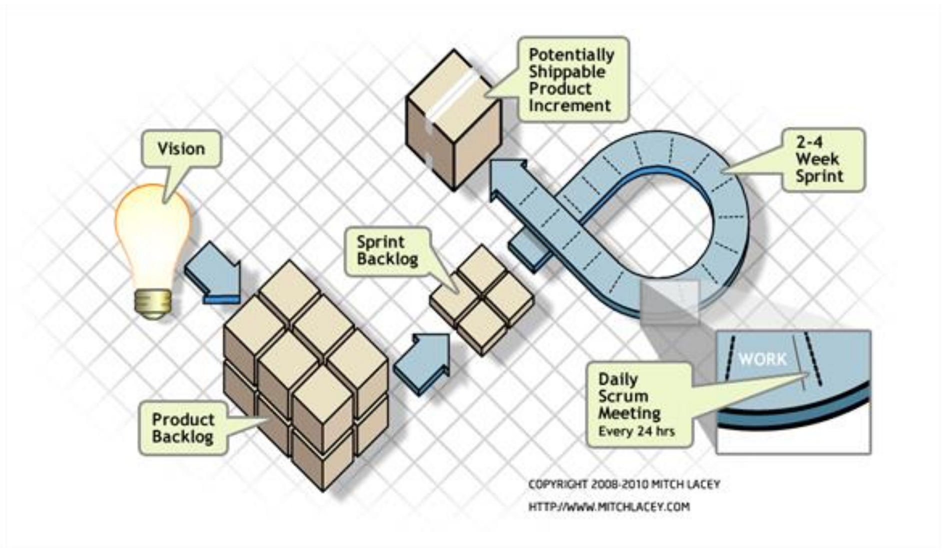

Vision
Product Backlog
Sprint Backlog
Potentially Shippable Product Increment
2-4 Week Sprint
WORK
Daily Scrum Meeting Every 24 hrs
COPYRIGHT 2008-2010 MITCH LACEY
HTTP://WWW.MITCHLACEY.COM

Figure 3-7: A graphical representation of the Scrum framework. Adopted from Lacey (Lacey, 2012).

The Scrum process is initiated when the client's vision is converted into the product backlog (Mnkandla, 2009; Lacey, 2012). The product backlog is a prioritised set of stories or functionality that is required by the client (Clutterbuck *et al.*, 2009; Lacey, 2012).

Each product backlog item is worked on by the development team in an iterative cycle called sprints (Clutterbuck *et al.*, 2009; Mnkandla, 2009; Lacey, 2012). Sprints typically last between 1 week and 4 weeks (Lacey, 2012). At a meeting called the sprint planning meeting the team decides which features to work on during a particular sprint (Mnkandla & Dwolatzky, 2004; Clutterbuck *et al.*, 2009; Lacey, 2012).

On a daily basis the team meets to update stakeholders about the progress of the system being developed and to remain focused on the goal (Mnkandla, 2009). At the end of every sprint a sprint review meeting is held to demonstrate the latest features of the product and to give the team an opportunity to get critical feedback (Mnkandla & Dwolatzky, 2004; Clutterbuck *et al.*, 2009; Lacey, 2012).

The Scrum framework consists of the following components:

Roles

- **Team**: Refer to the developers in the Scrum team committed to achieving the sprint goal (Williams, 2007a).
- **Scrum Master**: The Scrum is responsible for guiding and helps the team resolve issues that are blocking their progress (Scharff, 2011; Lacey, 2012).
- **Product Owner**: The product owner is in charge of the prioritised product backlog in the form of user stories (Scharff, 2011). The product owner also makes decisions regarding the approval of stories at the end of a sprint (Williams, 2007a).

Ceremonies

- **Sprint Planning**: During the sprint planning meeting the product owner creates and prioritises the product backlog (Williams, 2007a). Furthermore, during the sprint planning session the team also agrees on a sprint goal, which serves as the success criteria for the sprint (Williams, 2007a).
- **Sprint Review**: The sprint review provides an opportunity for the team to demonstrate its accomplishments during the sprint (Lacey, 2012).
- **Sprint Retrospect**: Retrospectives are important for the continuous improvement of the team (Lacey, 2012). Retrospectives also give the team the opportunity to reflect on how they worked together and how they can improve their efficiency, quality and velocity (Lacey, 2012).
- **Daily Scrum Meeting**: The Daily Scrum refers to a 15 minute meeting held daily (Williams, 2007a). Each team

member is required to answer the following three questions (Williams, 2007a):

  - What did you do yesterday?
  - What will you do today?
  - What is blocking you from completing your tasks?

Artefacts

- **Product Backlog**: The product backlog is a prioritized master list of business requirements that contains the vision of the product to be developed (Williams, 2007a; Lacey, 2012).
- **Sprint Backlog**: The sprint backlog is a list of task that the team needs to complete during the sprint (Lacey, 2012).
- **Burn down Charts**: The burn down chart is a graphical representation of the work remaining (Williams, 2007a; Lacey, 2012).

3.7.3 Feature Driven Development (FDD)

Feature driven development (FDD) focuses on delivering tangible functionality in 2 week iterations (Mnkandla, 2009) and is designed to be used in conjunction with other development activities (Mnkandla, 2009). FDD has the following artefacts and roles:

Artefacts

- **Feature list**: A set of features that the client deems useful (Williams, 2007a).

- **Design packages**: Refers to the notes, class diagrams, and sequence diagrams reports (Palmer & Felsing, 2002).

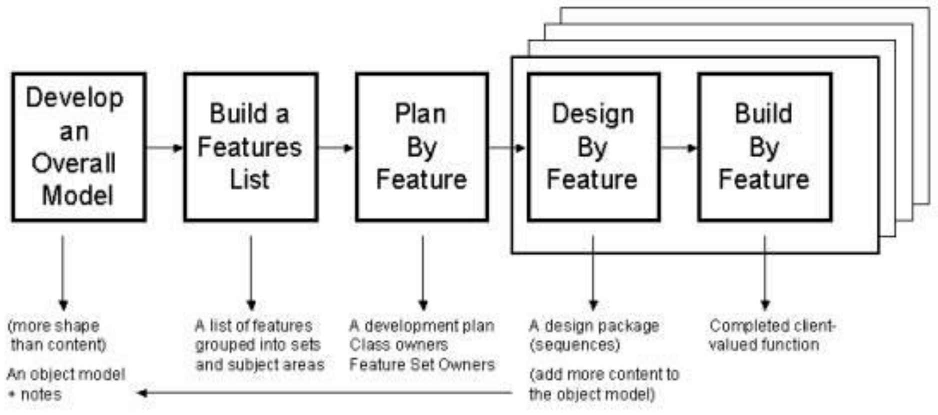
- **Track by feature**: A chart showing the dates when features will be released (Williams, 2007a).

- **“Burn Up” chart**: The chart shows work completed and project scope (Williams, 2007a).

Roles

- **Project manager**: The administrative lead for the project (Williams, 2007a).
- **Chief architect**: The person responsible for overall design (Williams, 2007a).
- **Development manager**: The person responsible for day to day development activities (Williams, 2007a).
- **Chief programmer**: An experienced developer that is the team leader (Williams, 2007a).
- **Class owner**: Responsible for designing, testing and documenting features (Williams, 2007a).
- **Domain experts**: A person(s) who has deep knowledge about the business (Williams, 2007a).
- **Feature teams**: Responsible for implementing features (Williams, 2007a).

FDD consist of five incremental processes:



Develop an Overall Model
Build a Features List
Plan By Feature
Design By Feature
Build By Feature
(more shape than content)
An object model + notes
A list of features grouped into sets and subject areas
A development plan Class owners Feature Set Owners
A design package (sequences)
(add more content to the object model)
Completed client-valued function

Figure 3-8: Graphical representation of the Feature Driven Development Model. Adopted from Palmer & Felsing (Palmer & Felsing, 2002).

- **Develop an overall model**: The team works to develop a high level object model of the problem domain (Goyal, 2007; Williams, 2007a).
- **Build feature lists**: Based on the requirements of the customer a list of features is developed for the business problem (Goyal, 2007; Williams, 2007a).
- **Plan by feature**: A plan detailing when features will be implemented is developed by the team (Palmer & Felsing, 2002; Williams, 2007a).
- **Design by feature**: The chief programmer chooses features to be developed and identifies owners who will be involved in developing those features (Goyal, 2007). The chief programmer also refines the object model while developers write the classes and methods (Goyal, 2007).
- **Build by feature**: The team implements the features outlined in the "Design by feature" process (Goyal, 2007).

3.8 Chapter Summary

Over the last decade numerous software development methodologies have emerged. Agile is one of these methodologies. It aims to find better ways of developing software based on 4 main ideologies:

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

The above statements do not mean that process, tools, documentation, negotiating and following a plan have no relevance. However, on the contrary less emphasis is placed on them.

The primary purpose of this chapter is to review the existing research. The chapter specifically answers questions related to the state of Agile, its popularity and the benefits and challenges associated with Agile adoption. It discussed in detail the impact of Agile process on an organisation from a culture, knowledge, developer and team size point of view. The chapter concludes by examining the practices associated with 3 popular Agile methods namely:

- Extreme Programming
- Scrum
- Feature Driven Development

After examining the literature, the researcher found that the problem statement and sub-problems were appropriately aligned. In the next chapter the theoretical framework for this study will be discussed in detail.

Chapter 4

THE GARTNER HYPE CYCLE

4.1 Introduction

The Gartner hype cycle (GHC) is an established and highly relevant model influencing the adoption strategies of large companies (Steinert & Leifer, 2010). Jackie Fenn, an analyst for Gartner, developed the concept of the "hype cycle" in 1995 (Fenn & Raskino, 2008). According to Fenn & Raskino new technologies often go through phases of enthusiasm and hype (Janes & Succi, 2012). Generally the more visible an innovation is in conversation, marketing buzz and in the media the more hype it gets (Fenn & Raskino, 2008). Human expectations are driven by three factors (Fenn & Raskino, 2008):

- Firstly, our social nature
- Our attraction to novelty
- Finally, our tendency to use shortcuts in making decisions under uncertainty

All three of these factors can cause potentially negative consequences when deciding whether to adopt an innovation. The Gartner hype cycle provides graphical representation of the maturity of an innovation or process and this can be used by organisations to solve adoption challenges or to exploit new opportunities (Fenn & Raskino, 2008).

4.2 Processes Behind the Hype Cycle

To get the bandwagon rolling, the media starts to talk about what people are doing with an innovation. Senior managers are invited to briefings where they are exposed to the latest technologies or management tools (Fenn & Raskino, 2008).

The threat of competitive advantage causes managers to pay careful attention to what the competition is adopting (Fenn & Raskino, 2008). As managers start to consider new technology they collect evidence for and against it (Fenn & Raskino, 2008). Human nature has a strong bias to seek out evidence that supports our preferred view. This biasness further compounds our decision making process (Fenn & Raskino, 2008).

As excitement about the new technology ramps up something is bound to give (Fenn & Raskino, 2008). Either the innovation will deliver on its promise or it will collapse under its own weight (Fenn & Raskino, 2008). History has shown that when people are most excited about an innovation or technology it rarely delivers on its promise (Fenn & Raskino, 2008).

Virtually all innovations need experience and time to realize their potential (Fenn & Raskino, 2008). After the excitement starts to wane the same factors that drove the hype up begin to drive it back down again (Fenn & Raskino, 2008). From this point onwards expectations can only be raised by the maturity of the innovation (Fenn & Raskino, 2008).

4.3 Stages in The Hype Cycle

The GHC provides a graphical representation of the adoption and maturity of applications and technologies (Fenn & Raskino, 2008). The cycle of hope and disappointment is aptly named the “hype cycle” because enthusiasm is primarily built on hope and hype (Fenn & Raskino, 2008). The figure below describes the GHC:

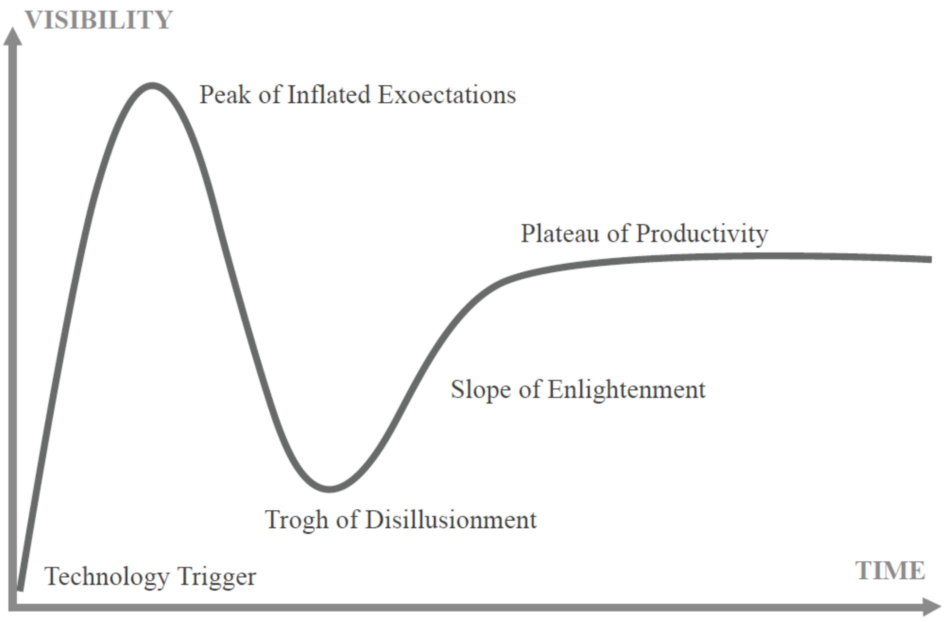


Figure 4-1: Graphical representation of the Gartner hype cycle. Adopted from Fenn & Raskino (Fenn & Raskino, 2008)

When innovation enters the market it starts raw and matures over time. Therefore, the horizontal axis represents the time while the vertical axis represents the visibility or the "hype" that the innovation has created (Fenn & Raskino, 2008).

The five stages of the GHC are described below (Fenn & Raskino, 2008):

Technology Trigger: The cycle starts when a product is launched or a breakthrough is made that generates interest in some innovation. As more and more people hear about the innovation a wave of buzz passed on the news (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

Peak of Inflated Expectations: Companies like to seek out innovation in order to jump ahead of their competitors. This creates a bandwagon effort as the innovation is pushed to its limits. Stories in the media capture and reinforce early successes (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

The Trough of Disillusionment: Over time less favourable stories start to emerge. Companies begin to realize that the potential value is not as it first seemed. The media begins to feature stories on the challenges rather than the opportunity for innovation. Innovation usually requires sufficient development, experimentation and patience before anything worthwhile can be delivered (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

Slope of Enlightenment: Early adaptors begin to experience benefits. Drawing on this experience understanding grows and the innovation matures over time. Best practices and methodologies are codified (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

Plateau of Productivity: As the benefits of the innovation are accepted a number of organisations begin to feel comfortable with the reduced levels of risk. A new surge in adoption begins and penetration levels accelerate (Fenn & Raskino, 2008). According to Fenn & Raskino innovations typically take between five to eight years to arrive at the plateau of productivity. However, there are exceptions for fast track and long fused innovations (Fenn & Raskino, 2008). Time between the peak of inflated expectations and the plateau of productivity is termed as the time-to-value gap and is expressed as "years to mainstream adoption" (Fenn & Raskino, 2008; Steinert & Leifer, 2010).

4.4 MAKING ADOPTION DECISIONS

Three factors generally play a role in an organisations adoption strategy:

- The value of the innovation to the organisation
- The maturity of innovation
- The risk tolerance of the organisation

Young innovations are often fraught with dissonance and group thinking, which can lead managers into misjudgement. The maturity of an innovation must be compared against its potential value (Fenn & Raskino, 2008).

Early adoption brings with it a high risk of failure. However, for innovations with high value this risk might be warranted. On the converse side when an innovation enters the Plateau of Productivity the risks are lower as organisations know more about the innovation. As such organisation might consider adopting these innovations if the payback is good (Fenn & Raskino, 2008).

The risk profile of an organisation will also play a role in its adoption decisions. Organisations tend to fit into one of the following risk profiles (Fenn & Raskino, 2008):

- **Type A**: Organisations that have an aggressive high risk profile with potentially high rewards. These organisations adopt early in the hype cycle.
- **Type B**: Organisations that have moderate risk profiles and moderate payoffs. These organisations adopt in the middle of the hype cycle and try to learn from “Type A” organisations.
- **Type C**: Refer to organisations that are cautious adopters and are averse to risk. These organisations adopt late in the hype cycle.

After a decade of studying the hype cycle Fenn & Raskino developed a set of best practices called the STREET processes (Fenn & Raskino, 2008). These processes help organisations select the right innovation at the right time and also lay down the foundation for its use. The stages in the STREET processes are described below (Fenn & Raskino, 2008):

Scope: At this stage an organisation decides what is valuable and how much of risk they are willing to take.

Track: At this stage an organisation seeks out and tracks the progress of relevant innovations on the hype cycle.

Rank: At this stage an organisation considers candidates by ranking potential innovations. The idea is to identify innovations that could benefit an organisation within their risk profile.

Evaluate: At this stage the organisation evaluates the top-raking innovations and makes a decision to choose an appropriate innovation based on the organisations risk profile.

Evangelize: At this stage the organisation must inspire, involve and educate people in order to obtain organisational support for the innovations they intend to adopt.

Transfer: An organisation must continue to inspire, involve and educate people in order to transfer responsibility to those who will implement or use the innovation.

4.5. The Problem with The Hype Cycle

According to Fenn & Raskino normal technologies take between five and eight years to complete the hype cycle (Fenn & Raskino, 2008). However, there are exceptions for fast track and long fuse innovations. Fast track innovations take between two to four years to complete the cycle and long fused innovations may take between two to three decades to go through several hypes and troughs (Roussel, 2003; Steinert & Leifer, 2010). Furthermore, many long fused innovations perpetually emerge and the hype takes these innovations repeatedly from peaks to troughs (Roussel, 2003). Analysis by Steinert & Leifer on some of the innovations in the GHC shows that these innovations were not moving at the average speed required to complete the hype cycle (Steinert & Leifer, 2010).

Another cause of concern is the axes defined in the GHC (Steinert & Leifer, 2010). The vertical axis is dependent on a specific market and technology and cannot be easily generalised. While the horizontal axis is dependent on time, which is not considered a good indicator as technology rarely evolves linearly over time (Steinert & Leifer, 2010). Amongst the other oddities found by Steinert & Leifer is the case where some innovations were not removed from the GHC consistently after passing through the final stages but simply disappear from analysis (Steinert & Leifer, 2010).

It has also been noted that the GHC does not necessarily conform to the actual news about an innovation as some news depict various peaks and troughs (Steinert & Leifer, 2010). However, it is worth noting that Adamuthe, Tomke & Thampi in their study "An Empirical Analysis of Hype-cycle: A Case Study of Cloud Computing Technologies" used news articles as an indicator for the hype cycle. Their analysis concluded that news articles can be used to effectively capture innovations in the first three phases of hype on the GHC (Adamuthe, Tomke & Thampi, 2015). As such this study will use news and related media items as a data source to plot Agile methods on the GHC. However, the limitations of the GHC need to be factored when considering the validity of the results.

Finally, Steinert & Leifer suggest that the GHC is mathematically flawed as there is no mathematical relationship between the technology s-curve and the hype (Steinert & Leifer, 2010). They suggest that this mathematical formula is indispensable. Furthermore, innovations need to be consistently followed through the GHC from the first to the fifth stage and phased out thereafter this would greatly enhance the model (Steinert & Leifer, 2010).

4.6 CHAPTER SUMMARY

The hype cycle is often used as a decision making tool that can be used to track the progress of innovations from its inception to maturity (Fenn & Raskino, 2008). This chapter describes the hype cycle processes and stages. Furthermore, it also discusses the use of the STREET framework when adopting new innovations and concludes with a summary of the problems associated with the model.

Chapter 5

RESEARCH METHODOLOGY

5.1 INTRODUCTION

This chapter outlines how the researcher implemented the research design described in chapter 1 of this study. This chapter discusses the methodology, that was used in this study, the study design and the geographical area in, which the study was conducted. Furthermore, it describes the population and sample. Finally, the instrument used to collect the data, including methods implemented to maintain validity and reliability of the instrument are also described.

5.2 THE QUANTITATIVE RESEARCH METHODOLOGY

A quantitative research methodology was implemented through the use of questionnaires. A snowball sampling technique was used to generate a list of contacts through information technology associations, groups, organisation and forums. These groups included:

1. The Institute of Information Technology Professionals South Africa
2. Agile user group
3. Small Enterprise Development Agency (SEDA)
4. Microsoft's MVP and Certified partner lists
5. Researches personal contacts within the industry
6. Various Chamber of Commerce
7. UKZN IT Procurement Database

Some of the organisations listed above were unable to provide the researcher with a list of their members due to their privacy policies. However, they did assist the researcher by sending out emails to their members asking them to participate in the study.

For organisations that supplied details of their affiliates a combined list of organisations was created. The University requires researchers to secure participation from organisations before applying for ethical clearance. As such the researcher used the combined list of organisations to contact the organisation via the use of electronic mail or telephonically in order to identify appropriate gatekeepers within the company to complete the gatekeeper's permission given letter. Once the gatekeeper letters were completed the researcher applied for ethical clearance. Finally, after ethical clearance approval questionnaires were distributed and retrieved using the following methods:

- Questionnaires were hand delivered and collected from prospective respondents.
- The questionnaires were administered via the internet by sending out a link to the electronic version of the questionnaire.

5.3 Data Capture and Analysis Procedures

Questionnaires were distributed as both hardcopies as well as softcopies via email (Refer to addendum 14 for a copy of the questionnaire). All hardcopy responses were captured onto Survey Monkey.

The results of the questionnaire were exported into Microsoft Excel and then sent to the statistician for further analysis. The statistician used the Statistical Package for the Social Sciences (SPSS) to analyse the data and determine statistical relationships, trends, draw graphs and cross-tabulate data.

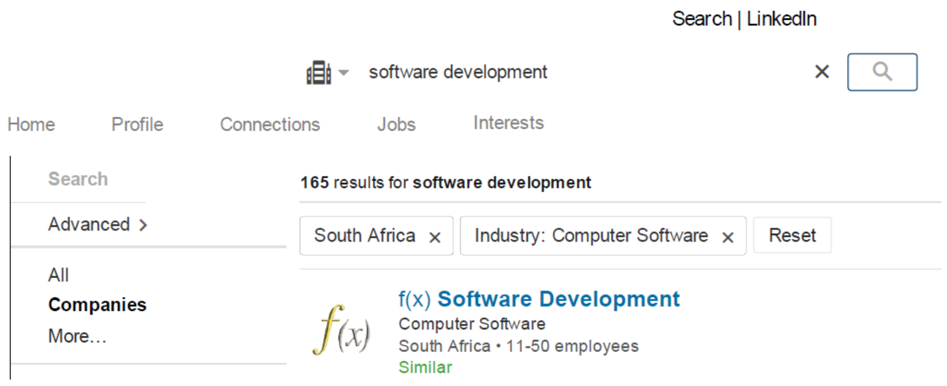
The Gartner hype cycle theoretical framework will be used to characterise the adoption of Agile methods in South Africa. This theory has been used by Gartner to characterise the over-enthusiasm and subsequent disappointment that typically happens with the introduction of new technologies (Fenn & Raskino, 2008).

5.4 RESEARCH PROCESS

During the data collection phase of the research, the researcher had to compile a list of information technology companies in South Africa. One of the problems experienced at the time when the list was being compiled was that the information technology sector in South Africa had no professional body representing its members.

As such compiling a list of information technology organisation in South Africa was a complex task. The researcher had to get in touch with various information technology associations, groups and forums including the researcher's personal contacts in order to compile a list.

To complicate the process further the population of software developers in South Africa is currently unknown and this made it difficult to determine the sample size. However, in order to provide some form of statistics the researcher took to LinkedIn to determine the number of software development organisation in South Africa. The researcher conducted a search on LinkedIn using the criteria "software development" and refined the search to include "South Africa" and the "Industry: Computer Software". Figure 5-1 below shows the search results:



165 results for software development

South Africa × Industry: Computer Software ×

f(x) Software Development
Computer Software
South Africa • 11-50 employees

Figure 5-1: Picture showing the number of software development organisations in South Africa on LinkedIn. Adopted from LinkedIn (LinkedIn.com, 2015).

The results show that one hundred and sixty-five companies were listed on LinkedIn. It must be noted that LinkedIn is not a definitive source of software development organisations in South Africa.

Once the list was completed the researcher had identified a total of one hundred organisations (100). The researcher drafted a gatekeeper's letter in line with the research ethics of the university and objective of transparency and voluntary participation (Refer to addendum 15 for the gatekeeper's letter). The benefits of the study were also outlined in the gatekeeper's letter. Each of the 100 organisations were contacted and invited to participate in this research.

Out of the one hundred organisations (100) contacted only twenty-five (25) organisations indicated that they would participate in the study. The researcher had encountered the following problems while trying to secure participation:

- Organisations regarded their practice of Agile processed as proprietary information and were reluctant to participate in the study.
- Organisations were not practicing Agile processes and had to therefore be excluded from the study.

- Organisations were outsourcing their software development activities and had to also be excluded from the study.

Gatekeepers who indicated that they were interested in participating in the study were required to complete a gatekeeper permission given letter (Refer to addendum 1). This process was completed via email or in person. The gatekeeper's permission given letters together with the ethical clearance documentation were sent to the university's research office for approval.

Once ethical clearance for this research had been approved (Refer to addendum 16) the gatekeepers were contacted via email and they were provided with the Survey Monkey link (Refer to addendum 14 for a copy of the questionnaire). Furthermore, they were asked to pass on the link to the appropriate personnel within their organisations.

After the data collection process had begun the researcher noticed that some of the sub-questions on the questionnaire had been numbered incorrectly. The incorrect numbering was not altered at that point in time because the survey was already published. Furthermore, the researcher felt that should the numbering need to be adjusted it would be done so in the fieldwork chapter of this study.

After a period of two months a report was extracted from survey monkey. This report showed that the survey had a poor response rate. A follow up email was sent to individual gatekeepers asking them to remind the appropriate personnel at their organisations to complete the questionnaire. Thereafter a subsequent report from Survey Monkey showed a total of 90 respondents had completed the questionnaire. However, of the sample of 90 respondents, five of them did not have experience with Agile and their records were excluded. The difficulties in securing responses and participation by organisations bare similarities to the challenges experienced by (Ramnath, 2011) when trying to secure respondents for his study.

A report by the IFC shows that a large part of the South African economy is driven by SMME's (IFC, 2013). As such one of the possible reasons for the poor response rate is that their software development teams are relatively small. Furthermore, not all of the software development teams were using Agile methods.

After identifying these challenges, the researcher contacted the statistician to determine if there was sufficient data to produce meaningful results. Upon investigation the statistician had indicated that there was sufficient data to produce meaningful results. Bearing in mind that the research process had to continue the researcher extracted the data from Survey Monkey and sent the data through to the statistician for processing. Thereafter a report with the complied results was sent back from the statistician.

The next chapter of this study will present the results from the research findings in line with the aims and objectives mentioned in chapter 1 of this study.

5.5 Chapter Summary

This chapter describes the approach that the researcher used when undertaking this study. It includes the research methodology, data collection, analysis and other considerations that were examined when this study was undertaken. A survey research strategy was deemed most appropriate for gaining insight into this study. Questionnaires were used to collect data and quantitative analysis was carried out. An account of the research process is also presented in this chapter. The research findings, analysis and interpretation of the findings will be presented in the next chapter.

Chapter 6

FIELDWORK, DATA PROCESSING, ANALYSIS & INTERPRETATION

6.1 Introduction

This chapter discusses how the research methodology defined in the previous chapter was used to analyse the sample data and ultimately answer the research questions stated at the beginning of this study. The chapter starts off by examining the data extraction process. Next the personal and business demographics of the sample are discussed and finally results are presented according to the research methodology.

6.2 Questionnaire Sections

The questionnaire is divided into the following sections:

- Personal Information
- The state of Agile in South Africa
- Barriers preventing Agile adoption in South Africa
- The impact of Agile on software development in South Africa

For the purposes of reporting the demographics section is divided into personal demographics and business demographics. Once the responses were captured the data was extracted from Survey Monkey. The extraction process is described in the next section.

6.3 Data Extraction and Analysis

Questionnaire response data in their raw format was downloaded from the Survey Monkey system in the form of an Excel spreadsheet. This spreadsheet was then sent to the statistician via email for analysis using Statistical Package for the Social Sciences (SPSS) software. The following statistical tests were performed on the sample:

- **A chi-square goodness of fit test**: The chi-square (pronounced "kai square") test is used to determine if any of the response selections are selected significantly more or less than the other response options (Daya, 2001). The chi-square is often used to compare the observed distribution under the null hypothesis. Under the null hypothesis categorization is assumed to be random and thus the cell frequencies are based on chance (Williams, 2007b).
- **Wilcoxon signed ranks test**: The Wilcoxon signed rank sum test a non-parametric that is used to determine whether the average value is significantly different from the central score of 3 (Shier, 2004). The score of 3 refers to the neutral response option (Shier, 2004).
- **Analysis of variance (ANOVA)**: ANOVA has been used since 1920 to test the mean differences between two or more means. (Lomax, 2013). Furthermore, it is used to determine whether a significant relationship exists between variables. There are three ways in which ANOVA can be used (Statistics Solutions, 2013):
    - **One Way**: Comparing data based on one independent variable.
    - **Two Way**: Comparing data based on two independent variables.
    - **N-Way**: Comparing data based on multiple independent variables.

Another aspect of statistics that is important to researchers is the shape of the distribution. Researchers are typically interested in how well a distribution can be approximated to the 'normal distribution' (StatSoft, 2013). In this regard the following terms are important (StatSoft, 2013):

- **Skewedness**: Refers to a deviation in symmetry for a distribution. A skewedness that is different from 0 is characterised as an asymmetrical distribution while perfectly symmetrical distributions have a skewness of 0 (Lowry, 2014). Furthermore, skewedness can be categorised as positively skewed and negatively skewed (Lowry, 2014):
    - **Positively Skewed**: Distributions that are heavy at the higher end of the range and light at the lower end can be described as a positively skewed distribution.
    - **Negatively Skewed**: On the converse a negatively skewed distribution has an elongated tail that extends to the left end of the range.
- **Kurtosis:** Measures whether data is flat or peak in a distribution. A kurtosis that is different from 0 could describe a distribution that is flatter or more peaked than usual. A normal distribution has a kurtosis of 0 (Lowry, 2014).

The result from the data analysis process is presented in the sections below.

6.4 Personal Demographics of Respondents

Respondents were asked to complete their personal demographical information. Information requested in this section includes the respondent’s age, gender and race. A summary of the personal demographics is shown in figure 6-1 below.

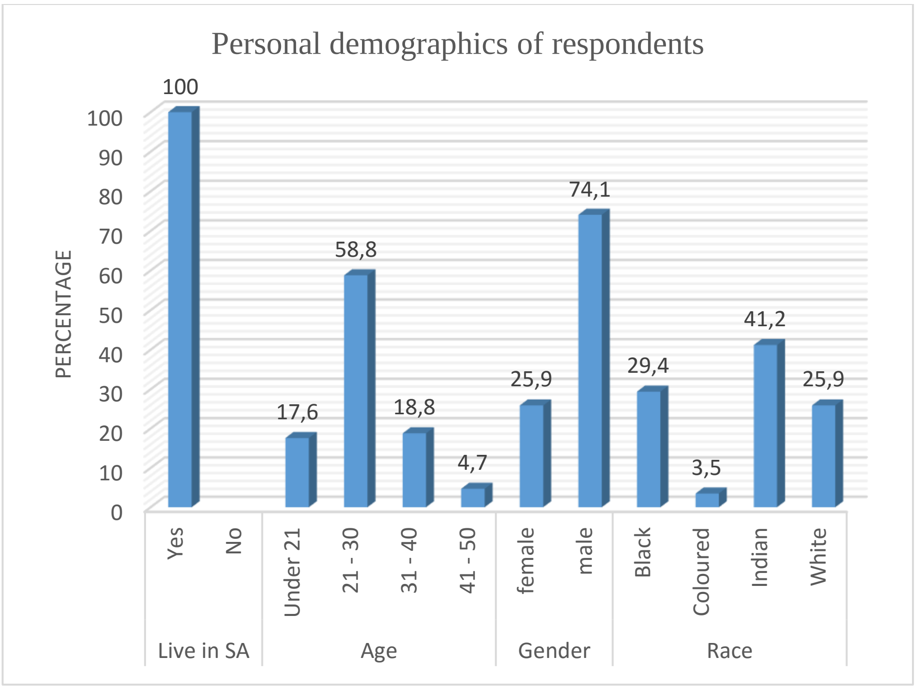


Figure 6-1: Bar chart showing personal demographics of respondents.

All respondents sampled are from South Africa with 58.8% of those respondents being between the ages of 21 and 30. Majority of the respondents were male with only 25.9% of them being female. Finally, the largest number of responses where received by Indian respondents followed by Black, White and Coloured respondents respectively.

6.5 Business Demographics of Respondents

The business demographics section details the business demographics of the sample. This section presents the statistics concerning the position of respondents at their organisations, the number of information technology employees, respondents working experience, organisation location and the organisation business sector. Results for each of the above sections is presented below.

6.5.1 Position of Respondents in The Sample

Respondents were asked to identify their position within the organisation in which they were employed. The results to this question is presented in figure 6-2 below.

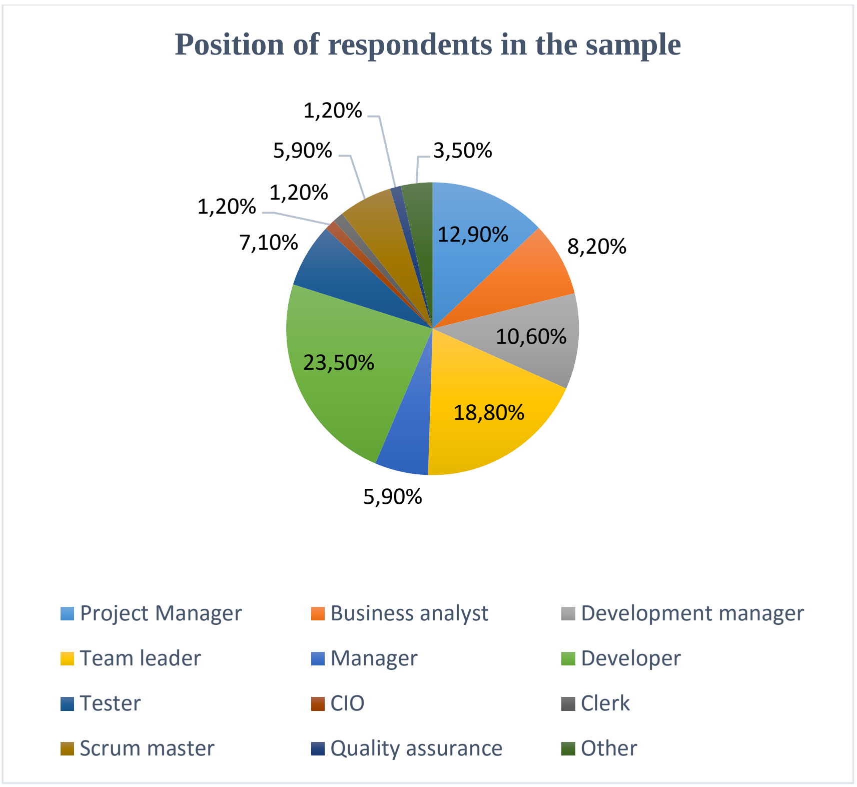


Figure 6-2: Pie chart showing position of respondents in the sample.

A significant number of the respondents from the sample were developers. This was followed by team leaders, project managers and development managers respectively.

6.5.2 Number of Information Technology Employees in an Organisation

Respondents were asked about the number of information technology workers in their organisations. The results from the analysis is presented in figure 6-3 below. The figure shows the number of information technology employees in an organisation.

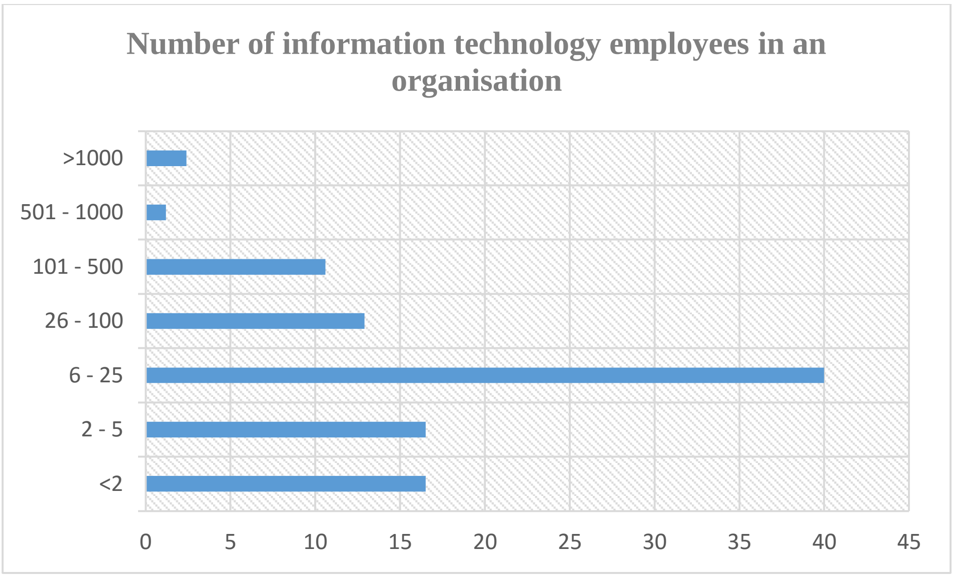


Figure 6-3: Horizontal bar chart showing number of information technology employees in an organisation.

Most respondents worked in organisations that had between 6 to 25 information technology employees. Information technology organisations with 2 to 5 team members followed and micro organisations with less than 2 information technology employees were amongst the most common responses.

6.5.3 Working Experience of Respondents

Another question under business demographics was the number of years of experience of respondents with Agile methods. Figure 6-4 below shows the working experience with Agile methods of the sample.

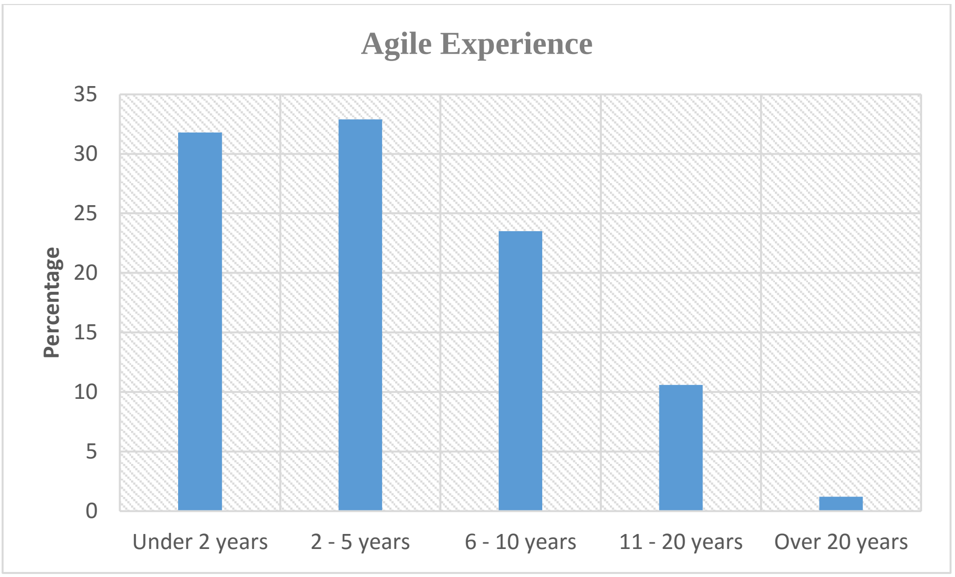


Figure 6-4: Vertical bar chart showing the working experience of respondents with Agile methods.

The results show that 35.3% of respondents have more than five years' experience with Agile methods and 32.9% of them have had between two and five years of experience. Finally, 31.8% percent of respondents have had under two years of experience with Agile methods. As such 68.2% of respondents have more than two year of experience with Agile methods.

6.5.3 Respondent Location

Analysed data about the location of respondents is shown in figure 6-5 below.

Province	Percent
Eastern Cape	1,2%
Gauteng	3,5%
KwaZulu-Natal	90,6%
Western Cape	4,7%

Figure 6-5: Table showing the location of respondents in South Africa.

Majority of the respondents in this sample were from KwaZulu-Natal this was followed by Western Cape, Gauteng and finally Eastern Cape respectively.

6.4.5 Organisation Business Sector

Finally, respondents were required to identify the business section in which their organisation operated. Figure 6-6 below shows the organisation business sectors of the sample.

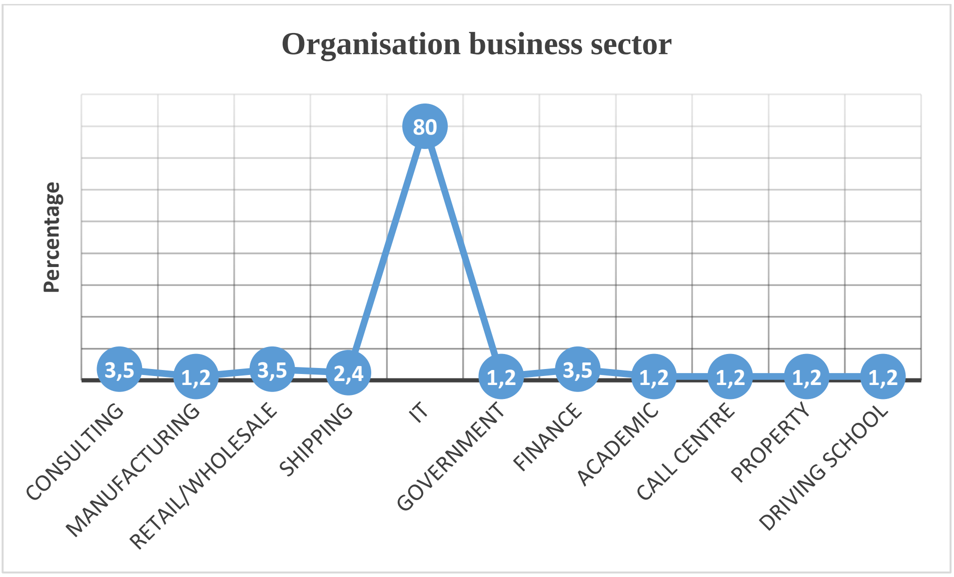


Figure 6-6: Line chart showing the organisation business sectors of the sample.

The sample consists of respondents across all of the business sectors with the majority of respondents operating within the information technology sector. The lack of data from other sectors makes it difficult to perform collaborative statistics on this data.

6.6 The State Of Agile In South Africa

Results presented in this section of the questionnaire focused on the state of Agile in South Africa and will be presented in the following sub-sections:

- The extent that organisations use Agile methods.
- The respondent's knowledge of Agile.

- The duration that the respondent has been practicing Agile.
- The size of Agile teams.
- The geographical distribution of Agile teams.
- The frequency of communication between stakeholders and the Agile team.
- The Type of Agile methodologies/frameworks used by respondents.
- The iteration length of Agile Methodologies.
- The organisation Agile adoption strategy.
- Suitable project types for Agile methods.

6.6.1 The Extent That Organisations Use Agile Methods

The first question in this section related to the extent to, which respondents practise Agile in their respective organisations. A summary of the results is presented in figure 6-7 below.

		Frequency	**Percent**	**Valid Percent**	**Cumulative Percent**
Valid	Always	21	24.7	24.7	24.7
	Most of the time	26	30.6	30.6	55.3
	Sometimes	12	14.1	14.1	69.4
	Rarely	26	30.6	30.6	100.0
	Total	85	100.0	100.0	

Figure 6-7: Table showing the extent that organisations use Agile.

Applying the chi-square goodness-of-fit test shows that there is no significant difference between the selections of the four response options in the table above. However, it is significant that nobody selected the ‘never’ option. This corroborates with the fact the only those with Agile experience participated in this study.

6.6.2 The Respondent's Knowledge of Agile

The second question asked respondents to rate their knowledge of Agile on a scale of 1 to 5. A summary of the results is presented in figure 6-8 and figure 6-9 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1	15	17.6	17.6	17.6
	2	16	18.8	18.8	36.5
	3	22	25.9	25.9	62.4
	4	27	31.8	31.8	94.1
	5	5	5.9	5.9	100.0
	Total	85	100.0	100.0	

Figure 6-8: Table showing the respondents knowledge of Agile.

	N	Mini-mum	Maxi-mum	Mean	Std. Devia-tion
Knowledge of Agile	85	1	5	2.89	1.205
Valid N (list-wise)	85				

Figure 6-9: Table showing the results of the Wilcoxon Signed ranks test with regards to respondent's knowledge of Agile.

Using the Wilcoxon Signed ranks test it was found that the average knowledge of respondents about Agile methods is 2.89. However, it must be noted that this is not significantly different from the central score of 3. As such no significant knowledge of the lack thereof can be interpreted with regards to this question.

6.6.3 The Duration That the Respondent Has Been Practicing Agile

The third question in this section relates to the duration that respondents have been practised Agile methods. A summary of the results is presented in figure 6-10 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	<1 year	45	52.9	52.9	52.9
	1 - 2 years	23	27.1	27.1	80.0
	3 - 4 years	8	9.4	9.4	89.4
	5 - 6 years	3	3.5	3.5	92.9
	7 - 8 years	3	3.5	3.5	96.5
	>8 years	3	3.5	3.5	100.0
	Total	85	100.0	100.0	

Figure 6-10: Table showing the experience of respondents with Agile methods.

The chi-square goodness-of-fit test shows that most respondents indicated that they are practicing AGILE for up to 2 years ($\chi 2(5, N=85) = 101.706$, $p<.0005$ (Refer to addendum 2 for further details).

6.6.4 The Size of Agile Teams

The fourth question in this section concerns the size of Agile teams. The results from the data analysis is presented in figure 6-11 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1 - 5	37	43.5	43.5	43.5
	6 - 10	37	43.5	43.5	87.1
	11 - 15	7	8.2	8.2	95.3
	16 - 20	2	2.4	2.4	97.6
	>30	2	2.4	2.4	100.0
	Total	85	100.0	100.0	

Figure 6-11: Table showing the size of Agile teams.

The chi-square goodness-of-fit test shows that the most numbers of respondents had team sizes between 1 to 10 members (Refer to addendum 3 for further details).

6.6.5 The Geographical Distribution of Agile Teams

The fifth question in this section relates to the geographical distribution of Agile teams. The analysed data is presented in figure 6-12 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Same room	39	45.9	45.9	45.9
	Same floor	11	12.9	12.9	58.8
	Same building	10	11.8	11.8	70.6
	Same campus	9	10.6	10.6	81.2
	Within driving distance of each other	9	10.6	10.6	91.8
	Within same time zone	5	5.9	5.9	97.6
	Different time zones	2	2.4	2.4	100.0
	Total	85	100.0	100.0	

Figure 6-12: Table showing the geographical distribution of Agile teams.

It is significant to note that according to the chi-square goodness-of-fit test most respondents had Agile teams in the same room (Refer to addendum 4 for further details).

6.6.6 The Frequency of Communication between Stakeholders and the Agile Team

The sixth question in this section concerns the frequency of communication amongst stakeholders and the Agile team. A summary of the results is presented in figure 6-13 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Regularly - daily	35	41.2	41.2	41.2
	Regularly - weekly	36	42.4	42.4	83.5
	At the start of an iteration	5	5.9	5.9	89.4
	As needed	9	10.6	10.6	100.0
	Total	85	100.0	100.0	

Figure 6-13: Table showing the frequency of communication in Agile teams.

According to the chi-square goodness-of-fit test most respondents indicated that they engaged with stakeholders on a daily or weekly basis (Refer to addendum 5 for further details).

6.6.7 The Type of Agile Methodologies/Frameworks used by Respondents

The seventh question relates to the type of Agile methodology/framework practiced by respondents. The results from the data analysis is presented in figure 6-14 below.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Scrum	71	83.5	83.5	83.5
	Hybrid Custom	4	4.7	4.7	88.2
	Kanban	3	3.5	3.5	91.8
	Extreme programming	3	3.5	3.5	95.3
	Lean	1	1.2	1.2	96.5
	Agile unified modelling	2	2.4	2.4	98.8
	K8	1	1.2	1.2	100.0
	Total	85	100.0	100.0	

Figure 6-14: Table showing the popular Agile methodologies/frameworks.

The most common Agile methodologies/frameworks as reported by respondents in this study is Scrum as indicated by the chi-square goodness-of-fit test (Refer to addendum 6 for further details).

6.6.8 The Iteration Length of Agile Methodologies

The eight question in this section concerns the length of iterations practiced by respondents. Figure 6-15 below presents the analysed data

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	1 - 7 days	28	32.9	32.9	32.9
	8 - 14 days	30	35.3	35.3	68.2
	15 - 21 days	7	8.2	8.2	76.5
	22 - 30 days	10	11.8	11.8	88.2
	over 30 days	10	11.8	11.8	100.0
	Total	85	100.0	100.0	

Figure 6-15: Table showing the iteration length of Agile methodologies.

Most respondents reported that they practiced between one and two week iterations according to the chi-square goodness-of-fit test (Refer to addendum 7 for further details).

6.6.9 The Organisation Agile Adoption Strategy

The ninth question in this section relates to the adoption strategy of the organisation. A summary of the result data is presented in figure 6-16 below.

		Frequency	**Percent**	**Valid Percent**	**Cumulative Percent**
Valid	Early adopter	27	31.8	31.8	31.8
	Fast follower	18	21.2	21.2	52.9
	Late adopter	40	47.1	47.1	100.0
	Total	85	100.0	100.0	

Figure 6-16: Table showing the organisation adoption strategy.

Most organisations reported that they were late adopters of Agile methodologies according to the chi-square goodness-of-fit test. (Refer to addendum 8 for further details).

6.6.9 Suitable project types for Agile methods

The final question in this section asked respondents to rank the suitability of Agile methods to the various software development sectors. A summary of the result data is presented in figure 6-17 below.

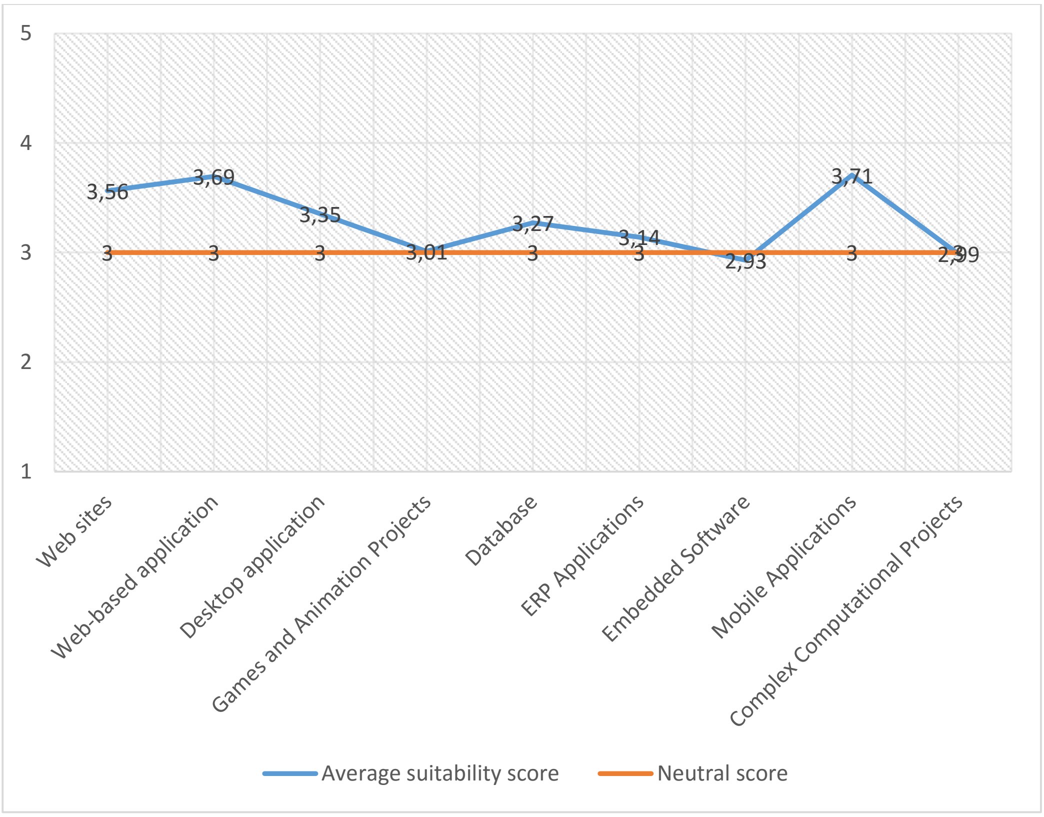


Figure 6-17: Line chart showing the type of projects that are suited to Agile methods.

According to the Wilcoxon signed ranks test web sites, web based applications, desktop applications and mobile applications are significantly suited to Agile methods. (Refer to addendum 9 for further details).

6.7 Barriers Preventing Agile Adoption in South Africa

Respondents were asked to identify barriers preventing Agile adoption in South Africa. The graph below presents the results from the Wilcoxon signed ranks test (Refer to addendum 10 for further details).

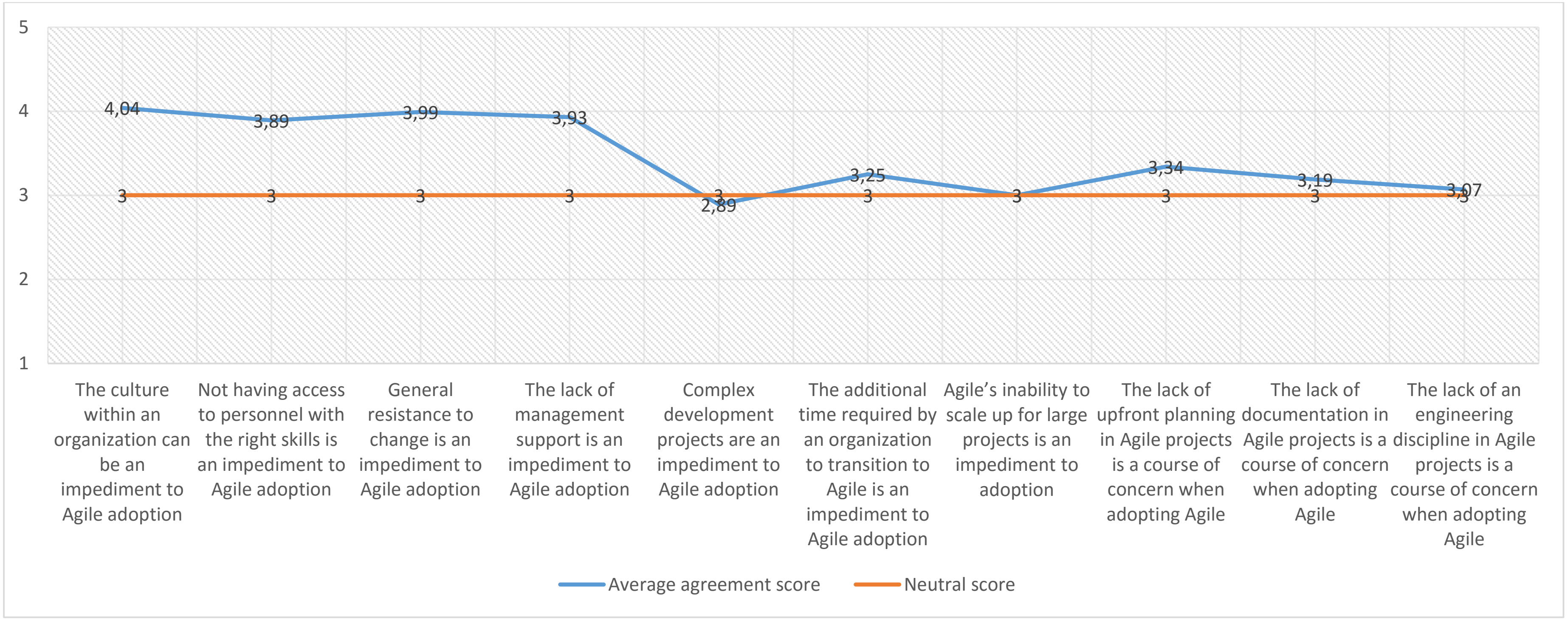


Figure 6-18: Line chart showing the barriers to Agile methods.

According to the data analysis the significant barriers that prevent Agile adoption are:

- The culture within an organisation
- Not having access to personnel with the right skills
- General resistance to change
- The lack of management support

6.8 The Impact of Agile on Software Development in South Africa

The final section of the questionnaire dealt with the impact of Agile software development in South Africa. The first question in this section asked respondents to characterise the state of their last Agile project. The results from the data analysis is presented in figure 6-19 below.

		Frequency	**Percent**	**Valid Percent**	**Cumulative Percent**
Valid	Too early to tell	11	12.9	12.9	12.9
	Successful	31	36.5	36.5	49.4
	Challenged	25	29.4	29.4	78.8
	Failed	14	16.5	16.5	95.3
	Don't know	4	4.7	4.7	100.0
	Total	85	100.0	100.0	

Figure 6-19: Table showing the state of respondent's last Agile projects.

Most respondents reported that their last Agile project was successful according to the results from the chi-square goodness-of-fit test (Refer to addendum 11 for further details).

The final question for this section asked respondents to rate the impact of Agile methods. The graph below presents the results from the Wilcoxon signed ranks test.

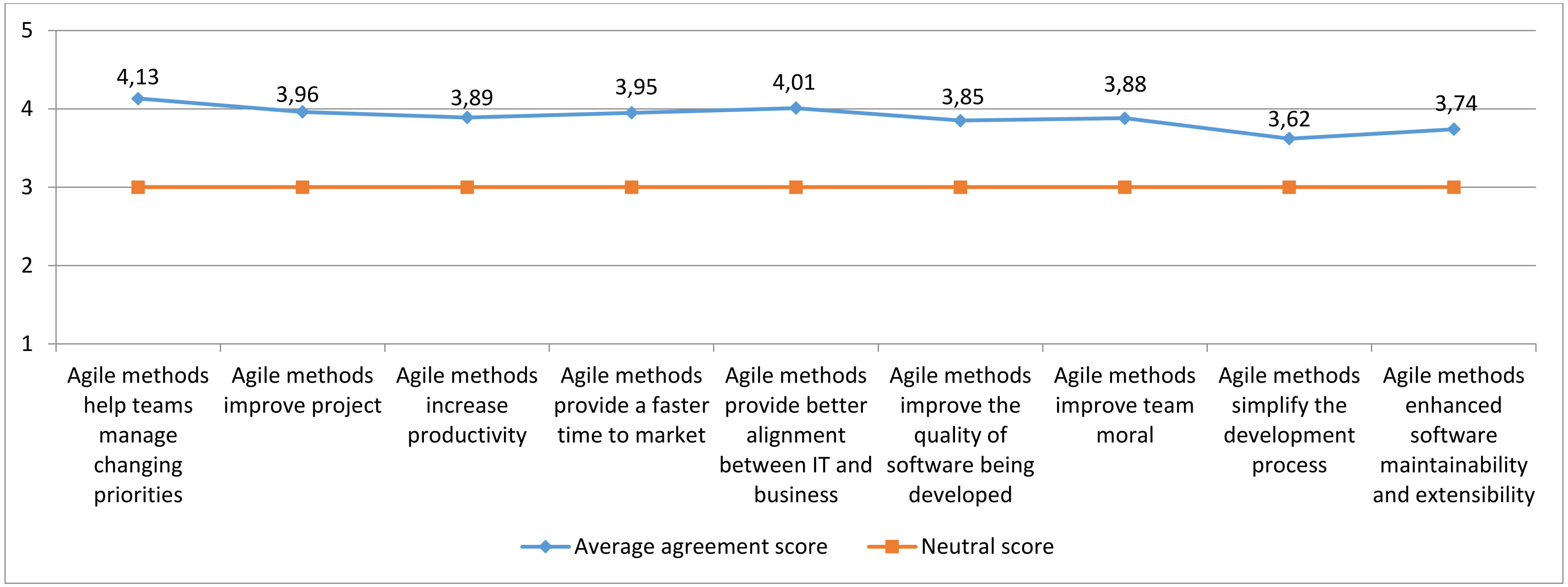


Figure 6-20: Line chart showing the impact of Agile methods in South Africa.

The analysis of the results showed that respondents agreed with the following statements regarding the impact of Agile software development:

- Agile methods help teams manage changing priorities.
- Agile methods improve project visibility.
- Agile methods increase productivity.
- Agile methods provide a faster time to market.
- Agile methods provide better alignment between IT and business.
- Agile methods improve the quality of software being developed.
- Agile methods improve team moral.
- Agile methods simplify the development process.
- Agile methods enhanced software maintainability and extensibility.

6.9 Bivariate Data Analysis

In this section the results from the bivariate data analysis is presented in the subsections below. The following questions were proposed by the researcher:

- Does the level of knowledge of respondents significantly affect the state of Agile projects?
- Are there any correlations between the knowledge of respondents and their perception of the barriers preventing Agile adoption?
- Does the level of experience of respondents affect the state of Agile projects?
- Are there any correlations between the knowledge of respondents and their perception of the benefits of Agile projects?

6.9.1 Does The Level of Knowledge of Respondents Significantly Affect the State of Agile Projects?

After critically comparing the question related to the respondent's level of knowledge of Agile and the state of their last Agile project, a significant difference in the perceived knowledge of the groupings and the state of respondents last agile project was noted (F (4,84) =15.317, p<.0005) . Figure 6-21 below presents the results from this analysis (For the questions, refer to Addendum 14 - Section B and Section C):

	N	Mean	Std. Deviation
Too early to tell	11	3.00	1.095
Successful	31	3.65	.798
Challenged	25	2.92	1.038
Failed	14	1.43	.646
Don't know	4	1.75	1.500
Total	85	2.89	1.205

Figure 6-21: Table showing the average knowledge scores for the different Agile project states.

Respondents who categorized their last Agile project as failed, rated their knowledge as lower than those who categorized their last Agile project as 'too early to tell', 'successful' or 'challenged'. Furthermore, those who stated that their last project was successful had more knowledge than those who didn't know the state of their last Agile project.

6.9.2 Are There Any Correlations Between the Knowledge of Respondents and their Perception of the Barriers Preventing Agile Adoption?

Next an analysis between the barriers of agile adoption and the perceived knowledge of candidates was done (For the questions, refer to Addendum 14 - Section B and Section C). The results are presented in figure 6-22 below:

Question	**Test**	**16** Knowledge of Agile
25.1 The culture within an organization can be an impediment to Agile adoption	Correlation Coefficient	.198
	Sig. (2-tailed)	.069
	N	85
25.2 Not having access to personnel with the right skills is an impediment to Agile adoption	Correlation Coefficient	.387**
	Sig. (2-tailed)	.000
	N	85
25.3 General resistance to change is an impediment to Agile adoption	Correlation Coefficient	.187
	Sig. (2-tailed)	.086
	N	85
25.4 The lack of management support is an impediment to Agile adoption	Correlation Coefficient	.233*
	Sig. (2-tailed)	.032
	N	85
25.5 Complex development projects are an impediment to Agile adoption	Correlation Coefficient	-.269*
	Sig. (2-tailed)	.013
	N	85
25.6 The additional time required by an organization to transition to Agile is an impediment to Agile adoption	Correlation Coefficient	-.173
	Sig. (2-tailed)	.113
	N	85
25.7 Agile's inability to scale up for large projects is an impediment to adoption	Correlation Coefficient	-.418**
	Sig. (2-tailed)	.000
	N	85
25.8 The lack of upfront planning in Agile projects is a cause of concern when adopting Agile	Correlation Coefficient	-.230*
	Sig. (2-tailed)	.034
	N	85
25.9 The lack of documentation in Agile projects is a cause of concern when adopting Agile	Correlation Coefficient	-.347**
	Sig. (2-tailed)	.001

Question	**Test**	**16** Knowledge of Agile
	N	85
25.10 The lack of an engineering discipline in Agile projects is a cause of concern when adopting Agile	Correlation Coefficient	-.454[**]
	Sig. (2-tailed)	.000
	N	85

Figure 6-22: Table showing the relationship between knowledge and the Agile adoption barriers.

The items in red above show significant correlations between knowledge and adoption barriers. Items with positive correlations show that higher knowledge is associated with more agreement whereas items with a negative correlation, with a minus sign, show that perceived knowledge is associated with disagreement.

Respondents with higher perceived knowledge agreed with the following agile adoption barriers:

- Not having access to personnel with the right skills is an impediment to Agile adoption.
- The lack of management support is an impediment to Agile adoption.

Whereas respondents with higher perceived knowledge disagreed with the following adoption barriers:

- Complex development projects are an impediment to Agile adoption.
- Agile's inability to scale up for large projects is an impediment to adoption.
- The lack of upfront planning in Agile projects is a cause of concern when adopting Agile.

- The lack of documentation in Agile projects is a cause of concern when adopting Agile.
- The lack of an engineering discipline in Agile projects is a cause of concern when adopting Agile.

6.9.3 Does The Level Experience of Respondents Affect the State of Agile Projects?

When comparing the respondent's experience against the state of their last Agile project it was found that there is a significant relationship between the two variables (Refer to Addendum 14 - Section B for the questions). For this analysis the years of experience had to be combined into categories. Figure 6-23 and figure 6-24 below shows the results.

	Too early to tell	**Suc-cessful**	**Chal-lenged**	**Failed**	**Don't know**	**Total**
< 1 Year	6 13.3%	8 17.8%	14 31.1%	14 31.1%	3 6.7%	45 100.0%
1-2 Years	4 17.4%	10 43.5%	8 34.8%	0 .0%	1 4.3%	23 100.0%
>2 Years	1 5.9%	13 76.5%	3 17.6%	0 .0%	0 ,0%	17 100.0%
Total	11 12.9%	31 36.5%	25 29.4%	14 16.5%	4 4.7%	85 100.0%

Figure 6-23: Table showing the respondent's experience vs. the state of their last Agile project.

Chi-Square Tests						
	Value	**df**	**Asymp. Sig. (2-sided)**	**Exact Sig. (2-sided)**	**Exact Sig. (1-sided)**	**Point Prob-ability**
Pearson Chi-Square	27.779[a]	8	.001	.000		
Likelihood Ratio	33.345	8	.000	.000		
Fisher's Exact Test	26.546			.000		
Linear-by-Linear Association	10.798[b]	1	.001	.001	.000	.000
N of Valid Cases	85					
a. 7 cells (46.7%) have expected count less than 5. The minimum expected count is .80.						
b. The standardized statistic is -3.286.						

Figure 6-24: Table showing the results of the Chi-Square tests in terms of respondent's experience vs. the state of their last Agile project.

The analysis shows a significant relationship exists between the respondent's experience and the state of their last agile project (Fishers (N=85) =, p<.0005). Most respondents with under one year of experience with Agile methods reported that their projects had failed while respondents with over 2 years of experience with Agile methods reported successful projects.

6.9.4 Are There Any Correlations Between the Knowledge of Respondents and Their Perception of the Benefits of Agile Projects?

A significant correlation was found between the knowledge of respondents and their perception of the benefits of Agile projects (For the questions, refer to Addendum 14 - Section D). Figure 6-25 presents the results from the comparison.

Question	**Test**	**16 Knowledge of Agile**
27.1 Agile methods help teams manage changing priorities	Correlation Coefficient	.401**
	Sig. (2-tailed)	.000
	N	85
27.2 Agile methods improve project visibility	Correlation Coefficient	.381**
	Sig. (2-tailed)	.000
	N	85
27.3 Agile methods increase productivity	Correlation Coefficient	.280**
	Sig. (2-tailed)	.009
	N	85
27.4 Agile methods provide a faster time to market	Correlation Coefficient	.350**
	Sig. (2-tailed)	.001
	N	85
27.5 Agile methods provide better alignment between IT and business	Correlation Coefficient	.163
	Sig. (2-tailed)	.136
	N	85

Question	**Test**	**16 Knowledge of Agile**
27.6 Agile methods improve the quality of software being developed	Correlation Coefficient	.244[*]
	Sig. (2-tailed)	.024
	N	85
27.7 Agile methods improve team moral	Correlation Coefficient	.231[*]
	Sig. (2-tailed)	.033
	N	85
27.8 Agile methods simplify the development process	Correlation Coefficient	.024
	Sig. (2-tailed)	.825
	N	85
27.9 Agile methods enhanced software maintainability and extensibility	Correlation Coefficient	-.022
	Sig. (2-tailed)	.841
	N	85

Figure 6-25: Table showing the respondents experience vs. the perceived impact of Agile methods.

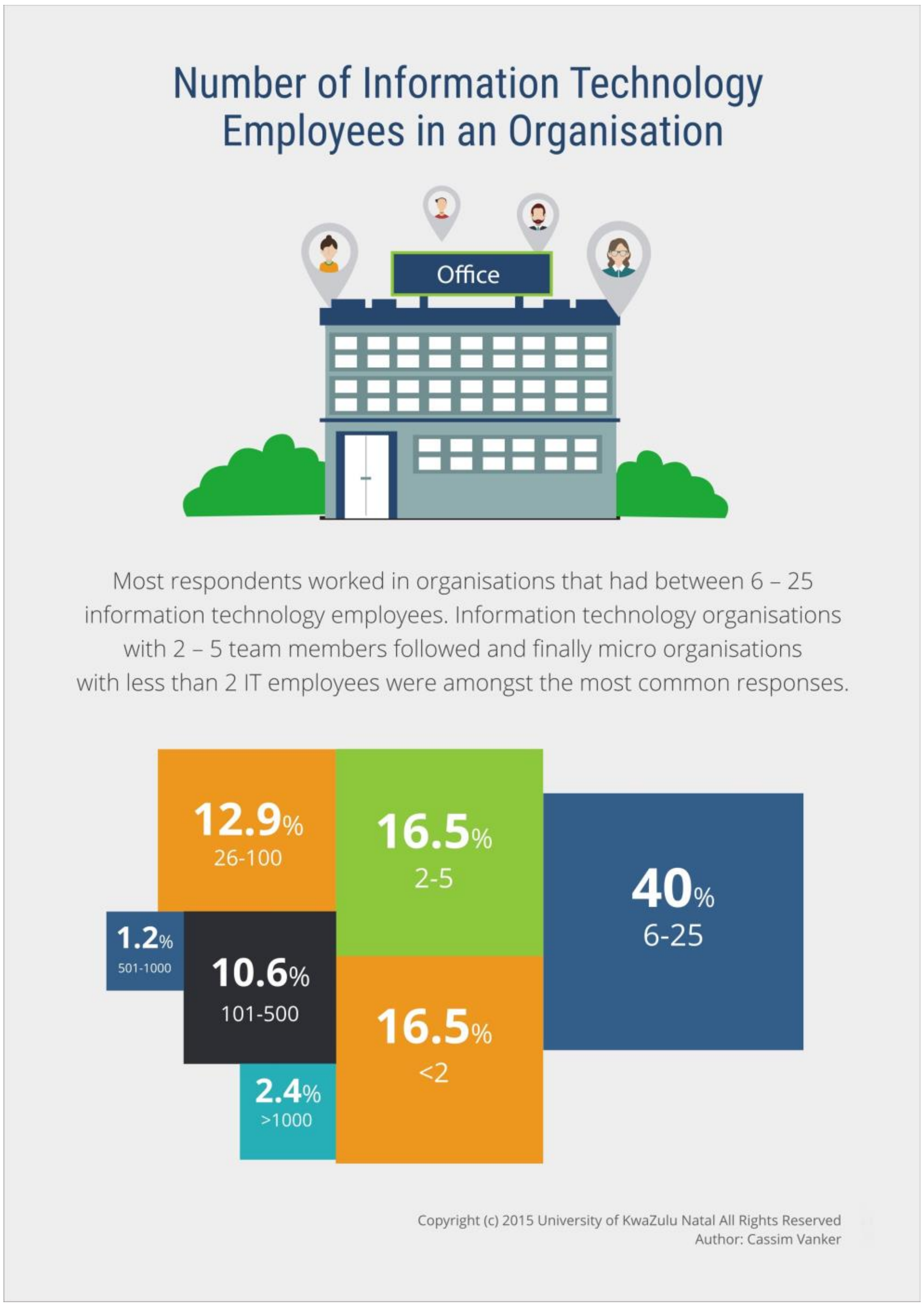
The items in red above show significant correlations between knowledge and agile benefits. Items with positive correlations show that higher knowledge is associated with more agreement.

Respondents with a high perceived knowledge of Agile were in agreement with the following statement on adoption benefits:

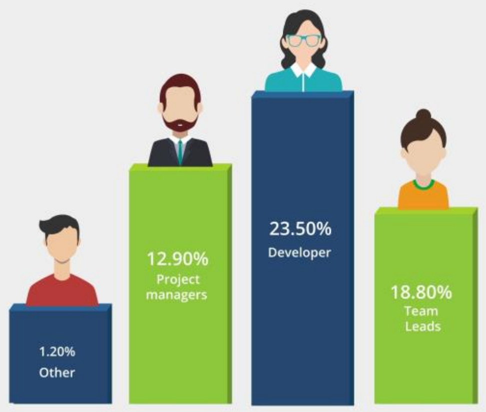
- Agile methods help teams manage changing priorities
- Agile methods improve project visibility
- Agile methods increase productivity
- Agile methods provide a faster time to market
- Agile methods improve the quality of software being developed
- Agile methods improve team moral

6.10 Graphical Report

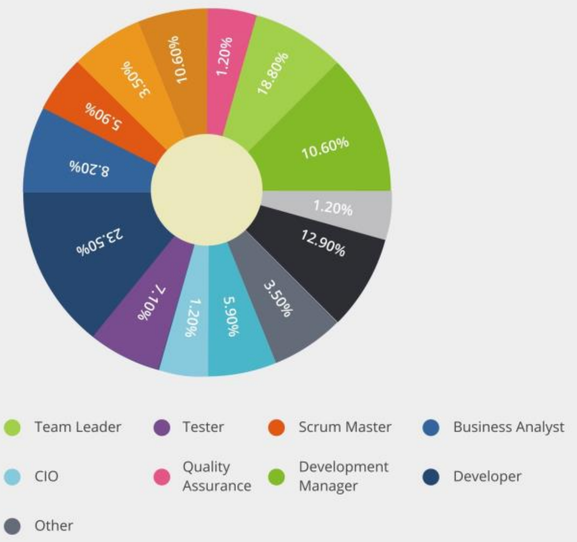
In this section a graphical report of the results is presented.

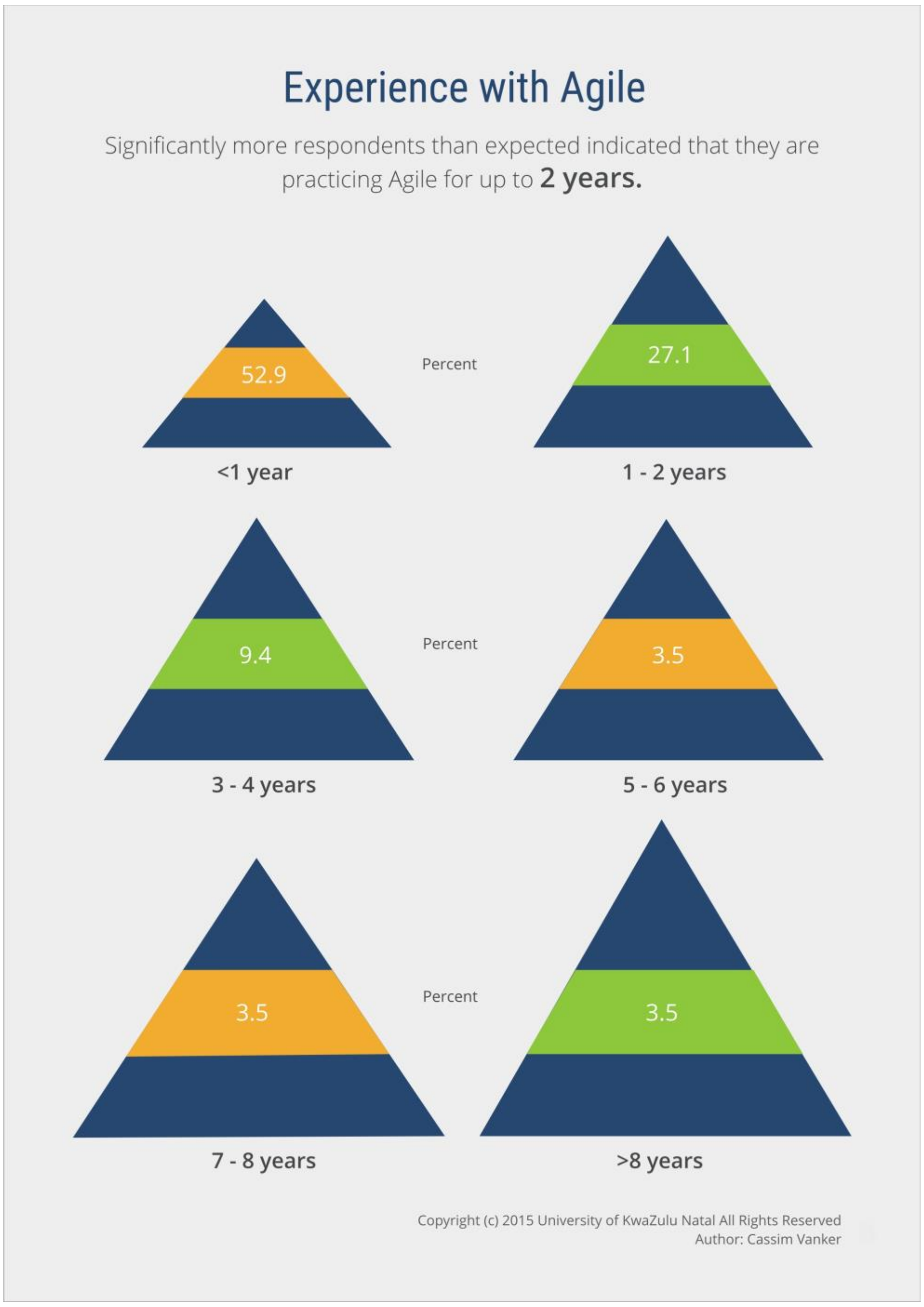


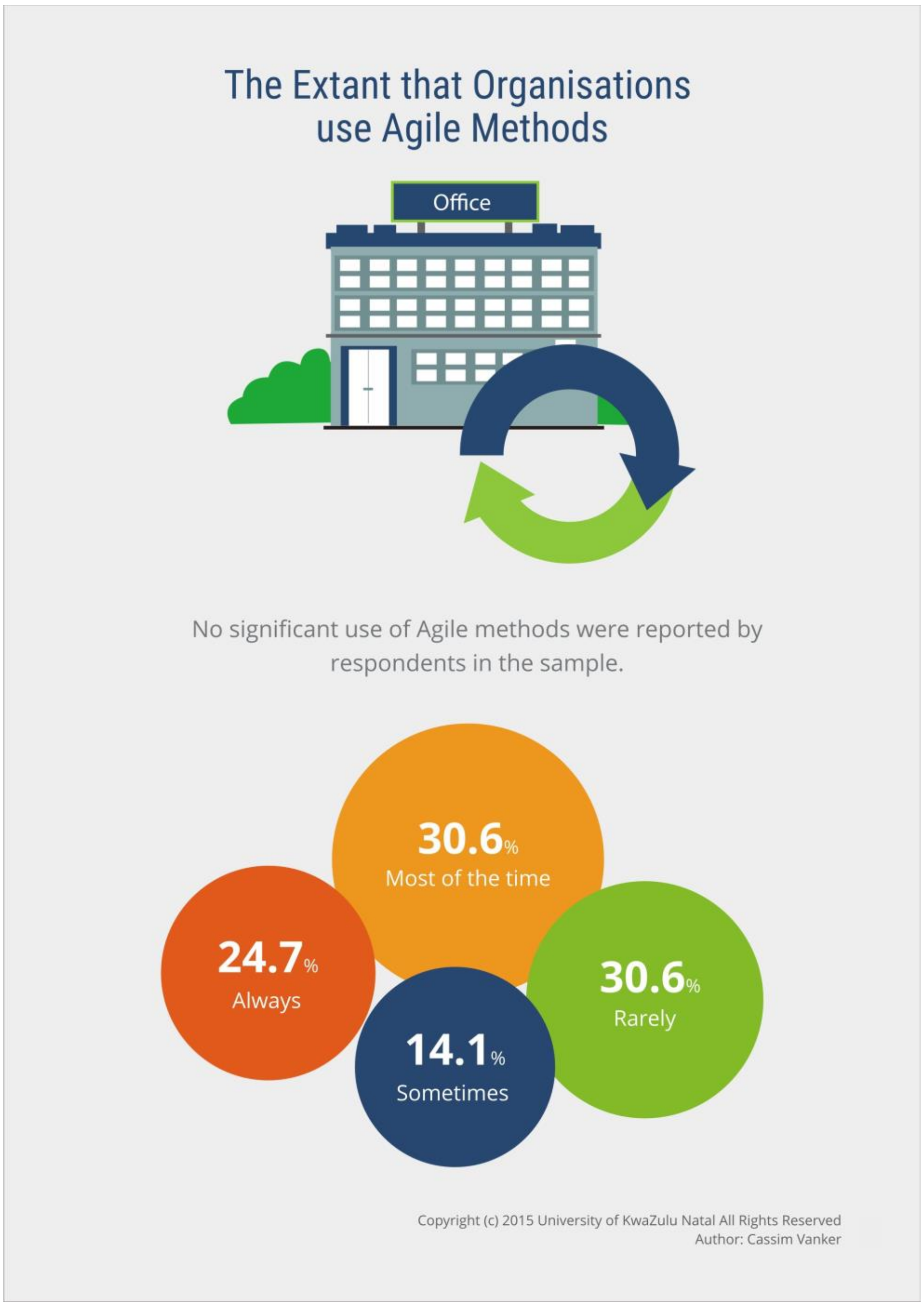
Position of Respondents in the Sample



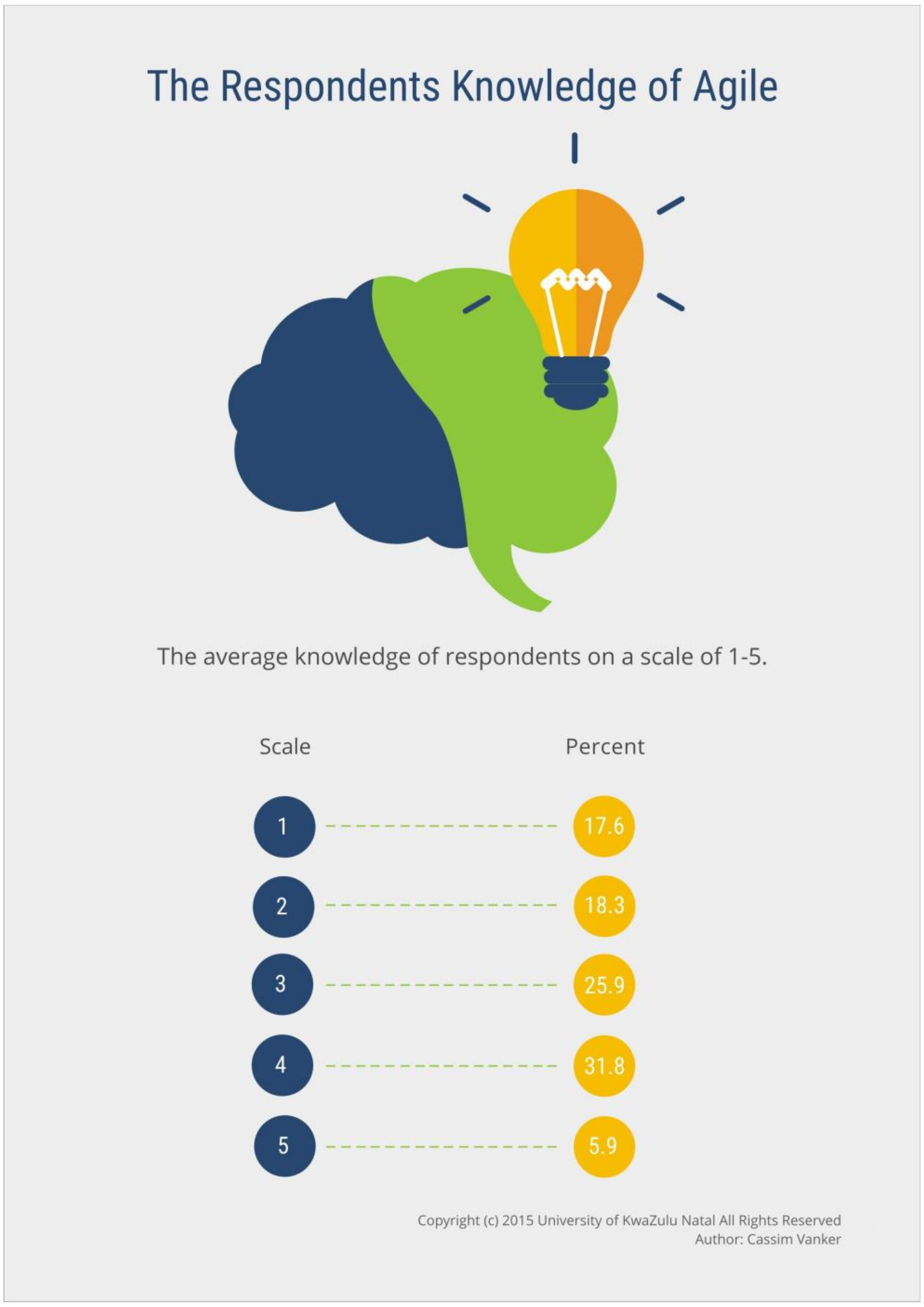
A significant number of the respondents from the sample were software developers. This was followed by team leads, project managers and development managers respectively.




Experience with Agile
Significantly more respondents than expected indicated that they are practicing Agile for up to **2 years.**
52.9
<1 year
Percent
27.1
1 - 2 years
9.4
3 - 4 years
Percent
3.5
5 - 6 years
3.5
7 - 8 years
Percent
3.5
>8 years
Copyright (c) 2015 University of KwaZulu Natal All Rights Reserved
Author: Cassim Vanker


The Extant that Organisations use Agile Methods

No significant use of Agile methods were reported by respondents in the sample.

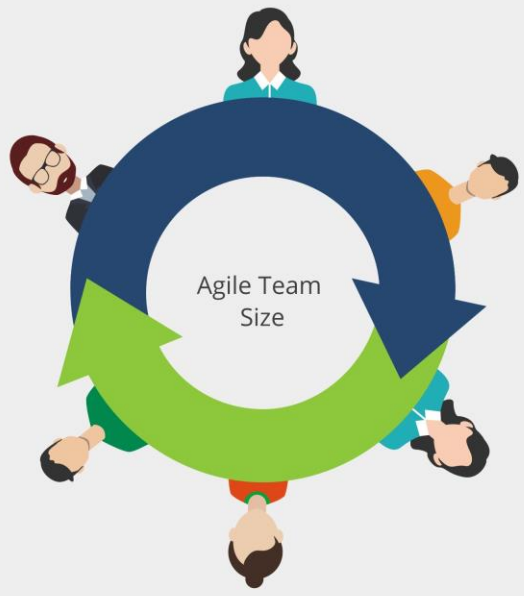


The Respondents Knowledge of Agile

The average knowledge of respondents on a scale of 1-5.

Scale	Percent
1	17.6
2	18.3
3	25.9
4	31.8
5	5.9

The Size of Agile Teams



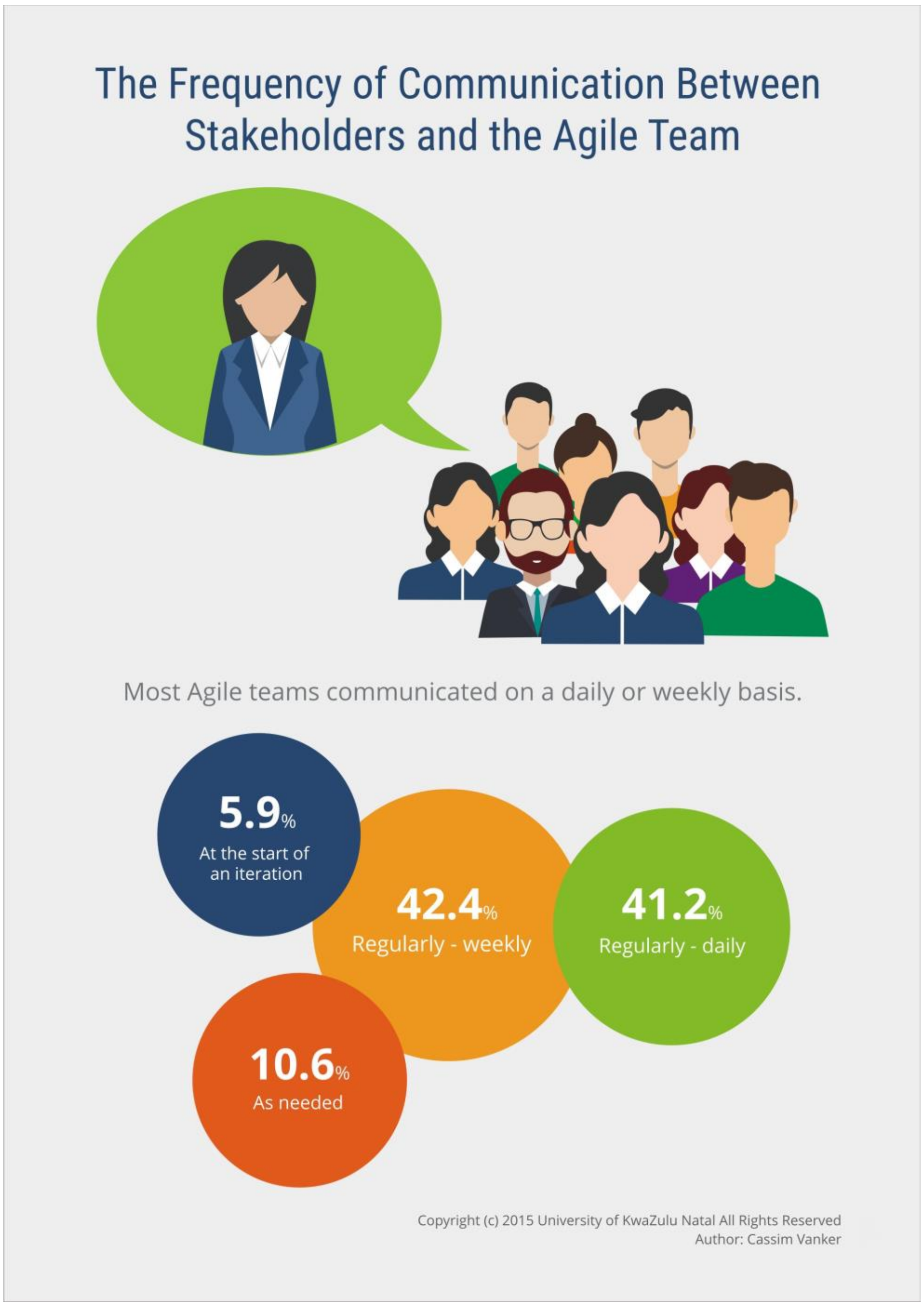
More than expected respondents had Agile team size between 1 to 5 members.

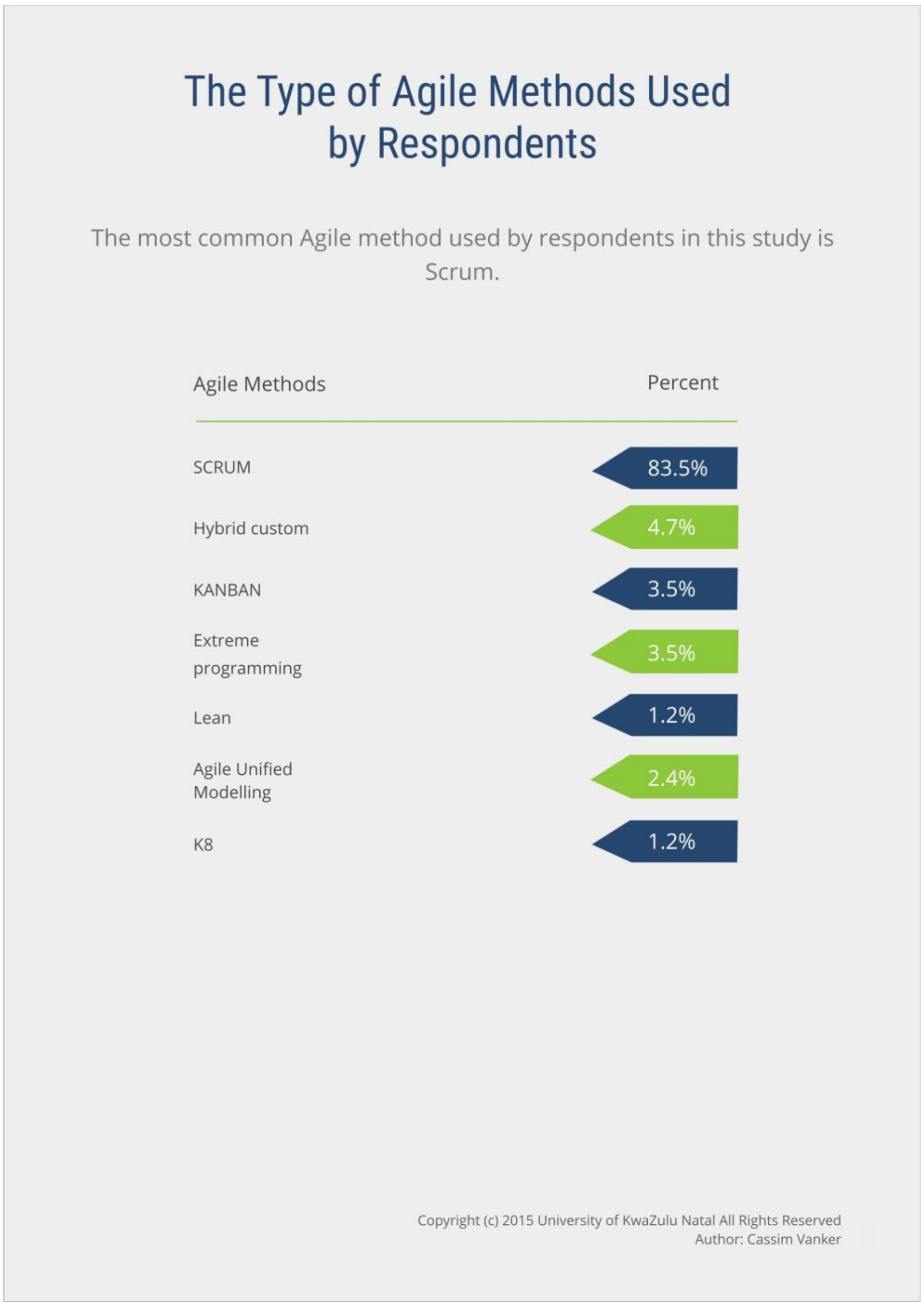
Scale	Percent
1 - 5	43.5
6 - 10	43.5
11 - 15	8.2
16 - 20	2.4
>30	2.4

The Geographical Distribution of Agile Teams

More than expected respondents have Agile teams in the same room.

Geographical Distribution	Percent
Same room	45.9
Same floor	12.9
Same building	11.8
Same campus	10.6
Within driving distance	10.6
Within same time zone	5.9
Different time zones	2.4

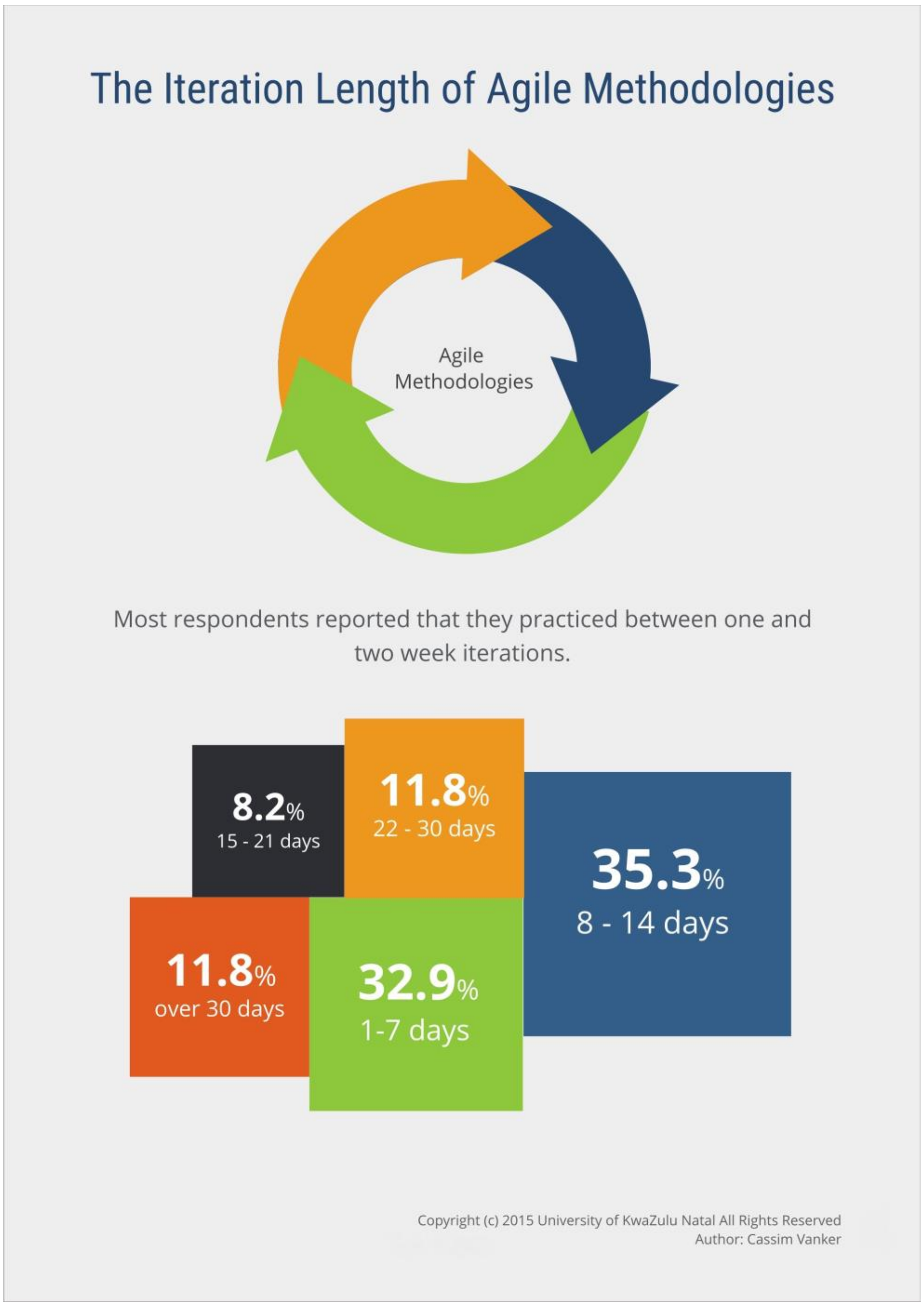

The Frequency of Communication Between Stakeholders and the Agile Team
Most Agile teams communicated on a daily or weekly basis.
5.9% At the start of an iteration
42.4% Regularly - weekly
41.2% Regularly - daily
10.6% As needed
Copyright (c) 2015 University of KwaZulu Natal All Rights Reserved
Author: Cassim Vanker

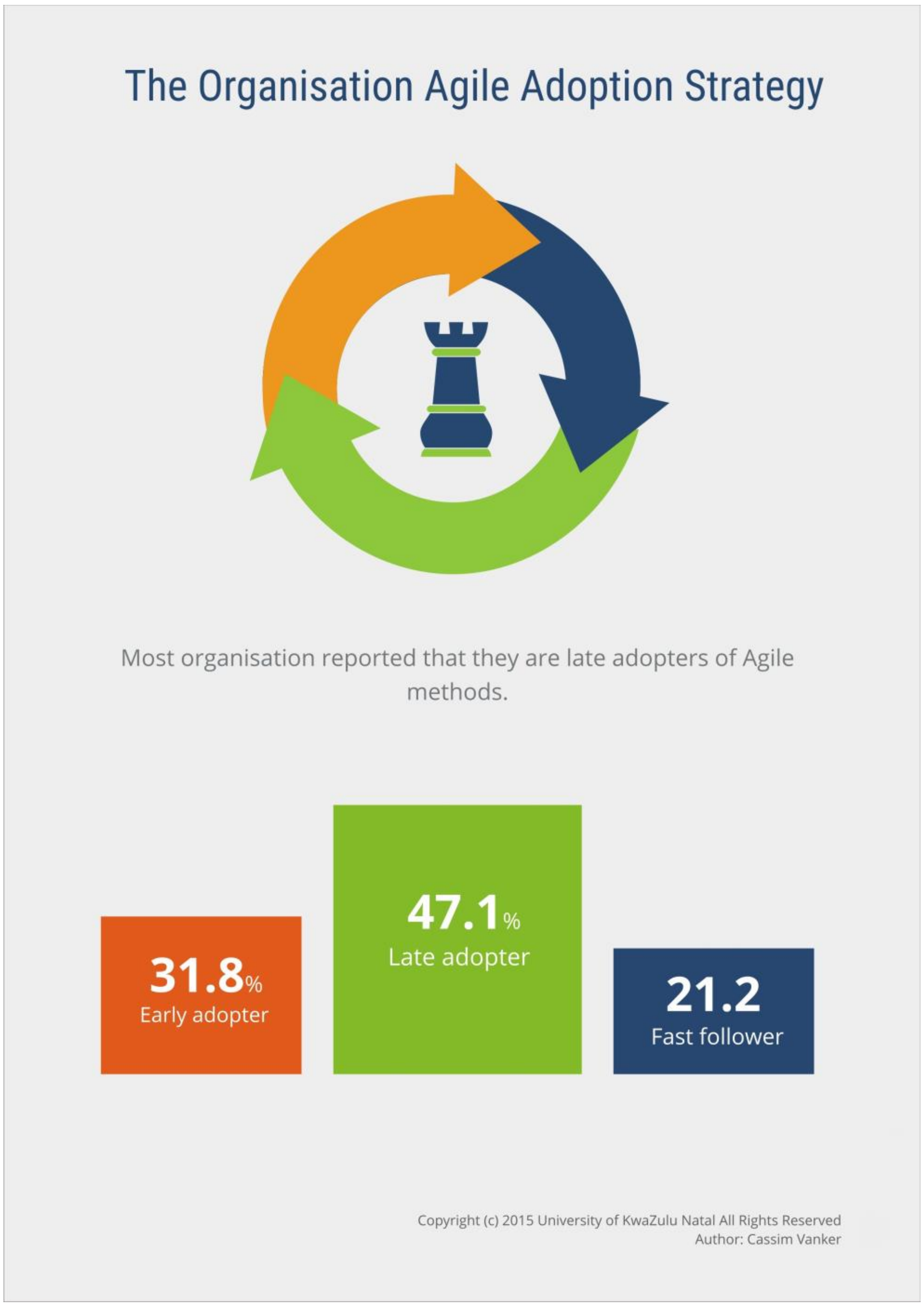


The Type of Agile Methods Used by Respondents

The most common Agile method used by respondents in this study is Scrum.

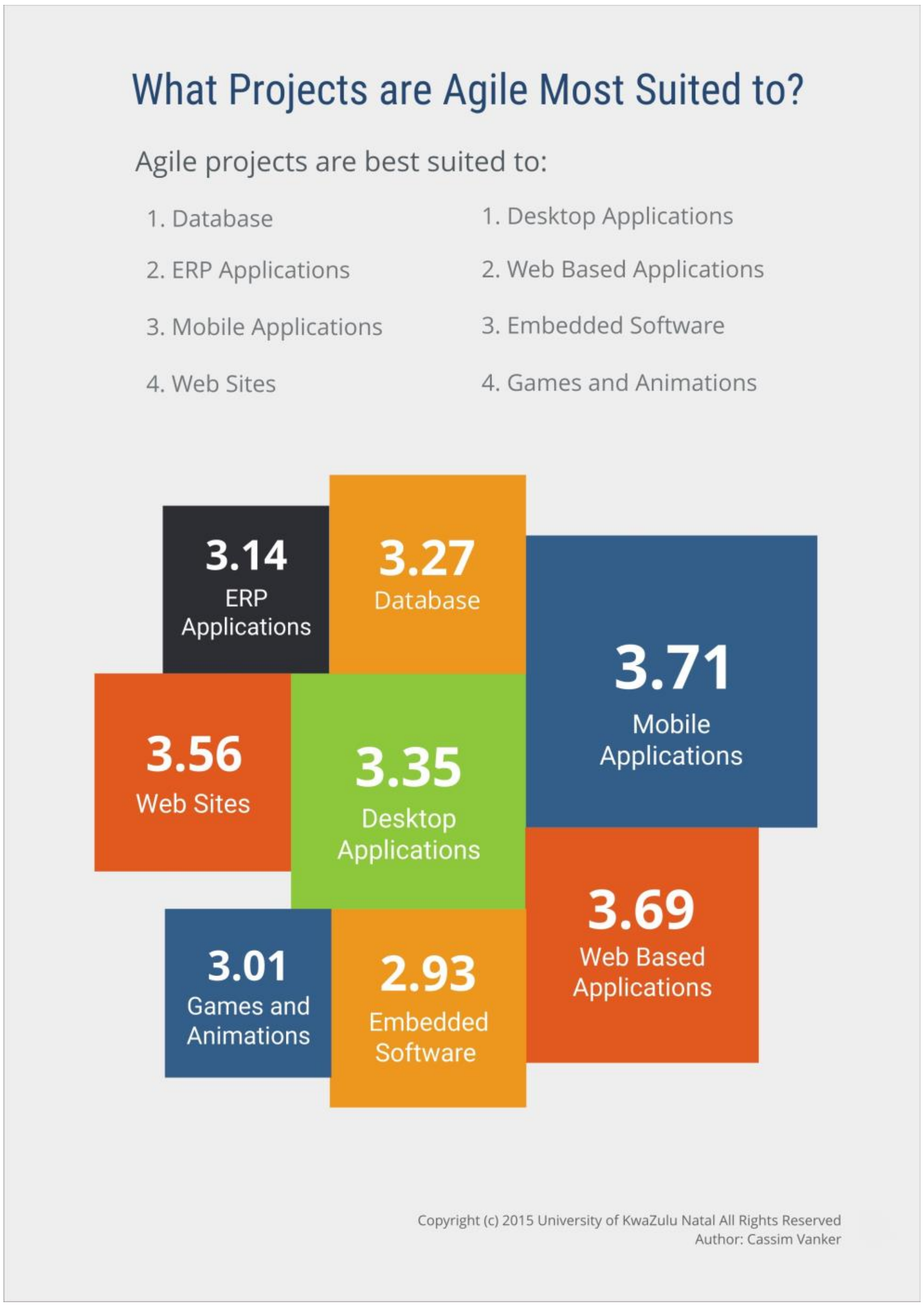
Agile Methods	Percent
SCRUM	83.5%
Hybrid custom	4.7%
KANBAN	3.5%
Extreme programming	3.5%
Lean	1.2%
Agile Unified Modelling	2.4%
K8	1.2%


The Iteration Length of Agile Methodologies
Agile Methodologies
Most respondents reported that they practiced between one and two week iterations.
8.2% 15 - 21 days
11.8% 22 - 30 days
35.3% 8 - 14 days
11.8% over 30 days
32.9% 1-7 days
Copyright (c) 2015 University of KwaZulu Natal All Rights Reserved
Author: Cassim Vanker



The Organisation Agile Adoption Strategy

Most organisation reported that they are late adopters of Agile methods.



What Projects are Agile Most Suited to?

Agile projects are best suited to:

1. Database	1. Desktop Applications
2. ERP Applications	2. Web Based Applications
3. Mobile Applications	3. Embedded Software
4. Web Sites	4. Games and Animations

6.11 The Extent to Which the Results Enables the Research to Answer the Research Questions

At the beginning of this study three research questions were proposed. These research questions are presented below as a reference point:

- What is the state of Agile software development in South Africa?
- What are the barriers preventing the adoption of Agile by South African organisations?
- What is the impact of Agile methodologies on the development of software in South Africa?

In the following sections, the researcher will answer these questions based on the results from the fieldwork.

6.11.1 What Is the State of Agile Software Development in South Africa?

The fieldwork showed that there is no conclusive evidence to suggest that organisations are extensively practicing Agile methods. On the contrary organisations are using a combination of methods to develop software. According to Schwaber & Sutherland the Ralph Stacy graph is a method that can be used to determine, which projects are suitable for Agile methods (Schwaber & Sutherland, 2012). Furthermore, projects with high risk and uncertainty are better aligned to Agile methods (Schwaber & Sutherland, 2012).

In line with the use of a variety of software development methods the fieldwork could not conclusively prove that respondents have or did not have significant knowledge of Agile methods.

Most respondents in this study indicated that they had been practicing Agile methods for two years. Comparing these results to the international study by VersionOne in 2014 shows that most respondents internationally had between 3 to 4 years of experience with Agile methods (VersionOne, 2014). This response indicates that South African organisations are behind in adopting Agile methods. This is further substantiated by respondents when they were asked about their organisation Agile adoption strategy. Most respondents indicated that they were late adopters.

The general size of Agile teams in South Africa is between 1 to 10 members with most of the teams collocated. Stakeholder engagement is on a daily or weekly basis. The most common Agile method practiced is Scrum and this is also in line with international trends. The VersionOne study in 2014 showed that 56% of respondents practiced Scrum, while the fieldwork shows that 83.5% of respondents practiced scrum (VersionOne, 2014).

Most respondents were practicing between one and two week iterations. Respondents also felt that Agile methods were best suited for web sites, web based applications, desktop applications and mobile applications. Furthermore, respondents suggested that their last Agile project could be characterised as successful. This characterised success of Agile projects is also in line with studies by the Standish Group, which shows that Agile projects are more likely to succeed than traditional projects (Schwaber & Sutherland, 2012).

6.11.2 What Are the Barriers Preventing the Adoption of Agile by South African Organisations?

According to the fieldwork organisations in South Africa are late adopters of Agile methods. Furthermore, the data analysis showed that:

- The culture within an organisation was a barrier to adoption.
- That not having access to skilled personnel is a barrier to adoption.
- General resistance to change is a barrier to adoption.
- The lack of management support is a barrier to adoption.

The international study by VersionOne in 2014 showed several similarities. The study showed that 44% of respondents reported that organisation culture was a barrier to Agile adoption and 35% reported that not having personnel with the right skill was a barrier to adoption (VersionOne, 2014). Furthermore, 34% suggested that general resistance to change was a barrier to adoption and finally 29% cited the lack of management support as a barrier to adoption (VersionOne, 2014). From the above we can see that internal barrier to Agile adoption and local barrier to Agile adoption are similar in nature.

The fieldwork also showed that knowledge and experience of Agile methods played a significant role in the characterisation of those projects. An interesting observation is that respondents with high perceived knowledge agreed with the following adoption barriers:

- Not having access to personnel with the right skills is an impediment to Agile adoption.
- The lack of management support is an impediment to Agile adoption.

However, they were in disagreement with the following adoption barriers:

- Complex development projects are an impediment to Agile adoption.
- Agile's inability to scale up for large projects is an impediment to adoption.
- The lack of upfront planning in Agile projects is a cause of concern when adopting Agile.
- The lack of documentation in Agile projects is a cause of concern when adopting Agile.

The level of experience also plays a significant role in the characterisation of Agile projects. Most respondents with under one year of experience with Agile methods reported that their projects had failed while respondents with over 2 years of experience with Agile methods reported successful projects. The results correlate with the survey by VersionOne in 2014 which reports that 44% of their respondents had indicated that the lack of experience with Agile methods is a leading cause of project failure (VersionOne, 2014).

6.11.3 What Is the Impact of Agile Methodologies On the Development of Software in South Africa?

The results of the fieldwork show that most Agile projects can be characterised as successful. There are also significant differences in respondent's perceived knowledge of Agile and the state of their last Agile project. Respondents who categorized their last Agile project as failed rated their knowledge as lower than those who categorized their last AGILE project as 'too early to tell'; 'successful' or 'challenged'. The VersionOne study in 2014 further supports this by reporting that 30% of their respondents had indicated that insufficient training was a leading cause of failure for Agile projects (VersionOne, 2014).

Furthermore, the fieldwork also showed that those who reported that their last Agile project was successful said that they had more knowledge than those who didn't know the state of their last Agile project.

Respondents also identified the following benefits of agile adoption:

- Agile methods help teams manage changing priorities.
- Agile methods improve project visibility.
- Agile methods increase productivity.
- Agile methods provide a faster time to market.
- Agile methods provide better alignment between IT and business.
- Agile methods improve the quality of software being developed.
- Agile methods improve team moral.
- Agile methods simplify the development process.
- Agile methods enhanced software maintainability and extensibility.

It is also worth noting that respondents with significantly higher perceived knowledge of Agile methods only agreed with the following benefits:

- Agile methods help teams manage changing priorities
- Agile methods improve project visibility
- Agile methods increase productivity
- Agile methods provide a faster time to market
- Agile methods improve the quality of software being developed
- Agile methods improve team moral

The international survey by VersionOne in 2014 also identified the above six benefits as amongst the leading benefits associated with Agile adoption (VersionOne, 2014).

6.12 CHAPTER SUMMARY

This chapter presents the results of the fieldwork conducted during this study. The chapter commences by outlining the different sections in the questionnaire. A brief explanation of the data extraction process is also provided followed by an explanation of the statistical tests that was used to analyse the data.

Thereafter the result of each question is presented in following sections:

- Personal Demographics.
- Business Demographics.
- The state of Agile in South Africa.
- Barriers preventing Agile adoption in South Africa.
- The impact of Agile on software development in South Africa.

Bivariate data analysis was conducted and those results are also presented in this chapter. Further statistical results of each test are also provided in the addendum. Finally, the chapter concludes by presenting answers to the research questions.

Chapter 7

THEORETICAL FRAMEWORK, LITERATURE REVIEW AND RESEARCH TOPIC

7.1 INTRODUCTION

In chapter 1 the research design framework for this study was presented together with a brief overview of the literature. In chapter 2 the different theoretical models were presented and an assessment was done on each model. Ultimately the Gartner hype cycle (GHC) was selected as being the most appropriate theoretical model. In chapter 4 a detail review of the Gartner hype cycle is presented. This chapter closes the loop by establishing a link between the chosen theoretical framework, the literature review and the research topic.

New innovations offer opportunities to provide solutions to unsettling problems (Janes & Succi, 2012). The release of these innovations usually trigger a phase of hype (Janes & Succi, 2012). However, when the dust settles these expectations often turn out as unrealistic (Fenn & Raskino, 2008). This phenomenon is described by Gartner as the hype cycle (GHC) (Janes & Succi, 2012).

The literature on the GHC shows that a number of large organisations are using the hype cycle to make decision about their adoption practices (Steinert & Leifer, 2010). As such one of the contributions of this study is to determine the location of Agile methods on the GHC.

Through this contribution the researcher hopes to provide South African organisations with better information so that they can make informed decisions with regards to Agile adoption.

7.2. THE TRIGGER

The GHC kicks off when an innovation generates significant attention in the field and this is called the technology trigger (Fenn & Raskino, 2008; Janes & Succi, 2012). For Agile methods this trigger was the release of the Agile manifesto in 2001 (Beck *et al.*, 2001a). In computer science the term Agile was not popular until the advent of agile software development (Janes & Succi, 2012). This is quite evident when you look at figure 7-1 below, which shows the number of hits associated with computer science terms that have been combined with Agile. The table is adapted from Janes & Succi and includes statistics from 2015 for comparison purposes.

Keywords	**No of hits in September 2015**	**No of hits in April 2012**
"Agile documentation"	1,283,840	35,100
"Agile management"	970,080	279,000
"Agile modeling" (sic)	1,431,740	117,000
"Agile product management"	1,298,670	323,000
"Agile testing"	1,003,600	716,000
"Agile web development"	1,086,910	364,000
"Agile database development"	531,850	186,000
"Agile ruby"	489,840	103,000
"Agile AJAX"	85,650	100,000
"Agile SOA"	110,710	79,600
"Agile architecture"	1,465,030	66,100
"Agile game development"	200,270	32,100

Figure 7-1: Table showing the search hits on Google for Agile related terms. Adapted from Janes & Succi (Janes & Succi, 2012; Google, 2015a).

The Agile manifesto was released in order to rally the industry behind agile practices (Fowler, 2005). According to Janes & Succi one of the reasons why the Agile manifesto prospered is because it was not defined precisely (Jeffries, 2010; Janes & Succi, 2012). Jeff Sutherland, one of the signatories of the manifesto, believes that each individual Agile methodology approaches the manifesto values in different ways. However, all of the methodologies have specific practices and processes that foster one or more of the manifesto values (Judy, 2012).

Furthermore, the flexibility of Agile methods brought new perspective to software development. It's ability to adapt to variations in the market and to provide value to the customer added to the hype (Janes & Succi, 2012). Agile gurus and consultants climbed on the bandwagon and created a contingent effect.

7.3. TRENDS

In order to get a clearer picture of the 'hype' around Agile a search was conducted on the term "Extreme Programming" using Google Trends. Furthermore, the search was conducted for the keyword "Extreme Programming" in the programming category between 2004 and 2015. It must be noted that Google Trends at the time the search was done only supplied data from 2004 onwards. A report with this data was extracted and is presented in the form of a graph below.

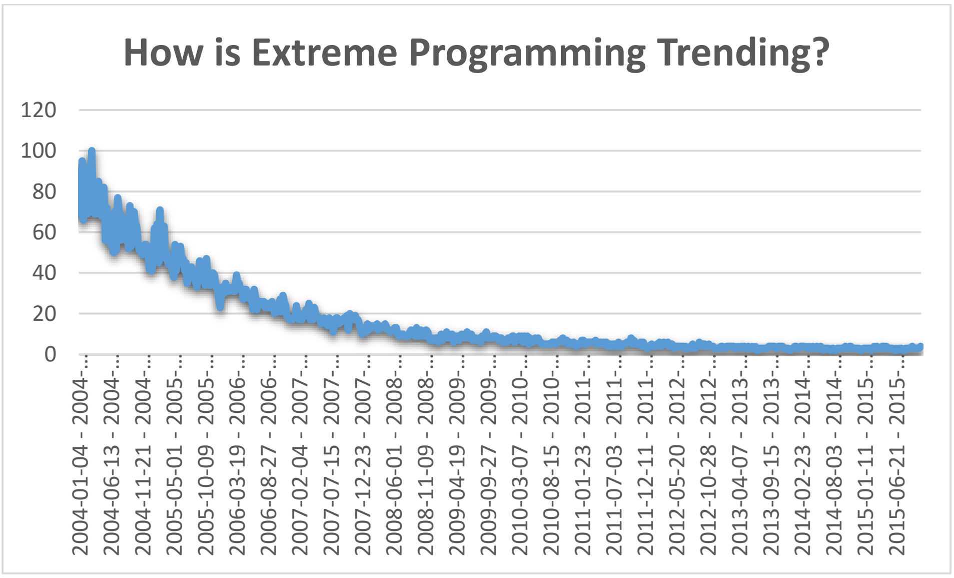


Figure 7-2: Line chart showing worldwide interest for Extreme Programming from 2004 to 2015. Adopted from Google Trends (Google, 2015b).

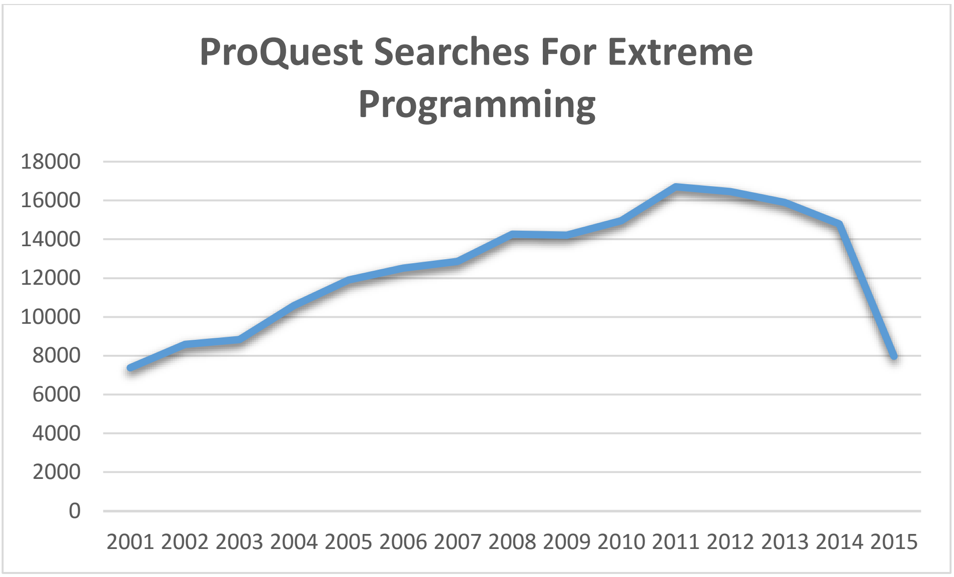


Figure 7-3: Line chart showing the results of ProQuest searches for Extreme Programming from 2001 to 2015. Adopted from ProQuest (ProQuest.com, 2015).

Looking at the above Google Trends statistics in figure 7-2 above, you will notice that the word "Extreme Programming" has been trending since 2004. The most number of searches for "Extreme Programming" was in 2004. However, since then there has been a decreasing trend in Google searches. This decreasing trend is also evident in the ProQuest searches shown in figure 7-3 above. According to Fenn & Raskino it takes between five to eight years for innovations to complete the hype cycle (Fenn & Raskino, 2008). However, an analysis of Agile methods on Gartner's hype cycle for application development from 2003 to 2015 shows the following:

	Sliding Into the Trough	**Climbing the Slope**	**Entering the Plateau**	**Not On Hype Cycle**
2015				X
2014		X		
2013	X			
2012	X			
2011	X			
2010	X			

	Sliding Into the Trough	**Climbing the Slope**	**Entering the Plateau**	**Not On Hype Cycle**
2009	X			
2008	X			
2007	X			
2006				X
2005		X		
2004		X		
2003			X	

Figure 7-4: Table showing Agile methods on the Gartner hype cycle from 2003 to 2015. Adapted from Gartner's hype cycle for application development (Gartner, 2003, 2004, 2005, 2006, 2007, 2008b, 2009; Blechar, 2010; Blechar & Finley, 2011; Finley, Wilson & Van Huizen, 2012; Wilson, Van Huizen & Prentice, 2013; Murphy, Wilson & Sobejana, 2014; Sobejana & Murphy, 2015).

When compiling the above table, the following was noted:

- In 2003 Agile methods were termed by Gartner as "Rapid Application Development/Agile Development".
- Between 2004 and 2010 Gartner called Agile methods "Agile Development Methodology".
- From 2011 onwards Agile methods were divided into "Project-Oriented Agile Development Methodology" and "Enterprise-Class Agile Development". Enterprise Agile refers to the application of Agile methods in an enterprise and according to Gartner this differs from Agile in smaller organisations (De Haan, 2011). As such this table only includes project oriented Agile development as this encapsulates the principles and process of Agile.

It is clearly evident from the above table that the plotting of Agile methods by Gartner on the GHC is inconsistent. In 2003 Gartner reported that Agile was entering the Plateau. However, in 2004 they reported that Agile methods were climbing the slope. In 2006 Agile methods seem to have vanished from the GHC. Furthermore, from 2007 to 2013 Gartner reported that Agile was sliding into the trough until it finally moved into the slope in 2014 and vanishes from the hype cycle in 2015. The table also shows that Agile methods have been in the hype cycle for more than eight years.

From figure 7-4 above it is evident that according to Gartner from 2007 Agile methods were in the trough of disillusionment (Gartner, 2007). The fieldwork of this study further supports the decreasing trend in the use of Agile methods as it shows that Agile is not the dominant method being used by companies in South Africa. On the contrary a variety of methods are being used by organisation in South Africa. Furthermore, contrary to Gartner's findings evidence support that fact that Agile methods are still in the trough of disillusionment and this will be presented in the next section.

7.4. THE TROUGH OF DISILLUSIONMENT

The publication of the Agile manifesto saw Agile methods proposed as the solution to the common software development problem (Janes & Succi, 2012). That is the demand for more flexibility in the software development process (Janes & Succi, 2012).

According to Janes & Succi software development is a lot like construction and over time a software house can predict the time and cost of development as new projects are similar to previous ones (Janes & Succi, 2012). Arguing that the benefits of Agile techniques might not necessarily be the reason for the success of a project.

An interesting argument provided by Fowler in his blog asserts that simple repetitive tasks are best managed using traditional techniques and tasks that require creativity are best managed using Agile techniques (Fowler, 2005). With this argument in mind Janes & Succi concludes that both traditional and Agile techniques have the right to coexist (Janes & Succi, 2012). However, the problem occurs when one method is used in a context where it does not belong.

It has also been reported that companies are often unaware of what Agile adoption really means (Laanti *et al.*, 2011). Thomas, a signatory of the Agile manifesto, in his blog post “Agile is dead” writes that the word “Agile” has been subverted to a point where it is meaningless (Thomas, 2014). He suggests that consultants and vendors have skewed the Agile manifesto by promoting activities that are process and tool intensive and placed emphasis on more planning activities than Agile requires.

The problem with Agile methods was expounded when Agile gurus used the benefits of Agile methods to further help evangelize the Agile hype (Janes & Succi, 2012). Geoff, aka "Gilligan” in a blog post put this thought through quite eloquently when he says (Preuss, 2006a: 1):

“There is a type of person that I call an Agilista. Agilistas are nothing more than opportunists that have jumped on the Agile bandwagon and they shout come down to the river and be healed. These Agilistas don't understand development, they don't know why Beck, Cockburn, Jeffries, Fowler, or whomever suggest specific practices. These Agilistas don't even know what the real problems are...”

In her opinion piece Preuss suggest that Agile has fallen victim of good and bad practices and the continued practice of bad Agile has led to Agile falling into the trough of disillusionment (Preuss, 2006b). Hazrati also suggest that over time Agile became “bad” perhaps because of the side effect of its increased popularity (Preuss, 2006a; Hazrati, 2011).

Janes & Succi suggest that a dark side of Agile emerged as the early subscribers of the Agile manifesto become zealot (Janes & Succi, 2012). They wanted to be better and interpreted the Agile manifesto as follows (Janes & Succi, 2012):

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

- Individuals and interactions ~~over~~ **and not** processes and tools
- Working software ~~over~~ **and not** comprehensive documentation
- Customer collaboration ~~over~~ **and not** contract negotiation
- Responding to change ~~over~~ **and not** following a plan"

"That is, ~~while~~ **since** there is **no** value in the items on the right, we value only the items on the left ~~more~~" (sic) (Janes & Succi, 2012).

According to Pietri and Moreira shortly after the hype Agile fell into a chasm (Pietri, 2011; Moreira, 2015). Geoffrey Moore in his book "Crossing The Chasm" spoke about the gap between early adopters and the rest of the market (Pietri, 2011; Moreira, 2015). Moore surmised that early adopters are comfortable with changing their behaviours in order to jump ahead of the competition (Pietri, 2011). In pursuit of innovation early adopters are followed by early majority, late majority and laggards (Linowes & Parker Hill Technology, 1991).

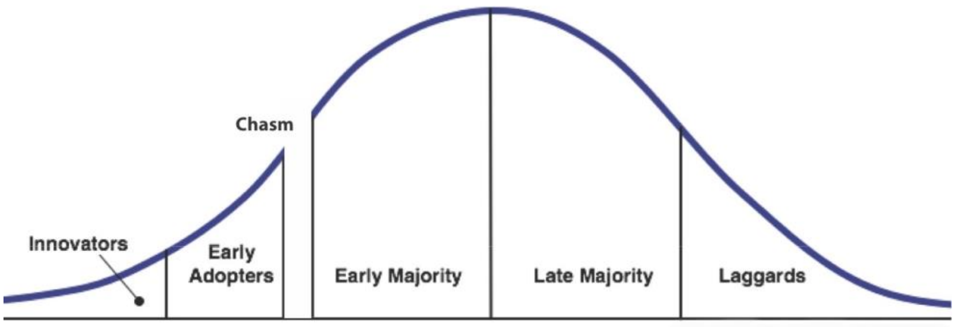


Figure 7-5: Graphic showing the Technology Adoption Life Cycle. Adopted from Linowes & Parker Hill Technology (Linowes & Parker Hill Technology, 1991).

Early majority adopters are pragmatists and want technology to enhance their established way of doing business (Linowes & Parker Hill Technology, 1991; Pietri, 2011). According to Moreira most organisations think they are practicing Agile; however, all they are doing is Fragile ("fake Agile")

(Moreira, 2015). The recent publication of the Anti-Agile manifesto is classic example of Fragile (Holub, 2014). According to the Anti-Agile manifesto:

"…Agile is simply the obfuscation of common sense…

- epics are really just projects
- stories are really just use cases
- sprints are really just work
- stand-ups are really just meetings
- iterations are really just versions
- backlogs are really just to do lists
- backlog grooming is really just planning
- burn-down charts are really just earned value charts
- velocity is really just output
- and that tasks, in fact, are really just tasks.

That is, while the concepts on the left are often presented as groundbreaking or unique, they are merely weakly defined versions of those on the right" (sic) (Holub, 2014). According to Holub the Anti-Agile manifesto contains "a lot of misinformation" and this does active damage to those trying to figure out Agile practices (Holub, 2014).

The Anti-Agile manifesto is a classic example of the sentiments amongst some of the Agile adopters as they try to align their existing process to Agile methods. However, Moreira suggest that in order for organisation to cross the chasm they must be willing to change their behaviour (Moreira, 2015). This is quite evident from the result of the fieldwork. The results show that respondents identified the culture within the organisation as well as the lack of management support as barriers to adoption. Studies have shown that the culture within an organisation has a significant impact on the decision making processes (Nerur *et al.*, 2005). According to Nerur *et al.*, organisations steeped in traditional methods are likely to experience several challenges when adopting Agile methods as Agile and traditional methods are grounded in opposite directions (Nerur *et al.*, 2005).

The role of management in Agile projects is also significantly different. Project managers move from a “planning and controlling” role to that of a facilitator (Nerur *et al.*, 2005). Nerur *et al.*, asserts that the biggest challenge for project managers when adopting Agile methods would be for them to relinquish control (Nerur *et al.*, 2005). Furthermore, changing the culture and mind-set of people is a formidable task and this makes the change to Agile difficult for many organisations (Nerur *et al.*, 2005).

It has also been reported that Agile methods are growing from the bottom up and therefore require executive management support (Schatz & Abdelshafi, 2005; Livermore, 2007; Atlas, 2009). Furthermore, performance appraisal activities in many organisations neglected to take into consideration Agile methods (Conboy *et al.*, 2011). This sometimes results in an incorrect appraisal of the team’s performance.

Trying to look back at why Agile got stuck the critics surmise that the concepts and practices that made Agile popular began to ultimately lead to its downfall. Preuss, Janes & Succi, Wilson, Bishop, Marshall, Makabee and Keeffe also suggested that Agile methods are in the trough of disillusionment (Preuss, 2006a,b; Janes & Succi, 2012; Wilson, 2012; Bishop, 2014; Makabee, 2014; Marshall, 2014; Keeffe III, 2015).

According to the above authors there are a number of reason as to why Agile is in the trough of disillusionment and these arguments are summarized below:

- The lack of understanding about Agile methods has led to incorrect practices (Nerur *et al.*, 2005; Bishop, 2014; Ottinger, 2014). Furthermore, there is a chasm between idea Agile and realized Agile (Dhurka, 2015).

- Agile practitioners are copying Agile practices rather than growing them (Preuss, 2006a). Agile teams need to foster Agile practices and values together in order to excel (Preuss, 2006a).
- In Agile methods requirements are solicited over a period of time. As such critics argue that developers do not have a complete picture and this could result in wasteful effort (Janes & Succi, 2012).

Furthermore, Janes & Succi suggests that the vast majority of project managers assume that software development is a project and should therefore be managed as such (Janes & Succi, 2012). They surmise that project managers regard Agile as a "strange beast" as it goes against the recognised standard in project management defined in the Project Management Body of Knowledge (PMBOK).The lack of understanding of Agile principles has further compounded the problem as Agile has often been described as (Janes & Succi, 2012):

- Giving developers the freedom to do what they want
- No upfront design or planning as time is only spent on code
- A lack of discipline and no documentation
- Provides inadequate preparation of development
- Ignores the consequences of risk by allowing stakeholders to change their requirements.

As mainstream support grew for Agile methods, it slowly became clear that the industries inflated expectations could not be met (Janes & Succi, 2012). The attention moved to other buzzed words such as Kanban and Lean (Janes & Succi, 2012). The above factors further suggest that the early majority adopters might not truly understand the ramifications of organisation wide adopting of Agile methods (Nerur *et al.,* 2005; Chandra Misra *et al.,* 2010).

Janes & Succi believe that if we want to be able to use Agile Methods effectively and efficiently one needs to understand "why it works" and "how it works” and “when it pays off” (Janes & Succi, 2012). As the answers to these questions will help Agile reach the plateau of productivity (Janes & Succi, 2012).

7.5. The Way Forward

An interesting article by Dhurka suggests that the way forward for Agile methods is for organisations to plot their own Agile hype cycles and confront the hypes and troughs and assess how to creatively push Agile process to the plateau of productivity (Dhurka, 2015). Dhurka defines her idea of the Agile hype cycle below (Dhurka, 2015):

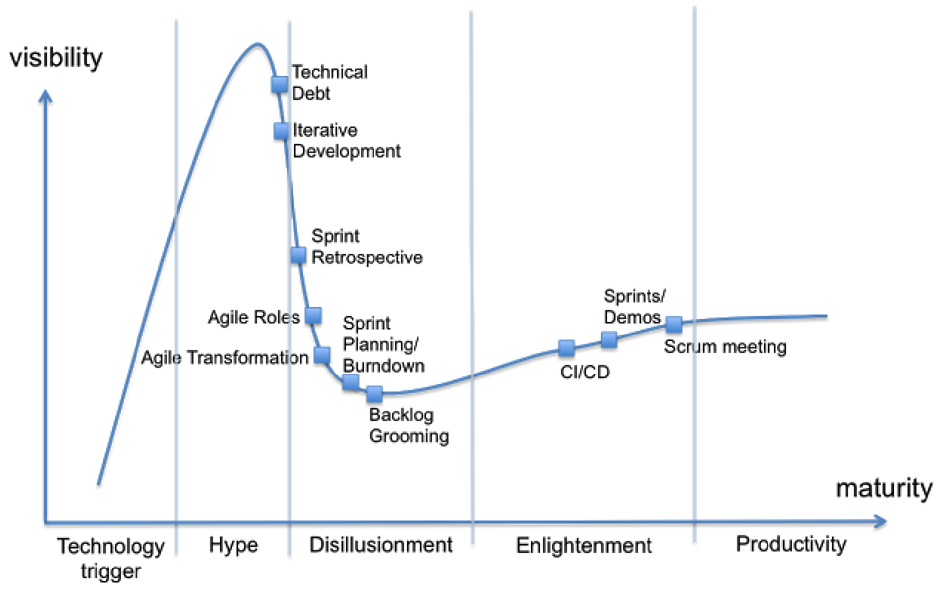


Figure 7-6: Graphic showing the Agile hype cycle. Adopted from Dhurka (Dhurka, 2015).

Agile Component	**Status**
Scrum meetings: In the slope of enlightenment	Teams have come to value and adapted the quick daily sync up to their needs.
Sprints Reviews: In the slope of enlightenment	Demonstrations help share information in teams.

Sprint Planning and Burn down: In the trough of disillusionment	Estimations are difficult to predict. Burn down charts are tedious to maintain.
Backlog Grooming: In the trough of disillusionment	There is no prescription on the amount to groom a backlog.
Sprint Retrospective: In the trough of disillusionment	Run at team level and have little influence on the organisation.
CI/CD: Slope of enlightenment	CI/CD has been accepted as a defacto deployment and release model.
Iterative Development: Peak of hype	The argument that iterative development promotes sloppy designs, which ultimately is costly over time.
Technical Debt: **Peak of hype**	The consequence of sloppy design or code.
Agile Roles Trough of disillusionment	Some Agile roles overlap with traditional titles and this can lead to unnecessary tension. Furthermore, organisations aren’t great at mapping Agile roles to appraisal goals and this stifles career growth.
Agile Transformation Trough of disillusionment	Ideal Agile transformation always seems beyond the reach of the organisation and this leaves the organisation in a transition fix.

Figure 7-7: Table showing the state of Agile processes on the Agile hype cycle. Adopted from Dhurka (Dhurka, 2015).

Dhurka concludes by suggesting that Agile methods have a bunch of good things about them but are also flawed (Dhurka, 2015). She suggest that the solution is for organisations to address these flaws to prevent themselves from being fixed in a transition state (Dhurka, 2015).

7.6. Chapter Summary

This chapter provided a link between the chosen theoretical framework, the literature review and the research topic. The chapter commences by describing the trigger event that leads to the release and hype of Agile methods. Using real world statistics, the chapter discusses the various arguments and trends related to Agile methods and ultimately pegs Agile methods as being in the trough of disillusionment. Finally, the chapter concludes by presenting a way forward for organisations to push Agile methods toward the plateau of productivity. The next chapter is the final chapter of this study. It describes the conclusions, limitations and recommendations for future research.

Chapter 8

CONCLUSIONS, LIMITATIONS AND RECOMMENDATIONS

8.1 INTRODUCTION

In 2001 a group of programmers got together to agree on a set of principles that make up the Agile manifesto (Beck *et al.*, 2001a). This Agile manifesto became the foundation for Agile methods around the world (Judy, 2012). The popularity of Agile methods began to grow and this is evident from a number of studies that showed the increased use of Agile methods (VersionOne, 2010, 2011, 2013).

As popularity grew, Agile adoption moved from early adopters to early mainstream adopters (Pietri, 2011). This caused a chasm to form between the early adopters and early mainstream adopters (Pietri, 2011). In 2007 Gartner announced that Agile methods were in the trough of disillusionment (Gartner, 2007). The fieldwork of this study further support this by showing that Agile methods are not the dominant software development method being used by organisation in South Africa.

Researchers and critics have cited a number of possible reason as to why Agile methods are in the trough. One of the objective of this study is to inform organisations about the state of Agile Software development in South Africa. As such it is important for practitioners to note the diminishing use of Agile methods and to take action in order to prevent Agile methods from becoming obsolete (Janes & Succi, 2012).

In the next section SWOT analysis is described, this is followed by a SWOT analysis of this study and the presentation of limitations and conclusions. Finally, the chapter concludes by presenting recommendations for future research.

8.2 What Is SWOT Analysis?

SWOT is an acronym for Strength, Weakness, Opportunities and Threats. It was created in 1965 by four professors at Harvard University (50MINUTES.COM, 2015). SWOT analysis is generally used as a decision making tool for strategic planning (Davey, 2010; 50MINUTES.COM, 2015). However, it has since been employed widely and in creative ways including academia (Davey, 2010). Each acronym in SWOT stands for the following:

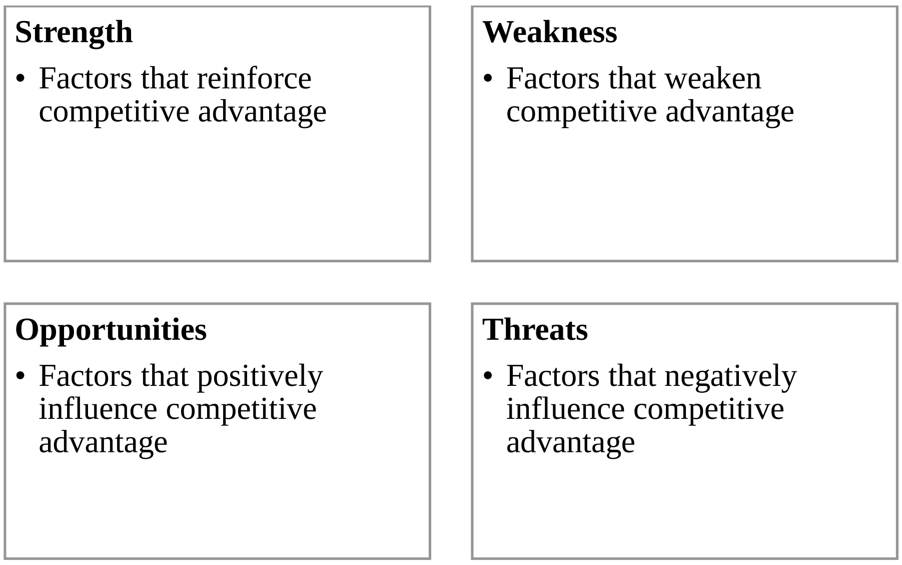

Strength
• Factors that reinforce competitive advantage
Weakness
• Factors that weaken competitive advantage
Opportunities
• Factors that positively influence competitive advantage
Threats
• Factors that negatively influence competitive advantage

Figure 8-1: Diagram defining SWOT. Adopted from 50MINUTES.COM (50MINUTES.COM, 2015).

According to Goodrich the outcome of a SWOT analysis is to accumulate data, which can be utilized to highlight the strengths and opportunities and also be used to address the weakness and threats (Goodrich, 2015).

8.3 SWOT Analysis For This Study

After applying SWOT analysis to this study the researcher compiled this report.

Strengths	**Weaknesses**
• This study can assist organisations intending to adopt Agile methods. • The use of electronic questionnaires provided convenience to respondents. As it could be completed at leisure. Furthermore, it reduced data capture errors.	• The number of respondents in this study was low. • The population of software developers in South Africa is not known. • The Gartner hype cycle (GHC) was used as the underpinning framework for this study. However, evidence provided in chapter 4 as well as chapter 7 of this study shows that the theoretical foundations of the GHC are flawed.
Opportunities	**Threats**
• Attract more respondents to the study. • Consider the use of another theoretical framework to underpin this study.	• Organisations regard Agile practices as proprietary and were not willing to participate or disclose information.

Figure 8-2: Diagram showing SWOT analysis for this study.

8.4 LIMITATIONS

The study focuses on organisations that develop software in South Africa. Although the results of the study were sufficient to produce meaningful data, the size of the sample should be considered when examining the validity of the results.

Furthermore, research on the validity of the GHC has concluded that theoretical foundation for the model is empirically flawed (Steinert & Leifer, 2010). Although the argument that Agile methods are in the trough of disillusionment is a significant one. The fact that Gartner has not disclosed the criteria used to evaluate innovations on the GHC is problematic and this has led to questions regarding the validity of the model (Steinert & Leifer, 2010).

8.5 Conclusions and Recommendations

The results from the Fieldwork show that Agile methods are being used by organisations in South Africa. However, there is no evidence to suggest that Agile methods are being preferred over other methods.

Evidence from Google Trends show that the number of searches on "extreme programing" is diminishing. This is suggestive of the fact that the popularity of Agile methods has decreased. One of the factors attributed to the downfall of Agile methods is the lack of understanding of Agile practices (Bishop, 2014; Ottinger, 2014). The fieldwork also showed differences in the interpretation of the barriers, impact and state of Agile methods amongst practitioners who had higher perceived knowledge than those with lower perceived knowledge. This is suggestive of the argument that knowledge of Agile practices can have a significant impact on the state of the adoption process.

This study also plotted Agile methods as being in the trough of disillusionment on the Gartner hype cycle (GHC). To avoid Agile methods from becoming obsolete organisations must understand the software development processes and its advantages and disadvantages (Janes & Succi, 2012).

Education, cultural and managerial aspects of Agile adoption needs to be further investigated in order to determine how to cross the chasm to mainstream Agile adoption.

Research on hypes and technology s-curves have established a foundation for the hype cycle (Steinert & Leifer, 2010). However, better grounding is needed into the existing body of knowledge (Steinert & Leifer, 2010). The

GHC is mathematically flawed as the combination of hype and the technology s-curve is missing a mathematical formulation. This is problematic as it is difficult to test the validity of the model (Steinert & Leifer, 2010). Further research is needed to develop a sound mathematical formulation of the model. This will enable the collection, analysis and reporting of the evolution of innovations in a structured and consistent way (Steinert & Leifer, 2010).

Another suggestion by Adamuthe, Tomke & Thampi is that the GHC should be used in conjunction with the technology life cycle (Adamuthe *et al.*, 2015). The combination of these models will allow for more effective interpretations (Adamuthe *et al.*, 2015).

It must be noted that it was not possible for the researcher to adequately test the validity of the results within the constraints of the theoretical framework as the researcher had to commit to the framework at the time of his application for ethical clearance. Furthermore, had the theoretical model been changed during the study the researcher would have had to reapply for ethical clearance. With the above critical assessment in mind the researcher would change the following processes in order to improve this study:

- Entice more respondents would to participate in the study.
- Due to above limitations in the GHC other theoretical frameworks or a combination of the GHC and the technology life cycle should be considered.
- The questionnaire would be modified in order to better aligned with the changes to the theoretical framework.

Further studies are also needed in order to determine if the skill level of Agile practitioners can have a significant impact on the state of Agile projects. Research has also shown that Scrum is a popular Agile method. However, not much studies have been conducted on the Scrum framework. Finally, work also needs to be done to determine how to push Agile methods out of the trough of disillusionment and into the plateau of productivity. As such this study can be used as a background to future research in these areas.

8.6 Chapter Summary

Agile methods have become popular since 2001 (Beck *et al.*, 2001a). However, as scepticism increased practitioners were not able to use Agile methods in an efficient and effective manner (Janes & Succi, 2012). In this chapter a SWOT analysis of the study was also presented. SWOT analysis was used by the researcher as a tool to identify shortcoming and improvements for future research.

While the data from this study is useful, areas of limitations need to be acknowledged. The low response rate and the use of an empirically flawed model needs to be factored when considering the results. The results from this study can be used as a platform for future research to determine how to push Agile methods into the plateau of productivity. The chapter finally concludes with recommendations for future research.

BIBLIOGRAPHY

50MINUTES.COM. 2015. *The SWOT Analysis: Develop strengths to decrease the weaknesses of your business*. 50MINUTES.COM.

Abrahamsson. 2003. Extreme programming: first results from a controlled case study. In *Proceedings of the 20th IEEE Instrumentation Technology Conference (Cat No 03CH37412) EURMIC-03*. IEEE. 259–266.

Abu-Dalbouh, H.M. 2013. A questionnaire approach based on the technology acceptance model for mobile tracking on patient progress applications. *Journal of Computer Science*. 9(6):763–770.

Adamuthe, A.C., Tomke, J. V & Thampi, G.T. 2015. An Empirical Analysis of Hype-cycle : A Case Study of Cloud Computing Technologies. *International Journal of Advanced Research in Computer and Communication Engineering*. 4(10):316–323.

Agile Alliance. 2013. *Collective Ownership*. Available at: http://guide.agilealliance.org/guide/cco.html [2014, November 09].

Agile In A Flash. 2009. *Agile In A Flash Blog*. Available at: http://agileinaflash.blogspot.com/2009/02/courage.html [2014, November 16].

Ahmed, A., Ahmad, S., Ehsan, N., Mirza, E. & Sarwar, S.Z. 2010. Agile software development: Impact on productivity and quality. In *2010 IEEE International Conference on Management of Innovation & Technology*. Singapore: IEEE. 287–291.

Ajzen, I. 1991. The theory of planned behavior. *Organizational Behavior and Human Decision Processes*. 50(2):179–211.

Ajzen, I. 2006. *Constructing a TpB Questionnaire : Conceptual and Methodological Considerations*. Vol. 2002.

Akbar, F. 2013. What affects students ' acceptance and use of technology? A test of UTAUT in the context of a higher-education institution in Qatar.

Aken, A. 2008. CHUNK: An Agile Approach to the Software Development

Life Cycle. *Journal of Internet Commerce*. 7(3):313–338.

Al-Jabri, I.M. & Sohail, M.S. 2012. Mobile banking adoption: application of diffusion of innovation theory. *Journal of Electronic Commerce Research*. 13(4):379–391. Available at: http://ssrn.com/abstract=2523623.

Ambler, S.W. 2003. *Agile Database Techniques: Effective Strategies for the Agile Software Developer*. 1st ed. Wiley.

Ambler, S.W. 2006. Imperfectly agile: You too can be agile! Just be smart about how you apply agile concepts. *Dr Dobbs Journal*. 31(10):82–84. Available at: http://www.nxtbook.com/nxtbooks/cmp/ddj1006/index.php?startid=82.

Ambler, S.W. 2012. *Agile/Lean Documentation: Strategies for Agile Software Development*. Vol. 4. Available at: http://www.agilemodeling.com/essays/agileDocumentation.htm#CRUFT.

Angeles, R. 2014. Using the Technology-Organization-Environment Framework for Analyzing Nike's "Considered Index" Green Initiative, a Decision Support System-Driven System. *Journal of Management and Sustainability*. 4(1):96–113.

Archibald, M.M. & Clark, A.M. 2014. Twitter and nursing research: how diffusion of innovation theory can help uptake. *Journal of Advanced Nursing*. 70(3):e3–e5.

Armitage, C.J. & Conner, M. 2001. Efficacy of the Theory of Planned Behaviour: a meta-analytic review. *The British journal of social psychology / the British Psychological Society*. 40(Pt 4):471–99. Available at: http://www.ncbi.nlm.nih.gov/pubmed/11795063.

Atlas, A. 2009. Accidental Adoption: The Story of Scrum at Amazon.com. In *2009 Agile Conference*. IEEE. 135–140.

Attarzadeh, I. & Ow, S.H. 2008. Project Management Practices: The Criteria for Success or Failure Project Management Practices: The Criteria for Success or Failure. *IBIMA publishing*. 1(28):234–241. Available at: http://www.ibimapublishing.com/journals/CIBIMA/volume1/v1n28.pdf.

Auvinen, J., Back, R., Heidenberg, J., Hirkman, P. & Milovanov, L. 2005. *Improving the Engineering Process Area at Ericsson with Agile Practices. A Case Study*. Turku Centre for Computer Science.

Awa, H.O., Ojiabo, O.U. & Emecheta, B.C. 2015. Integrating TAM, TPB and TOE frameworks and expanding their characteristic constructs for e-commerce adoption by SMEs. *Journal of Science and Technology Policy Management*. 6(1):76–94.

Awad, M.A. 2005. *A Comparison between Agile and Traditional Software Development Methodologies*. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.464.6090&rep=rep1&type=pdf.

Babcock, C. 2011. *Where Agile Development Fails: IT Operations*. Available at: http://www.informationweek.com/applications/where-agile-development-fails-it-operations/d/d-id/1101370? [2015, October 19].

Bagozzi, R. 2007. The Legacy of the Technology Acceptance Model and a Proposal for a Paradigm Shift. *Journal of the Association for Information Systems p*. 8(4):244–254.

Baird, S. 2002. *Extreme Programming Practices in Action*. Available at: http://www.informit.com/articles/article.aspx?p=30187&seqNum=13 [2014, October 25].

Baudson, C. & Beck, K. 2012. *What is sustainable pace?* Available at: http://sustainablepace.net/what-is-sustainable-pace [2014, September 25].

Beck, K. 1999. *Extreme Programming Explained: Embrace Change*. Addison-Wesley Professional.

Beck, K., Beedle, M., Bennekum, A. Van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. & Thomas, D. 2001a. *Manifesto for Agile Software Development*. Available at: http://agilemanifesto.org/ [2012, April 06].

Beck, K., Beedle, M., Bennekum, A. Van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. & Thomas, D. 2001b. *Principles behind the Agile Manifesto*. Available at: http://agilemanifesto.org/principles.html [2012, April 06].

Bedoll, R. 2003. A Tail of Two Projects: How "Agile" Methods Succeeded After "Traditional" Methods Had Failed in a Critical System-Development Project. In *Extreme Programming and Agile Methods - XP/Agile Universe*. Vol. 2753. 25–34.

Benbasat, I. & Barki, H. 2007. Quo vadis, TAM? *Journal of the Association for Information Systems*. 8(4):211–218.

Benefield, G. 2008. Rolling Out Agile in a Large Enterprise. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008)*. IEEE. 461–461.

Bernstein, L. 2010. Characterizing people as non-linear, first-order components in software development, is written by Alistair A.R. Cockburn and published in Humans and Technology, HaT Technical Report 1999.03, Oct 21, 1999. *ACM SIGSOFT Software Engineering Notes*. 35(4):33.

Bishop, R. 2014. *Agile Is Dead: The Angry Developer Version*. Available at: http://rubiquity.com/2014/03/12/agile-is-dead-angry-developer.html [2015, October 28].

Blechar, M. 2010. *Hype Cycle for Application Development 2010*. Available at: https://www.gartner.com/doc/1412014/hype-cycle-application-development- [2015, November 28].

Blechar, M. & Finley, I. 2011. *Hype Cycle for Application Development*

2011. Available at: https://www.gartner.com/doc/1753116/hype-cycle-application-development- [2015, November 28].

Blose, I. 2008. Lessons Learned from Distributed Agile Software Projects: A Case-Based Analysis. *Communications of the Association for Information Systems*. 23(34):619–632. Available at: http://aisel.aisnet.org/cais/vol23/iss1/34.

Boehm, B. & Turner, R. 2003. Using risk to balance agile and plan- driven methods. *Computer*. 36(6):57–66.

Boehm, B.W. & Papaccio, P.N. 1988. Understanding and controlling software costs. *IEEE Transactions on Software Engineering*. 14(10):1462–1477.

Brooks, F.P. 1995. *Mythical Man-Month, The: Essays on Software Engineering, Anniversary Edition*. Anniversar ed. Addison-Wesley Professional. Available at: http://www.amazon.de/dp/0201835959.

Burback, R. 1998. Software Engineering Methodology: The Watersluice. Stanford University.

Bustard, D., Wilkie, G. & Greer, D. 2013. The Maturation of Agile Software Development Principles and Practice: Observations on Successive Industrial Studies in 2010 and 2012. In *2013 20th IEEE International Conference and Workshops on Engineering of Computer Based Systems (ECBS)*. Scottsdale, AZ: IEEE. 139–146.

Cao, L. & Ramesh, B. 2008. Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software*. 25(1):60–67.

Cao, L., Mohan, K., Xu, P. & Ramesh, B. 2009. A framework for adapting agile development methodologies. *European Journal of Information Systems*. 18(4):332–343.

Centres Medical and Medicaid Services. 2008. *SELECTING A DEVELOPMENT APPROACH*. Available at: http://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf

[2014, February 24].

Chandra Misra, S., Kumar, V. & Kumar, U. 2010. Identifying some critical changes required in adopting agile practices in traditional software development projects. *International Journal of Quality & Reliability Management*. 27(4):451–474.

Charette, R.N. 2005. Why software fails. *IEEE Spectrum*. 42(9):36–43.

Chuang, S.-W., Luor, T. & Lu, H.-P. 2014. Assessment of institutions, scholars, and contributions on agile software development (2001–2012). *Journal of Systems and Software*. 93(7):84–101.

Clutterbuck, P., Rowlands, T. & Seamons, O. 2009. A Case Study of SME Web Application Development Effectiveness via Agile Methods. *Electronic Journal Information Systems Evaluation*. 12(1):13–26.

Cockburn, A. & Highsmith, J. 2001. Agile software development, the people factor. *Computer*. 34(11):131–133.

Cohn, M. 2012. *Agile Succeeds Three Times More Often Than Waterfall*. Available at: https://www.mountaingoatsoftware.com/blog/agile-succeeds-three-times-more-often-than-waterfall.

Cohn, M. & Ford, D. 2003. Introducing an Agile Process to an Organization. *Computer*. 36(6):74–78.

Conboy, K., Coyle, S., Wang, X. & Pikkarainen, M. 2011. People over Process: Key Challenges in Agile Development. *IEEE Software*. 28(4):48–57.

Copeland, L. 2001. *Extreme Programming*. Available at: http://www.computerworld.com/s/article/66192/Extreme_Programming.

Coram, M. & Bohner, S. 2005. The Impact of Agile Methods on Software Project Management. In *12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05)*. IEEE. 363–370.

Crotty, M. 1998. *The Foundations of Social Research: Meaning and Perspective in the Research*. SAGE Publications Ltd.

Crowther, D. & Lancaster, G. 2008. *Research Methods.* 2nd ed. Routledge.

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.

Davenport, T.H. 2005. *Thinking for a Living: How to Get Better Performances And Results from Knowledge Workers.* Harvard Business Press.

Davey, G. 2010. Visual Anthropology: Strengths, Weaknesses, Opportunities, Threats. *Visual Anthropology*. 23(4):344–352.

Davis, F.D. 1989. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*. 13(3):319–340.

Daya, S. 2001. Chi-square test. *Evidence-based Obstetrics & Gynecology*. 3(1):3–4.

Dhurka, D.K. 2015. *Agile Transformation 2015 and the Gartner's Hype Cycle*. Available at: http://deepadhurka.blogspot.co.za/2015/08/agile-transformation-2015-and-gartners.html [2015, October 31].

DSDM Consortium. 2013. *Agile and the concept of "Multi-skilled" Teams*. Available at: http://dsdm.org/dig-deeper/articles/agile-and-concept-%E2%80%9Cmulti-skilled%E2%80%9D-teams [2014, September 25].

Duka, D. 2011. *Complexity in Systems Development Projects.* Available at: http://www.derailleurconsulting.com/blog/complexity-and-noise-in-systems-development-projects [2012, November 04].

Duka, D. 2012. Agile experiences in software development. In *MIPRO, 2012 Proceedings of the 35th International Convention*. Opatija: IEEE. 692–697.

Dybâ, T. & Dingsøyr, T. 2009. What Do We Know about Agile Software

Development? *IEEE Software*. 26(5):6–9.

Dybå, T. & Dingsøyr, T. 2008. Empirical studies of agile software development: A systematic review. *Information and Software Technology*. 50(9-10):833–859.

Dybå, T., Arisholm, E., Sjøberg, D.I.K., Hannay, J.E. & Shull, F. 2007. Are Two Heads Better than One? On the Effectiveness of Pair Programming. *IEEE Software*. 24(6):12.

Edmonds, W.A. & Kennedy, T.D. 2012. *An Applied Reference Guide to Research Designs: Quantitative, Qualitative, and Mixed Methods*. SAGE Publications, Inc.

Erickson, J., Lyytinen, K. & Siau, K. 2005. Agile Modeling, Agile Software Development, and Extreme Programming. *Journal of Database Management*. 16(4):88–100.

Eveleens, J.L. & Verhoef, C. 2010. The rise and fall of the Chaos report figures. *IEEE Software*. 27(1):30–36.

Fenn, J. & Raskino, M. 2008. *Mastering the hype cycle. How to choose the right innovation at the right time*. Harvard Business School.

Finley, I., Wilson, N. & Van Huizen, G. 2012. *Hype Cycle for Application Development 2012*. Available at: https://www.gartner.com/doc/2098316/hype-cycle-application-development- [2015, November 28].

Ford. 1914. *Henry Ford's $5-a-Day Revolution*. Available at: http://corporate.ford.com/news-center/press-releases-detail/677-5-dollar-a-day [2014, September 25].

Fowler, M. 2005. *The New Methodology*. Available at: http://www.martinfowler.com/articles/newMethodology.html [2012, April 08].

Fowler, M. & Highsmith, J. 2001. *The Agile Manifesto*. Available at: http://www.drdobbs.com/open-source/the-agile-manifesto/184414755 [2014, September 14].

Gartner. 2003. *Hype Cycle for Application Development 2003*. Available at:

https://www.gartner.com/doc/396291/hype-cycle-application-development- [2015, November 28].

Gartner. 2004. *Hype Cycle for Application Development 2004*. Available at: https://www.gartner.com/doc/452212/hype-cycle-application-development- [2015, November 28].

Gartner. 2005. *Hype Cycle for Application Development 2005*. Available at: https://www.gartner.com/doc/483709/hype-cycle-application-development- [2015, November 28].

Gartner. 2006. *Hype Cycle for Application Development 2006*. Available at: https://www.gartner.com/doc/499813/hype-cycle-application-development- [2015, November 28].

Gartner. 2007. *Hype Cycle for Application Development 2007*. Available at: https://www.gartner.com/doc/508590/hype-cycle-application-development- [2015, November 28].

Gartner. 2008a. *Five Reasons Organizations Fail to Adopt Agile Methods*. Gartner. Available at: https://www.gartner.com/doc/666707/reasons-organizations-fail-adopt-agile [2014, February 15].

Gartner. 2008b. *Hype Cycle for Application Development 2008*. Available at: https://www.gartner.com/doc/715410/hype-cycle-application-development- [2015, November 28].

Gartner. 2009. *Hype Cycle for Application Development 2009*. Available at: https://www.gartner.com/doc/1087912/hype-cycle-application-development- [2015, November 28].

Gerber, M.E. 2010. *The E-Myth Enterprise: How to Turn a Great Idea into a Thriving Business.* 1st ed. HarperBusiness.

Goodrich, R. 2015. *SWOT Analysis: Examples, Templates & Definition*. Available at: http://www.businessnewsdaily.com/4245-swot-analysis.html [2015, November 03].

Google. 2015a. *Google Keyword Planner*. Available at: https://adwords.google.co.za/KeywordPlanner [2015, November

16].

Google. 2015b. *Google Trends*. Available at: https://www.google.co.za/trends/ [2015, October 28].

Goyal, S. 2007. Feature Driven Development Agile Techniques for Project Management Software Engineering. In *Agile Techniques for Project Management and Software Engineering*. Technical University Munich. Available at: http://csis.pace.edu/~marchese/CS616/Agile/FDD/fdd.pdf.

GP-Training.Net. 2013. *The Stacey Matrix*. Available at: http://www.gp-training.net/training/communication_skills/consultation/equipoise/complexity/stacey.htm [2013, March 01].

Grant, C. & Osanloo, A. 2014. UNDERSTANDING, SELECTING, AND INTEGRATING A THEORETICAL FRAMEWORK IN DISSERTATION RESEARCH: CREATING THE BLUEPRINT FOR YOUR "HOUSE". *Administrative Issues Journal Education Practice and Research*. 4(2):12–26.

De Haan, J. 2011. *Enterprise Agile: 3 characteristics*. Available at: http://www.theenterprisearchitect.eu/blog/2011/11/16/enterprise-agile-3-characteristics/ [2015, November 28].

Hadar, I., Sherman, S., Hadar, E. & Harrison, J.J. 2013. Less is more: Architecture documentation for agile development. In *2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*. IEEE. 121–124.

Hazrati, V. 2008. *Does Sustainable Pace mean a 40 hour week?* Available at: http://www.infoq.com/news/2008/05/sustainable-pace [2014, September 26].

Hazrati, V. 2011. *Is Agile in the Trough of Disillusionment?* Available at: http://www.infoq.com/news/2011/03/agile-trough-of-disillusionment [2015, October 31].

Highsmith, J. & Cockburn, A. 2001. Agile software development: the business of innovation. *Computer*. 34(9):120–127.

Hoda, R., Noble, J. & Marshall, S. 2010. Organizing self-organizing teams. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10*. Vol. 1. New York, New York, USA: ACM Press. 285.

Holub, A. 2014. *The Anti Anti Agile Manifesto*. Available at: http://www.drdobbs.com/architecture-and-design/the-anti-anti-agile-manifesto/240168963 [2015, November 17].

IFC. 2013. *IFC Job Study*. Available at: http://www.ifc.org/wps/wcm/connect/0fe6e2804e2c0a8f8d3bad7a9dd66321/IFC_FULL+JOB+STUDY+REPORT_JAN2013_FINAL.pdf?MOD=AJPERES [2015, March 14].

Inc.com. 2010. *How to Use Online Tools for Customer Surveys*. Available at: http://www.inc.com/guides/2010/07/how-to-use-online-tools-for-customer-surveys.html [2015, November 10].

Ionel, N. 2009. Agile Software Development Methodologies: An Overview of the Current State of Research. *Annals of the University of Oradea, Economic Science Series*. 18(4):381–385. Available at: http://content.ebscohost.com/ContentServer.asp?T=P&P=AN&K=485895648&S=R&D=bth&EbscoContent=dGJyMNHX8kSeqK44zdnyOLCmr0qeprVSsqm4SbWWxWXS&ContentCustomer=dGJyMPGosk+xq65QuePfgeyx44Dt6fIA.

James, L. 2008. *Software Engineering*. PHI.

Janes, A.A. & Succi, G. 2012. The dark side of agile software development. In *Proceedings of the ACM international symposium on New ideas, new paradigms, and reflections on programming and software - Onward! '12*. New York, New York, USA: ACM Press. 215.

Jansen, A., Avgeriou, P. & van der Ven, J.S. 2009. Enriching software architecture documentation. *Journal of Systems and Software*. 82(8):1232–1248.

Jeffries, R. 2010. *Beyond Agile: New Principles?* Available at: http://ronjeffries.com/xprog/articles/beyond-agile-new-principles/

[2015, November 17].

Jiang, L. & Eberlein, A. 2009. An analysis of the history of classical software development and agile development. In *2009 IEEE International Conference on Systems, Man and Cybernetics*. San Antonio, TX: IEEE. 3733–3738.

Jones, S., Murphy, F., Edwards, M. & James, J. 2008. Doing things differently: advantages and disadvantages of web questionnaires. *Nurse Researcher*. 15(4):15–26.

Jonker, J. & Pennink, B. 2009. *The Essence of Research Methodology*. Berlin, Heidelberg: Springer Berlin Heidelberg.

Judy, K.H. 2012. Agile Values, Innovation and the Shortage of Women Software Developers. In *2012 45th Hawaii International Conference on System Sciences*. IEEE. 5279–5288.

Kalermo, J. & Rissanen, J. 2002. Agile software development in theory and practice. University of Jyväskylä. Available at: http://www.cs.jyu.fi/sb/Publications/KalermoRissanen_MastersThesis_060802.pdf.

Keeffe III, E.B. 2015. *Visualizing an Enterprise-Class Agile*. Available at: http://mobileenterprise.edgl.com/tech-spotlight/Visualizing-an-Enterprise-Class-Agile100880 [2015, November 01].

Klopper, R. & Lubbe, S. 2012. Using matrix analysis to achieve traction, coherence, progression and closure in problem-solution oriented research. In *International Conference on Information Resources Management (CONF-IRM)*. CONF-IRM 2012 Proceedings. 1–13. Available at: http://www.finance.ukzn.ac.za/docs/18.3 SpEd 4/18 Klo.pdf.

Korhonen, K. 2013. Evaluating the impact of an agile transformation: a longitudinal case study in a distributed context. *Software Quality Journal*. 21(4):599–624.

Kulatunga, K.J., Amaratunga, R.D.G. & Haigh, R.P. 2007. Researching construction client and innovation: methodological perspective. In

7th International Postgraduate Conference in the Built and Human Environment. Salford: University of Salford. 479–488. Available at: http://usir.salford.ac.uk/9847/.

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.

Lacey, M. 2012. *The scrum field guide: Practical advice for your first year*. 1st ed. Addison-Wesley Professional.

Lambkin, M. & Day, G.S. 1989. Evolutionary Processes in Competitive Markets: Beyond the Product Life Cycle. *Journal of Marketing*. 53(3):4–20.

Layman, L., Williams, L. & Cunningham, L. 2004. Exploring Extreme Programming in Context: An Industrial Case Study. In *Agile Development Conference*. IEEE. 32–41.

Leau, Y.B., Loo, W.K., Tham, W.Y. & Tan, S.F. 2012. Software Development Life Cycle AGILE vs Traditional Approaches. In *2012 International Conference on Information and Network Technology*. Vol. 37. Singapore. 162–167. Available at: http://ipcsit.com/vol37/030-ICINT2012-I2069.pdf.

Lin, C.-C. 2013. Exploring the relationship between technology acceptance model and usability test. *Information Technology and Management*. 14(3):243–255.

Lindvall, M., Basili, V.R., Boehm, B.W., Costa, P., Dangle, K., Shull, F., Tesoriero, R., Williams, L.A. & Zelkowitz, M. V. 2002. Empirical Findings in Agile Methods. In *Proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods - XP/Agile Universe 2002*. London, UK, UK: Springer-Verlag. 197–207. Available at: http://dl.acm.org/citation.cfm?id=647276.722332.

LinkedIn.com. 2015. *LinkedIn Search Results*. Available at:

https://www.linkedin.com/vsearch/c?orig=FCTD&rsid=902317231447664514317&keywords=software development&trk=vsrp_companies_sel&trkInfo=VSRPsearchId%3A902317231447664498683,VSRPcmpt%3Atrans_nav&f_CCR=za%3A0&openFacets=N,CCR,JO,I&f_I=4 [2015, June 01].

Linowes, J.S. & Parker Hill Technology. 1991. *A Summary of " Crossing the Chasm".* Available at: http://ewthoff.home.xs4all.nl/Weppage documents/Summary Crossing the Chasm.pdf.

Livermore, J. 2007. Factors that impact implementing an agile software development methodology. In *Proceedings 2007 IEEE SoutheastCon*. Richmond, VA: IEEE. 82–86.

Loftus, C. & Ratcliffe, M. 2005. Extreme programming promotes extreme learning? *ACM SIGCSE Bulletin*. 37(3):311.

Lomax, R.G. 2013. Statistical Accuracy of iPad Applications: An Initial Examination. *The American Statistician*. 67(2):105–108.

Lowry, R. 2014. *VassarStats: Website for Statistical Computation*. Available at: http://vassarstats.net/index.html [2015, September 12].

Luo, X., Warkentin, M. & Li, H. 2013. Understanding technology adoption trade-offs: A conjoint analysis approach. *Journal of Computer Information Systems*. 53(3):65–74. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle: Understanding+technology+adoption+trade-offs:+A+conjoint+analysis+approach#0.

Lycett, M., Macredie, R.D., Patel, C. & Paul, R.J. 2003. Migrating agile methods to standardized development practice. *Computer*. 36(6):79–85.

Makabee, H. 2014. *The End of Agile: Death by Over-Simplification*. Available at: http://effectivesoftwaredesign.com/2014/03/17/the-end-of-agile-death-by-over-simplification/ [2015, October 28].

Mannaro, K., Melis, M. & Marchesi, M. 2004. Empirical Analysis on the Satisfaction of IT Employees Comparing XP Practices with Other

Software Development Methodologies. In *Extreme Programming and Agile Processes in Software Engineering SE - 19*. Vol. 3092. J. Eckstein & H. Baumeister, Eds. (Lecture Notes in Computer Science). Springer Berlin Heidelberg. 166–174.

Marshall, B. 2014. *I Don't Want Agile Back*. Available at: https://flowchainsensei.wordpress.com/2014/03/11/i-dont-want-agile-back/ [2015, October 28].

Maruping, L.M., Zhang, X. & Venkatesh, V. 2009. Role of collective ownership and coding standards in coordinating expertise in software project teams. *European Journal of Information Systems*. 18(4):355–371.

Masrom, M. 2007. Technology Acceptance Model and E-learning. In *12th International Conference on Education*. Malaysia: Universiti Brunei Darussalam. 1–10.

Maurer, F. & Martel, S. 2002. Extreme programming. Rapid development for Web-based applications. *IEEE Internet Computing*. 6(1):86–90.

Maxwell, J.A. 2012. *Qualitative Research Design: An Interactive Approach (Applied Social Research Methods)*. 3rd ed. SAGE Publications, Inc.

Mazni, O., Syed-Abdullah, S.-L. & Yasin, A. 2011. The Impact of Agile Approach on Software Engineering Teams. *American Journal of Economics and Business Administration*. 3(1):12–17.

Melymuka, V. 2012. *TeamCity 7 Continuous Integration Essentials*. Packt Publishing.

Mirnalini, K. & Raya, V.R. 2010. Agile - A software development approach for quality software. In *2010 International Conference on Educational and Information Technology*. Chongqing: IEEE. V1–242 – V1–244.

Mnkandla, E. 2009. About software engineering frameworks and methodologies. In *AFRICON 2009*. Vol. 1087. IEEE. 1–5.

Mnkandla, E. & Dwolatzky, B. 2004. A survey of agile methodologies. *SAIEE Africa Research Journal*. 95(4):236–247.

Moe, N.B., Dingsøyr, T. & Dybå, T. 2008. Understanding self-organizing teams in agile software development. In *Proceedings of the Australian Software Engineering Conference, ASWEC*. Perth, WA: IEEE. 76–85.

Moore, S.M. 2006. Theoretical Framework. In *Encyclopedia of Nursing Research*. 2nd ed. New York, NY, USA: Springer Publishing Company, LLC.

Moreira, M. 2015. *Agile Adoption Roadmap: Have you Crossed the Agile Chasm?* Available at: http://ewthoff.home.xs4all.nl/Weppage documents/Summary Crossing the Chasm.pdf [2015, November 01].

Moreira, M.E. 2013. *Being Agile: Your Roadmap to Successful Adoption of Agile*. 1st ed. Apress.

Mukherji, P. & Albon, D. 2009. *Research Methods in Early Childhood: An Introductory Guide*. 1st ed. SAGE Publications Ltd.

Münch, J., Armbrust, O., Kowalczyk, M. & Soto, M. 2012. *Software Process Definition and Management*. Springer-Verlag Berlin Heidelberg.

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? In *2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement*. IEEE. 75–84.

Murphy, T.E., Wilson, N. & Sobejana, M. 2014. *Hype Cycle for Application Development 2014*. Available at: https://www.gartner.com/doc/2810920/hype-cycle-application-development- [2015, November 28].

Naderuzzaman, M., Rabbi, F. & Beg, A. 2011. An improved & adaptive software development methodology. *Computer Engineering and Intelligent*. 2(3):35–41. Available at: http://iiste.org/Journals/index.php/CEIS/article/view/248.

Nagler, R. 2012. *Extreme Programming in Perl*. Available at: http://www.extremeperl.org/f/extremeperl.pdf [2014, March 03].

Nerur, S. & Balijepally, V. 2007. Theoretical reflections on agile development methodologies. *Communications of the ACM*. 50(3):79–83.

Nerur, S., Mahapatra, R. & Mangalaraj, G. 2005. Challenges of migrating to agile methodologies. *Communications of the ACM*. 48(5):72–78.

Nkhoma, M. & Dang, D. 2013. Contributing factors of cloud computing adoption: a technology-organisation-environment framework approach. *International Journal of Information Systems and Engineering (IJISE)*. 1(1):38–49. Available at: http://www.ftms.edu.my/ascent2014/IS/4.pdf.

Noll, J. 2007. *A Survey of Empirical Studies of Extreme Programming*. Santa Clara.

Nord, R.L. & Tomayko, J.E. 2006. Software architecture-centric methods and agile development. *IEEE Software*. 23(2):47–53.

Nordberg, M.E. 2003. Managing code ownership. *IEEE Software*. 20(2):26–33.

Oliveira, T. & Martins, M.F. 2011. Literature Review of Information Technology Adoption Models at Firm Level. *Electronic Journal of Information Systems Evaluation*. 14(1):110–121.

Orlikowski, W.J. & Baroudi, J.J. 1991. Studying Information Technology in Organizations: Research Approaches and Assumptions. *Information Systems Research*. 2(1):1–28.

Osherove, R. 2013. *The Art of Unit Testing: with .NET Examples*. 2nd ed. Manning Publications.

Ottinger, T. 2014. *I Want Agile Back*. Available at: http://agileotter.blogspot.co.za/2014/02/i-want-agile-back.html [2015, October 28].

Overhage, S. & Schlauderer, S. 2012. Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of

Developer Perceptions in Scrum Projects. In *2012 45th Hawaii International Conference on System Sciences*. IEEE. 5452–5461.

Paasivaara, M. & Lassenius, C. 2003. Collaboration practices in global inter-organizational software development projects. *Software Process: Improvement and Practice*. 8(4):183–199.

Palmer, S.R. & Felsing, J.M. 2002. *Guide to Feature-Driven Development.* 1st ed. Prentice Hall.

Pathirage, C., Amaratunga, R. & Haigh, R. 2008. The role of philosophical context in the development of research methodology and theory. *The Built & Human Environment Review*. 1(1):1–10. Available at: http://usir.salford.ac.uk/id/eprint/9681.

Pichler, R. 2006. Agile Estimating and Planning by Mike Cohn. *Journal of Product Innovation Management*. 23(6):588–589.

Pietri, W. 2011. *Agile's Second Chasm*. Available at: http://agilefocus.com/2011/02/21/agiles-second-chasm-and-how-we-fell-in/ [2015, November 01].

Poole, C.J., Murphy, T., Huisman, J.W. & Higgins, A. 2001. Extreme maintenance. In *Proceedings IEEE International Conference on Software Maintenance. ICSM 2001*. IEEE Comput. Soc. 301–309.

Porter, C.E. & Donthu, N. 2006. Using the technology acceptance model to explain how attitudes determine Internet usage: The role of perceived access barriers and demographics. *Journal of Business Research*. 59(9):999–1007.

Preuss, D.H. 2006a. *Do Agile Practices Make it an Agile Project?* Available at: http://www.infoq.com/articles/agile-values-and-practices [2015, October 30].

Preuss, D.H. 2006b. *Opinion: Take Agile Off Your Resume*. Available at: http://www.infoq.com/news/yegge-good-bad-agile [2015, October 31].

ProQuest.com. 2015. *ProQuest Searh Results*. Available at: http://dut.summon.serialssolutions.com/ [2015, November 25].

Proulx, M. 2010. *Gartner Predicts 2010: Agile and Cloud Impact Application Development Directions*. Available at: http://analytical-mind.com/2010/03/09/gartner-predicts-2010-agile-and-cloud-impact-application-development-directions/ [2012, April 08].

Ramnath, V. 2011. The level of adoption and effectiveness of software development methodologies in the software development industry in South Africa. University of Pretoria. Available at: http://hdl.handle.net/2263/30536.

Rao, K.N., Naidu, G.K. & Chakka, P. 2011. A study of the Agile software development methods, applicability and implications in industry. *International Journal of Software Engineering and its Applications*. 5(2):35–46.

Rasmusson, J. 2003. Introducing xp into greenfield projects: lessons learned. *IEEE Software*. 20(3):21–28.

Rehan, S., Assudani, M., Shrivastav, R. & Deshmukh, S. 2014. Challenges of Software Development Team in Agile Development. In *International Conference on Advances in Engineering & Technology*. Vol. 5. 1–8.

Rogers, E.M. 2003. *Diffusion of Innovations*. 5th ed. Free Press.

Roussel, A.-M. 2003. *New Hype Cycles Provide Key Evaluation Tool for Investors*. Available at: https://www.gartner.com/doc/396553/new-hype-cycles-provide-key [2015, October 31].

Rubin, E. & Rubin, H. 2011. Supporting agile software development through active documentation. *Requirements Engineering*. 16(2):117–132.

Rumpe, B. & Schröder, A. 2002. Quantitative survey on extreme programming projects. In *Third International Conference on Extreme Programming and Flexible Processes in Software Engineering*. Alghero, Italy: RWTH Aachen University. 95–100. Available at: http://www.sse-tubs.de/~rumpe/publications/ps/XP02.RumpeSchroeder.Study.pdf.

Runeson, P. 2006. A survey of unit testing practices. *IEEE Software*. 23(4):22–29.

Saleh, H. 2013. *JavaScript Unit Testing.* 1st ed. Packt Publishing.

Saunders, B.M. & Tosey, P. 2013. The Layers of Research Design. *Rapport: The Magazine for NLP Professionals*. 58–59.

Saunders, M., Lewis, P. & Thornhill, A. 2012. *Research Methods for Business Students*. Pearson Custom Publishing.

Scharff, C. 2011. Guiding global software development projects using Scrum and Agile with quality assurance. In *2011 24th IEEE-CS Conference on Software Engineering Education and Training (CSEE&T)*. Vol. 24. IEEE. 274–283.

Schatz, B. & Abdelshafi, I. 2005. Primavera Gets Agile: A Successful Transition to Agile Development. *IEEE Software*. 22(3):36–42.

Schwaber, K. 2004. *Agile Project Management with Scrum*. 1st, Ed. Microsoft Press.

Schwaber, K. & Sutherland, J. 2012. *Software in 30 Days: How Agile Managers Beat the Odds, Delight Their Customers, And Leave Competitors In the Dust*. New Jersey: John Wiley Sons.

Schwaber, C., Leganza, G. & D'Silva, D. 2007. *The truth about agile processes - Frank answers to frequently asked questions*. Available at: http://dev.www.rallydev.com/sites/default/files/The_Truth_About_Agile_Processes_Forrester_white_paper.pdf [2014, March 15].

Senapathi, M. & Srinivasan, A. 2011. Understanding Post-Adoptive Agile Usage -- An Exploratory Cross-Case Analysis. In *2011 AGILE Conference*. IEEE. 117–126.

Sharp, H. & Robinson, H. 2004. An Ethnographic Study of XP Practice. *Empirical Software Engineering*. 9(4):353–375.

Shier, R. 2004. *The Wilcoxon signed rank sum test*. Available at: http://www.statstutor.ac.uk/resources/uploaded/wilcoxonsignedranktest.pdf.

Shore, J. & Warden, S. 2007. The Art of Agile Development. 409. Available at: http://books.google.com/books?id=g_ji7cRb--UC&pgis=1.

Sobejana, M. & Murphy, T.E. 2015. *Hype Cycle for Application Development 2015*. Available at: https://www.gartner.com/doc/3101023/hype-cycle-application-development- [2015, November 28].

Soiferman, K.L. 2010. *Compare and Contract Inductive and Deductive Research Approaches*. Available at: http://files.eric.ed.gov/fulltext/ED542066.pdf [2015, February 28].

Sonnenwald, D.H., Maglaughlin, K.L. & Whitton, M.C. 2001. Using innovation diffusion theory to guide collaboration technology evaluation: work in progress. In *Proceedings Tenth IEEE International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises. WET ICE 2001*. IEEE Comput. Soc. 114–119.

Sriram, R. & Mathew, S.K. 2012. Global software development using agile methodologies: A review of literature. In *2012 IEEE International Conference on Management of Innovation & Technology (ICMIT)*. IEEE. 389–393.

Standish Group. 1995. *The CHAOS Report*. Available at: https://www.projectsmart.co.uk/white-papers/chaos-report.pdf [2014, March 01].

Statistics Solutions. 2013. *ANOVA (Analysis of Variance)*. Available at: http://www.statisticssolutions.com/manova-analysis-anova/ [2014, September 12].

StatSoft, I. 2013. *Electronic statistics textbook*. Available at: http://www.statsoft.com/textbook/ [2015, May 10].

Stavru, S. 2014. A critical examination of recent industrial surveys on agile method usage. *Journal of Systems and Software*. 94(August):87–97.

Steinert, M. & Leifer, L. 2010. Scrutinizing Gartner's Hype Cycle Approach. In *PICMET 2010 Proceedings*. Phuket: IEEE. 1–13.

Stephenson, G. 2003. The Somewhat Flawed Theoretical Foundation of the Extension Service. *Journal of Extension*. 41(4):1–11.

Stoica, M., Mircea, M. & Ghilic-Micu, B. 2013. Software Development: Agile vs. Traditional. *Informatica Economica.* 17(4):64–76.

Sudhakar, P., Farooq, A. & Patnaik, S. 2012. Measuring productivity of software development teams. *Serbian Journal of Management*. 7(1):65–75.

Suryaningrum, D. 2012. Assessing individual performance on information technology adoption: A new model. *Global Journal of Business Research*. 6(4):111–125. Available at: http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2146065.

Taiwo, A.A. & Downe, A.G. 2013. The theory of user acceptance and use of technology (UTAUT): A meta-analytic review of empirical findings. *Journal of Theoretical and Applied Information Technology*. 49(1):48–58. Available at: http://www.jatit.org/volumes/Vol49No1/7Vol49No1.pdf.

Takeuchi, H. & Nonaka, I. 1986. The new new product development game. *Harvard Business Review*. 64(1):137–146.

Thomas, D. 2014. *Agile Is Dead (Long Live Agility).* Available at: http://pragdave.me/blog/2014/03/04/time-to-kill-agile/ [2015, March 08].

Thomas, H.R. & Raynar, K. a. 1997. Scheduled Overtime and Labor Productivity: Quantitative Analysis. *Journal of Construction Engineering and Management*. 123(2):181–188.

Thoughtworks.com. 2011. *Agile Adoption In India 2011*. Available at: http://www.thoughtworks.com/articles/agile-adoption-india-2011 [2012, May 05].

Tobbin, P. 2012. Towards a model of adoption in mobile banking by the unbanked: a qualitative study. *Info*. 14(5):74–88.

Tornatzky, L.G. & Fleischer, M. 1990. *The processes of technological innovation*. Lexington, MA: Lexington Books.

Turk, D., France, R. & Rumpe, B. 2014. Assumptions Underlying Agile Software Development Processes. *Journal of Database Management*. 16(4):62–87. Available at: http://arxiv.org/abs/1409.6610.

UK Essays. 2013. *Explanation Of The Concept Of Research Onion Psychology Essay*. Available at: http://www.ukessays.com/essays/psychology/explanation-of-the-concept-of-research-onion-psychology-essay.php [2015, March 29].

Venkatesh, V., Morris, M.G., Davis, G.B. & Davis, F.D. 2003. User Acceptance of Information Technology: Toward a Unified View. *MIS Quarterly*. 27(3):425–478.

Venkatesh, V., Thong, J.Y.L. & Xu, X. 2012. Consumer Acceptance and Use of Informations Technology: Extending the Unified Theory of Acceptance and Use of Technology. *MIS Quarterley*. 36(1):157–178.

VersionOne. 2010. *5th Annual State of Agile Development Survey Final summary report*. Available at: http://www.infoq.com/resource/vcr/1276/file/2010_State_of_Agile_Development_Survey_Results.pdf.

VersionOne. 2011. *6th Annual State of Agile Development Survey Final summary report*. Available at: https://www.versionone.com/pdf/2011_State_of_Agile_Development_Survey_Results.pdf.

VersionOne. 2013. *8th Annual State of Agile Development Survey Final summary report*. Available at: http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf [2014, March 03].

VersionOne. 2014. *9th Annual State Of Agile Development Survey Final summary report*. Available at: https://www.versionone.com/pdf/state-of-agile-development-survey-ninth.pdf [2015, November 11].

Vijayasarathy, L. & Turk, D. 2012. Drivers of agile software development use: Dialectic interplay between benefits and hindrances. *Information and Software Technology*. 54(2):137–148.

Vinekar, V., Slinkman, C.W. & Nerur, S. 2006. Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View. *Information Systems Management*. 23(3):31–42.

Williams, L. 2007a. *A Survey of Agile Development Methodologies*. Available at: http://www.cems.uwe.ac.uk/~pchatter/2011/isd_hk/AgileMethods.pdf [2014, November 16].

Williams, S.E. 2007b. Chi-Square Test for Goodness of Fit. In *Encyclopedia of Measurement and Statistics*. 2455 Teller Road, Thousand Oaks California 91320 United States of America: Sage Publications, Inc. 134–136.

Williams, L. & Cockburn, A. 2003. Agile software development: it's about feedback and change. *Computer*. 36(6):39–43.

Williams, L., Kessler, R.R., Cunningham, W. & Jeffries, R. 2000. Strengthening the case for pair programming. *IEEE Software*. 17(4):19–25.

Williams, L., Kudrjavets, G. & Nagappan, N. 2009. On the Effectiveness of Unit Test Automation at Microsoft. In *2009 20th International Symposium on Software Reliability Engineering*. 81–89.

Wilson, N. 2012. *The Trough of Disillusionment*. Available at: http://blogs.gartner.com/nathan-wilson/the-trough-of-disillusionment/ [2015, October 28].

Wilson, N., Van Huizen, G. & Prentice, B. 2013. *Hype Cycle for Application Development 2013*. Available at: https://www.gartner.com/doc/2560015/hype-cycle-application-development- [2015, November 28].

Wray, S. 2010. How Pair Programming Really Works. *IEEE Software*. 27(1):50–55.

Youssef, M.A. 1992. Agile amnufacturing: a necessary condition for competing in global markets. *Industrial Engineering*. 24(12):18–20.

ADDENDUM 1:

PERMISSION-GIVER LETTER

The adoption of Agile software development methodologies by organizations in South Africa

Researcher: Mr C Vanker
Supervisor: Prof. Rembrandt Klopper
Co-Supervisor: Mr Karunagaran Naidoo
School of Management, IT and Governance
Discipline of Information Systems and Technology
University of KwaZulu-Natal

PERMISSION TO *CONDUCT A RESEARCH QUESTIONAIRE*

As the_______________________________________ (Designation)

at____ (Organisation)

I hereby grant permission to the researcher to administer a questionnaire on the above topic in this organisation.

Name: ____

Signature:

____

Date: / /

Addendum 2:

17. How long have you been practicing Agile?			
	Observed N	Expected N	Residual
<1 year	**45**	**14.2**	**30.8**
1 - 2 years	**23**	**14.2**	**8.8**
3 - 4 years	8	14.2	-6.2
5 - 6 years	3	14.2	-11.2
7 - 8 years	3	14.2	-11.2
>8 years	3	14.2	-11.2
Total	85		

Test Statistics	
	17. How long have you been practicing Agile?
Chi-Square	**101.706**[a]
df	**5**
Asymp. Sig.	**.000**

a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 14.2.

Addendum 3:

18. How large was your last Agile team? (Please include all IT/systems staff and stakeholder representatives such as product owners or product managers.)?				
	Category	Observed N	Expected N	Residual
1	**1 - 5**	**37**	**14.2**	**22.8**
2	**6 - 10**	**37**	**14.2**	**22.8**
3	11 - 15	7	14.2	-7.2
4	16 - 20	2	14.2	-12.2
5		0	14.2	-14.2
6	>30	2	14.2	-12.2
Total		85		

Test Statistics	
	18. How large was your last Agile team? (Please include all IT/systems staff and stakeholder representatives such as product owners or product managers.)?
Chi-Square	**112.294**[a]
df	**5**
Asymp. Sig.	**.000**

a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 14.2.

Addendum 4:

19. How is your team geographically distributed?			
	Observed N	Expected N	Residual
Same room	**39**	**12.1**	**26.9**
Same floor	11	12.1	-1.1
Same building	10	12.1	-2.1
Same campus	9	12.1	-3.1
Within driving distance of each other	9	12.1	-3.1
Within same time zone	5	12.1	-7.1
Different time zones	2	12.1	-10.1
Total	85		

Test Statistics	
19. How is your team geographically distributed?	
Chi-Square	**74.188**[a]
df	**6**
Asymp. Sig.	**.000**
a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 12.1.	

Addendum 5:

20. How often do team members communicate with stakeholders?			
	Observed N	Expected N	Residual
Regularly - daily	**35**	**21.3**	**13.8**
Regularly - weekly	**36**	**21.3**	**14.8**
At the start of an iteration	5	21.3	-16.3
As needed	9	21.3	-12.3
Total	85		

Test Statistics	
20. How often do team members communicate with stakeholders?	
Chi-Square	**38.624**[a]
df	**3**
Asymp. Sig.	**.000**
a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 21.3.	

Addendum 6:

21. In your last Agile project what type of methodology/framework did you follow?			
	Observed N	Expected N	Residual
SCRUM	**71**	**12.1**	**58.9**
Hybrid custom	4	12.1	-8.1
KANBAN	3	12.1	-9.1
Extreme programming	3	12.1	-9.1
Lean	1	12.1	-11.1
Agile unified modelling	2	12.1	-10.1
K8	1	12.1	-11.1
Total	85		

Test Statistics	
21. In your last Agile project what type of methodology/framework did you follow?	
Chi-Square	**333.435**[a]
df	**6**
Asymp. Sig.	**.000**
a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 12.1.	

Addendum 7:

22. What was the iteration length of you last Agile project?			
	Observed N	Expected N	Residual
1 - 7 days	**28**	**17.0**	**11.0**
8 - 14 days	**30**	**17.0**	**13.0**
15 - 21 days	7	17.0	-10.0
22 - 30 days	10	17.0	-7.0
over 30 days	10	17.0	-7.0
Total	85		

Test Statistics	
22. What was the iteration length of you last Agile project?	
Chi-Square	**28.706**[a]
df	**4**
Asymp. Sig.	**.000**
a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 17.0.	

Addendum 8:

23 How would you describe your company’s Agile adoption strategy (Select ONE option only)?			
	Observed N	Expected N	Residual
Early adopter	27	28.3	-1.3
Fast follower	18	28.3	-10.3
Late adopter	**40**	**28.3**	**11.7**
Total	85		

Test Statistics	
	23 How would you describe your company’s Agile adoption strategy (Select ONE option only)?
Chi-Square	**8.635**[a]
df	**2**
Asymp. Sig.	**.013**

a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 28.3.

Addendum 9:

Suitable project types for Agile methods

	N	Mean	Std. Deviation
24.1 Web sites	**85**	**3.56**	1.367
24.2 Web-based application	**85**	**3.69**	1.319
24.3 Desktop application	**85**	**3.35**	1.369
24.4 Games and Animation Projects	85	3.01	1.452
24.5 Database	85	3.27	1.295
24.6 ERP Applications	85	3.14	1.207
24.7 Embedded Software	85	2.93	1.343
24.8 Mobile Applications	**85**	**3.71**	1.335
24.9 Complex Computational Projects	85	2.99	1.401

Test Statistics									
	threes - 24.1 Web sites	threes - 24.2 Web-based application	threes - 24.3 Desktop application	threes - 24.4 Games and Animation Projects	threes - 24.5 Database	threes - 24.6 ERP Applications	threes - 24.7 Embedded Software	threes - 24.8 Mobile Applications	threes - 24.9 Complex Computational Projects
Z	**-3.784**[a]	**-4.406**[a]	**-2.561**[a]	-.109[a]	-1.908[a]	-.954[a]	-.708[b]	**-4.118**[a]	-.180[a]
Asymp. Sig. (2-tailed)	**.000**	**.000**	**.010**	.913	.056	.340	.479	**.000**	.857

a. Based on positive ranks.

b. Based on negative ranks.

c. Wilcoxon Signed Ranks Test

Addendum 10:

Barriers Preventing Agile Adoption

	threes - 25.1 The culture within an organization can be an impediment to Agile adoption - Strongly Disagree	threes - 25.2 Not having access to personnel with the right skills is an impediment to Agile adoption - Strongly Disagree	threes - 25.3 General resistance to change is an impediment to Agile adoption - Strongly Disagree	threes - 25.4 The lack of management support is an impediment to Agile adoption - Strongly Disagree	threes - 25.5 Complex development projects are an impediment to Agile adoption - Strongly Disagree
Z	**-7.320**[a]	**-6.849**[a]	**-7.616**[a]	**-6.822**[a]	-1.030[b]
Asymp. Sig. (2-tailed)	**.000**	**.000**	**.000**	**.000**	.303

	threes - 25.6 The additional time required by an organization to transition to Agile is an impediment to Agile adoption - Strongly Disagree	threes - 25.7 Agile's inability to scale up for large projects is an impediment to adoption - Strongly Disagree	threes - 25.8 The lack of up-front planning in Agile projects is a cause of concern when adopting Agile - Strongly Disagree	threes - 25.9 The lack of documentation in Agile projects is a cause of concern when adopting Agile - Strongly Disagree	threes - 25.10 The lack of an engineering discipline in Agile projects is a cause of concern when adopting Agile - Strongly Disagree
Z	**-2.473**[a]	-.088[b]	**-2.781**[a]	-1.511[a]	-.453[a]
Asymp. Sig. (2-tailed)	**.013**	.930	**.005**	.131	.651

Addendum 11:

26. How would you characterise your last Agile project?			
	Observed N	Expected N	Residual
Too early to tell	11	17.0	-6.0
Successful	**31**	**17.0**	**14.0**
Challenged	25	17.0	8.0
Failed	14	17.0	-3.0
Don't know	4	17.0	-13.0
Total	85		

Test Statistics	
	26. How would you characterise your last Agile project?
Chi-Square	**27.882**[a]
df	**4**
Asymp. Sig.	**.000**
a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 17.0.	

Addendum 12:

Analysis with Wilcoxon Signed Ranks Test

27.The impact of Agile Methods.					
	threes - 27.1 Agile methods help teams manage changing priorities - Strongly Disagree	threes - 27.2 Agile methods improve project visibility - Strongly Disagree	threes - 27.3 Agile methods increase productiv-ity - Strongly Disagree	threes - 27.4 Agile methods provide a faster time to market - Strongly Disagree	threes - 27.5 Agile methods provide better alignment between IT and business - Strongly Disagree
Z	**-7.598[a]**	**-6.729[a]**	**-6.902[a]**	**-7.630[a]**	**-7.113[a]**
Asymp. Sig. (2-tailed)	**.000**	**.000**	**.000**	**.000**	**.000**

	threes - 27.6 Agile meth-ods improve the quality of software being devel-oped - Strongly Disagree	threes - 27.7 Agile meth-ods improve team moral - Strongly Disagree	threes - 27.8 Agile meth-ods simplify the develop-ment pro-cess - Strongly Disagree	threes - 27.9 Agile meth-ods en-hanced soft-ware main-tainability and extensi-bility - Strongly Disagree
Z	**-6.144[a]**	**-6.838[a]**	**-5.064[a]**	**-5.632[a]**
Asymp. Sig. (2-tailed)	**.000**	**.000**	**.000**	**.000**

Addendum 13:

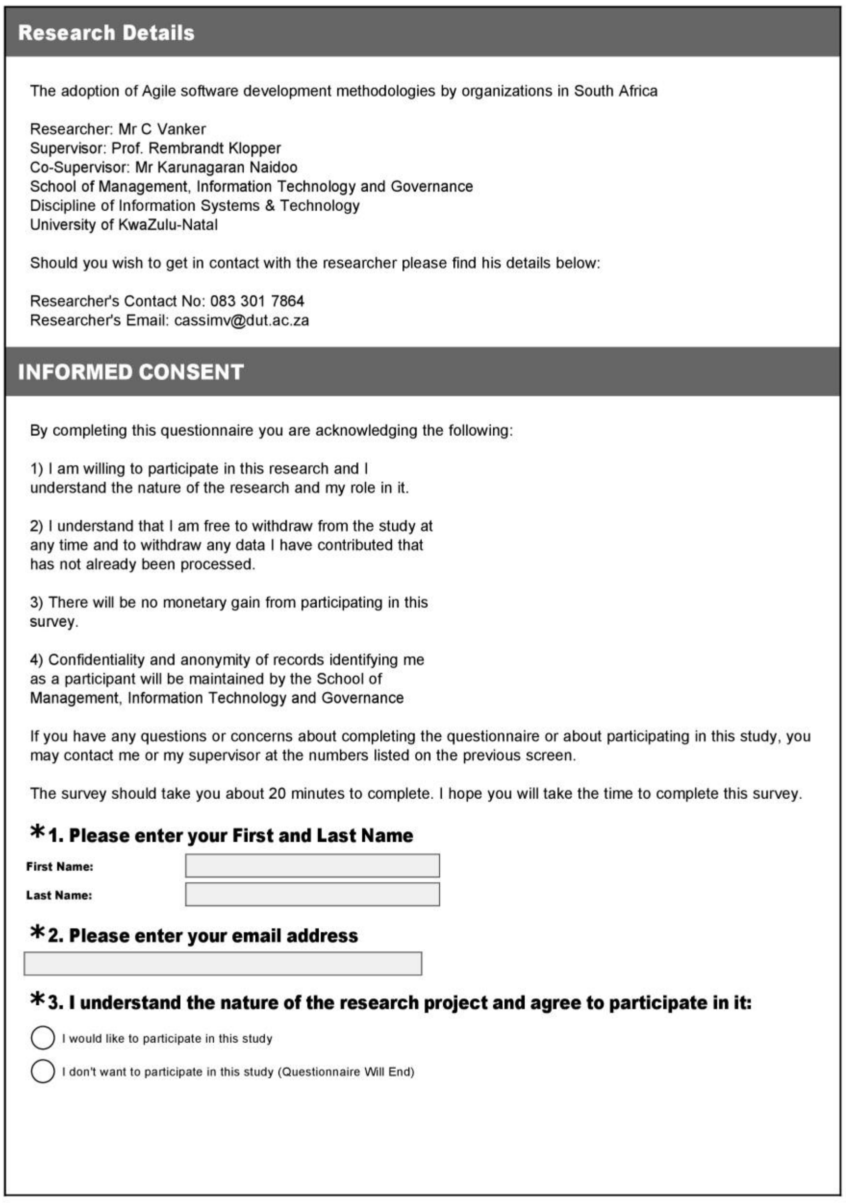
References	Software Engineering	Agile Methodologies	Popularity Of Agile	Adoption Benefits	Adoption Barriers	Total
Rubin and Rubin (2010)					1	1
Abrahamsson (2003)				1		1
Agile Alliance (2013)					1	1
Ahmed et al (2010)		1			1	2
Aken (2008)	1					1
Ambler (2006)					1	1
Ambler (2012)					1	1
Atlas (2009)					1	1
Auvinen et al (2005)				1		1
Babcock (2011)					1	1
Baudson (2012)					1	1
Beck (2000)					1	1
Beck et al (2001)	1	1			1	3
Bedoll (2003)				1		1
Benefield (2008)					1	1
Blose (2008)					1	1
Boehm and Papaccio (1998)	1					1
Boehm and Turner (2011)					1	1
Brooks(1995)					1	1
Bustard et al (2013)			1			1
Cao and Ramesh (2008)	1					1
Cao et al (2009)					1	1
Charette (2005)					1	1
Chuang et al (2014)	1					1
Clutterbuck et al (2009)	1	1				2
Cockburn and Highsmith (2001)			1			1
Pichler (2006)					1	1
Cohn and Ford (2003)					1	1

References	Software Engineering	Agile Methodologies	Popularity Of Agile	Adoption Benefits	Adoption Barriers	Total
Conboy et al (2011)					1	1
Copeland (2001		1				1
Coram and Bohner (2005)					1	1
Daneva et al (2012)					1	1
DSDM Consortium (2013)					1	1
Duka (2012)		1				1
Dyba and Dingsoyr (2008)	1	1	1	1	1	5
Dyba et al (2007)					1	1
Erickson et al (2005)		1				1
Fowler (2005)	1					1
Fowler and Highsmith (2001)					1	1
Goyal (2008)		1				1
Highsmith and Cockburn (2001)	1					1
Hoda et al (2010)					1	1
Inform IT (2002)					1	1
Jansen et al (2009)					1	1
Jiang and Eberlein (2009)	1	1				2
Korhonen (2011)				1	1	2
Laanti et al (2011)				1	1	2
Lacey (2012)		1				1
Layman et al (2004)				1		1
Leau et al (2012)	1					1
Lindvall et al (2002)					1	1
Livermore (2007)					1	1
Loftus and Ratcliffe (2005)		1				1
Mangalaraj et al (2005)				1	1	2
Mannaro et al (2004)				1		1
Maruping et al (2009)					1	1
Maurer and Martel (2002)				1		1

References	Software Engineering	Agile Methodologies	Popularity Of Agile	Adoption Benefits	Adoption Barriers	Total
Melymuka (2012)					1	1
Mirnalini and Venkata (2010)		1				1
Misra et al (2010)					1	1
Mnkandla (2009)	1					1
Mnkandla and Dwolatzky (2004)		1				1
Moe et al (2008)					1	1
Moreira (2013)					1	1
Murphy et al (2013)	1			1	1	3
Naftanaila (2009)	1		1			2
Nagler (2004)					1	1
Nerur and Balijepally (2007)					1	1
Noll (2007)		1				1
Nord and Tomayko (2006)					1	1
Nordberg III (2003)					1	1
Omar et al (2011)					1	1
Osherove (2009)					1	1
Paasivaara and Lassenius (2003)			1		1	2
Palmer and Felsing (2002)		1				1
Poole et al (2001)				1		1
Randolph and Raynar (1977)					1	1
Rasmussen (2003)					1	1
Reneson (2006)					1	1
Rumpe and Schröder (2002)			1			1
Saleh (2009)					1	1
Schatz and Abdelshafi (2006)				1	1	2
Schwaber et al (2007)				1		1

References	**Software Engineering**	**Agile Methodologies**	**Popularity Of Agile**	**Adoption Benefits**	**Adoption Barriers**	**Total**
Senapahti and Srinivasan (2012)				1		1
Sharp and Robinson (2004)				1		1
Shore (2007)					1	1
Sriram and Mathew (2012)				1		1
Stavru (2014).	1		1			2
Stoica et al (2013)		1			1	2
Syed et al (2014)					1	1
Takeuchi and Nonaka (1986)					1	1
Turk et al (2014)		1			1	2
Versionone (2010)					1	1
Versionone (2011)			1			1
Versionone (2013)			1			1
Vijayasarathy and Turk (2012)	1	1			1	3
Vinekar et al (2006)	1			1	1	3
WikiQuotes (2014)					1	1
Williams (2007)	1	1			1	3
Williams and Cockburn (2003)	1					1
Williams et al (2009)					1	1
Wray (2010)					1	1
Youssef (1992)		1				1

Addendum 14:



Research Details

The adoption of Agile software development methodologies by organizations in South Africa

Researcher: Mr C Vanker
Supervisor: Prof. Rembrandt Klopper
Co-Supervisor: Mr Karunagaran Naidoo
School of Management, Information Technology and Governance
Discipline of Information Systems & Technology
University of KwaZulu-Natal

Should you wish to get in contact with the researcher please find his details below:

Researcher's Contact No: 083 301 7864
Researcher's Email: cassimv@dut.ac.za

INFORMED CONSENT

By completing this questionnaire you are acknowledging the following:

1) I am willing to participate in this research and I understand the nature of the research and my role in it.

2) I understand that I am free to withdraw from the study at any time and to withdraw any data I have contributed that has not already been processed.

3) There will be no monetary gain from participating in this survey.

4) Confidentiality and anonymity of records identifying me as a participant will be maintained by the School of Management, Information Technology and Governance

If you have any questions or concerns about completing the questionnaire or about participating in this study, you may contact me or my supervisor at the numbers listed on the previous screen.

The survey should take you about 20 minutes to complete. I hope you will take the time to complete this survey.

***1. Please enter your First and Last Name**

First Name:

Last Name:

***2. Please enter your email address**

***3. I understand the nature of the research project and agree to participate in it:**

○ I would like to participate in this study

○ I don't want to participate in this study (Questionnaire Will End)

Page 1

***4. Do you live in South Africa ?**

◯ Yes

◯ No (Questionnaire Will End)

Agile Experience

This questionnaire is ONLY intended for respondents with Agile experience.

***5. Have you had experience on an Agile Project ?**

◯ Yes

◯ No (Questionnaire Will End)

Section A: Information About Your Self

***6. Please tick your age group in the appropriate block?**

◯ Under 21 years

◯ 21-30 years

◯ 31-40 years

◯ 41-50 years

◯ 51-69 years

***7. Your gender:**

◯ Female

◯ Male

***8. Please tick your racial grouping (As legislated by parliament):**

◯ Black

◯ Coloured

◯ Indian

◯ White

◯ Other (please specify)

***9. What is your current position in the company?**

- ◯ Project Manager
- ◯ Business Analyst
- ◯ Development Manager
- ◯ Team Leader
- ◯ Manager
- ◯ Another Position:

***10. How many Information Technology employees are there in your company?**

- ◯ Under 2 employees
- ◯ 2-5 employees
- ◯ 6-25 employees
- ◯ 26-100 employees
- ◯ 101-500 employees
- ◯ 501-1000 employees
- ◯ Over 1000 employees

***11. How many years of working experience do you have in your current position?**

- ◯ Under 2 years
- ◯ 2-5 years
- ◯ 6-10 years
- ◯ 11-20 years
- ◯ Over 20 years

***12. Where are you located?**

- ◯ Eastern Cape
- ◯ Free State
- ◯ Gauteng
- ◯ KwaZulu-Natal
- ◯ Limpopo
- ◯ Mpumalanga
- ◯ North West
- ◯ Northern Cape
- ◯ Western Cape

***13. Which sector is your company primarily in?**

- ◯ Mining
- ◯ Construction
- ◯ Agriculture
- ◯ Consulting
- ◯ Manufacturing
- ◯ Retail
- ◯ Shipping
- ◯ Information Technology
- ◯ Government
- ◯ Finance
- ◯ Academic
- ◯ Other (please specify)

Section B: The State of Agile In South Africa

***14. To what extent does your company use Agile methodologies/frameworks?**

Always	Most Of The Time	Sometimes	Rarely	Never
◯	◯	◯	◯	◯

***15. Rate your knowledge of Agile on a scale of 1 (not at all knowledgeable) to 5 (extremely knowledgeable)?**

***16. How long have you been practicing Agile?**

- ◯ Under 1 year
- ◯ 1-2 years
- ◯ 3-4 years
- ◯ 5-6 years
- ◯ 7-8 years
- ◯ Over 8 years

***17. How large was your last Agile team? (Please include all IT/systems staff and stakeholder representatives such as product owners or product managers.)?**

- ◯ 1-5 team members
- ◯ 6-10 team members
- ◯ 11-15 team members
- ◯ 16-20 team members
- ◯ 21-30 team members
- ◯ Over 30 team members

***18. How is your team geographically distributed?**

- ◯ All team members are in the same room
- ◯ All team members are on the same floor
- ◯ All team members are in the same building
- ◯ All team members are on the same campus
- ◯ All team members are within driving distance of each other
- ◯ All team members are within the same time zone
- ◯ Team members are in different time zones

***19. How often do team members communicate with stakeholders?**

- ◯ Regularly - on a daily basis
- ◯ Regularly - on a weekly basis
- ◯ Only at the start of an iteration
- ◯ Only as needed throughout the project

***20. In your last Agile project what type of methodology/framework did you follow?**

◯ SCRUM

◯ Hybrid Custom

◯ KANBAN

◯ SCRUM-BAN

◯ Crystal Clear

◯ Feature-Driven Development

◯ Extreme Programming

◯ Lean

◯ Agile Unified Process

◯ Agile Modelling

◯ Dynamic Systems Development Method

◯ Other (please specify)

***21. What was the iteration length of you last Agile project?**

◯ 1-7 Days

◯ 8-14 Days

◯ 15-21 Days

◯ 22-30 Days

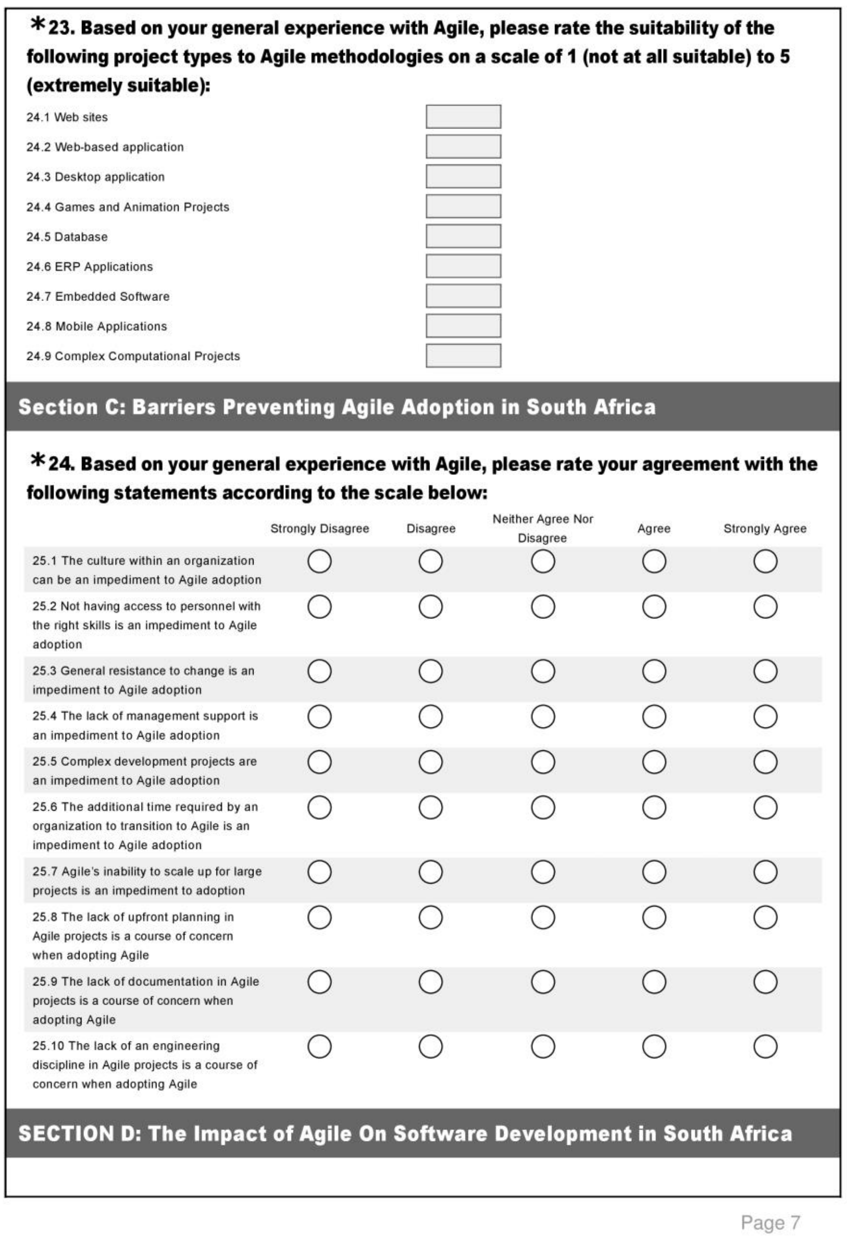
◯ Over 30 Day

***22. How would you describe your company's Agile adoption strategy (Select ONE option only)?**

◯ Our company was an early adopter of Agile process

◯ Our company was a fast follower of Agile process

◯ Our company was a late adopter of Agile process



***23. Based on your general experience with Agile, please rate the suitability of the following project types to Agile methodologies on a scale of 1 (not at all suitable) to 5 (extremely suitable):**

24.1 Web sites	
24.2 Web-based application	
24.3 Desktop application	
24.4 Games and Animation Projects	
24.5 Database	
24.6 ERP Applications	
24.7 Embedded Software	
24.8 Mobile Applications	
24.9 Complex Computational Projects	

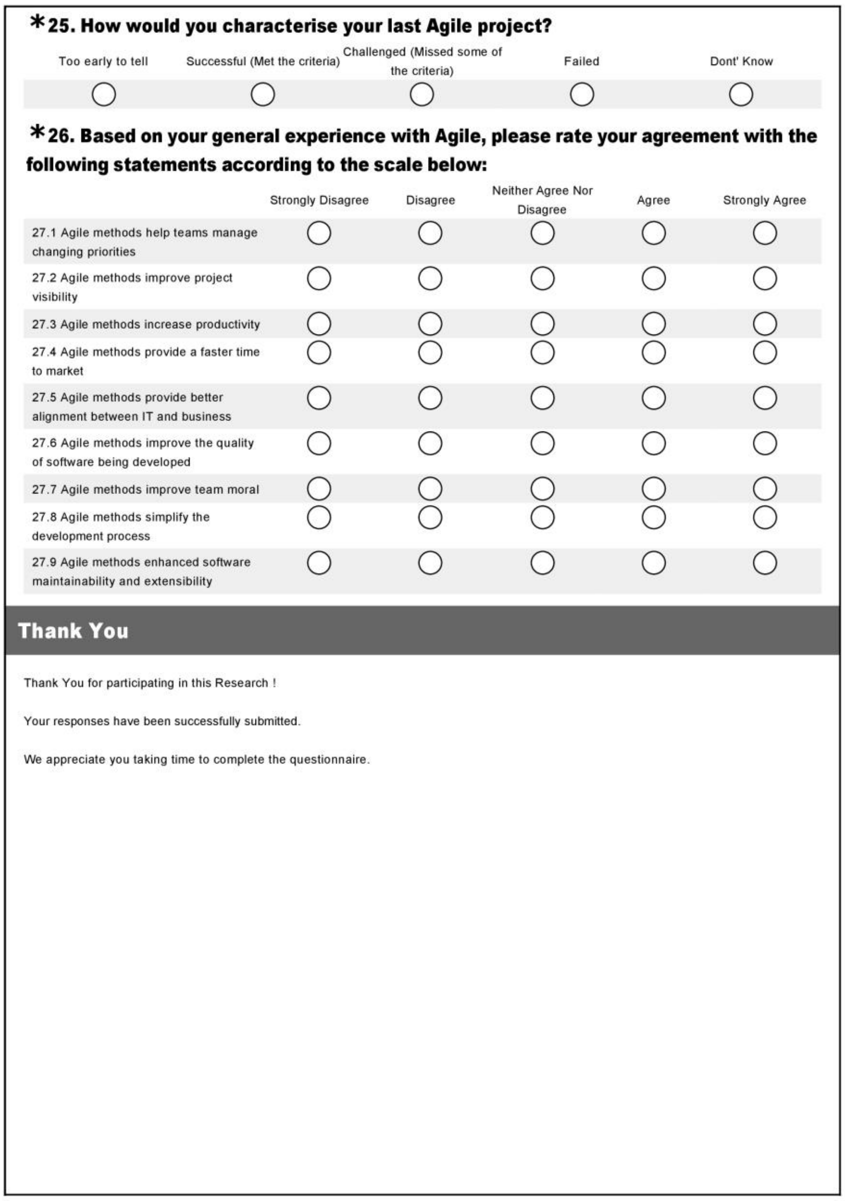
Section C: Barriers Preventing Agile Adoption in South Africa

***24. Based on your general experience with Agile, please rate your agreement with the following statements according to the scale below:**

	Strongly Disagree	Disagree	Neither Agree Nor Disagree	Agree	Strongly Agree
25.1 The culture within an organization can be an impediment to Agile adoption	○	○	○	○	○
25.2 Not having access to personnel with the right skills is an impediment to Agile adoption	○	○	○	○	○
25.3 General resistance to change is an impediment to Agile adoption	○	○	○	○	○
25.4 The lack of management support is an impediment to Agile adoption	○	○	○	○	○
25.5 Complex development projects are an impediment to Agile adoption	○	○	○	○	○
25.6 The additional time required by an organization to transition to Agile is an impediment to Agile adoption	○	○	○	○	○
25.7 Agile's inability to scale up for large projects is an impediment to adoption	○	○	○	○	○
25.8 The lack of upfront planning in Agile projects is a course of concern when adopting Agile	○	○	○	○	○
25.9 The lack of documentation in Agile projects is a course of concern when adopting Agile	○	○	○	○	○
25.10 The lack of an engineering discipline in Agile projects is a course of concern when adopting Agile	○	○	○	○	○

SECTION D: The Impact of Agile On Software Development in South Africa

Page 7



***25. How would you characterise your last Agile project?**

Too early to tell	Successful (Met the criteria)	Challenged (Missed some of the criteria)	Failed	Dont' Know
○	○	○	○	○

***26. Based on your general experience with Agile, please rate your agreement with the following statements according to the scale below:**

	Strongly Disagree	Disagree	Neither Agree Nor Disagree	Agree	Strongly Agree
27.1 Agile methods help teams manage changing priorities	○	○	○	○	○
27.2 Agile methods improve project visibility	○	○	○	○	○
27.3 Agile methods increase productivity	○	○	○	○	○
27.4 Agile methods provide a faster time to market	○	○	○	○	○
27.5 Agile methods provide better alignment between IT and business	○	○	○	○	○
27.6 Agile methods improve the quality of software being developed	○	○	○	○	○
27.7 Agile methods improve team moral	○	○	○	○	○
27.8 Agile methods simplify the development process	○	○	○	○	○
27.9 Agile methods enhanced software maintainability and extensibility	○	○	○	○	○

Thank You

Thank You for participating in this Research !

Your responses have been successfully submitted.

We appreciate you taking time to complete the questionnaire.

Addendum 15:

Mr Cassim Vanker
P.O. Box 37300
Overport
4091

Wednesday, 01 May 2013

Dear Sir/Madam

I am a student at the University of KwaZulu-Natal, currently studying towards a Masters in Commerce (MCOM) in Information Systems & Technology. As such, I am required to complete a research study and the topic that I have chosen for my study is "*The adoption of agile software development methodologies by organizations in South Africa."*

The research will be conducted through the use of a questionnaire, which would be administered to Project Managers, Scrum Masters or personnel that manage agile software development projects within your organization. As a participant in this study, please be advised that:

1) This research will be conducted on a voluntary, anonymous basis. No company or individual will be identified by name and only aggregated results will appear in print.

2) The research is of a constructive, cooperative nature, and respondents would be able to withdraw at any time.

3) Research will only commence after ethical clearance has been obtained from the University's ethical clearance committee.

The ethical clearance policy of the University requires that I apply for ethical clearance once I have confirmed participants for my study. As such, I would highly appreciate it if you can complete the letter of permission (a template has been provided). Should you require a copy of my draft questionnaire please don't hesitate to contact me.

In conclusion, this study is important as it will give a broad overview of the current state of agile processes among organization in South Africa. It will also provide valuable information to those organizations intending to make the process shift.

Should you have any questions, please feel free to contact me or my supervisors, the relevant contact details are listed below.

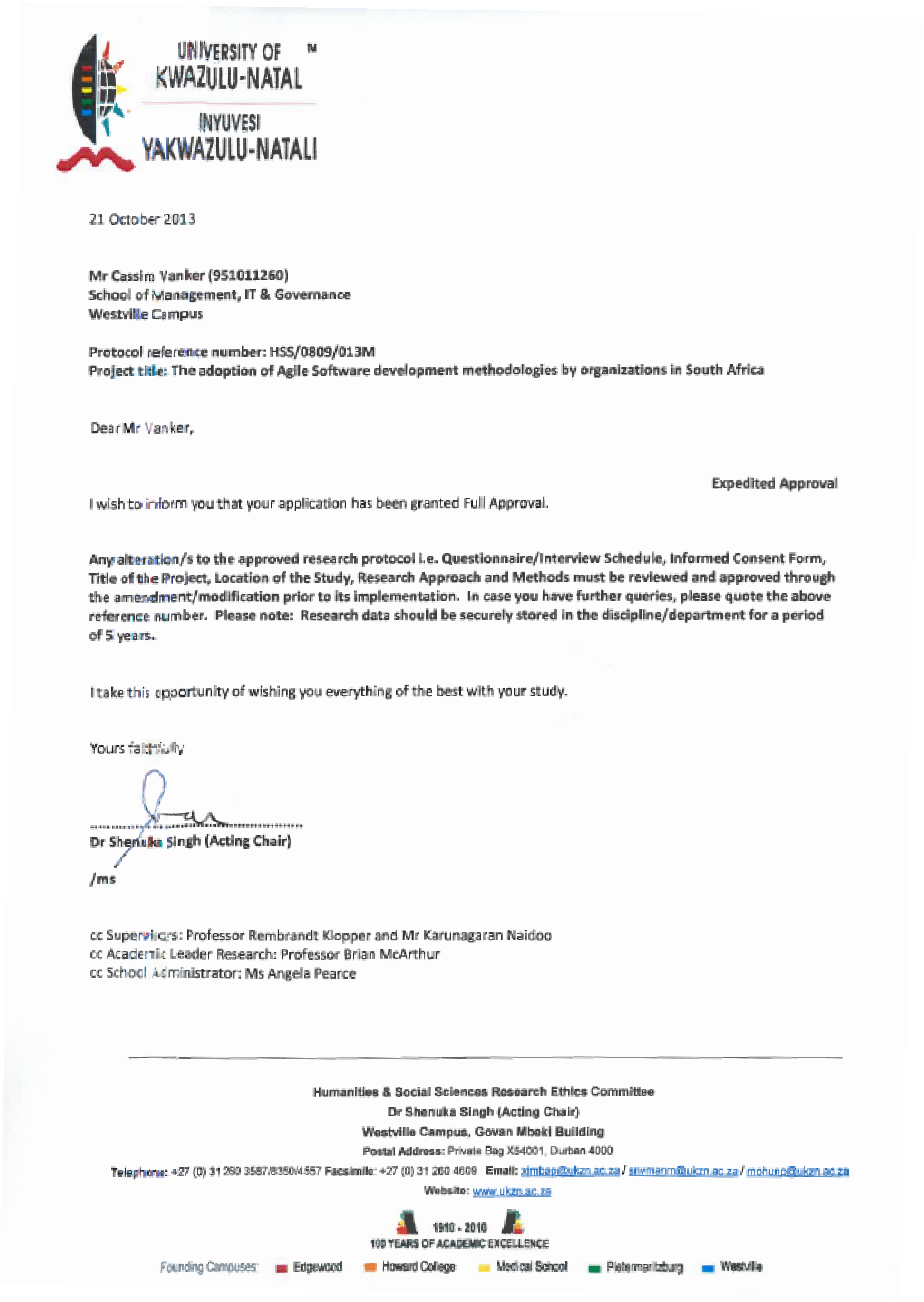
Kind Regards,

Cassim Vanker

Mr C. Vanker
Tel: (031) 373-5596
Home: (031)208-9720
Mobile: (083) 301-7864
Mailto: cassimv@dut.ac.za

Supervisor: Prof. Rembrandt Klopper
mail to: rklopper@gmail.com
Co-Supervisor: Mr Karunagaran Naidoo
Mailto: Naidook82@ukzn.ac.za
Tel: 031-260 2216

Addendum 16:



UNIVERSITY OF KWAZULU-NATAL ™
INYUVESI YAKWAZULU-NATALI

21 October 2013

Mr Cassim Vanker (951011260)
School of Management, IT & Governance
Westville Campus

Protocol reference number: HSS/0809/013M
Project title: The adoption of Agile Software development methodologies by organizations in South Africa

Dear Mr Vanker,

Expedited Approval

I wish to inform you that your application has been granted Full Approval.

Any alteration/s to the approved research protocol i.e. Questionnaire/Interview Schedule, Informed Consent Form, Title of the Project, Location of the Study, Research Approach and Methods must be reviewed and approved through the amendment/modification prior to its implementation. In case you have further queries, please quote the above reference number. Please note: Research data should be securely stored in the discipline/department for a period of 5 years.

I take this opportunity of wishing you everything of the best with your study.

Yours faithfully

[illegible]
....
Dr Shenuka Singh (Acting Chair)

/ms

cc Supervisors: Professor Rembrandt Klopper and Mr Karunagaran Naidoo
cc Academic Leader Research: Professor Brian McArthur
cc School Administrator: Ms Angela Pearce

Humanities & Social Sciences Research Ethics Committee
Dr Shenuka Singh (Acting Chair)
Westville Campus, Govan Mbeki Building
Postal Address: Private Bag X54001, Durban 4000
Telephone: +27 (0) 31 260 3587/8350/4557 **Facsimile:** +27 (0) 31 260 4609 **Email:** ximbap@ukzn.ac.za / snymanm@ukzn.ac.za / mohunp@ukzn.ac.za
Website: www.ukzn.ac.za

1910 - 2010
100 YEARS OF ACADEMIC EXCELLENCE

Founding Campuses: 🟥 Edgewood 🟧 Howard College 🟨 Medical School 🟩 Pietermaritzburg 🟦 Westville