Investigating critical success factors in agile systems development projects

Ruhan Wagener

20850352

Dissertation submitted in partial fulfilment of the requirements for the degree M.Com Computer Science and Information Systems in the Faculty of Natural Sciences of the Potchefstroom Campus of the North-West University

Supervisor: Prof. H.M. Huisman

November 2012

Acknowledgements

This study would not have been possible without the help of my family and friends. I would like to thank my parents in particular for helping wherever they could and always supporting me in my endeavours. To each and every respondent that took part in this study by taking the time to complete a questionnaire, thank you. Without your participation, this study would not have been possible.

A big thank you to my supervisor, Professor Magda Huisman for her continued support, financially, personally, and academically. To my language editor, Professor Annette Combrink, thank you very much for all the hard work and dedication. I would also like to thank the staff of the Ferdinand Postma Library for their friendly assistance and guidance during my research. Finally, thank you to all my fellow masters students and colleagues for their assistance and motivation throughout this study.

Abstract

This study investigates the critical success factors involved in agile systems development projects. Various systems development methodologies and project management methodologies are presented with their underlying principles, strengths and weaknesses. Thereafter the critical success factors adopted from the work of Chow and Cao (2007) are presented.

A positivistic research paradigm was chosen for data collection and analysis. The survey method was chosen for data collection. A questionnaire was sent to multiple respondents in a predominantly agile work environment, which resulted in a total of 129 respondents in various business sectors.

The results were analysed by implementing multiple correlation and regression statistics as well as descriptive statistics. The results show that there are in fact 16 critical success factors that have a direct impact on the success of agile systems development projects. Agile systems development methodologies have been increasing in use during the last 3 years, and most organisations are implementing some form of project management methodology. The first recommendation is based on the findings that strong customer involvement and the appropriate management of the agile process with a satisfactory amount of documentation resulted in greater process success. Therefore, organisations should encourage these critical success factors when implementing an ASDM as this has a positive effect on the project outcome.

The appropriate management of the agile process with a satisfactory amount of documentation, the application of good design practices and technical knowledge to a project, and a cooperative organizational culture instead of hierarchical are three of the key critical success factors that were positively related to the success of the product. By focussing on these critical success factors, the success of the entire project can be predicted.

Keywords: Critical success factors, agile, system development methodologies, project management, information technology

Opsomming

Hierdie studie het gefokus op ‘n ondersoek van die kritieke suksesfaktore wat onderliggend is aan agile-stelselsontwikkelingsprojekte. Verskeie stelselsontwikkelingsmetodologieë en projekbestuurmetodologieë is aangebied in samehang met hulle onderliggende beginsels, sterktes en swakhede. Daarna is die kritieke suksesfaktore, soos ontleen uit die werk van Chow and Cao (2007) aangebied vir oorweging.

‘n Positivistiese navorsingsparadigma is gekies vir die data-insameling en ontleding. Die opname-metode is gebruik vir data-insameling. ‘n Vraelys is na veelvuldige respondente gestuur wat in ‘n oorwegend agile werkomgewing werk, en dit het gelei tot ‘n aantal van 129 respondente uit verskeie besigheids omgewings.

Die resultate is ontleed met die gebruik van veelvoudige korrelasie- en regressie-statistieke. Die uitslae het getoon dat daar in werklikheid 16 kritieke suksesfaktore is wat ‘n direkte impak het op die sukses van agile stelselsontwikkelingsprojekte. Agile stelselsontwikkelingsmetodologieë is toenemend gebruik in die afgelope drie jaar, en meeste organisasies implementeer een of ander vorm van projekbestuursmetodologie. Die eerste aanbeveling word gemaak op grond van die bevinding dat sterk kliënte betrokkenheid en die gepaste bestuur van die agile proses met genoegsame dokumentasie gelei het tot ‘n groter sukses van die proses. Organisasies behoort hierdie kritieke suksesfaktore aan te spreek wanneer ‘n agile stelselsontwikkelingsmetodologie geïmplementeer word omrede dit ‘n positiewe uitwerking het op die uitkoms van die projek.

Die gepaste bestuur van die agile proses met genoegsame dokumentasie, die toepassing van goeie ontwerp praktyke en tegniese kennis op ‘n projek en ‘n samewerkende in plaas van hierargiese kultuur in die organisasie is drie van die deurslaggewende kritieke sukses faktore wat geïdentifiseer is. Deur op hierdie kritieke suksesfaktore te fokus, kan die sukses van die hele projek voorspel word.

Sleutelwoorde: Kritieke suksesfaktore, agile, stelselontwikkelingmetodologieë, projekbestuur, inligtingstegnologie

Table of Contents

		Page
	Acknowledgements	i
	Abstract	ii
	Opsomming	iii
Chapter 1	**The problem statement**	**1**
1.1	Introduction	1
1.2	Problem statement	1
1.3	Research objectives	11
1.4	Research methodology	11
1.5	Chapter outline	12
1.6	Summary	13
Chapter 2	**Critical success factors information technology projects implementing ASDMs and PMMs**	**14**
2.1	Introduction	14
2.2	A general introduction to Agile Methods	14
2.3	History of Agile Methods	16
2.3.1	The Agile Manifesto	16
2.3.2	History of the Agile Manifesto	17
2.3.3	Principles of the Agile Manifesto	18
2.4	Effectiveness of Agile Systems Development Methodology	19
2.4.1	Agile Methodology Characteristics	19
2.4.2	Factors with a negative impact on ASDMs	19
2.5	Advantages of Agile Methodologies	19
2.6	Disadvantages of Agile Methodologies	20
2.7	Timeline of Agile Methodologies	21
2.7.1	Lean Software Development	21
2.7.2	Scrum	22
2.7.3	Dynamic Systems Development	22
2.7.4	Rapid Application Development	22
2.7.5	Extreme Programming	22
2.7.6	Feature-Driven Development	23
2.7.7	Crystal	23
2.8	Different Agile Methodologies	23
2.8.1	Lean Software Development	24
2.8.2	Scrum	27
2.8.3	Dynamic Systems Development	31
2.8.4	Rapid Application Development	33
2.8.5	Extreme Programming	36
2.8.6	Feature-Driven Development	40

2.8.7	Crystal	43
2.9	Comparison of Agile Systems Development Methodologies	45
2.9.1	Comparison with other methods	45
2.9.2	Lightweight vs Heavyweight methodologies	49
2.10	Project Management Methodologies	50
2.11	The Project Management Body of Knowledge	53
2.12	PRINCE2	57
2.13	Comparison between PRINCE2 and PMBOK	62
2.14	Critical success Factors in Agile Systems Development Projects	66
2.15	Critical evaluation of ASDM and PMM against critical success factors	68
2.16	Conclusion	76
Chapter 3	**Research methodology – a positivistic approach**	**77**
3.1	Introduction	77
3.2	Research paradigm	77
3.3	Research strategies and data-generation methods associated with the positivistic research paradigm	81
3.4	Data-analysis methods associated with the research strategies used in positivism	84
3.5	Implementation for this study	87
3.6	Conclusion	89
Chapter 4	**Research results and analysis**	**90**
4.1	Introduction	90
4.2	Research method	90
4.2.1	Development and testing of the questionnaire	90
4.2.2	Data collection	92
4.2.3	Measurement of the research variables	95
4.2.4	The use of systems development methodologies	96
4.2.5	The use of project management methodologies	100
4.2.6	Measurement and reliability of critical success factor variables	104
4.2.7	Critical success factors	105
4.2.8	Project outcome (effectiveness)	116
4.2.9	Success of the process	117
4.2.10	Success of the product	128
4.3	Summary	141
Chapter 5	**Conclusion and recommendations**	**142**
5.1	Introduction	142
5.2	Contributions of this research	142
5.2.1	Determine critical success factors in agile systems development projects	142

5.2.2	Evaluate ASDMs against the various critical success factors	144
5.2.3	Evaluate PMMs against the various critical success factors	145
5.2.4	Study the use/effectiveness of agile systems development methodologies in the industry	147
5.2.5	Study the use/effectiveness of project management methodologies in the industry	147
5.3	Practical implications	148
5.4	Limitations and recommendations for future research	148
5.5	Conclusion	149
6	**References**	**150**
	Appendix A – Research Questionnaire	**157**

List of Tables

		Page
1.1	Literature study	4
2.9.1	Comparison between Agile SDMs and Traditional SDMs	47
2.9.2	Comparison between the different ASDMs	48
2.10.1	Performance gaps	52
2.13.1	Comparison of PMBOK areas of knowledge and PRINCE2 components	63
2.13.2	Comparison of PMBOK areas of knowledge and PRINCE2 principles, themes and processes	63
2.13.3	Comparison of PMBOK process groups and PRINCE2 processes	65
2.15.1	Evaluation of ASDMs against various critical success factors	70
2.15.2	Evaluation of PMMs against various critical success factors	74
3.2.1	Comparison of research paradigms	81
3.3.1	Relative strengths of survey and experimentation	83
3.5.1	Response rate per company	88
4.2.1.1	Information captured by questionnaire	91
4.2.2.1	Business area of respondent's organisation	93
4.2.2.2	Respondent profile	94
4.2.2.3	Respondent's work delegation	95
4.2.4.1	Systems development methodologies	96
4.2.4.2	Type of Systems Development Methodologies used	97
4.2.4.3	Proportion of projects applying SDM knowledge	98
4.2.4.4	Proportion of people applying SDM knowledge	98
4.2.4.5	Horizontal use of SDM	98
4.2.4.6	Systems Development Methodologies - strictness of use	99
4.2.4.7	Systems Development Methodologies- Time of use	100
4.2.5.1	Project Management Methodologies - vertical use	101
4.2.5.2	Proportion of projects applying PMM knowledge	101
4.2.5.3	Proportion of people applying PMM knowledge	102
4.2.5.4	Project Management Methodologies - Horizontal use	102
4.2.5.5	Project Management Methodologies - Strictness of use	103
4.2.5.6	Project Management Methodologies - Time of use	103
4.2.7.1	Critical success factors in information technology projects	105
4.2.7.2	Top 10 critical success factors in information	107

	technology projects	
4.2.7.3	Lowest 10 critical success factors in information technology projects	108
4.2.7.4	Critical success factors reliability measures	114
4.2.7.5	Critical success factors descriptive statistics	115
4.2.8.1	Project size	116
4.2.8.2	Project outcome	117
4.2.9.1	Process success reliability measures	119
4.2.9.2	Process success descriptive statistics	119
4.2.9.3	Correlation statistics on process success	119
4.2.9.4	Standard regression - Process and Project Management Quality	121
4.2.9.5	Forward stepwise regression - Process and Project Management Quality	123
4.2.9.6	Forward stepwise regression summary - Process and Project Management Quality	124
4.2.9.7	Backward stepwise regression - Process and Project Management Quality	125
4.2.9.8	Standard regression - Excellence of the project	125
4.2.9.9	Forward stepwise regression - Excellence of the project	126
4.2.9.10	Forward stepwise regression summary - Excellence of the project	127
4.2.9.11	Backward stepwise regression - Excellence of the project	127
4.2.10.1	Product success reliability measures	129
4.2.10.2	Product success descriptive statistics	130
4.2.10.3	Correlation statistics on product success	130
4.2.10.4	Standard regression - Product quality and user-friendliness	132
4.2.10.5	Forward stepwise regression - Product quality and user-friendliness	134
4.2.10.6	Forward stepwise regression summary - Product quality and user-friendliness	135
4.2.10.7	Backward stepwise regression - Product quality and user-friendliness	136
4.2.10.8	Standard regression – Sustainability of the developed system	136
4.2.10.9	Forward stepwise regression - Sustainability of the developed system	138
4.2.10.10	Forward stepwise regression summary - Sustainability of the developed system	139
4.2.10.11	Backward stepwise regression - Sustainability of the	140

developed system
5.2.1 The critical success factors in IT projects 143
5.2.2 The critical success factors not addressed by ASDMs 145
5.2.3 The critical success factors not addressed by PMMs 146

List of Figures

Figure 2.6.1	Timeline of ASDMs	21
Figure 2.7.2.2	Scrum Flow	27
Figure 2.7.5.1	XP Lifecycle	37
Figure 2.7.6.1	FDD Process Model	40
Figure 2.7.7.1	Dimension of Crystal Methodologies	43
Figure 2.11.1	The five project management process groups	55
Figure 2.11.2	The nine project management knowledge areas	57
Figure 2.12.1	The use of PRINCE2 components and techniques in the processes	61
Figure 3.2.1	Research Paradigm Characteristics	79
Figure 3.4.1	Epistemological assumptions underlying qualitative research	86
Figure 3.4.2	Epistemological assumptions underlying qualitative and quantitative research	87

Chapter 1

The problem statement

1.1 Introduction

In this chapter the problem statement and the research objectives of the research study are defined. A definition and introduction to agile systems development methodologies (ASDM) and project management methodologies (PMM) are presented, as well as an overview of the critical success factors in agile systems development projects. The last section of this chapter outlines the remaining chapters of this study.

1.2 Problem statement

According to the CHAOS Summary 2009, 32% of all projects succeeded, meaning that they were delivered on time, within the budget and with all the required features and functions; 44% were challenged, implying late delivery, over budget, and not meeting the user requirements. A massive 24% of all projects failed which were either cancelled before being completed or delivered and never used. This failure rate is the highest in the last decade (CHAOS Summary, 2009).

In the United States, a 2008 Government Accountability Office report stated that over 400 U.S. government agency IT projects, with an estimated collective value of $25 billion, suffer from poor planning and underperformance (Schwalbe, 2010). According to surveys on 8000 projects, most project failures are due to stakeholder problems (Ceschi *et al*., 2005). One of the top reasons for failure is a result of poor communication between the development team and the customer (Ceschi *et al*., 2005).

It is evident from these statements above that there are severe problems regarding IT projects. Companies and individuals incur massive financial losses due to projects failing or being cancelled prior to being completed.

The traditional approach with regard to the management of complex projects is to use a project management methodology to improve the system development process. System development projects implementing ASDMs should be able to exist in an environment where project management methodologies are used (Karlström & Runeson, 2006). This implies that ASDMs and PMMs should be able to coexist when a system is being developed in order to deliver a better product.

The financial impact of failed information technology projects is critical to any organization and its sustainability in the corporate environment. It is therefore of great importance that the failure of information technology projects be kept to an absolute minimum.

During this study the critical success factors for agile systems development projects will be investigated by means of a survey focusing on agile systems development methodologies and project management methodologies in particular in order to find solutions to the problems identified earlier in this chapter.

Some typical critical success factors of agile systems development projects include culture, people, time, budget, scope, user acceptance and communication (Kerzner, 2009; Misra *et al.,* 2009). Contrary to system development methodologies being implemented, system development has not had a consistent success rate, often resulting in delayed, failed, abandoned or rejected projects. The challenge to information technology professionals is to find a way of improving system development. Current trends in this area of research lean towards the critical success factors in agile systems development methods (Chow & Cao, 2007), which will be critically addressed in this study.

In the next section, an explanation of agile systems development projects is given as well as an introduction to certain critical success factors for agile systems development projects in general. Special focus is put on agile systems development methodologies and project management methodologies.

"Agile" refers to a certain readiness for motion or the quality of being agile. It implies a certain factor of dexterity in a specific situation (Abrahamsson *et al.*, 2002). When saying that an organization or company is agile, it implies that such an entity should be more responsive to sudden changes. Agile systems development projects are projects which involve the implementation of an ASDM.

With an ASDM, the emphasis is on the working system or working part of that system. Together with face-to-face communication, ASDMs produce less documentation than any other methods. Agile systems development methodologies provide organizations with the ability to rapidly produce IS solutions (Highsmith, 1999; Sutherland & van den Heuvel, 2002).

The following ASDMs will be studied in the subsequent chapters: Lean Software Development (Chan & Thong, 2009; Dyba & Dingsoyr, 2008), Scrum (Chan & Thong, 2009; Abrahamsson *et al.*, 2002), Dynamic Systems Development (Ketter *et al.,* 2009; Dyba & Dingsoyr, 2008), Rapid Application Development (CASEMaker, 2000), Extreme Programming (Chan & Thong, 2009; Ketter *et al.,* 2009), Feature Driven Development (Abrahamsson *et al.*, 2002; Dyba & Dingsoyr, 2008), and Crystal (Chan & Thong, 2009; Abrahamsson *et al.*, 2002). These are the leading methodologies currently in practice, which merits their review.

The second part of the study will focus on project management methodologies. Firstly, a proper definition of project management needs to be formulated in order to fully understand the scope thereof:

Project Management is the application of various tools and techniques to steer and direct all the different resources that are relevant toward the accomplishment of a uniquely defined task. Such a task must be completed within time, cost and quality constraints and each task requires a unique application or mix of the available tools and techniques that need to be applied in order to successfully complete the task (Atkinson, 1999).

The following project management methodologies will be studied in the following chapters: PMBOK® (Cleland, 1995) and PRINCE2 (Wideman, 2002; Siegelaub, 2006).

There is very little academic literature available regarding the critical success factors in agile systems development projects, making this study of great value in the particular field. *Table* 1.1 below supports the reason to investigate the critical success factors in agile systems development projects as it shows the lack of academic literature on the specific subject.

Table 1.1 contains the various combinations of keywords used on all the various academic databases with the number of search results matching these unique keywords. The number of search results that apply to this specific study is supplied in brackets and these results were restricted to a period between 2005 and 2012. The search results explicitly show a lack of academic literature regarding this subject of study.

***Table* 1.1 - Literature study**

Keywords:	Agile develop* AND project manage*	Combining agile method* AND project manage*	Agile method* AND project manage*	Project manage* method* AND Agil*	Critical success factor* AND agile project*
Science Direct	17 (1)	0 (0)	13 (1)	12 (1)	1 (1)
	1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008)		1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*
Scopus	52 (6)	3 (2)	31 (6)	33 (6)	3 (1)
	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010)	1. Combining agile software projects and large-scale organizational agility (Kettunen & Laanti, 2008)	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*	1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*
	2. Agile architecture interactions (Madison, 2010)				
	3. Project management in plan-based and Agile companies (Ceschi *et al*., 2005)	2. Combining Agile methods with stage-gate project management (Karlström & Runeson,	2. The impact of agile principles on market-driven software product development (Fogelström *et al*., 2010)	2. The impact of agile principles on market-driven software product development (Fogelström *et al*., 2010)	
	4. Integrating agile software				

Scopus continued...	development into stage-gate managed product development (Karlström & Runeson, 2006)	2005)	3. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	*repeat article* 3. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	
	5. The role of project management in ineffective decision making within agile software development projects (McAvoy & Butler, 2009)		4. Integrating agile software development into stage-gate managed product development (Karlström & Runeson, 2006) *repeat article*	4. Integrating agile software development into stage-gate managed product development (Karlström & Runeson, 2006) *repeat article*	
	6. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*		5. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005) *repeat article*		
			6. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	5. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005)	

				repeat article 6. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	
IEEE Xplore	11 (1)	0 (0)	4 (1)	4 (1)	0 (0)
	1. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*		1. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	1. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	
EBSCOhost	23 (1)	1 (1)	23 (5)	5 (0)	1 (1)
	1. The role of project management in ineffective decision making within agile software development projects (McAvoy & Butler, 2009) *repeat article*	1. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005) *repeat article*	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*		1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*
			2. The role of project management in ineffective decision		

EBSCOhost continued...			making within agile software development projects (McAvoy & Butler, 2009) *repeat article*		
			3. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*		
			4. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*		
			5. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005) *repeat article*		
ISI Web of Knowledge	40 (5)	1 (1)	26 (5)	29 (5)	2 (1)
ISI Web of Knowledge continued...	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*	1. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005) *repeat article*	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*	1. Distributed agile: Project management in a global environment (Lee & Yong, 2010) *repeat article*	1. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*

	2. The role of project management in ineffective decision making within agile software development projects (McAvoy & Butler, 2009) *repeat article*		2. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	2. Identifying some important success factors in adopting agile software development practices (Misra, Kumar & Kumar, 2009)	
	2. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*		3. Integrating agile software development into stage-gate managed product development (Karlström & Runeson, 2006) *repeat article*	3. A survey study of critical success factors in agile software projects (Chow & Cao, 2008) *repeat article*	
	4. Integrating agile software development into stage-gate managed product development (Karlström & Runeson, 2006) *repeat article*		4. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	4. Integrating agile software development into stage-gate managed product development (Karlström & Runeson, 2006) *repeat article*	
	5. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*		5. Combining Agile methods with stage-gate project management (Karlström & Runeson, 2005) *repeat article*	5. Project management in plan-based and Agile companies (Ceschi *et al*., 2005) *repeat article*	

(***repeat article – means that the same article was found in multiple academic databases)***

From *Table* 1.1 above, it is clear that there is very little academic literature available on the research subject. Many of the databases consulted yielded the same articles, in which case the entry was marked as a repeat article. Only 10 unique articles were found. This lack of academic literature, supports the problem statement and reason for this study to be performed.

Formal studies regarding the critical success factors in agile systems development projects have not been done based on recent searches in peer reviewed academic literature or practitioner literature (Chow & Cao, 2007). Some examples of critical success factors in agile systems development projects include:

- a committed sponsor or manager;
- face-to-face communication;
- a facility with a proper agile-style work environment;
- strong customer commitment and presence;
- managers who are knowledgeable in the agile process.

These are just some of the critical success factors that will be investigated in this study.

1.3 Research objectives

The primary objective of this study is to investigate the critical success factors involved in agile systems development projects.

In order to achieve the primary objective, there are certain secondary objectives that need to be achieved, e.g. one has to:

- Determine critical success factors in agile systems development projects;
- Evaluate ASDMs against the various critical success factors;
- Evaluate PMMs against the various critical success factors;
- Study the use/effectiveness of agile systems development methodologies in the industry; and
- Study the use/effectiveness of project management methodologies in the industry.

1.4 Research methodology

This study implemented a positivistic research paradigm by means of a survey. Questionnaires were distributed to over 25 companies and 175 individuals to

collect data. Of these 175, there were 129 respondents, resulting in a response rate of 74%. The questions targeted different aspects of each individual with the main focus being their perceptions regarding the critical success factors of agile systems development projects . Background information such as profession and the sector of the particular organisation formed part of the first section of the questionnaire.

In section 2 the use of system development methodologies (which focuses on the end-product) and project management methodologies (which focuses on the systems development process) were measured. In section 3, 43 critical success factors in IT projects were measured. The project outcome was finally measured in section 4 to establish how effective the systems development methodology and project management methodology being implemented by that particular company is. The results were measured using statistical methods on which the recommendations are based. The results and recommendations are presented in later chapters.

1.5 Chapter outline

Chapter 1
Introduction and problem statement

This chapter introduces the focus of this research. The problem statement is highlighted and the research method of investigating the problem is also discussed.

Chapter 2
Critical success factors impacting on information technology projects implementing ASDMs and PMMs

This chapter outlines the critical success factors involved when reviewing information technology projects. A thorough introduction is given regarding agile systems development methodologies and project management methodologies.

Chapter 3
Research methodology

This chapter discusses the methods through which this research was executed. A general overview of different research methods are discussed and supporting arguments for the chosen method are given.

Chapter 4
Results and analysis

A critical analysis is performed on the data which was collected from the questionnaires. Using various statistical methods, the results are reviewed and certain correlations are drawn.

Chapter 5
Conclusion and recommendations

In this final chapter, the results of the study are discussed and recommendations are made by the author. The main purpose of the study was to investigate the critical success factors in agile systems development projects. This was accomplished by reviewing existing literature on ASDMs and PMMs and deriving a list of critical success factors by which the success of these various methodologies were measured in the industry.

1.6 Summary

This chapter outlines the problem statement and stated the various objectives of the research study. The research contribution is discussed and the research methodology presented. In the next chapter, the critical success factors in information technology projects implementing ASDMs and PMMs are investigated.

Chapter 2

Critical success factors impacting on information technology projects implementing ASDMs and PMMs

2.1 Introduction

This chapter explains what agile systems development methodologies are before thoroughly analysing agile systems development methodologies and project management methodologies. The final part of this chapter will focus on the critical success factors concerned with agile systems development projects.

2.2 A general introduction to agile methodologies

"Agile" refers to a certain readiness for motion or the quality of being agile. It implies a certain factor of dexterity in a specific situation (Abrahamson *et al.*, 2002). When saying that an organization or company is agile, it implies that such an entity is more responsive to sudden changes.

The trend towards agile systems development was the most significant development in the field of systems development in the recent past (Leffingwell, 2007). Agile systems development emphasizes the development of a system in an agile and flexible environment. It enables the entire project to be completed in a compressed time frame and in an efficient way, compared to traditional methods of systems development. Agile systems development requires continuous gathering of requirements and continuous updates throughout the project life-cycle. Agile systems development can very easily be integrated with the business process (Leffingwell, 2007).

Agile development refers to a group of methodologies based on iterative development where prerequisites and solutions evolve through the collaboration of different teams. Agile methods generally enhance a disciplined project management process, encouraging frequent inspection and adaptation, a leadership philosophy that promotes teamwork and allows for rapid development of high quality systems delivering in customer needs and business goals (Leffingwell, 2007).

There are many agile systems development methodologies which vary in many ways. Most of these methodologies focus on promoting development, teamwork and process adaptability throughout the project's life-cycle (Leffingwell, 2007).

Let's suppose the entire project is a puzzle. These agile methods divide tasks into small pieces which fit together in the end to finish the puzzle. Hence, project success. Agile methods require minimal planning and break away from endless documentation and extensive long-term planning (Leffingwell, 2007).

How does it work? Agile methodologies use processes called iterations. Iterations are short time frames called "timeboxes" and these iterations generally last from one to four weeks. During each iteration, a specific team works through a full software development life cycle involving planning, requirements analysis, design, coding, unit testing and acceptance testing when the working product is presented to the stakeholders. This improves the flexibility of the project as it allows for adaptive change. The stakeholders produce the necessary documentation as required and although any single iteration may not add enough functionality for market release, the goal is to have an available release with minimal bugs at the end of each iteration (Leffingwell, 2007).

Agile methods emphasize face-to-face communication rather than focusing on written documents. This is only possible when the team is all in one location. When the geographical location of team members becomes a problem, they communicate daily through videoconferencing, e-mail or other electronic resources (Abrahamson *et al.*, 2002).

Teams are generally small in order to make communication and collaboration easier. Larger development projects may be handled by several teams working towards the same goal. In such a case it is important to coordinate priorities across teams.

Each agile team contains a customer representative who is appointed by stakeholders to act on their behalf. Such a person makes a personal commitment to be available to development staff to answer mid-iteration problem questions. At the end of each iteration the stakeholders and customer representative review the progress of the project keeping in mind that their goal is to maximize the return on

investment while aligning with customer needs and business goals. Most agile methods use a daily routine during which team members meet face-to-face. This specifically includes the customer representative and any interested stakeholders to observe. Team members briefly discuss what they did the day before, what they are planning for the current day and what problems they foresee. This face-to-face communication prevents problems from being hidden (Abrahamson *et al.*, 2002).

With an agile method, the emphasis is on the working system or working part of that system. Together with face-to-face communication, agile methods produce less documentation than any other methods.

Agile software development methodologies provide organizations with the ability to rapidly evolve IS solutions (Highsmith, 1999; Sutherland & van den Heuvel, 2002).

2.3 History of Agile Methods

In this section the history of agile systems development methodologies is studied, starting with the Agile Manifesto, which is the birthplace of ASDMs.

2.3.1 The Agile Manifesto

The Manifesto for Agile Software Development (Beck *et al*., 2001) has been uncovering ways to improve software development as a whole and through this process they have come to value a few things:

- **Individuals and interactions** over processes and tools

  The agile movement focuses on the human aspect and the relationships between developers and stakeholders as opposed to institutionalized processes and development tools (Abrahamson *et al.*, 2002).

- **Working software** over comprehensive documentation

  The main objective of the project team is to produce tested working software on a continuous basis. Developers in an agile environment are encouraged to keep the code simple and basic to ensure the least amount of documentation. (Abrahamson *et al.*, 2002)

- **Customer collaboration** over contract negotiation

  The cooperation between the developers and the clients is preferred to strict contracts, keeping in mind the importance of thoroughly drafted contracts. The negotiation process is seen as a means of achieving and also maintaining a viable relationship (Abrahamson *et al.*, 2002).

- **Responding to change** over following a plan

  Everyone involved in the project need to be informed, competent and empowered to review possible adjustment needs that could emerge during the development life-cycle (Abrahamson *et al.*, 2002).

The Agile Manifesto regards the items in bold to be more valuable than the other items in plain text (Beck *et al*., 2001).

2.3.2 History of the Agile Manifesto

The Agile Manifesto came about when seventeen people came together at a ski resort in Utah searching for common ground. Representatives from Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development, Pragmatic Programming, and others interested in finding a feasible alternative to documentation driven, heavyweight software development processes were among those present (Beck *et al*., 2001).

Alistair Cockburn was initially concerned about the thoughts of the other participants and didn't think that this particular group of people could agree on anything substantive. After their meeting he expressed his satisfaction on the final phrasing of the Manifesto and felt that this was something substantive that they all agreed upon (Beck *et al*., 2001).

Bob Martin believes that Agile Methodologists are about "mushy" stuff and delivering quality products to their customers. This is done in an environment where people are valued as the most important part of the operation and are not seen as just an asset to the organization. The "mushy" stuff in essence refers to values and culture. The Agile movement is not against methodologies in general (Beck *et al*., 2001).

With this movement, the founders are trying to restore a balance in the corporate world. To illustrate this they state that they embrace documentation, but not hundreds of pages of never-maintained and rarely-used volumes. Planning is encouraged whilst recognizing the limits of planning in a turbulent environment (Beck *et al*., 2001).

The common goal of the Agile Alliance is to make people think about software development, methodologies and organizations in more agile ways. (Highsmith, 2001)

2.3.3 Principles of the Agile Manifesto

The Agile Manifesto follows the following principles (Beck *et al*., 2001):

- The highest priority is the satisfaction of the customer through the continuous delivery of valuable software; and then to
- Embrace changing requirements at any stage in the development process. This gives the customer a competitive advantage;
- Deliver working software at frequent intervals varying in weeks or months with preference to the shortest time scale;
- Business people and developers must join forces and work together each day throughout the duration of the project;
- Rely on motivated staff and individuals to get the job done giving them the environment and support they require;
- Implement face-to-face conversation to convey information to and within a development team;
- Progress is primarily measured through working software;
- All participating parties should be able to maintain a constant pace for an indefinite period of time keeping in mind that agile processes promote sustainable development;
- To enhance agility, pay attention to technical excellence and good design;
- Simplicity is essential;
- The best architectures, requirements, and designs are an immediate result of self-organizing teams.;

- Regular team meetings where the team can reflect on how they can be more effective are useful to adjust their behaviour accordingly.

2.4 Effectiveness of Agile Systems Development Methodology

In this section, the characteristics, suitability, advantages, and disadvantages of ASDMs are stated and discussed.

2.4.1 Agile Methodology Characteristics

Despite the diversity of the various agile development methodologies, they can further be characterized as being:

- Incremental (development of small pieces in a rapid cycle);
- Cooperative (a close relationship between the customer and the developer);
- Straightforward (easily modified, user-friendly when learning to use and sufficiently documented); and
- Adaptive (agile methodologies can easily adapt to last minute changes).

2.4.2 Factors with a negative impact on ASDMs

There are some factors that can have a negative impact on an agile system development project. These include the following (Schaaf, 2007):

- Development on a large scale (more than twenty developers);
- Geographical location of teams are distributed;
- Forcing a team to use agile processes in their development; and
- Using an agile approach in a mission critical system where failure is not an option.

2.5 Advantages of Agile Methodologies

These are some advantages in using ASDMs:

- There is a relatively shorter time period between conception and delivery dates (Parthasarathy & Rangarajan, 2008);

- Every step or iteration need not be perfect since work and enhancements are done throughout the project life-cycle (Parthasarathy & Rangarajan, 2008);
- There is a relatively shorter time needed to return on investment (ROI) (Parthasarathy & Rangarajan, 2008);
- Defects can be corrected and enhancements can be made even after release, until the project concludes completely. This can be done at a low incremental cost (Parthasarathy & Rangarajan, 2008);
- Changes to the business process can be incorporated into an upcoming software release at low costs (Parthasarathy & Rangarajan, 2008);
- Since all project resources are fully engaged for the entire project life-cycle, the use of resources is optimized (Parthasarathy & Rangarajan, 2008);
- An efficient and fast-paced method (Parthasarathy & Rangarajan, 2008);
- ASDMs are suitable to create reliable and safety-critical systems (Lindvall, 2002);
- ASDMs necessitate less formal training compared to traditional methods (Lindvall, 2002); and
- Warning signals are evident in Agile projects and can easily be spotted due to frequent communication with stakeholders (Lindvall, 2002).

2.6 Disadvantages of Agile Methodologies

These are some disadvantages to using ASDMs:

- The number of iterations to achieve an error free system may be more than that of the traditional method (Parthasarathy & Rangarajan, 2008);
- An agile method can only be successful in a trustworthy and cooperative environment. Anything less can cause the entire project to stall or fail completely (Parthasarathy & Rangarajan, 2008);
- Agile methods require resources with a lot of maturity, knowledge and experience. Lack thereof can cause a simple project to fail if implemented in an agile scenario (Parthasarathy & Rangarajan, 2008);
- Agile methods require a flexible work schedule and open lines of communication between all the project resources and organizational resources to ensure project success (Parthasarathy & Rangarajan, 2008);

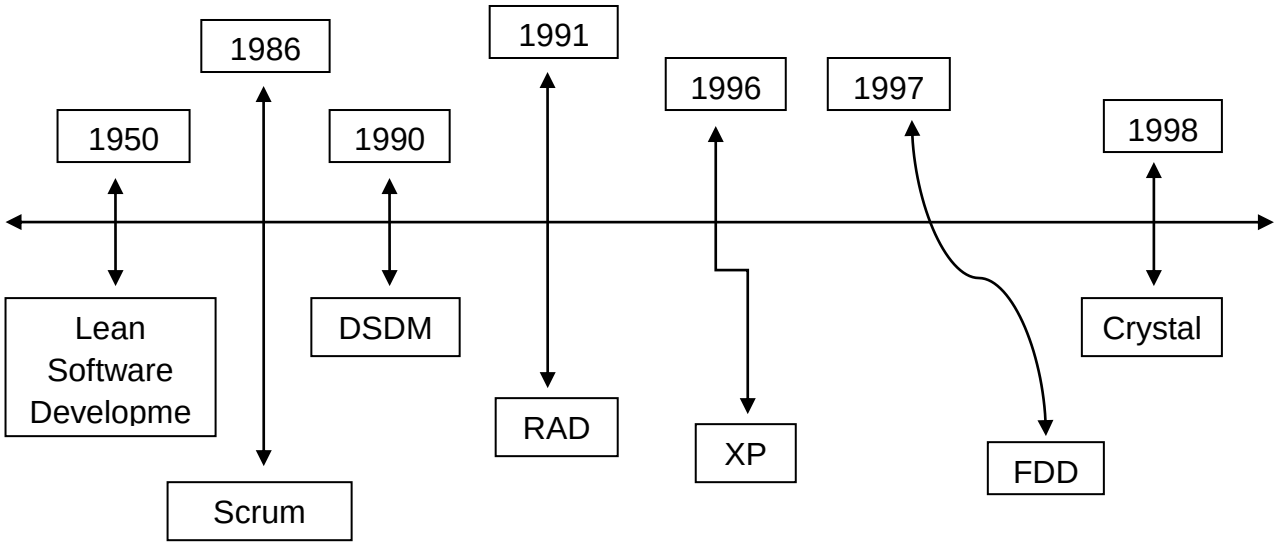
- Agile methods are not the remedy for all failed projects and should never be treated as such. Each project in an organization may warrant a different treatment, some may benefit from the agile model, others would be more successful using a traditional method (Parthasarathy & Rangarajan, 2008); and
- Team size is an issue in the sense that more people make communication harder. There is adequate experience regarding small teams, but much less regarding larger teams (Lindvall, 2002).

Now that a general understanding of where ASDMs come from has been established, we will look at each individual ASDM in more depth based on a timeline of when each ASDM has been developed.

2.7 Timeline of Agile Methodologies

The following timeline was developed by integrating the different ASDMs according to their history and year in which their development started.

Figure 2.7.1 – Timeline of ASDMs



2.7.1 Lean Software Development (1950)

This methodology has its roots in Lean Manufacturing which was a new concept introduced by the Japanese in the 1950s. The Japanese business, Toyota® Motor Company, changed their way of testing at the end of a production cycle, to testing at

each station which forms part of the cycle. This major change resulted in the number of components assembled with the same defect occurring less often. The implication: less rework and lower costs (Windholtz, 2005; Chan & Thong, 2009; Dyba & Dingsoyr, 2008).

2.7.2 Scrum (1986)

The term "Scrum" was first mentioned in an article by Takeuchi and Nonaka (1986) where an adaptive, quick, self-organizing product development process, which originated from Japan, is disclosed. The term "scrum" originates from a strategy in the sport of rugby which denotes "getting an out-of-play ball back into the game" by applying teamwork (Schwaber & Beedle, 2002; Chan & Thong, 2009; Abrahamsson *et al.*, 2002).

2.7.3 Dynamic Systems Development (1990s)

The Dynamic Systems Development Methodology originated in the United Kingdom around the 1990s. It was derived from rapid application development practices and it is also an extension of these practices. DSDM is seen in the light of a framework, more than a method. The DSDM has a reputation of having the best training and documentation of all the agile methods available in practice (Cohen *et al.,* 2004; Ketter *et al.,* 2009; Dyba & Dingsoyr, 2008).

2.7.4 Rapid Application Development (1991)

In the mid-80s, Boehm and Gilb's work led to the formulation of a new methodology called Rapid Iterative Production Prototyping (RIPP). In 1991, James Martin extended RIPP to create what is known today as Rapid Application Development (RAD). (CASEMaker, 2000)

2.7.5 Extreme Programming (1996)

Extreme Programming was derived from the waterfall method where a client would specify his requirements at the beginning of the project after which it would be developed, not taking into account that the client's needs might change during the development (Beck *et al.,* 2001). The discovery was made that making changes in

these long life cycles led to a lot of extra costs. To solve this problem, the life cycles needed to be shortened. Before XP, this newly developed method was called iterative development. Many people contributed to developing XP from then on over the following years (Beck, 1999; Chan & Thong, 2009; Ketter *et al.,* 2009).

2.7.6 Feature-Driven Development (1997)

This methodology was developed for the first time by Jeff De Luca in 1997. The reason for developing FDD was to meet the specific needs of a bank in Singapore. The software development team comprised 50 people and had to accomplish the project in fifteen months. De Luca's model consisted of five processes:

- Development of an overall model;
- Building the feature list;
- Planning by feature;
- Designing by feature; and
- Building by feature.

FDD is most suitable for new projects starting afresh, projects upgrading and enhancing existing source code, and projects that aim to develop the next version of an existing application. The adoption by an organization can best be done by a gradual process as the project progresses (Abrahamson *et al.*, 2002; Dyba & Dingsoyr, 2008).

2.7.7 Crystal (1998)

This is a very new methodology with very little literature available. It consists of a family of methods, each with the same underlying core values and principles. (Chan & Thong, 2009; Abrahamsson *et al.*, 2002)

2.8 Different Agile Methodologies

The following section introduces different ASDMs and explains each one in more detail.

2.8.1 Lean Software Development

2.8.1.1 Description

The Lean Software Development Methodology is built on seven basic principles that will be discussed in the following sections.

- **Eliminate Waste**

Taiichi Ohno (1988) called the Toyota Production System a management system for "the absolute elimination of waste." Kumar (2005) added that the first step in lean thinking is to understand the concept of "value" and what activities and resources are necessary to create it. Since no worker would like to consider what they do as a waste, the task of determining what “value” is and what adds value needs to be done at a high level by upper management. (Kumar, 2005)

The seven types of manufacturing wastes, according to Poppendieck and Poppendieck (2006) are Overproduction, Inventory, Extra processing steps, Motion, Defects, Waiting, and Transportation.

The seven wastes of software development (based on the Manufacturing Wastes) are Overproduction (Extra features), Inventory (Requirements), Extra processing steps (Extra Steps), Motion (Finding Information), Defects (Bugs not caught by tests), Waiting (Waiting for decisions, including customers), and Transportation (Handoffs) (Poppendieck & Poppendieck, 2006).

The project teams should try to avoid these wastes at all costs as they are commonly produced during the software development process.

- **Build Quality in**

The goal here is to build quality into the software code from the start, not to test it in later as stated by Poppendieck and Poppendieck (2006). You don’t focus on minimizing defects in a system, you avoid the creation of defects in the first place. It takes a highly disciplined organization and team to achieve that. Defects need to be caught as soon as possible and production can be totally halted if need be to achieve this.

Poppendieck and Poppendieck (2006) made the statement that shorter production lines are needed that are defect free and if a line has a defect, the entire line should be dropped and reworked it until it is 100% defect free. Today there are tools available to keep most defects out of your code as it is being developed. Developers who are using test-driven-development write unit tests and acceptance tests before they write the necessary code. They integrate both code and tests into the system as often as possible and run these tests to see that no defects have been introduced to the system. If the new code does not pass the tests they do not add new code until the problem is solved or the code is removed.

- **Create knowledge**

Software development is a continuous learning process. The best way to improve a software development environment is to encourage learning. The learning process is sped up by making use of short iteration cycles. Each cycle should be coupled with refactoring and integration testing. Feedback can be improved by short feedback sessions with customers when determining the current phase of development and future improvements. (Poppendieck & Poppendieck, 2006).

- **Defer commitment**

This principle can be seen as a Just-in-Time principle. Most of the researchers in this report point to this fact. They state that important decisions should be made only at the last responsible moment. By dragging out the time it takes to make the decision you create more time to gather the necessary information required to make the best decision and choose the best option. Not all decisions should be deferred, but all decisions should try to be made reversible so that we can easily take a few steps back if the right decision was not made at the time (Poppendieck & Poppendieck, 2006).

- **Deliver fast**

The sooner the product is delivered (without any defects) the sooner feedback can be gathered from the customer so that any necessary improvements can be made or new requirements can be added to meet the customers' needs. Shorter

iterations improve learning and communication. There is more than one technique that can be applied to achieve this goal, but it will not be discussed in this report as it is a subject matter of its own.

- **Respect people**

Some sources call this phase by another name (empowering the people), but it is essentially the same principle. This principle is very simple to implement if you work in a team where every vote counts and no one person is regarded more important than the next. It's not to say that Mr. X's proposal is weaker than Mr. Y's just because Mr. Y works longer for the company than Mr. X. Everyone must get a fair chance to be heard, otherwise you might lose out on a brilliant plan or strategy.

- **Optimize the whole**

A lean organization optimizes the whole value stream, from the time it receives a requirement from a customer until the software is deployed and the requirement met. If an organization is to focus on optimizing less than the entire value stream, we can just about guarantee that the overall value stream will suffer (Poppendieck & Poppendieck, 2006). Drucker (1999) noted that a company that maintains a single management system throughout the value stream will see a cost advantage of 25% to 30% over their competitors. There is a sizable amount of money to be saved from structuring contracts, outsourcing agreements, and cross-functional interactions with correct incentives. This in turn makes sure everyone is focused on optimizing the whole.

2.8.1.2 Strengths and weaknesses

The greatest strength of this SDM is that lesser defects are produced in the code while costs are cut to boost total profit. The drawback to this approach is that the whole production line should be rebuilt and set up according to the principles to achieve the goals set by this SDM. But if this drawback is met head on, a permanent advantage will be gained as the transition needs to be done only once and then just maintained (Poppendieck & Poppendieck, 2006).

2.8.2 Scrum

2.8.2.1 Description

Scrum is a simple management framework for incremental product development that makes use of robust teams of about seven people each. Scrum teams use fixed length iterations, called Sprints, approximately 14 – 30 days long. Each Sprint contains the traditional phases as used in the Waterfall methodology. They attempt to build a potentially shippable, tested product increment at each iteration phase. Sometimes only three Sprints are required to reach the end-goal, but it can be either eight or more (James, 2009).

Let's take a look at what is involved when making use of the Scrum Methodology. We will start off by pointing to the flow of the whole process.

2.8.2.2 Scrum Flow

Scrum's incremental, iterative approach (see *Figure 2.7.2.2*) gives the ability to develop a working, useable product as soon as possible and allows for quicker user feedback than as with the traditional Waterfall method. (James, 2009)

Figure 2.8.2.2 – Scrum Flow (James, 2009)

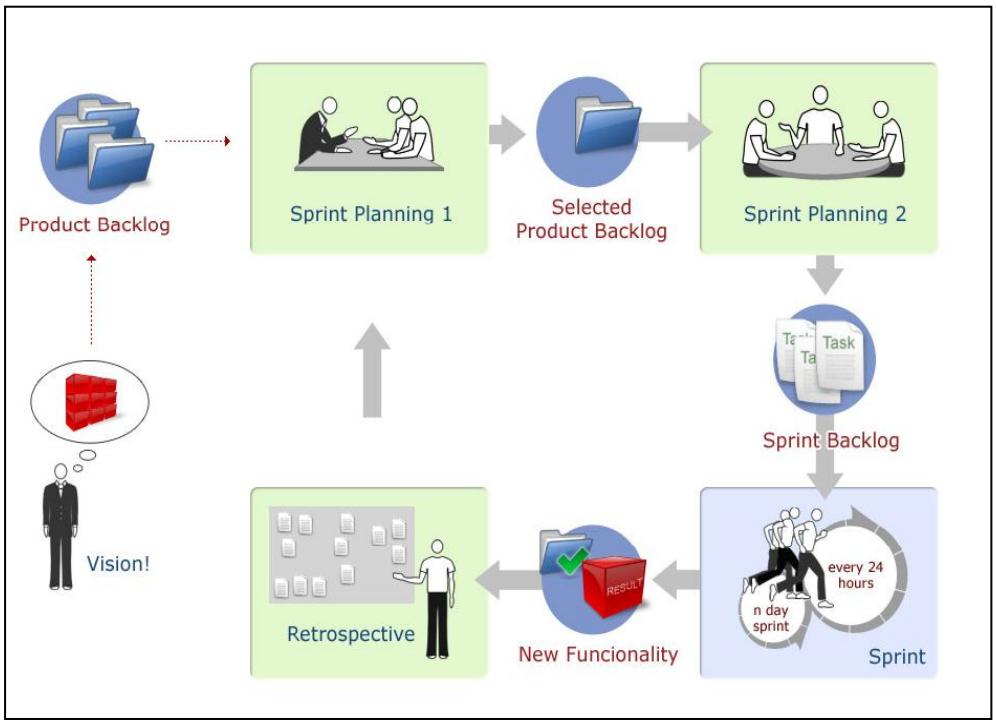


Figure 2.8.2.2 illustrates each step to be taken during iteration. At the beginning of an iteration cycle we first go to the Product Backlog, which is a description of all the requirements and features that has to be met in the end-product. The Product Backlog is a living list which means that it is definitely subject to change in the future. Thus the Scrum Team need to go over the list at the beginning of each iteration phase to secure their commitment for the next iteration. From this list the new features are generated which need to be developed during the next iteration. (Abrahamson *et al.,* 2002). The new features are then put onto the Sprint Backlog, which is a stable list that only changes at the beginning of an iteration phase. It is essentially a list of Product Backlog items that has to be implemented at this iteration (Abrahamson *et al.,* 2002).

This process is repeated until all requirements are met, implying that the Product Backlog list is empty, and no problems are encountered after final testing is completed.

2.8.2.3 Scrum roles

There are six roles in Scrum, each with different tasks and purposes. The six roles are:

- Scrum Master - is responsible for ensuring that the project commences according to the Scrum principles and that the deadlines are met. The Scrum Master interacts with the Scrum Team, Customer and Management during the life of the project.
- Product Owner - is responsible for the product Backlog list. He/she has full power over the Product Backlog and turns the issues in the Backlog into features to be developed.
- Scrum Team - is the project team that decides on the necessary actions and organize themselves to reach the goals at the end of each sprint.
- Customer - participates in the tasks related to the Product Backlog items that need to be enhanced.
- Management - is in charge of final decision making concerning the project. They also set the goals, specify the requirements and help to select the Product Owner.

- User - the person or people that will be making use of the new product after development is done (Schwaber & Beedle, 2002).

2.8.2.4 Scrum phases

Scrum consists of three phases which include the following: pre-game, development and post-game (Abrahamson *et al.,* 2002). Schwaber and Beedle (2002) gave this description of each phase:

- Pre-game (consisting of two phases)
    - Planning - A description of the system to be developed is included and the Product Backlog list is created. The list of requirements is prioritized and the amount of work needed to implement them is estimated. This phase also includes a description of the project team, tools, and other resources, risk assessment and controlling issues, training needs and verification management approved.
    - Architecture level design - This phase contains the high-level design and the architectural planning based on the Product Backlog items. A design review meeting is called at this phase to cover the proposals for the implementation and decisions are based on this review.
- Development - This phase is the agile part of the whole process where the Sprints take place. The different environmental and technical variables identified in Scrum are managed by means of various Scrum techniques during the Sprints. Take note that there may be more than one team involved in building the next increment.
- Post-game - This phase is started only when the Product Backlog list is empty and no new requirements can be added to the list. The system is now ready to be released. Planning for the release should have been done during the post-game phase.

2.8.2.5 Strengths and weaknesses

Schwaber and Beedle (2002) noted that Scrum can be adapted to manage any engineering practices that are currently being used in a business or organization which is a very good plus point for Scrum, but they also stated that when implemented, the Scrum will in fact change the job description of the people involved, and this can cause a lot of confusion as to who must do what at the beginning. Another great strength of the Scrum is that it works extremely well on new and existing projects.

2.8.3 Dynamic Systems Development

2.8.3.1 Description

This model was developed in the United Kingdom around the 1990s. It was derived from rapid application development practices and it is also an extension of these practices. DSDM is seen in the light of a framework, more than a method. The DSDM has a reputation of having the best training and documentation of all the agile methods available in practice. The DSDM lifecycle consists of the following major phases:

- Pre-project;
- Feasibility study;
- Business study;
- Functional model iteration;
- Design-and-build iteration;
- Implementation; and
- Post-project (Cohen *et al.,* 2004).

The DSDM addresses common issues that agile development methodologies are commonly faced with. It firstly states explicitly the difference between DSDM and traditional methods with respect to flexible requirements. According to DSDM, traditional views imply that functionality remains relatively fixed, although time and resources are allowed to vary. With DSDM, this point of view is reversed which allows for functionality to vary over the life cycle of the project as new things are discovered throughout the development process. While functionality is allowed to vary, time boxes are applied to maintain control over the project (Cockburn & Highsmith, 2001).

DSDM also addresses the common issue of documentation, or the lack thereof, which is a constant criticism of agile methodologies. With DSDM, prototypes are used to collect information rather than lengthy documents. The DSDM recommends only 15 work products from the development phases of which several are prototypes. Unlike rigid methodologies, DSDM does not provide endless documentation for its 15 work products. Instead, DSDM work product guidelines

present a brief description, listing of the purposes, and a few quality criteria questions for every work product (Cockburn & Highsmith, 2001).

The DSDM encourages the project manager to focus on sustaining progress, getting agreement on requirement priorities, managing customer relationships, and supporting the team culture and motivation (Cockburn & Highsmith, 2001).

2.8.3.2 Uses and Principles of DSDM

One of the basic fundamentals of DSDM is that nothing is done perfectly the first time, although 80 percent of the solution can be produced in 20 percent of the time it would take to produce the end product. According to Pareto's Law, the final 20 percent of the project absorbs the majority of the problem-solving effort. Along with this, there are some other underlying principles which dictate how a project is taken. There are two versions, one which contains thirteen principles (version 1) and the other which contains nine principles (version 2). Version 2 will now be disclosed:

- Active user involvement in the system development process;
- DSDM team members need to be empowered to make decisions;
- The focus in the project is on the frequency of product delivery rather than on activities;
- Every deliverable is fit for its business purpose;
- Iterative and incremental development is a very powerful method to build strong systems;
- Changes are always reversible and no requirements are frozen;
- Base-lining of high-level system requirements;
- Testing is integrated throughout the development cycle,
- A collaborative and co-operative approach is essential in the development process (Howard, 1997).

2.8.3.3 Strengths and weaknesses

The DSDM Consortium has developed a holistic approach to system development providing a philosophy and a lifecycle supported by the necessary controls to

ensure project success. There are rumours that DSDM will become a standard for Rapid Application Developments. This method is based on guidelines and principles, not recipes or “blueprints” that have to be followed to the letter. Interest in this agile method is continuously growing internationally and will continue to prosper in the future (Cohen *et al.,* 2004).

2.8.4 Rapid Application Development

2.8.4.1 Description

Rapid Application Development (RAD) is a software development methodology that thrives on minimizing planning and maximizing prototyping. In his book that coined the term RAD, James Martin wrote, "Rapid Application Development (RAD) is a development lifecycle designed to give much faster development and higher-quality results than those achieved with the traditional lifecycle. It is designed to take the maximum advantage of powerful development software that has evolved recently" (CASEMaker, 2000).

Other experts such as Professor Clifford Kettemborough defined RAD as "a method of developing computer systems which combines Computer-Assisted Software Engineering (CASE) tools and techniques, user-driven prototyping, and strict project delivery time limits into a potent, tested, reliable formula of the highest quality and productivity. RAD drastically raises the quality of finished systems while reducing the time it takes to build them” (CASEMaker, 2000).

The name "Rapid Application Development" precisely describes what the methodology is, a process through which system development is accelerated. RAD makes it possible to produce quality products faster, saving valuable resources. RAD consists of four critical aspects: people, tools, methodology, and management. The absence of any of these components will slow down the development process (CASEMaker, 2000).

- Methodology

The RAD development methodology provides a means for developing systems at a quicker rate and at the same time minimizing cost. The basic fundamentals are:

- Use of workshops to gather user requirements;
- Use of evolutionary prototyping;
- Use of best techniques and specify the sequence of tasks to make the technique more effective;
- Use of time-boxes to speed up development process;
- Providing guidelines for success and highlight pitfalls to avoid.

- People

Individuals with the right technical know-how are of paramount importance to the success of RAD. Best available tools are required to develop applications fast. However, they do not guarantee success. Teamwork is more important than just individual brilliance. RAD allows each person involved to play several different roles. During the requirements planning and user design stages, key end users should be available to take part in workshops. As the system is being constructed, the construction team, accomplish detailed design and code generation, they should be ready to move fast. At the end of the development cycle, the cutover team, which handles training and cutover, should be ready to move quickly.

- Management

Management must be totally committed to RAD in order to manage the change in organizational culture. Managers ought to motivate users and IT staff, select and manage SWAT teams, and demonstrate through the use of performance measurements the goals of RAD (speed, quality, and productivity).

Management should be attentive to human motivation. Managers should try and change the mind-sets of the people especially those who are resistant to change. RAD is at the other end of the pendulum of development methods (in comparison to where conventional development methods are) and management has to start on a small scale if they are to introduce new rapid development techniques. Emphasis

must also be put on the importance of thorough quality training in the use of tools. Good training with tools that are exciting to use can have a great impact on the attitude of IT professionals, as well as ensure the uninterrupted success of the RAD project (CASEMaker, 2000).

- Tools

RAD methodology makes use of computerized power tools and human techniques to achieve the goals of high-speed and high quality. The power tools utilized in RAD are CASE tools.

A basic rule for RAD tools is that diagrams are frequently used whenever possible as an aid to clear thinking. Diagrams are used to represent planning information, overview of systems, data models, process models, detailed designs, and program structures. RAD tools must generate executable code.

2.8.4.2 RAD critics

Some people may question whether RAD really does live up to its expectations. Everyone seems to have adopted the RAD approach at some point in their systems development careers but it doesn't always meet expectations or deliver promised results (Reilly & Camel, 1995).

The RAD methodology bypasses the design phase which implies that there is little software reuse and consequently producing a brittle and hard to maintain system.

2.8.4.3 Strengths and weaknesses

One advantage of using RAD is that it promotes a strong collaborative atmosphere and dynamic gathering of requirements. For fast development, the business owners actively participate in the prototyping phase, by writing test cases and performing unit testing. Thus, the use of CASE tools aid to increase the speed of development and increase quality of the product through the involvement of the user in the analysis and design stages.

For every action there is an equal but opposite reaction as Newton stated in his law. RAD has reduced scalability due to the fact that developed applications start as prototypes and evolve into a finished application. RAD also has reduced features due to Time-boxing, where features are pushed to later versions in order to finish a release in a short amount of time (CASEMaker, 2000).

2.8.5 Extreme Programming

2.8.5.1 Programming

Extreme Programming (XP) focuses on delivering immediate benefits to the end user/customer. This can be done due to the very short development life cycles and the fact that at the end of each cycle, a usable piece of software is delivered (Abrahamsson & Koskela, 2004).

XP can be described as a software development discipline because it has a set of things you have to do to be doing XP (Beck, 1999).

In the following section we will look at how XP works, as well as what the values and principles are.

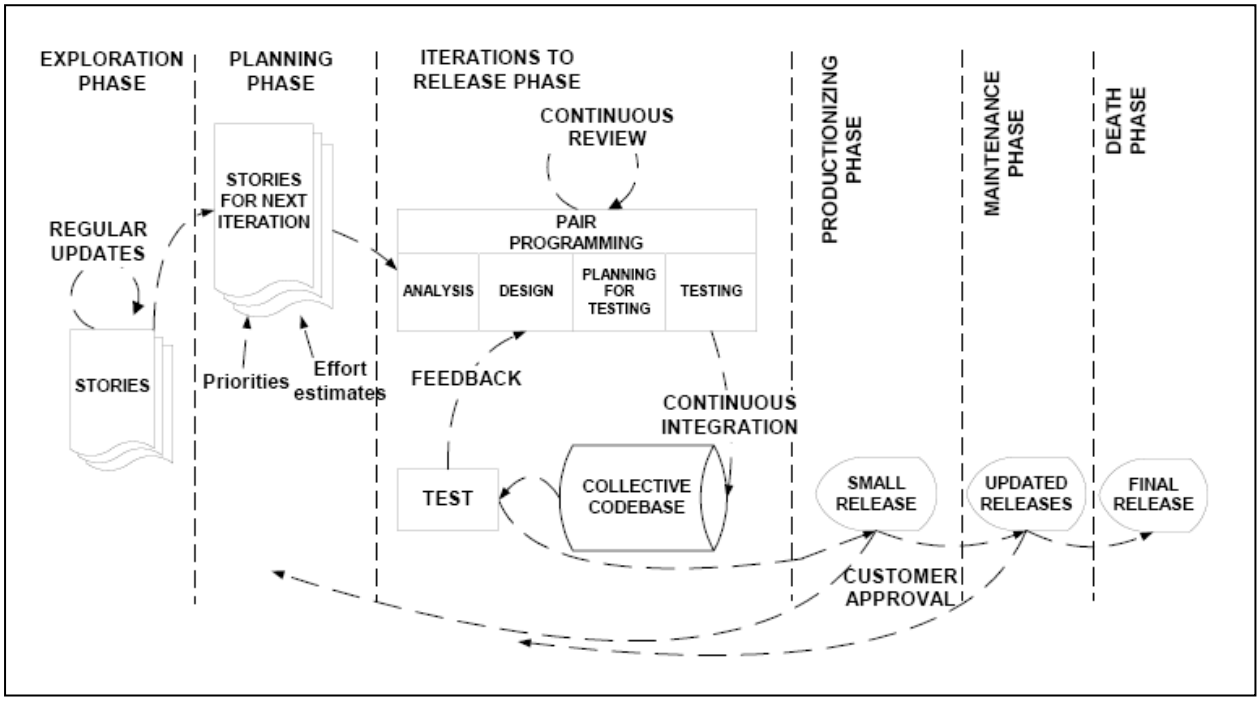
- XP Lifecycle

  The XP lifecycle consists of 5 phases and they are illustrated in *figure 2.8.5.1* below. (Abrahamson *et al.*, 2002)

  These phases are:

  - Exploration;
  - Planning;
  - Iteration to release;
  - Productionizing;
  - Maintenance and death.

Figure 2.8.5.1 – XP Lifecycle (Abrahamson *et al*., 2002)



2.8.5.2 Values and principles of XP

According to Beck (1999) XP is based on four values namely:

- Communication

This is one of the most important values because without effective communication in a project, someone can forget to pass along an important message or not get all the requirements from the customer. This could have catastrophic consequences for the project which is why XP uses practices that cannot be implemented without the use of good communication.

- Simplicity

The project leader needs to ask the question: "What is the simplest thing that could possibly work?" The reason for this question is: the simpler the system you're

working on, the less communication there has to be which can lead to more effective communication. It could also result in the project needing fewer programmers, a situation which also benefits communication because there are fewer channels through which information must travel.

- Feedback

Feedback can be done in a lot of different time-scales, like days, weeks, or months. The more often the feedback, the smoother the communication channels will become. Testing is much easier in simpler systems which enables feedback to occur much faster.

- Courage

In some situations you might need to just delete all the code you did during that day and start all over again. This time you might end up with a much cleaner solution that is less prone to bugs. Another example of courage would be when someone on the team comes up with a radical idea that will slash the complexity of the system in half and then actually implementing that idea.

According to Beck (1999) the principles involved in XP are:

- Rapid feedback

In XP, we try to get feedback as quickly as possible, interpret it and put whatever we learned from this back into the system as fast as we can.

- Assumed simplicity

98% of the problems in the system can be solved with ridiculous simplicity. When this happens you save time and then the extra time can be spent solving the other 2% that is extremely complex.

- Incremental change

It is hard to make big changes to a system all at once, but it is easy to make smaller changes over a period of time.

- Quality work

People enjoy doing good work and this will lead to project success. If you are not happy with your work then you won't do it well and the project will fail.

2.8.5.3 Strengths and weaknesses

One of the major strengths of XP is that it can change and adapt to changing business needs. If a change occurs this will not increase the cost by a lot like it does in conventional programming because here we focus on solving small problems at a time.

In XP, some of its strengths are also its weaknesses because:

- The small team structure means that it will not be possible to do projects that require more than 20 programmers because 20 is the recommended maximum. The fact that they break every problem down into smaller problems that can be solved in a couple of weeks means that huge problems that cannot be broken down, cannot be solved using XP. Furthermore, the fact that XP relies on people working closely together means that all the people has to be in the same building. This means you won't be able to use virtual teams with XP and this could increase the cost of the project due to relocating staff (Beck, 1999).

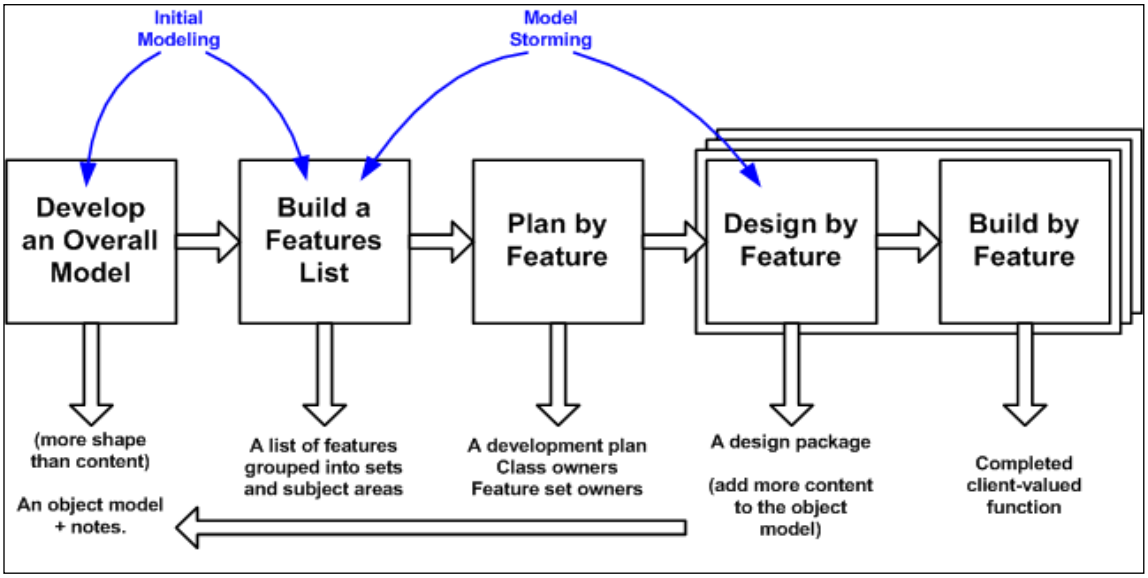
2.8.6 Feature-Driven Development

2.8.6.1 Description

Feature-Driven Development (FDD) software development methodology is a model-driven, short-iteration process comprising five activities. These various activities aid in accurately monitoring the state of the software development project as well as defining targets that mark the progress made on each feature.

Figure 2.8.6.1 below represents the meta-process model for these activities. The first three sequential activities establish an overall model shape. The final activities are iterations for each feature.

Figure 2.8.6.1 – FDD Process Model (Palmer & Felsing, 2002)



- **Develop an overall model**

Before the development of the overall model commences, the domain experts (see roles and responsibilities) should already be aware of the scope, context and requirements regarding the new system to be built (Palmer & Felsing, 2009).

FDD does not explicitly point out how requirements should be gathered or managed. This remains the responsibility of the domain experts. These domain experts then

present other team members as well as the chief architect with a "walkthrough" where they are informed about the high-level description of the system (Abrahamson *et al.,* 2002).

- **Build a features list**

The above-mentioned walkthroughs and existing requirement documentation lay the foundation on which the features list for the system being developed can be built. Included in this list are all the functions as required by the client. The various features are ranked and consist of so-called major feature sets. These major feature sets are then further divided into more feature sets. The features list is reviewed by users and sponsors alike to ensure the validity and completeness thereof (Abrahamson *et al.,* 2002).

- **Plan by feature**

This includes creating a high-level plan, where the feature sets are sequenced according to their importance and assigned to Chief Programmers. Schedules and milestones can be set for feature sets (Abrahamson *et al.,* 2002).

- **Design by feature and build by feature**

A select group of features is chosen from the various feature sets to form feature teams chosen by class owners. This step involves many iterative procedures, during which the chosen features are developed. Any single iteration should last a few days, but never more than two weeks. After the completion of a successful iteration, the completed features are added to the main build while new features are selected from the feature set (Abrahamson *et al.,* 2002).

When applying FDD to a project, the main focus is on the design and building phases. The FDD methodology does not cover the entire scope of systems development, although its designed to be incorporated with other activities in the development process (Palmer & Felsing, 2002).

FDD monitors the quality aspects throughout the development process and includes frequent, tangible deliveries. This is done in conjunction with accurate monitoring of the project's progress (Abrahamson *et al.,* 2002).

2.8.6.2 Roles and responsibilities

Roles are classified under three categories: key roles, supporting roles and additional roles (Palmer & Felsing, 2002). There are six key roles consisting of a project manager, chief architect, development manager, chief programmer, class owner and domain experts. The five supporting roles are release manager, language lawyer, build engineer, toolsmith, and system administrator. There are three more roles which are part of any project: the testers, deployers, and technical writers.

Note that any particular team member can play multiple roles, and any single role can also be shared among many people (Palmer & Felsing, 2002).

2.8.6.3 Strengths and weaknesses

FDD, just like all methodologies, is designed to enable teams to deliver results faster without compromising quality. FDD is a highly iterative process and results oriented. It focuses on people/ users rather than large documentation. FDD does away with the traditional approach of separating the domain and business analysts from designers and implementers. Instead, the analysts are taken out of their abstractions and put in the same room as implementers and users.

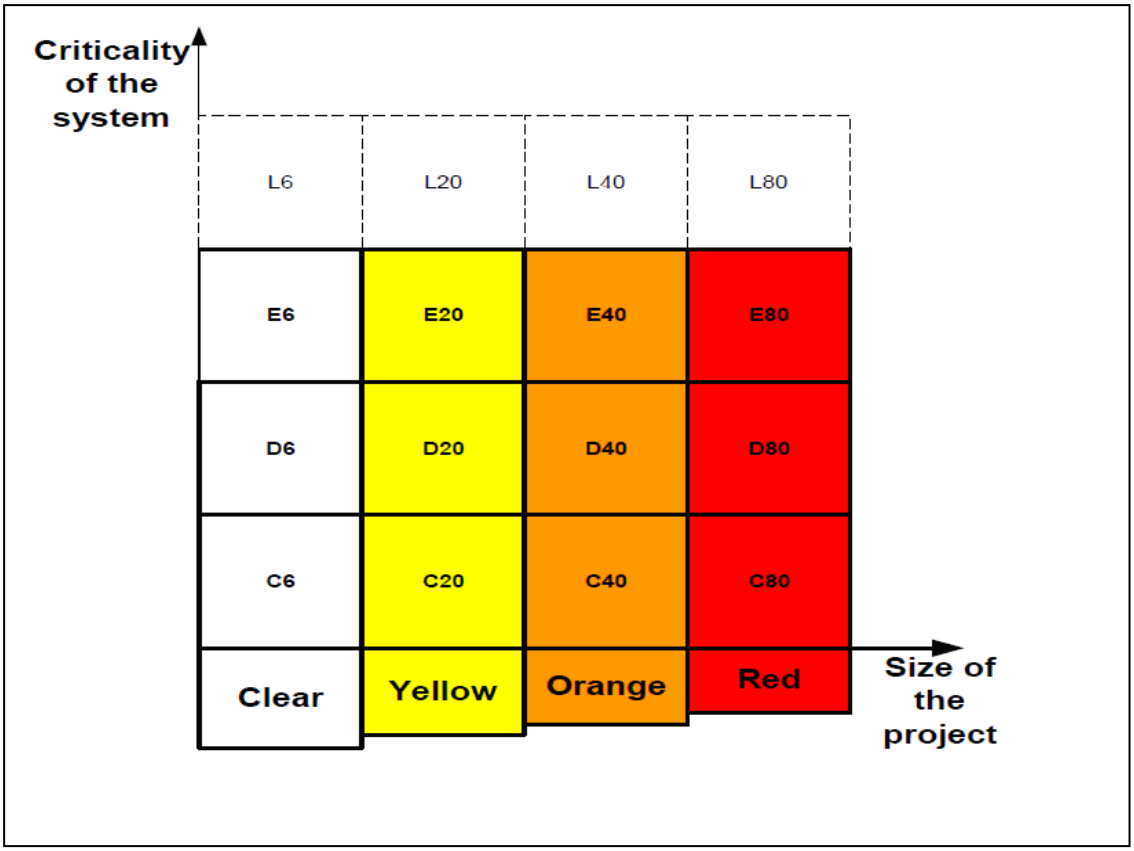
From the research conducted, very few weaknesses centring on FDD were discovered. However, there are some limitations to the methodology that can be attributed to relativity of its use. FDD is designed to scale to much larger team sizes. The limiting factor is the number of available Chief Programmers. Chief Programmer teams have been proven in practice to scale well to much larger project teams (Abrahamson *et al.,* 2002).

2.8.7 Crystal

2.8.7.1 Description

The Crystal family of methodologies includes a number of different methods for the selection of the methodology best suited for any individual project. Aside from this, the Crystal approach also includes principles for tailoring the methodologies to fit the varying circumstances that different projects might present (Abrahamson *et al.,* 2002).

Figure 2.8.7.1 – Dimension of Crystal Methodologies (Abrahamson *et al*., 2002)



Each member of the Crystal family is marked with a colour that indicates the "heaviness" of the particular methodology. The darker the colour the heavier the methodology. The Crystal methodology suggests choosing the appropriate colour of methodology for a project according to the size and criticality of that particular

project (refer to *figure 2.8.7.1*). Larger projects are more likely to require better coordination and heavier methodologies than smaller ones. The higher the criticality is of the system being developed, the more strictness is needed.

The character symbols in *Figure 2.8.7.1* indicate a potential loss caused by a system failure (i.e. the criticality level): **C**omfort (C), **D**iscretionary money (D), **E**ssential money (E) and **L**ife (L) (Cockburn, 2002a). For example, a system crash due to defects (indicated by criticality level C) causes a loss of comfort to the user whereas defects in a life critical system could literally cause loss of life. The dimensions of criticality and size are represented by a symbol in *Figure 2.8.7.1*. For example, D6 denotes a project with a maximum number of six people, delivering a system where the maximum criticality is discretionary money.

Certain rules, features and values are common to each Crystal method. Firstly, the projects always use incremental development cycles where the maximum increment length is four months, but preferably between one and three months (Cockburn, 2002).

The emphasis lies on communication and the cooperation of people. Crystal methodologies do not limit any development practices, tools or work products, while also allowing the adoption of for example XP or Scrum practices (Cockburn, 2002). The Crystal approach comprises objectives for reducing interim work products and shaping conventions within a methodology for individual projects and developing them as the project evolves.

There are currently three main Crystal methodologies which comprise: Crystal Clear, Crystal Orange and Crystal Orange Web (Cockburn, 2002). The first two of these have been used or experimented with in practice.

2.8.7.2 Crystal policies and standards

These are the practices that need to be applied throughout the development process. Both the Crystal Clear and Crystal Orange methodologies suggest the following policy standards (Cockburn, 2002).

- Incremental delivery on a regular basis;

- Progress tracking by using milestones which are based on software deliveries and major decisions rather than endless written documents
- Direct user involvement;
- Automated regression testing of functionality;
- Two user viewings per release;
- Workshops for product- and methodology tuning at the beginning and in the middle of each increment.

The only difference between the policy standards of these two methodologies is that Crystal Clear suggests incremental delivery within a two to three month period whereas with the Crystal Orange methodology the increments can be extended to a maximum of four months. The policy standards of both these Crystal methodologies are obligatory, but can however be replaced with equivalent practices of other methodologies such as XP or scrum (Cockburn, 2002).

2.8.7.3 Strengths and weaknesses

Crystal Clear and Crystal Orange do not define any specific practices or techniques to be used by the project members in their various development tasks. The adoption of practices from other methodologies such as XP or Scrum is allowed in Crystal to replace some of its own practices such as reflection workshops for instance. While only two of the four Crystal methodologies exist, Alistair Cockburn continues to develop his family of Crystal methodologies to cover different types of projects and problem domains. Beyond this, no research can be identified (Cockburn, 2002).

2.9 Comparison of Agile Systems Development Methodologies

In this section, ASDMs will be compared in various ways to other models and methodologies.

2.9.1 Comparison with other methodologies

Agile methodologies are often seen as being “undisciplined” or “unplanned”. A more accurate distinction is that all methods exist in a continuum from “adaptive” to “predictive” where agile methods lie on the adaptive side of the equation. Agile

methods allow for quick adaptation to changing realities. When the project needs suddenly change, the project team needs to change as well. An adaptive team would focus on short term goals and development issues in the short term. Predictive methods however plan the future in detail. Predictive teams know exactly what tasks are planned for the entire life cycle of the development process. Agile development techniques have much in common with Rapid Application Development. Agile development is different to other models in the way that time periods are measured in weeks rather than months and work is performed in a highly collaborative manner (Boehm & Turner, 2004).

***Table* 2.9.1 – Comparison between Agile SDMs and Traditional SDMs (Vinekar *et al.*, 2006)**

	Agile Systems Development Methodologies	**Traditional Systems Development Methodologies**
Management and organizational	Leadership and collaboration	Command and control
	Cooperative	Autonomous
	Flexible	Disciplined
	Manager as facilitator	Manager as planner
	Tacit knowledge	Explicit knowledge
	Team reward systems	Individual reward systems
People	Collaborative work	Individual work
	Multi-disciplinary skills	Specialized skills
	Pluralist decision-making	Managerial decision making
	High customer involvement	Low customer involvement
	Small teams	Large teams
Process	People centric	Process centric
	Speculative	Standardized
	Assess progress	Measure progress
	Evolutionary development	Life-cycle development
	Write tests prior to code	Write code prior to tests
	Individual approach to projects	Unified approach to projects
	Adaptable	Pre-planned
	Iterative	Linear
	Short durations	Long durations
Technology	Object oriented	Structured or object oriented
	Tools for iteration	Standardized tools

Table 2.9.1 above compares agile systems development methodologies and traditional systems development methodologies on 4 different levels namely management and organizational, people, process, and technology. These elements are described as they are applied within the different methodologies and compared to each other on the same level. It is eminently clear that ASDMs are much more flexible and people centric than traditional systems development methodologies.

***Table* 2.9.2 – Comparison between the different ASDMs (Adapted from Abrahamsson *et al*., 2002)**

Method Name	Key Points	Special Features	Identified Shortcomings
Lean Software Development	Focuses on the elimination of waste and determining value up front.	High quality products resulting from the use of this methodology.	This methodology requires a very specific setup and rigorous following of the set out principles.
Scrum	Independent, small, self-organizing development teams, 30-day release cycles.	Enforce a paradigm shift from the "defined and repeatable" to the "new product development view of Scrum".	While Scrum details in specific how to manage the 30-day release cycle, the integration and acceptance tests are not detailed.
Dynamic Systems Development Method	Application of controls to RAD, use of time-boxing, empowered DSDM teams, active consortium to steer the method development.	First truly agile systems development method, use of prototyping, several user roles: "ambassador", "visionary" and "advisor".	While the method is available, only consortium members have access to white papers dealing with the actual use of the method.
Rapid Application Development	Development of systems in a very short time. Saves resources and promotes productivity. Delivers systems of high quality	The use of time-boxes and dynamic gathering of requirements.	Due to the use of time-boxes, the output of this methodology is limited. Some features are pushed to later versions in order to finish in a short amount of time.
Extreme Programming	Customer driven development, small teams, daily builds.	Refactoring - the ongoing redesign of the system to improve its performance and responsiveness to	While individual practices are suitable for many situations, overall view and management

		change.	practices are given. less attention.
Feature Driven Development	Five-step process, object-oriented component (i.e., feature) based development. Very short iterations: from hours to 2 weeks.	Method simplicity, design and implement the system by features, object modelling.	FDD focuses only on design and implementation. Needs other supporting approaches.
Crystal	Family of methods, each with the same underlying core values and principles. Techniques, roles, tools and standards vary.	Method design principles. Ability to select the most suitable method based on project size and criticality.	Too early to estimate. Only two or four suggested methods exist.

Table 2.9.2 above, compares the different ASDMs based on their key points, special features, and identified shortcomings. In many cases, the amount of documentation and literature available on a specific ASDM, was identified as a shortcoming. Most of the special features referred to result in a higher quality product and ease of use during the development phase.

2.9.2 Lightweight vs. Heavyweight methodologies

Any methodology can be classified as lightweight or heavyweight, depending on the emphasis of the following factors:

- Planning and documentation
- Well-defined phases in the development process
- The composition of the development team
- The use of well-elaborated modelling and coding techniques (Blum, 1996; Highsmith, 1999; Krutchen, 2001).

Heavyweight or traditional system development methodologies promote upfront overall planning of the roles to be played, activities to be performed, and artefacts to be produced (Germain & Robillard, 2005). Proponents of these methodologies argue that planning leads to lower overall cost, timely product delivery, and better software

quality. Traditional methodologies such as waterfall methodology and RUP (Rational Unified Process) to some extent are classified in this category (Germain & Robillard, 2005).

Lightweight or agile system development methodologies put extreme emphasis on delivering working code or product while downplaying the importance of formal processes and comprehensive documentation. Proponents of these methodologies argue that by putting more emphasis on “actual working code,” software development processes can adapt and react promptly to changes and demands imposed by their volatile development environment (Krutchen, 2001).

ASDMs like Extreme Programming or Scrum would fall under lightweight system development methodologies.

2.10 Project Management Methodologies

In the next section of this chapter, an overview of project management methodologies is given, followed by a detailed study of two PMMs used in the industry.

Firstly a proper definition of project management needs to be formulated in order to understand the scope thereof:

Project Management is the application of various tools and techniques to steer and direct all the different resources that are relevant toward the accomplishment of a uniquely defined task. Such a task must be completed within time, cost and quality constraints and each task requires a unique application or mix of the available tools and techniques that need to be applied in order to successfully complete the task (Atkinson, 1999).

According to Reiss (1993) a project is “a human activity that achieves a clear objective against a time scale” which implies that project management is a combination of management and planning as well as adapting to change.

In the various descriptions and definitions of project management the cost, time and quality constraints remain immanent and of greatest importance. Project

management is continuously evolving and very difficult to ultimately define. There are certain factors that have been created over the years although certain suggestions towards changing the success criteria have remained unchanged (Atkinson, 1999).

The basic constraints, which are cost, time and quality, are suggested to be the success criteria according to Oisen (1971) although this list is reduced to just two parameters, time and budget, when assuming the view of the customer (Wright, 1997) .

Most authors agree that cost, time and quality should be used as the success criteria, although not exclusively. To determine whether the project is going to plan there are certain temporary criteria to be applied in the delivery stage. These temporary criteria are used to measure the progress to date of the relevant project (Atkinson, 1999).

Performance measurement is essential in a project making use of cross-functional teams. Due to the fact that project management does not use traditional or functional teams, performance measurement can be reckoned as an important success factor (Meyer, 1994).

The basic constraints are unique in every project, for instance in projects for life critical systems, quality would be of utmost importance. The main focus in this case would be whether the project delivers a quality product instead of focusing on sound processes (Atkinson, 1999).

The question arises as to who should decide and sign off on the criteria? According to Struckenbruck (1987) the four most important stakeholders, comprising the project manager, top management, customer-client, and the team members, should decide on the criteria.

Customers and users are considered to be stakeholders in an information systems project and therefore their opinions about criteria for success should also be considered when assessing the project. Deane *et al*. (1997) attempted to align project outcomes with customer needs, which proved to be potential gaps which

could result in the project being ineffective if the performance was incorrect. Five possible levels of project performance gaps were identified and are represented in *Table* 2.10.1 below.

***Table* 2.10.1 – Performance Gaps (Atkinson, 1999)**

Actual project outcome needed by the customer
Gap 1
Desired project outcome as described by the customer
Gap 2
Desired project outcome as perceived by the project team
Gap 3
Specific project plan developed by the project team
Gap 4
Actual project outcome delivered to the customer
Gap 5
Project outcome as perceived by the customer

From *Table* 2.10.1 above, we can derive the fact that there is a performance gap between the actual project outcome that the customer needs, and the project outcome that the customer desires. The second performance gap is between the desired project outcome from the customer point of view, as opposed to the desired project outcome perceived by the project team. The third performance gap is between this aforementioned perception and the specific project plan that the project team develops. There is a fourth gap between the project plan developed by the project team, and the actual project outcome delivered to the customer. The final performance gap lies between the actual project outcome delivered to the customer, and the customer's perception as to what the project outcome should be.

One of the main concerns in the field is the number of projects failing or not meeting user requirements. In 2000, a study was done by the Standish Group in which they found 10 factors for project success. Among these, the second and third most important were "user involvement" and an "experienced project manager". Based on this, the conclusion can be made that most projects fail due to people and project management issues as opposed to technical ones. (Ceschi *et al.*, 2005)

Agile system development methodologies (ASDMs) have been developed precisely for this very reason, as the core principles of ASDMs are human-centred (Ceschi *et al.*, 2005).

2.11 The Project Management Body of Knowledge (PMBOK)

This project management methodology identifies the generally accepted best practice body of project management knowledge, providing a generalised language for project managers and uniform standards with regard to project management quality, excellence, and professionalism. Due to the fact that the PMBOK is a documented standard stating how project managers ought to construct and define their success, a certain level of authenticity is provided to project management education (Buckle & Thomas, 2003).

The focus of the PMBOK has always been on the hard skills which are deemed necessary for managing projects, relegating soft skills to the background. The main emphasis of the PMBOK is on the delivery of hard concepts such as technical knowledge, scientific management principles, the usage of various tools and techniques, and concrete outputs (Bourne & Walker, 2004).

The question is why a body of knowledge is important? In essence, the project management body of knowledge identifies and defines all the elements regarding project management in which competent project management professionals should be knowledgeable. The body of knowledge therefore states the ontology of the profession, which is the set of words, relationships and meanings, describing the underlying philosophy of project management (Morris *et al*., 2000).

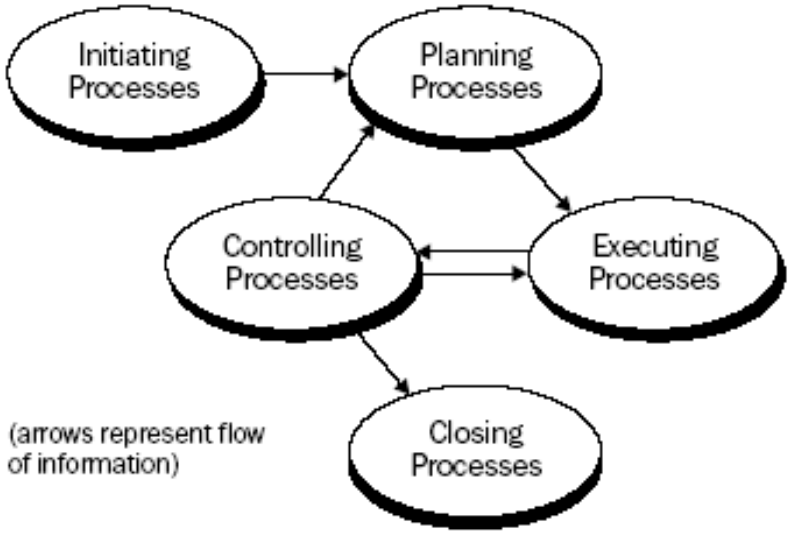
One of the main purposes of the PMBOK is to advance and improve the effectiveness of communications among all the technologies involved in project management. PMBOK also facilitates the development of a mutual understanding of an accepted body of knowledge for the profession. There are many opportunities to add additional strength to the PMBOK by implementing thorough descriptions of the leadership perspective in theory and in practice of project management. Here are a few preliminary suggestions:

- Define, describe, and set forth a common vision of leadership in the context of managing projects and other team endeavours;
- Describe how leadership cuts across the "warp and woof" of the PMBOK, with particular reference to human resources management and communications management;
- Formulate the idea and demonstrate that the opportunities for leadership by team leaders and team members alike have never been better, considering the current economic situation in various enterprises
- Present the differences, if they exist, between the leadership required for an enterprise as opposed to where teams are used (Cleland, 1995).

According to the Project Management Institute (PMI) (2008), there are five project management process groups illustrated in PMBOK:

- Initiating Process Group;
- Planning Process Group;
- Execution Process Group;
- Monitoring and Controlling Process Group; and
- Closing Process Group.

Figure 2.11.1 – The five project management process groups (PMI, 2008)



These five project management process groups in figure 2.11.1 above are required for any project although they should not be interpreted as being project phases. In either large scale or complex projects, these process groups could be repeated for each phase in the project. There are a total of 42 project management processes mapped into the 5 main project management process groups.

There are nine knowledge areas of project management in PMBOK:

- Project integration management

This involves the coordination of all other project management knowledge areas throughout the lifecycle of a project.

- Project scope management

This includes the processes involved when defining and directing which work should be included in a project.

- Project time management

This involves the processes required to ensure the timely completion of a project.

- Project cost management

This includes all the required processes to ensure that a project is completed within a specified budget.

- Project quality management

The purpose of this knowledge area is to ensure that the project satisfies the requirements for which it was undertaken.

- Project human resource management

This knowledge area includes all processes involved with making the most effective use of the people that form part of the project.

- Project communications management

The main goal of project communications management is to ensure timely and appropriate generation, collection, storage and disclosure of project information.

- Project risk management

This knowledge area involves the identification, analysis, and response to risks throughout the project lifecycle with the best interests of meeting the project objectives in mind.

- Project procurement management

This knowledge area is concerned with the acquisition of goods and/or services from an external source. It is also referred to as outsourcing (PMI, 2008).

Figure 2.11.2 – The nine project management knowledge areas (PMI, 2008)

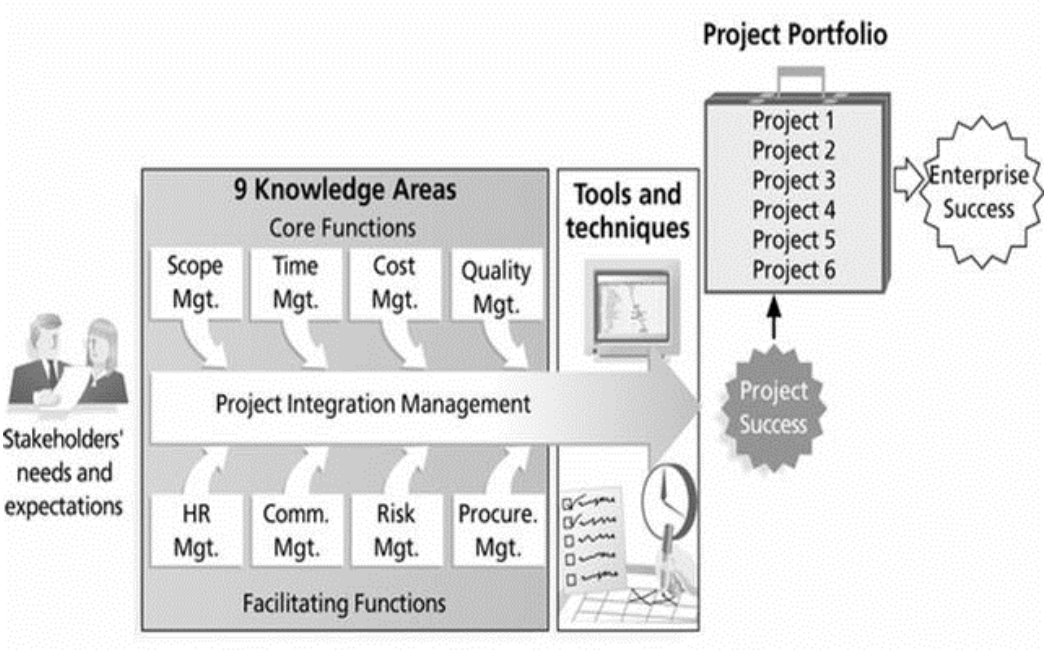


Figure 2.11.2 above depicts the nine project management knowledge areas.

According to PMBOK, project management knowledge areas are the identified areas of project management defined by their knowledge requirements and described in terms of their component processes, practices, inputs, outputs, tools, and techniques (Yeong, 2009).

These knowledge areas are very similar to the PRINCE2 themes mentioned in the next section of this chapter.

2.12 PRINCE2

The term “PRINCE” refers to PRojects IN Controlled Environments. This methodology is a structured method to ensure effective project management which is greatly influenced by information technology although it can be adapted to fit any project. PRINCE2 introduces a list of reasons as to why projects fail, and the methodology attempts to remove these causes (Wideman, 2002).

PRINCE2 consists of seven principles, seven themes and seven processes (OGC, 2009).

The seven PRINCE2 principles are:

- Continued business justification: A PRINCE2 project has continued business justification,
- Learn from experience: PRINCE2 project teams learn from previous experience. Lessons are sought after and recorded, where after action is taken throughout the lifecycle of the project,
- Defined roles and responsibilities: A PRINCE2 project has defined and agreed roles and responsibilities within an organization structure that engages the business, user and supplier stakeholder interests,
- Manage by stages: A PRINCE2 project is planned, monitored and controlled on a per stage basis,
- Manage by exception: A PRINCE2 project has defined tolerances for each project objective to establish limits of delegated authority,
- Focus on products: A PRINCE2 project focuses on the definition and delivery of products, in particular their quality requirements,
- Tailor to suit the project environment: PRINCE2 is tailored to suit the project's environment, size, complexity, importance, capability and risk.

The seven PRINCE2 themes are:

Business Case: It addresses how the idea is formulated into a feasible investment proposition for the organization and how project management maintains the focus on the organization's objectives throughout the project; *Organization*: It describes the roles and responsibilities in the temporary PRINCE2 project management team required to manage the project effectively; *Quality*: It explains how the outline is developed so that all participants understand the quality attributes of the products to be delivered; *Plans*: It complements the Quality theme through the description of various steps required to develop plans and the PRINCE2 techniques that should be applied; *Risk*: It addresses how project management manages the uncertainties in its plans and in the wider project management; *Change*: It describes how project management assesses and acts upon issues which have a potential impact on any of the business aspects of the project; *Progress*: It explains the decision-making

process for approving plans, the monitoring of actual performance and the escalation process if events do not go according to plan

The seven PRINCE2 Processes are:

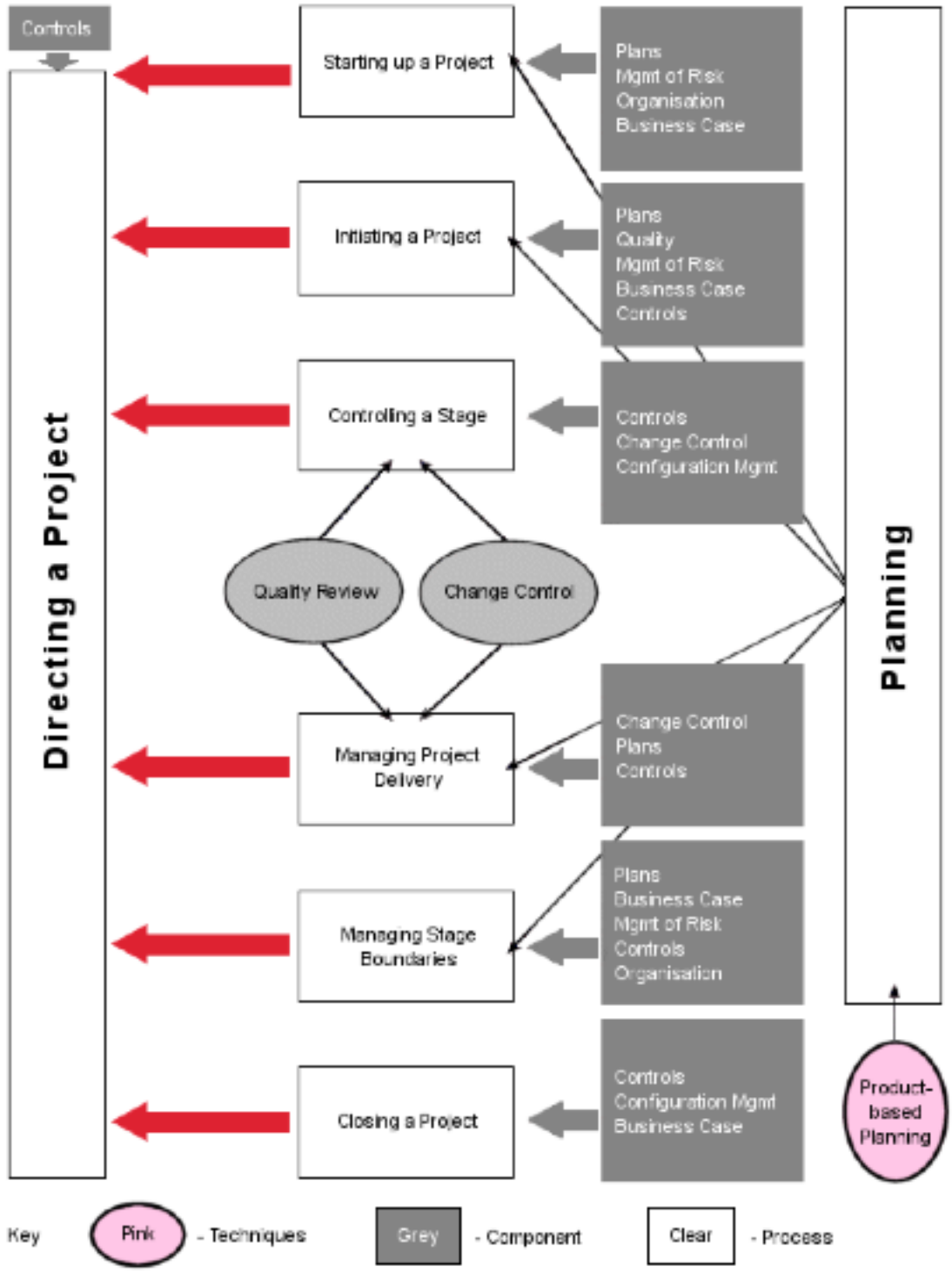
Starting up a Project: The purpose of this process is to ensure that the prerequisites for initiating a viable and worthwhile project are in place; *Directing a Project*: The purpose of this process is to enable the project board to be accountable for the project's success through the making of key decisions and exercising overall control while delegating the day-to-day management of the project to the project manager; *Initiating a Project*: The purpose of this process is to establish solid foundations for the project, enabling the organization to understand the work that needs to be done to deliver the project's products before committing to a significant spend; *Controlling a Stage*: The purpose of this process is to assign work to be done, monitor such work, deal with issues, report progress to the project board, and take corrective actions to ensure that the stage remains within tolerance; *Managing Product Delivery*: The purpose of this process is to control the link between the project manager and team manager(s), by placing formal requirements on accepting, executing and delivering project work; *Closing a Project*: The purpose of this process is to provide a fixed point at which the acceptance for the product resulting from the project is confirmed, and to recognize that objectives set out in the original project initiation documentation have been achieved (or approved changes to the original objectives have been achieved), or that the project has nothing more to contribute.

The PRINCE2 project management methodology acts as a bridge between the customers, users and suppliers, which brings these various parties together on the project board through common project management language. PRINCE2 management products are not necessarily documents, but rather useful information sets which enable the project board and project manager to take action and make decisions (Yeong, 2009).

Although the methodology lays down the basic principles and processes as well as how they link together, the application thereof needs to be scaled to fit the size and needs of the specific project to which PRINCE2 is applied (Wideman, 2002).

The PRINCE2 methodology is project life-cycle based consisting of eight major processes, each with their respective sub-processes, which brings the total number of processes to 45. There are also three techniques involved in PRINCE2 namely “Product Based Planning”, “Quality Review”, and “Change Control” (Wideman, 2002). Figure 2.12.1 below depicts the various components and techniques involved in the PRINCE2 processes.

Figure 2.12.1 - The use of PRINCE2 components and techniques in the processes (Wideman, 2002)



2.13 Comparison between PRINCE2 and PMBOK

These two methodologies are very similar in many ways and are based on the same grounds, although PRINCE2 is not as comprehensive as the PMBOK. The principles of the PMBOK is applied in PRINCE2, as it should be with any project management methodology. The focus of PRINCE2 is mainly on the elements identified to be a crucial part of successfully assessing and completing a project. Essentially, a process is constructed tying these crucial elements together which results in project risk being reduced substantially. PRINCE2 also provides techniques to support these constructs. "The Guide to the PMBOK" requires the practitioner to apply a project management methodology which in turn, is provided by PRINCE2 (Siegelaub, 2006).

PRINCE2 intends to organize and focus project management knowledge in order to make it suitable to apply in most project environments. In order to successfully work with this methodology, a certain level of experience is needed to complete the PRINCE2 package. The main strength of PRINCE2 lies in its common-sense approach (Siegelaub, 2006).

***Table* 2.13.1 - Comparison of PMBOK areas of knowledge and PRINCE2 components (Siegelaub, 2006)**

PMBOK® Knowledge Area	*Comparable PRINCE2™ Components*
Integration	Combined Processes and Components, Change Control
Scope, Time, Cost	Plans, Business Case
Quality	Quality, Configuration Management
Risk	Risk
Communications	Controls
Human Resources	Organization
Procurement	Not Covered

In *Table* 2.13.1 the different areas of knowledge in PMBOK are compared to the components in PRINCE2. The underlying difference between the PMBOK and PRINCE2 is that the PMBOK is a knowledge based project management methodology covering wide proven practices and areas while PRINCE2 provides a more prescriptive or process oriented approach for the project manager to apply to projects. Generally, the PMBOK is much more comprehensive whilst PRINCE2 has a more realistic nature. PMBOK addresses the question of "What" in project management, whereas PRINCE2 tells the "How" in project management.

Table 2.13.2 below depicts the PMBOK knowledge areas as opposed to the different principles, themes and processes of PRINCE2.

***Table* 2.13.2 - Comparison of PMBOK areas of knowledge and PRINCE2 principles, themes and processes (Yeong, 2009)**

PMBOK Knowledge Areas	**PRINCE2 Principles / Themes / Processes**
Project Integration Management	[Principles]: 7 Principles [Themes]: Organization, Plans, Change, Progress [Processes]: 7 Processes
Project Scope Management	[Principles]: 7 Principles

	[Themes]: Business Case, Plans, Change [Processes]: 7 Processes
Project Time Management	[Principles]: Continued business justification, Learn from experience, Manage by stages, Tailor to suit the project environment [Themes]: Plans, Change, Progress [Processes]: 7 Processes
Project Cost Management	[Principles]: Continued business justification, Learn from experience, Tailor to suit the Project environment [Themes]: Plans, Change, Progress [Processes]: 7 Processes
Project Quality Management	[Principles]: Continued business justification, Learn from experience, Focus on products, Tailor to suit the Project environment [Themes]: Quality, Plans, Change [Processes]: 7 Processes
Project Human Resource Management	[Principles]: Learn from experience, Defined roles and responsibilities, Tailor to suit the Project environment [Themes]: Organization, Plans [Processes]: 7 Processes
Project Communication Management	[Principles]: Learn from experience, Tailor to suit Project environment [Themes]: Plans, Change, Progress [Processes]: 7 Processes
Project Risk Management	[Principles]: Learn from experience, Manage by exception, Tailor to suit Project environment [Themes]: Risk, Change, Progress [Processes]: 7 Processes
Project Procurement Management	[Principles]: Continue business justification, Learn from experience, Focus on products, Tailor to suit Project environment [Themes]: Business case, Plans, Change, Progress [Processes]: Initiating up a Project, Managing Product Delivery

For each PMBOK knowledge area, there are 7 processes involved with PRINCE2. Plans, change and progress are some of the eminently important themes involved in PRINCE2 for each of the knowledge areas in PMBOK. One of the key principles in PRINCE2, is learning from previous experience. This principle was repeated multiple times when comparing to the knowledge areas in PMBOK.

***Table* 2.13.3 - Comparison of PMBOK process groups and PRINCE2 processes (Yeong, 2009)**

PMBOK Process Groups	**PRINCE2 Processes**
Initiating Process Group	Starting up a Project Directing a Project
Planning Process Group	Initiating a Project
Executing Process Group	Managing Product Delivery Managing a Stage Boundary
Monitoring & Controlling Process Group	Directing a Project Controlling a Stage Managing Product Delivery Managing a Stage Boundary
Closing Process Group	Managing a Stage Boundary Closing a Project

In *Table* 2.13.3 the various PMBOK process groups are compared to the PRINCE2 processes. The PMBOK process groups are unique for each instance, whereas some PRINCE2 processes are repeated across multiple instances. An example of such a PRINCE2 process is managing a stage boundary, which occurs in 3 processes.

Both PMBOK and PRINCE2 have over 15 years of history in project management discipline. They are equally accepted across the world by project management professionals. Both approaches should not be viewed as if they were in competition against each other, but rather to complement the strengths and weaknesses of both knowledge and processes.

2.14 Critical Success Factors in Agile Systems Development Projects

Very few studies have been done regarding the critical success factors involved in agile systems development. There have however been various case studies and research theories documenting the failures and successes in the implementation of ASDMs and projects involving ASDMs. By reviewing these failures and successes in academic literature, we could identify the possible success factors involved with agile systems development projects. By acknowledgement of the failures involved, we learn how to avoid some serious pitfalls that critically contribute to the success of the project (Chow & Cao, 2007).

According to Reel (1999) there are 10 signs of software development project failure, of which at least 7 can be determined even before the design phase of the project. There are various challenges around agile systems development projects, namely management challenges, people, processes, and technology dimensions of migrating to agile projects (Boehm & Turner, 2005; Nerur *et al*., 2005).

Considering the above-mentioned literature, project failures can be categorised as follows according to Chow and Cao (2007):

- Organisational

  This category includes the following attributes: lack of executive sponsorship, lack of management commitment, organizational culture too traditional, organizational culture too political, organizational size too large, and the lack of agile logistical arrangements

- People

  This category includes the following attributes: lack of necessary skills-set, lack of project management competence, lack of team work, resistance from groups or individuals, and a bad customer relationship.

- Process

This category includes the following attributes: poorly defined project scope, poorly defined project requirements, poorly defined project planning, lack of agile progress tracking mechanism, lack of customer presence, and a poorly defined customer role.

- Technical

This category includes the following attributes: lack of complete set of correct agile practices, and the inappropriateness of technology and tools.

The success of research of agile systems development projects is predominantly based on various case studies and observations of agile systems development projects and practices. Highsmith (2002) reports from direct experience regarding agile systems development projects whilst Schatz and Abdelshafi (2005) contribute with their Primavera case study and Karlstrom and Runeson (2005) give insight from the Star-Gate case study. Koch (2005) also makes a compilation of a wide range of success factors regarding agile systems development projects. Based on the above mentioned academic literature, the critical success factors in agile systems development projects can be categorised as follows:

- Organisational

This category includes the following success factors: strong executive support, a committed sponsor or manager, a cooperative organizational culture instead of hierarchical, strong oral culture placing high value on face-to-face communication, an organization where agile methodology is universally accepted, collation of the whole team, a facility with proper agile-style work environment, a reward system appropriate for agile systems.

- People

This category includes the following success factors: team members with high competence and expertise, team members with great motivation, managers knowledgeable in the agile process, managers who have a light-touch or adaptive management style, coherent, self-organizing teamwork, a good customer relationship.

- Process

This category includes the following success factors: following agile-oriented requirement management process, following agile-oriented project management process, following agile-oriented configuration management process, strong communication focus with daily face-to-face meetings, honouring regular working schedule with no overtime, strong customer commitment and presence, customer having full authority.

- Technical

This category includes the following success factors: well-defined coding standards up front, pursuing simple design, rigorous refactoring activities, the right amount of documentation, regular delivery of software, delivering the most important features first, doing correct integration testing, and providing appropriate technical training to teams.

- Project

This category includes the following success factors: project nature being non-life-critical, project type being of variable scope with emergent requirements, projects with dynamic, accelerated schedule, projects with a small team, projects with no multiple independent teams, projects with up-front cost evaluation done, projects with up-front risk analysis done, the project could handle unexpected crises and deviations from plan, the project was efficiently coordinated and successfully integrated activities, documents and personnel, sufficient testing was done during the project, the project had clearly defined business priorities, the project had a clear vision and objectives, project management was good, the project applied appropriate systems development methodology knowledge (Chow & Cao, 2007; Huisman, 2000)

2.15 Critical evaluation of ASDM and PMM against critical success factors

In this section, the various ASDMs and PMMs are measured against the critical success factors.

Table 2.15.1 below was constructed by applying the theoretical knowledge gathered from the academic literature in the previous sections of this chapter. This table compares the different ASDMs and the measure to which they apply to the different critical success factors constructed earlier in this chapter. The following philosophy was used to score these ASDMs: If the critical success factor is deemed to be addressed by the ASDM 80% of the time or higher, a “yes” was awarded, if the critical success factor was addressed between 30% and 80% of the time, a “partially” was awarded, and in cases where the critical success factor was addressed 30% of the time or less, a “no” was awarded. The following scale was used to score each individual ASDM: Yes – 1, Partially – 0.5, No – 0. The term “partially” is represented by a “P” in the table below.

Table 2.15.1 - Evaluation of ASDMs against various critical success factors

ASDM	LSD	SCRUM	DSDM	RAD	XP	FDD	CRYSTAL
Critical Success Factor							
Strong executive support	**YES**	**YES**	**YES**	**YES**	**P**	**YES**	**YES**
Committed sponsor or manager	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Cooperative organizational culture instead of hierarchical	**NO**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Oral culture placing high value on face-to-face communication	**P**	**YES**	**P**	**P**	**YES**	**NO**	**P**
Organizations where agile methodology is universally accepted	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**P**
Collation of the whole team	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**P**
Facility with proper agile-style work environment	**P**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Reward system appropriate for agile	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Team members with high competence and expertise	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Team members with great motivation	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Managers knowledgeable in the agile process	**YES**	**YES**	**YES**	**YES**	**YES**	**P**	**YES**
Managers who have light-touch or adaptive management style	**P**	**YES**	**YES**	**YES**	**YES**	**P**	**P**
Coherent, self-organizing teamwork	**YES**	**YES**	**YES**	**YES**	**YES**	**P**	**P**
Good customer relationship	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Following agile-oriented requirement management process	**P**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**

Following agile-oriented project management process	**P**	**YES**	**YES**	**YES**	**YES**	**P**	**YES**
Following agile-oriented configuration management process	**P**	**YES**	**YES**	**YES**	**YES**	**P**	**YES**
Strong communication focus with daily face-to-face meetings	**NO**	**YES**	**NO**	**NO**	**YES**	**NO**	**P**
Honouring regular working schedule - no overtime	**P**	**P**	**P**	**P**	**P**	**P**	**P**
Strong customer commitment and presence	**NO**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Customer having full authority	**NO**	**P**	**P**	**NO**	**P**	**NO**	**P**
Well-defined coding standards up front	**YES**	**P**	**P**	**NO**	**P**	**YES**	**YES**
Pursuing simple design	**NO**	**YES**	**YES**	**YES**	**YES**	**P**	**P**
Rigorous refactoring activities	**NO**	**YES**	**YES**	**NO**	**YES**	**P**	**YES**
Right amount of documentation	**YES**	**P**	**NO**	**NO**	**P**	**P**	**P**
Regular delivery of software	**P**	**YES**	**YES**	**YES**	**YES**	**P**	**YES**
Delivering most important features first	**Yes**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Correct integration testing	**YES**	**YES**	**P**	**NO**	**P**	**YES**	**YES**
Appropriate technical training to team	**YES**	**P**	**P**	**YES**	**P**	**YES**	**YES**
Project nature being non-life-critical	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Project type being of variable scope with emergent requirement	**P**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
Projects with dynamic, accelerated	**NO**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**

schedule							
Projects with small team	**P**	**YES**	**YES**	**YES**	**YES**	**NO**	**P**
Projects with no multiple independent teams	**P**	**YES**	**YES**	**YES**	**YES**	**YES**	**P**
Projects with up-front cost evaluation done	**YES**	**P**	**P**	**NO**	**NO**	**P**	**YES**
Projects with up-front risk analysis done	**YES**	**P**	**P**	**NO**	**NO**	**P**	**YES**
The project could handle unexpected crises and deviations from plan	**P**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
The project was efficiently coordinated and successfully integrated activities, documents and personnel	**YES**	**YES**	**YES**	**P**	**YES**	**YES**	**YES**
Sufficient testing was done during the project	**YES**	**P**	**P**	**NO**	**YES**	**P**	**YES**
The project had clearly defined business priorities	**YES**	**YES**	**P**	**YES**	**YES**	**YES**	**P**
The project had a clear vision and objectives	**YES**	**P**	**P**	**YES**	**YES**	**YES**	**P**
Project management was good	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
The project applied appropriate systems development methodology	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**	**YES**
FINAL SCORE	**30**	**38.5**	**35.5**	**32.5**	**37.5**	**32.5**	**36**

Based on the scores from *Table* 2.15.1 above, Scrum addresses the most critical success factors, closely followed by XP and Crystal.

Table 2.15.2 below was constructed by applying the theoretical knowledge gathered from the academic literature in the previous sections of this chapter. This *Table* compares the different PMMs and the measure to which they apply to the different

critical success factors constructed earlier in this chapter. The following philosophy was used to score these PMMs: If the critical success factor is deemed to be addressed by the PMM 80% of the time or higher, a “yes” was awarded, if the critical success factor was addressed between 30% and 80% of the time, a “partially” was awarded, and in cases where the critical success factor was addressed 30% of the time or less, a “no” was awarded. The following scale was used to score each individual PMM: Yes – 1, Partially – 0.5, No – 0.

***Table* 2.15.2 - Evaluation of PMMs against various critical success factors**

PMM	PMBOK	PRINCE2
Critical Success Factor		
Strong executive support	**YES**	**YES**
Committed sponsor or manager	**YES**	**YES**
Cooperative organizational culture instead of hierarchical	**PARTIALLY**	**PARTIALLY**
Oral culture placing high value on face-to-face communication	**PARTIALLY**	**PARTIALLY**
Organizations where agile methodology is universally accepted	**PARTIALLY**	**PARTIALLY**
Collation of the whole team	**YES**	**YES**
Facility with proper agile-style work environment	**PARTIALLY**	**NO**
Reward system appropriate for agile	**PARTIALLY**	**PARTIALLY**
Team members with high competence and expertise	**YES**	**YES**
Team members with great motivation	**YES**	**YES**
Managers knowledgeable in the agile process	**YES**	**YES**
Managers who have light-touch or adaptive management style	**YES**	**YES**
Coherent, self-organizing teamwork	**YES**	**YES**
Good customer relationship	**YES**	**YES**
Following agile-oriented requirement management process	**PARTIALLY**	**PARTIALLY**
Following agile-oriented project management process	**YES**	**PARTIALLY**
Following agile-oriented configuration management process	**YES**	**PARTIALLY**
Strong communication focus with daily face-to-face meetings	**NO**	**NO**
Honouring regular working schedule - no overtime	**PARTIALLY**	**PARTIALLY**
Strong customer commitment and presence	**PARTIALLY**	**PARTIALY**
Customer having full authority	**NO**	**NO**
Well-defined coding standards up front	**PARTIALLY**	**YES**

Pursuing simple design	**PARTIALLY**	**YES**
Rigorous refactoring activities	**NO**	**NO**
Right amount of documentation	**YES**	**YES**
Regular delivery of software	**YES**	**YES**
Delivering most important features first	**YES**	**YES**
Correct integration testing	**YES**	**YES**
Appropriate technical training to team	**YES**	**PARTIALLY**
Project nature being non-life-critical	**YES**	**YES**
Project type being of variable scope with emergent requirement	**YES**	**YES**
Projects with dynamic, accelerated schedule	**NO**	**PARTIALLY**
Projects with small team	**PARTIALLY**	**PARTIALLY**
Projects with no multiple independent teams	**PARTIALLY**	**PARTIALLY**
Projects with up-front cost evaluation done	**YES**	**YES**
Projects with up-front risk analysis done	**YES**	**YES**
The project could handle unexpected crises and deviations from plan	**PARTIALLY**	**PARTIALLY**
The project was efficiently coordinated and successfully integrated activities, documents and personnel	**YES**	**YES**
Sufficient testing was done during the project	**YES**	**YES**
The project had clearly defined business priorities	**YES**	**YES**
The project had a clear vision and objectives	**YES**	**YES**
Project management was good	**YES**	**YES**
The project applied appropriate systems development methodology	**YES**	**YES**
FINAL SCORES	**32.5**	**32**

According to the scores in *Table* 2.15.2 above, PMBOK very narrowly outweighs PRINCE2 when it comes to addressing critical success factors.

ASDMs and PMMs address various critical success factors in different ways which can be seen from the analysis done in *Table* 2.15.1 and *Table* 2.15.2 in the previous sections of this chapter. The use of an ASDM could greatly impact the success of a project, depending on many variables on a project-to-project basis. The ASDM

focuses on the processes involved in a project, whereas the PMM is centred around the management of these various processes. When applying ASDM and PMM knowledge, the critical success factors are addressed in a much more thorough fashion, preventing the project from easily going of course. If no ASDM or PMM knowledge is applied to a project, there is little or no control and the project could easily deviate from the initial scope and business requirements.

2.16 Conclusion

There are undoubtedly many advantages in adopting an Agile development approach, although Agile methodologies are not the remedy to all problems in the business environment. Traditional methods still have their rightful place in the industry, especially when thoroughness and error proofing take preference over time and money savings. By using both PRINCE2 and PMBOK, project managers are applying best practice from both worlds. An objective assessment of all possible development strategies must be made for each project before an organization can choose the approach best suited for the project under consideration. If Agile systems development is the right solution for a particular project, integrating it with continuous process improvement and taking into consideration the various critical success factors involved, will produce optimal results.

Chapter 3
Research Methodology – A positivistic approach

3.1 Introduction

In this chapter the research methodology will be presented along with its underlying principles. The research paradigm, research method, data collection and data analysis will be discussed in this chapter.

3.2 Research paradigm

The most fundamental set of assumptions adopted by a professional community that allows every member of such a community to share similar perceptions and engage in commonly shared practices is called a “paradigm” (Hirschheim & Klein, 1989).

For this particular study, the positivistic research paradigm was chosen. At the heart of positivism lies Karl Popper's two-part differentiation between "scientific theories" and "myth". A scientific theory is one of which the predictions can be empirically falsified, meaning it can be proven wrong. Based on this theorem, a scientific theory could be a risky endeavour, the risk being that the theory might be thrown out if it cannot be supported by the applicable data in that situation. The core of positivism is the Falsification Principle. Basically we conclude that experience can show theories to be wrong, but can never prove them right. The underlying principle here is that the accuracy or correctness of theories can never be shown (Straub *et al.*, 2004).

When relating science to positivism, it is all about solving problems. It is not about matching a theory to a specific observation. Observation is subjective and therefore it is almost always possible to choose and select data supporting almost any theory on condition that the researcher looks for confirming examples. In essence, scientific theory, from a positivist point of view, is about trying to falsify predictions of the theory, i.e. prove them to be wrong (Straub *et al.*, 2004).

The positivist approach uses truly scientific methods when gathering data. The logical positivist, also known as "empiricist", philosophy assumes that science is

objective and places emphasis on thorough measurement and hypothesis testing. This model of research is characterized mostly by a statistical method of research and states that any statements or proposals are only meaningful if they can be empirically verified (Brown, 1977). In other words, all information must be proven scientifically.

The problem with a scientific theory is that it is accepted until proven otherwise. It is at best extensively corroborated and is never verified because a future observation may contradict it (Straub *et al.*, 2004).

This posed as a significant problem to positivists as they could no longer accept the fundamental propositions of the system as unconditional truths (the reason being that these could not be empirically tested). Positivists were thus forced to alter their point of view and tolerate some modifications to their theory.

The positivist philosophy has many limitations, especially when applied to social sciences. Positivism, based on the deductive statistical method, generalizes a universal statement of truth forthcoming out of observations of a certain number of positive instances. This strict inductive approach is often inadequate because speculation and creation of a hypothesis are essential in conjunction with the systematic procedure of theory building (Leong, 1985).

Secondly, the positivistic approach has its roots in pure observation, which is near impossible in research since observations are always subject to measurement faults (Anderson, 1983).

Finally, the positivistic approach assumes that knowledge is derived from an objective interpretation of assumptions, without any of the possible subjective opinions or prior knowledge of the scientist playing a role. Taking these arguments into account, it is clear that the positivistic (empirical) approach emphasizes pure rigour (Bharadwaj, 2000).

The absolute dominance of the positivistic approach to information systems research has led to criticism that information systems research has frequently sacrificed relevance for rigour.

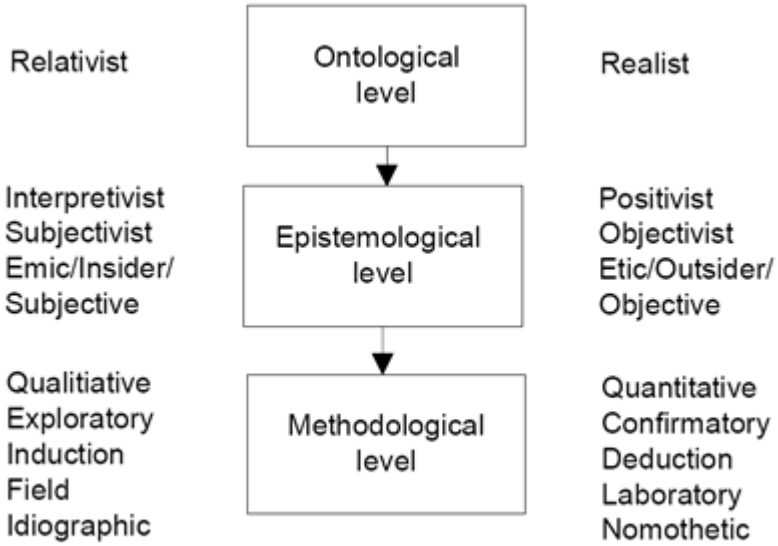
Another disadvantage of the positivistic approach is when applied to practical problems the problem scope is narrowed down to those aspects that are researchable by standard quantitative methods.

Relying exclusively on the statistical testing of hypothesis has been criticized in the social sciences as providing calamitous effects (Cook & Campbell, 1979).

By ignoring the subjective and inter-subjective dimensions such as power, politics, and other socially constructed variables, the purely positivistic approach simply fails to produce deeper insights into information systems phenomena (Klein & Lyytinen, 1984).

Figure 3.2.1 below depicts the characteristics of a research paradigm.

Figure 3.2.1: Research Paradigm Characteristics (Khazanchi & Munkvold, 2002)



The term *paradigm* has the following key characteristics:

- Ontology, i.e., the theory or study of existence (being). For example, ontological assumptions in the conduct of inquiry within a paradigm might specifically characterize the nature of reality;

- Epistemology, i.e., a theory of knowledge that deals with the nature of knowledge, the scope thereof, and provides a set of criteria for evaluating knowledge claims and establishing whether such claims are warranted; and
- Methodology, i.e., a procedure through which knowledge will be generated (Khazanchi & Munkvold, 2002).

These key characteristics are depicted in figure 3.2.1 above.

Table 3.2.1 below provides a summarised comparison of the various research paradigms measured against the ontological assumptions, the epistemological assumptions, the relationship between theory and practice, and the role played by the researcher in each paradigm. The positivist approach is the most objective approach, whereas the interpretivist approach is more interactive and could be more subjective. Critical research is transformative and requires the most involvement by the researcher, also making it the most subjective research paradigm.

***Table* 3.2.1 – Comparison of research paradigms (Khazanchi & Munkvold, 2002)**

.	Positivist[1]	Interpretivist	Critical Research
Ontological Assumptions	"Naive Realism" in which an understandable reality is assumed to exist, driven by immutable natural laws. True nature of reality can only be obtained by testing theories about actual objects, processes or structures in the real world.	Relativist; the social world is produced and reinforced by humans through their action and interaction	Historical realist; social reality is historically constituted; human beings, organizations, and societies are not confined to existing in a particular state
Epistemological Assumptions	• Verification of hypothesis through rigorous empirical testing • Search for universal laws or principles • Tight coupling among explanation, prediction and control	• Understanding of the social world from the participants' perspective, through interpretation of their meanings and actions • Researchers' prior assumptions, beliefs, values, and interests always intervene to shape their investigations	• Knowledge is grounded in social and historical practices • Knowledge is generated and justified by a critical evaluation of social systems in the context of researchers' theoretical framework adopted to conduct research
Relationship between Theory and Practice	• It is possible to discover universal laws that govern the external world	• Generative mechanisms identified for phenomena in the social sciences should be viewed as 'tendencies', which are valuable in explanations of past data but not wholly predictive for future situations	• Generalizations point to regularities of process rather than cross-sectional differences • Generalization in critical research focuses on the "totality" of relationships • There can be no theory-independent collection and interpretation of evidence to conclusively prove or disprove a theory
Role of the Researcher	Objective, impartial observer, passive, value-neutral	Interactive; the researcher interacts with the human subjects of the enquiry, changing the perceptions of both parties	Transformative; initiating change in social relations and practices, helping to eliminate the bases of alienation and domination

3.3 Research Strategies and Data Generation Methods Associated with the Positivistic Research Paradigm

There are various strategies involved with the positivistic research paradigm which will be covered in this section.

<u>The survey method</u>

The survey approach is a collective term that refers to a group of methods where the emphasis is placed on quantitative analysis. An example of such methods would be questionnaires, interviews or by collecting data from published statistics. These

various forms of data that have been collected are then analysed using statistical techniques.

Surveys test different hypotheses on data which is gathered systematically from a well-known population by means of a questionnaire which is administered in writing or orally. This method is mostly applied when studying large samples, hypothesis development or testing, and description (Wynekoop & Russo, 1997).

Through an in-depth study of organisations, the survey approach seeks to discover the different relationships commonly found across organisations and henceforth provide statements that can be generalised in the object of study. A disadvantage of the survey approach is that it often provides information that is only relevant at a certain point in time which in turn yields very little information on the underlying meaning of the data. As a researcher, many variables of interest could be immeasurable through the survey method (Gable, 1994).

In contrast to some disadvantages, surveys can accurately document the norm, identify extreme outcomes, and portray associations between the different variables in a sample. Vidich and Shapiro reason that without any survey data, the observer would only be enabled to make reasonable guesses concerning his area of ignorance in an effort to reduce preconception (Vidich & Shapiro, 1955:31) .

For a survey to produce successful results, it must contain the right set of questions that are asked in the right way. Survey research is very inflexible to discoveries that can be made throughout the data-collection process. This is true in the sense that when applying the survey method there is little one can do upon discovering that some crucial items were omitted from the questionnaire, or by realizing that a question is perceived as ambiguous or not understood by respondents. The ideal is for the researcher to have a clear picture of what a specific answer should be prior to starting a survey (Gable, 1994).

The experimentation method

Another efficient research strategy that correlates with positivism is experimentation. Laboratory research (experimentation) refers to various treatments and independent

variables which are controlled in an artificial setting. This is used for hypothesis testing and simulations (Wynekoop & Russo, 1997).

While it is easier to extend research from the laboratory to the field, one should also make an attempt to experimentally test various propositions that were proven non-experimentally. A clear indication is found in the desire to combine survey and experimentally designed methods. A clever observation is made in psychology where an unfortunate division is made between survey and experimental researchers. Psychologists generally do not understand both methods and tend to discredit the opposite method from the one in which they specialize. In the psychological field surveys are viewed as descriptive whereas experimentation is viewed as hypothesis-testing. According to this statement, the survey method is inferior to experimentation (Gutek, 1991a:321).

Through a combination of the strengths of survey research with that of experimental research, a superior piece of research can be yielded. Using different methods in appropriate situations is a good idea to promote thorough and accurate research.

***Table* 3.3.1: Relative strengths of survey and experimentation (Gable, 1994)**

	Survey	**Experimentation**
Controllability	Medium	High
Deductibility	Medium	High
Repeatability	Medium	High
Generalisability	High	Medium
Discoverability	Medium	Low
Representability	Medium	Low

Table 3.3.1 depicts the relative strengths of the survey – and experimentation research methods. Although the experimentation research method has more “High” ratings compared to the survey research method, it also has 2 “Low” ratings, which do not occur on the survey side of things. This shows that the survey research method is more consistent and flexible in most situations of study. Surveys are also most suited for reaching larger audiences spread across multiple geographical locations.

According to Wynekoop and Russo (1997) there are nine research methods for studying systems development methodologies. These methods include normative writings, laboratory research, surveys, field inquiries, case studies, action research, interpretive research, and descriptive research and practice descriptions.

Due to the nature of the research problem of this study, non-empirical research methods such as normative writings and descriptive research are inadequate. Furthermore, this study covers a vast number of agile systems development methodologies rather than focusing on just one or only a few agile systems development methodologies. After taking into account all of these factors, it became clear that only a survey method would suffice for this study.

The decision was therefore made to implement a survey as the research method. There are numerous reasons for this decision. It firstly allows us to investigate the critical success factors in agile systems development projects in a large number of organizations and individuals. These organizations and individuals are widely spread across South Africa, making the study that much more feasible. Many researchers suggest that surveys should be used when studying large samples (Huisman, 2000; Wynekoop & Russo, 1997). In a situation where the respondents are widely dispersed, which is also the case regarding this study, the use of a survey is appropriate (Huisman, 2000).

In addition to the above-mentioned statements, surveys test different hypotheses on data which is gathered systematically from a well-known population by means of a questionnaire which is administered in writing or orally. This method is mostly applied when studying large samples, hypothesis development or testing, and description (Wynekoop & Russo, 1997).

3.4 Data analysis methods associated with the research strategies used in Positivism

Quantitative positivist research refers to a set of methods and techniques allowing information system researchers to answer various questions with regard to the interaction between humans and computers. We will review two specific cornerstones on which this method of research is built, the first cornerstone being

the emphasis on quantitative data. With reference to this cornerstone, these methods and techniques specialize in vast quantities in the sense that numbers represent many values and different levels of theoretical constructs and concepts and the interpretation of such numbers is regarded as strong scientific evidence showing how a specific phenomenon functions (Straub *et al.*, 2004).

The presence of quantities dominates quantitative positivist research in such a fashion that statistical tools and packages form an essential part of the researcher's toolkit. Various sources from which data springs forth are of much less concern in the identification of an approach as being quantitative positivistic than the fact that empirically derivative numbers lie at the core of the scientific evidence assembled.

A quantitative positivist researcher could utilise archival data or accumulate it through structured interviews. In essence the researcher is ultimately motivated exclusively by the numerical outputs and how to derive meaning from them (Straub *et al.*, 2004).

The second cornerstone, referring to the emphasis on positivist philosophy, is also built on numerical analysis. Positivist philosophy defines a scientific theory as one that can be falsified (Straub *et al.*, 2004).

Generally positivist or post-positivist researchers prefer a quantitative approach when validating the various research models they propose.

"A useful perspective on what quantitative, positivist research methods are, conceptually, can be gained by seeing them in the context of Myers' framing in the ISWorld Qualitative Research Section" (Straub *et al.*, 2004).

Figure 3.4.1: Epistemological assumptions underlying qualitative research (Straub *et al.*, 2004)

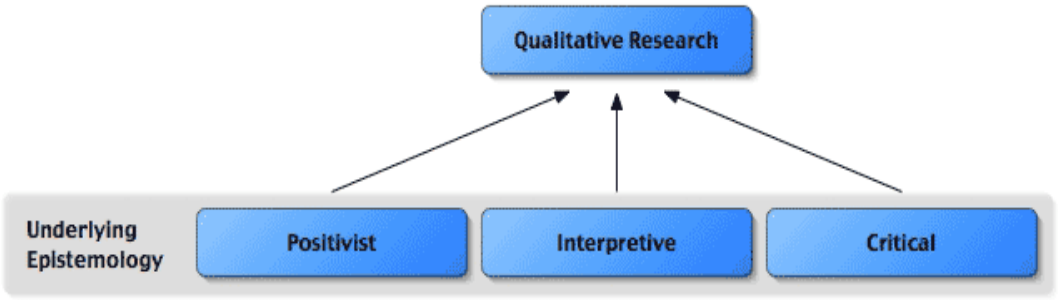
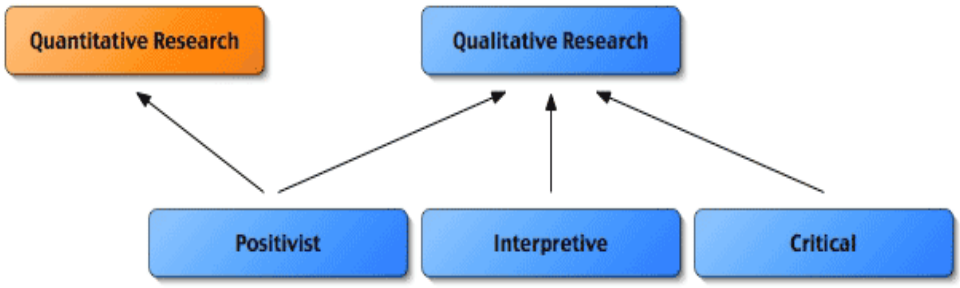


Figure 3.4.1 clearly shows how, for qualitative research specifically, the basic epistemological positions to choose from are threefold, referring to positivistic, interpretive, or critical methods. However, with reference to quantitative research, the interpretive and critical positions are not applicable or meaningful, only the positivist paradigm applies to quantitative research.

Therefore the positivist epistemology relies on a vast amount of scientific methods which produce numerical and alphanumeric data.

These assumptions for both quantitative and qualitative research can be represented as in Figure 3.4.2

Figure 3.4.2: Epistemological assumptions underlying qualitative and quantitative research (Straub *et al*,. 2004)



3.5 Implementation for this study

To ensure an objective perspective on the subject matter as well as scientifically accurate measurements, positivism was chosen as the ideal research paradigm. To reach the largest audience, the survey method was used as research strategy and a questionnaire was compiled with special attention as to the right set of questions to be asked.

The questionnaire mainly consisted of close-ended questions, using the five-point Likert scale, as it is the most frequently used response scale (Conger, 1994). The measurability of close-ended questions is much higher as the responses can be quantified and analysed. The majority of these questions were derived from instruments developed by literature by Huisman, and Chow and Cao (Huisman, 2000; Chow & Cao, 2007).

Questionnaires were distributed to over 25 companies and 175 individuals to collect data. These companies and respondents were chosen due to their background and knowledge in their respective fields and most of the chosen companies were implementing ASDMs. Of these 175 individuals, there were 129 respondents, resulting in a response rate of 74%. The questionnaires were mainly distributed by hand and some through email. The companies who did not respond within one month from the date of distribution were contacted to remind them of the questionnaire. The vast majority of questionnaires were distributed by hand and collected via face-to-face meetings with the various respondent companies. The distribution of the questionnaires was closely monitored and a stringent follow-up

process was followed. Companies who returned 10 or more surveys were promised an incentive as a token of appreciation for their participation and additional effort to help with this study. This process resulted in a very high response rate. *Table* 3.5.1 below summarizes the number of surveys returned by each company:

***Table* 3.5.1 – Response rate per company**

	N questionnaires returned	**% of Total**
Company 1	2	1.55%
Company 2	1	0.78%
Company 3	4	3.10%
Company 4	3	2.33%
Company 5	15	11.63%
Company 6	8	6.20%
Company 7	4	3.10%
Company 8	25	19.38%
Company 9	21	16.28%
Company 10	2	1.55%
Company 11	6	4.65%
Company 12	8	6.20%
Company 13	1	0.78%
Company 14	6	4.65%
Company 15	5	3.88%
Company 16	11	8.53%
Company 17	7	5.43%

Due to the fact that anonymity was promised in the survey, the companies cannot be named.

The questions targeted different aspects of each individual with the main focus being their perceptions regarding the critical success factors of agile systems development projects . Background information such as profession and the sector of the particular organisation formed part of the first section of the questionnaire.

In section 2 the use of system development methodologies (which focuses on the end-product) and project management methodologies (which focuses on the systems development process) were measured. In section 3, 43 critical success factors in IT projects were measured. The project outcome was finally measured in section 4 to establish how effective the systems development methodology and project management methodology being implemented by that particular company is. The results were measured using statistical methods and Statistica software version 10 on which the recommendations are based.

3.6 Conclusion

The main viewpoint is that positivism can be seen as an essential tool in the arsenal of any researcher although it should not be viewed as an exclusive tool.

Chapter 4
Research results and analysis

4.1 Introduction

Research conducted into the critical success factors in agile systems development projects can be centred on a large number of research approaches. These research approaches are dependent on the research question, the research target, the theoretical direction of the study and the research method used.

In chapter 1, the problem statement was outlined which identified some important inadequacies in the field of systems development in general. This was followed by chapter 2 where several critical success factors were identified and agile systems development methodologies were discussed in detail as well as project management methodologies. In chapter 3 the research method of the study was presented.

The purpose of this chapter is to present the research results. Firstly, the research method will be discussed with regard to the development and testing of the questionnaire, followed by a discussion of the measurement of the different research variables for investigating the critical success factors in agile systems development projects will be presented. This will be followed by the various statistical results generated from the surveys. Special attention will be given to the reliability of the research variables.

4.2 Research method

4.2.1 Development and testing of the questionnaire

For a survey to produce successful results, it must contain the right set of questions that are asked in the right way. In order to decide on what information should be captured by the questionnaire, the research problem, research questions and conceptual research model of the study were considered. A literature study was performed in order to identify validated instruments that could be used for capturing data.

A single questionnaire was developed for gathering the data. The questionnaire was structured in 4 sections. Section 1 gathered some background information regarding the IS department. Section 2 gathered information about the systems development methodology and project management methodology used in the IS department. Section 3 gathered information regarding the critical success factors in IS projects and section 4 measured the success of the process and project respectively.

The information captured by the questionnaire is summarized in Table 4.2.1.1 below:

Table 4.2.1.1 – Information captured by questionnaire

Variable	**Questionnaire**	**Number of items measured**
Background information		
Individual role	✓	1
Education level	✓	1
Systems development experience	✓	1
Organizational sector	✓	1
Task delegation	✓	1
Systems development methodology		
Methodology used - Type - Intensity of use	✓	Respondent specific Intensity was measured on a 5-point Likert scale
Period of use	✓	1
Horizontal use of SDM	✓	2 (Huisman, 2000)
Strictness of use	✓	1
Project management methodology		
Methodology used - Type - Intensity of use	✓	Respondent specific Intensity was measured on a 5-point Likert scale

Period of use	✓	1
Horizontal use	✓	2
Strictness of use	✓	1
Critical success factors		
Various success factors involved in IS projects	✓	43 items measured on a 5-point Likert scale. These items were adopted from the work of Chow and Cao (2007)
Project Outcome		
Size of project	✓	1
Product outcome	✓	1
Product success	✓	8 items measured on a 5-point Likert scale (Huisman, 2000)
Process success	✓	11 items measured on a 5-point Likert scale (Huisman, 2000)

4.2.2 Data collection

The survey was conducted across South Africa between July and November 2011. Multiple leading organizations in the IT industry were contacted via telephone, email and face to face meetings. The questionnaires were distributed with a cover letter describing the purpose of the study and guaranteeing the confidentiality of the respondents. Organizations that did not respond within the deadline were sent a reminder. The profiles of the participating organizations and individual respondents are summarised in Table 4.2.2.1 and Table 4.2.2.2 respectively.

Table 4.2.2.1 – Business area of respondent's organisation

	N	**%**
Business area		
System/Software development	27	20.77
Telecommunication	2	1.53
Manufacturing	12	9.23
Mining	1	0.77
Consulting	22	16.9
Financial/Banking/Insurance	49	37.7
Government	9	6.93
Other	7	5.4
Missing	1	0.77

The majority (37,7%) of the participants came from the financial sector and another 20,77 percent from system/software development. Nearly 17 percent of all respondents indicated consulting as their business area.

***Table* 4.2.2.2 - Respondent profile**

	N	**%**
Education		
Senior certificate (High School)	3	2.3
Certificate or diploma	75	57.7
University or Technikon degree	38	29.23
Honours or Master's degree	12	9.23
PhD degree	1	0.77
Other	0	0
Missing	1	0.77
Title		
IS manager	14	10.77
Project manager	7	5.4
Team leader	14	10.77
Systems analyst	9	6.92
Analyst/Programmer	17	13.1
Programmer	20	15.35
Other	48	36.92
Missing	1	0.77
Experience in systems development		
None	1	0.77
Less than 1 year	21	16.16
1-2 years	34	26.16
3-5 years	19	14.6
5-10 years	18	13.84
More than 10 years	36	27.7
Missing	1	0.77

An astounding 96,16 percent of the respondents had completed further education in the form of a certificate or diploma, a university or technikon degree, or an Honours or Master's degree. A total of 38,46 percent of the respondents had completed their

tertiary education at a technikon or university. These figures clearly indicate that the respondents are highly qualified.

Fifteen percent of all respondents stated programmer as their title, whilst thirteen percent stated analyst/programmer. A total of 27 percent of the respondents are managers or team leaders and 37 percent stated *other* as their title.

Nearly one-third of the respondents have more than 10 years of experience in their respective fields, fourteen percent have 5 to 10 years of experience and nearly fifteen percent have 3 to 5 years of experience. These figures indicate that the vast majority of the respondents are highly skilled.

***Table* 4.2.2.3 – Respondent's work delegation**

	Valid N	**Mean %**
Activities		
Systems planning, analysis and design	129	24
Programming	129	21
Testing	129	12
Installation	129	12
User training	129	4
Project management	129	9
Other	129	18

In *Table* 4.2.2.3 we can see that 24 percent of the respondents are performing systems planning, analysis and design as part of their everyday tasks whilst 21 percent are involved in programming tasks. Only 9 percent of the respondents are performing project management tasks from day to day.

4.2.3 Measurement of research variables

This study implemented a survey and collected data from IT professionals across all kinds of organisations. The collected data was analysed using Statistica (version 10) software.

4.2.4 The use of systems development methodologies

To measure the use of systems development methodologies, we focused on the vertical use along with the type of SDM use, horizontal use, the strictness of use, and for how long the specific methodology has been in use.

Vertical use of SDM

Vertical use is concerned with which components of the methodology are being used and to what extent they are used, in other words, the intensity of use of systems development methodologies. To measure vertical use, the respondents were presented with examples of traditional- and agile systems development methodologies, and they were asked to name the SDM used in their organization, and to indicate the intensity of use in their IS department using these methods (if any) on a scale of 1 (very infrequently) to 5 (very often). Since the possibility exists that an IS department can use multiple methodologies, the maximum of all indicated methods was selected to form vertical use of SDM. *Table 4.2.4.1* below indicates that XP, Waterfall, IE and Object-orientated are the most popular SDMs.

Table 4.2.4.1 - Systems development methodologies

	Total N	**Mean**	**%**
Methodology			
Object-orientated	15	3.33	10.14
Waterfall	20	3.7	13.5
Scrum	2	3	1.35
XP	24	3.67	16.22
Prototype	5	2.6	3.38
RAD	8	4.75	5.41
Agile	9	3.44	6.08
IE	17	3.53	11.49
Own	11	4.64	7.43
None	37	4.97	25

As stated in previous chapters, we can differentiate between traditional and agile systems development methodologies. The list of traditional methodologies that form part of this study include Waterfall, Object-orientated and IE (Information Engineering). The following systems development methodologies are considered to be Agile: Prototyping, Scrum, XP, RAD and Agile. A total of 32.44 percent of the respondents are implementing ASDMs whereas 35.13 percent still use traditional systems development methodologies. A quarter of the respondents do not use any form of systems development methodologies and 7.43 percent use their own systems development methodology.

Table 4.2.4.2 – Type of Systems Development Methodologies used

	Total N applied	**%**
Methodology		
Agile	48	48
Traditional	52	52

A total of 48 percent of respondents are applying ASDMs whereas 52 percent are applying traditional systems development methodologies.

Horizontal use of SDM

The horizontal use is concerned with how widely the systems development methodology is being used across the organisation. The horizontal use was calculated by determining the average of two items, namely the proportion of projects developed in the IS department by applying systems development methodology knowledge and the proportion of people in the IS department that apply systems development methodology knowledge regularly.

Table 4.2.4.3 – Proportion of projects applying SDM knowledge

	Total N applied	**%**
Proportion		
None	21	16.28
1 – 25%	9	6.98
26 – 50%	40	31
51 – 75%	28	21.71
Over 75%	31	24.03

Table 4.2.4.3 above shows the proportion of projects to which SDM knowledge is applied in every respondent's organisation.

Table 4.2.4.4 – Proportion of people applying SDM knowledge

	Total N applied	**%**
Proportion		
None	19	14.73
1 – 25%	12	9.3
26 – 50%	39	30.23
51 – 75%	34	26.36
Over 75%	25	19.38

Table 4.2.4.4 above shows the proportion of people who apply SDM knowledge in every respondent's organisation

Table 4.2.4.5 – Horizontal use of SDM

	Total N applied	**%**
Proportion		
None	19	14.73
1 – 25%	9	6.98
26 – 50%	36	27.91
51 – 75%	35	27.13
Over 75%	30	23.25

Table 4.2.4.5 above shows the horizontal use in the respondents' organisations

The majority of respondents are implementing their SDM knowledge widely across their respective organisation. Over 50 percent of the respondents are implementing their SDM knowledge more than 51 percent of the time. Only 14.73 percent of the respondents aren't implementing their respective SDM widely across their organisation.

Strictness of use of SDM

The strictness of use measures how rigorously the systems development methodology is followed in the organisation.

***Table* 4.2.4.6 - Systems Development Methodologies - strictness of use**

	Total N applied	**% guideline**	**% adapted**	**% rigorous**
Methodology				
Agile	45	40	46.67	13.33
Traditional	35	40	37.14	22.86

As seen from the *Table* 4.2.4.6 above, most respondents adapt their respective systems development methodology on a project-to-project basis or use it as a guideline. Almost 50 percent of the time, agile organisations adapt their respective methodology on a project-to-project basis, whereas traditional organisations adapt 37 percent of the time. Mostly traditional systems development methodologies are rigorously followed. Agile systems development methodologies are still being adapted on a frequent basis, eventhough they were designed to be flexible.

Time of use of SDM

This measurement is concerned with measuring how long the specific systems development methodology has been in use within the organisation.

Table 4.2.4.7 – Systems Development Methodologies- Time of use

	Total N applied	**Less than 3 years %**	**3-5 years %**	**5–10 years %**	**More than 10 years %**
Methodology					
Agile	33	33.33	30.3	24.24	12.12
Traditional	22	13.64	31.82	31.82	22.73

From the *Table* 4.2.4.7 above we can derive that most of the respondents' implementations have been in place for less than 3 years or between 3 and 5 years. Respondents implementing traditional systems development methodologies have been using their respective methodology for longer periods of time.

4.2.5 The use of project management methodologies

To measure the use of project management methodologies, we focused on the vertical use, horizontal use, the strictness of use, and for how long the specific methodology has been in use.

Vertical use of PMM

Vertical use is concerned with which components of the methodology are being used and to what extent they are used, in other words, the intensity of use of project management methodologies. To measure vertical use, the respondents were presented with examples of project management methodologies, and they were asked to name the PMM used in their organization, and to indicate the intensity of use in their IS department using these methods (if any) on a scale of 1 (very infrequently) to 5 (very often). Since the possibility exists that an IS department can use multiple methods, the maximum of all indicated methods was selected to form vertical use of PMM.

***Table* 4.2.5.1 - Project Management Methodologies - vertical use**

	Total N	**%**	**Mean**
Methodology			
PMBOK	23	25.56	3.17
PRINCE2	67	74.44	3.45

A quarter of the respondents are using the PMBOK methodology and nearly 75 percent are using PRINCE2.

Horizontal use of PMM

The horizontal use is concerned with how widely the project management methodology is being used across the organisation. The horizontal use was calculated by determining the average of two items, namely the proportion of projects developed in the IS department by applying project management methodology knowledge and the proportion of people in the IS department that apply project management methodology knowledge regularly.

Table 4.2.5.2 – Proportion of projects applying PMM knowledge

	Total N applied	**%**
Proportion		
None	17	13.18
1 – 25%	17	13.18
26 – 50%	24	18.6
51 – 75%	33	25.58
Over 75%	38	29.46

Table 4.2.5.2 above shows the number of projects to which PMM knowledge is applied in every respondent's organisation. PMM knowledge is applied to more than 55 percent of projects that respondents were involved in.

Table 4.2.5.3 – Proportion of people applying PMM knowledge

	Total N applied	**%**
Proportion		
None	20	15.5
1 – 25%	16	12.4
26 – 50%	44	34.11
51 – 75%	27	20.93
Over 75%	22	17.05

Table 4.2.5.3 above shows the number of people who apply PMM knowledge in every respondent's organisation. In the majority of respondent's organisations, the number of people applying PMM knowledge are between the 26 – and 75 percent bracket.

Table 4.2.5.4 - Project Management Methodologies - Horizontal use

	Total N applied	**%**
Proportion		
None	17	13.18
1 – 25%	15	11.63
26 – 50%	24	18.6
51 – 75%	42	32.56
Over 75%	31	24.03

Table 4.2.5.4 above shows the horizontal use of PMMs in respondents' organisations. Over 56 percent of respondents are implementing PMM knowledge widely across their respective organisation, whilst 13.18 percent of respondents aren't implementing PMM knowledge widely across their organisation.

Strictness of use of PMM

The strictness of use measures how rigorously the project management methodology is followed in the organisation.

***Table* 4.2.5.5 - Project Management Methodologies - Strictness of use**

	Total N applied	**% guideline**	**% adapted**	**% rigorous**
Methodology				
PMBOK	23	21.74	47.83	30.43
PRINCE2	66	33.33	36.36	30.30

As seen from the Table above, most respondents adapt their respective project management methodology on a project-to-project basis. In the cases where respondents were using PMBOK, nearly 48 percent of their projects are adapted and in cases where respondents were using PRINCE2, 36 percent of their projects are adapted. In both cases where PMBOK or PRINCE2 are used, nearly a third of the projects apply their respective project management methodology knowledge rigorously.

Time of use of PMM

This measurement is concerned with measuring how long the specific project management methodology has been in use within the organisation.

Table 4.2.5.6 - Project Management Methodologies - Time of use

	Total N applied	**Less than 3 years %**	**3-5 years %**	**5–10 years %**	**More than 10 years %**
Methodology					
PMBOK	23	23.8	19.05	38.1	19.05
PRINCE2	66	17.24	20.69	24.14	37.93

From Table 4.2.5.6 above, we can derive that more than 57 percent of the respondents using PMBOK, have been using it for more than 5 years. More than half of the respondents using PRINCE2, have been using it for more than 5 years. A total of almost 38% of respondents implementing PRINCE2, have been using it for more than 10 years.

4.2.6 Measurement and reliability of critical success factor variables

The measurement of various critical success factors in information technology projects (section 3 of the questionnaire) was done using a multitude of items. This resulted in a large data set, which was reduced by performing factor analysis using the principal-components method with varimax normalised rotation on the data. The rudimentary idea behind factor analysis is to express two or more variables through a single factor. In order to determine the number of factors to retain, the Kaiser criterion was used. This criterion is widely used and states that only factors with Eigen values (variances extracted by each factor) greater than 1 are retained. The resulting number of factors is smaller than the initial number of items that were used to measure the research variables. For each item, its factor loadings on the different factors were analysed, and the item was grouped together with the factor where it had the highest loading.

The reliability of these factors is very important. Each factor should be a reliable measure of its corresponding research variable. Reliability is the extent to which an instrument is free of measurement errors (Conger, 1994). A measurement is reliable if it mostly reflects a true score, relative to the error. This means that if a reliable instrument were given to the same group of people on multiple occasions, the same answers would be obtained. Reliability analysis was performed on the items of each of the factors identified using Cronbach's coefficient alpha as the index of reliability, because it is used most commonly. If all items in the factor are perfectly reliable and measure the same thing, then Cronbach's coefficient alpha is equal to 1. There is no exact threshold for reliability, but researchers have resorted to using a Cronbach alpha of 0,6 to estimate reliability (Roberts *et al*., 1998; Huisman, 2000). For exploratory research studies, it is agreed that a coefficient alpha level of 0,5 could be deemed acceptable (Nunally, 1967).

Due to the lack of empirical research on the critical success factors in information technology projects and the exploratory nature of this study, a value of 0,5 will be used for Cronbach's alpha in conjunction with an inter-item correlation between the range of 0,2 and 0,55 (Clark & Watson, 1995).

After performing the reliability analysis, the items of each factor were studied and an explanatory name was created to represent the factor.

4.2.7 Critical success factors

During the study, a total of 43 critical success factors were presented. *Table* 4.2.7.1 below depicts the mean results for these 43 critical success factors:

Table 4.2.7.1 – Critical success factors in information technology projects

Total N applied = 129						
Critical Success Factor	**1**	**2**	**3**	**4**	**5**	**Mean**
Strong executive support	5	21	36	41	26	3,48
Committed sponsor or manager	2	14	50	44	19	3,5
Cooperative organizational culture instead of hierarchal	10	31	41	38	9	3,04
Oral culture placing high value on face-to-face communication	6	15	30	42	36	3,67
Agile methodology is universally accepted in the organization	10	40	48	23	8	2,84
Collocation of the whole team	3	10	52	47	17	3,5
Facility with proper agile-style work environment	7	39	44	26	13	2,99
Reward system appropriate for the agile process	15	24	45	37	8	2,99
Team members with high competence and expertise	0	2	36	70	21	3,85
Team members with great motivation	2	5	26	66	30	3,91
Managers knowledgeable in the agile process	7	27	51	35	9	3,09
Managers who have light-touch or adaptive management style	5	11	53	56	4	3,33
Coherent, self-organizing teamwork	0	8	43	67	11	3,63
Good customer relationship	9	7	25	56	32	3,74
Following agile-oriented requirement management process	12	42	49	23	3	2,71
Following agile-oriented project	13	35	49	26	6	2,82

management process						
Following agile-oriented configuration management process	12	40	48	22	7	2,78
Strong communication focus with daily face-to-face meetings	7	15	30	46	31	3,61
Honouring regular working schedule - no overtime	20	30	42	34	3	2,77
Strong customer commitment and presence	4	13	40	52	20	3,55
Customer having full authority	23	35	44	23	4	2,61
Well-defined coding standards up front	6	12	51	50	10	3,36
Pursuing simple design	4	8	46	66	5	3,47
Rigorous refactoring activities	6	30	53	36	4	3,02
Right amount of documentation	5	31	47	33	13	3,14
Regular delivery of software	2	15	39	61	12	3,51
Delivering most important features first	5	7	35	74	8	3,57
Correct integration testing	0	8	39	67	15	3,69
Appropriate technical training to team	2	13	43	59	12	3,51
Project nature being non-life-critical	13	14	58	39	5	3,07
Project type being of variable scope with emergent requirement	0	16	73	38	2	3,2
Projects with dynamic, accelerated schedule	2	8	74	41	4	3,29
Projects with small team	0	20	49	46	14	3,42
Projects with no multiple independent teams	6	23	51	43	6	3,16
Projects with up-front cost evaluation done	6	23	37	53	10	3,29
Projects with up-front risk analysis done	5	20	58	38	8	3,19
The project could handle unexpected crises and deviations from plan	4	8	44	65	8	3,5
The project was efficiently coordinated and successfully integrated activities, documents and personnel	1	14	61	42	11	3,37

Sufficient testing was done during the project	2	9	42	66	10	3,57
The project had clearly defined business priorities	4	10	48	56	11	3,47
The project had a clear vision and objectives	3	6	46	57	17	3,61
Project management was good	4	7	47	60	11	3,52
The project applied appropriate systems development methodology knowledge.	4	15	51	51	8	3,34

The Table below represents the top 10 critical success factors collected from the survey in descending order:

Table 4.2.7.2 – Top 10 critical success factors in information technology projects

Rating	Critical Success Factor	Mean
1	Team members with great motivation	3,91
2	Team members with high competence and expertise	3,85
3	Good customer relationship	3,74
4	Correct integration testing	3,69
5	Oral culture placing high value on face-to-face communication	3,67
6	Coherent, self-organizing teamwork	3,63
7	Strong communication focus with daily face-to-face meetings	3,61
7	The project had a clear vision and objectives	3,61
8	Sufficient testing was done during the project	3,57
9	Strong customer commitment and presence	3,55
10	Project management was good	3,52

From these results in Table 4.2.7.2 above, we can see that the most critical success factor is team members with great motivation, closely followed by team members with high competence and expertise. From these results it seems clear that the most

critical part of an information technology project lies with the team members involved in the project. Face-to-face communication was also rated as one of the most critical success factors, which is one of the key attributes in most agile systems development methodologies. The importance of project management can also be seen from this top 10, emphasizing the significance of using a project management methodology.

Table 4.2.7.3 below depicts the lowest 10 critical success factors:

Table 4.2.7.3 –Lowest 10 critical success factors in information technology projects

Rating	Critical Success Factor	Mean
1	Customer having full authority	2,61
2	Following agile-oriented requirement management process	2,71
3	Honouring regular working schedule - no overtime	2,77
4	Following agile-oriented configuration management process	2,78
5	Following agile-oriented project management process	2,82
6	Agile methodology is universally accepted in the organization	2,84
7	Reward system appropriate for the agile process	2,99
7	Facility with proper agile-style work environment	2,99
8	Rigorous refactoring activities	3,02
9	Cooperative organizational culture instead of hierarchal	3,04
10	Project nature being non-life-critical	3,07

From Table 4.2.7.3 above, we can see that the least critical success factor in information technology projects is the customer having full authority, closely followed

by the following of an agile-oriented requirement management process and the honouring of a regular working schedule with no overtime.

After performing a statistical factor analysis, these 43 critical success factors were narrowed down to 16. The summary below shows the factor loadings and how each new factor was calculated:

Factor 1 – Environment suitable for Agile

Cronbach Alpha: 0.884	**Average inter-item correlation: 0.587**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Agile methodology is universally accepted in the organization	.445	0.883
Facility with proper agile-style work environment	.645	0.866
Managers knowledgeable in the agile process	.668	0.880
Following agile-oriented requirement management process	.837	0.858
Following agile-oriented project management process	.850	0.844
Following agile-oriented configuration management process	.842	0.851

Factor 2 – Strong customer involvement

Cronbach Alpha: 0.520	**Average inter-item correlation: 0.266**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Good customer relationship	.670	0.350
Strong customer commitment and presence	.509	0.456
Customer having full authority	.301	0.443

Factor 3 – Appropriate management of the Agile process with satisfactory amount of documentation

Cronbach Alpha: 0.720	**Average inter-item correlation: 0.472**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Right amount of documentation	.579	0.651
The project was efficiently coordinated and successfully integrated activities, documents and personnel	.740	0.567
Project management was good	.470	0.678

Factor 4 – Reward system appropriate for the Agile process

Original factor number	**Loading**
Reward system appropriate for the agile process	Not applicable

Factor 5 – Frequent delivery of usable software meeting frequently changing requirements

Cronbach Alpha: 0.703	**Average inter-item correlation: 0.393**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Regular delivery of software	.580	0.593
Delivering most important features first	.486	0.731
Project type being of variable scope with emergent requirement	.787	0.603
Projects with dynamic, accelerated schedule	.681	0.623

Factor 6 – Proper planning and emphasis on face-to-face communication

Cronbach Alpha: 0.733	**Average inter-item correlation: 0.436**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Oral culture placing high value on face-to-face communication	.599	0.703
Strong communication focus with daily face-to-face meetings	.602	0.691
Projects with up-front cost evaluation done	.767	0.652
Projects with up-front risk analysis done	.780	0.648

Factor 7 – Project had a clear vision and was well driven by executives

Cronbach Alpha: 0.776	**Average inter-item correlation: 0.491**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Strong executive support	.772	0.747
Committed sponsor or manager	.740	0.712
The project had clearly defined business priorities	.738	0.710
The project had a clear vision and objectives	.622	0.723

Factor 8 – Adaptive manager and project team

Cronbach Alpha: 0.581	**Average inter-item correlation: 0.324**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Managers who have light-touch or adaptive management style	.791	0.583
Coherent, self-organizing teamwork	.597	0.389
The project could handle unexpected crises and deviations from plan	.428	0.469

Factor 9 – Competent team in one location

Cronbach Alpha: 0.524	**Average inter-item correlation: 0.367**
Original factor number	**Loading**
Collocation of the whole team	-.675
Team members with high competence and expertise	-.728

Factor 10 – Sufficient integration testing

Cronbach Alpha: 0.687	**Average inter-item correlation: 0.524**
Original factor number	**Loading**
Correct integration testing	.403
Sufficient testing was done during the project	.428

Factor 11 – Suitable team size

Cronbach Alpha: 0.557	**Average inter-item correlation: 0.387**
Original factor number	**Loading**
Projects with small team	-.585
Projects with no multiple independent teams	-.804

Factor 12 – Good design practices and technical knowledge applied to the project

Cronbach Alpha: 0.730	**Average inter-item correlation: 0.406**	
Original factor number	**Loading**	**Cronbach alpha if deleted**
Well-defined coding standards up front	.614	0.650
Pursuing simple design	.732	0.652
Appropriate technical training to team	.658	0.674
The project applied appropriate systems development methodology knowledge.	.419	0.701

Factor 13 – Regular schedule was honoured with no overtime

Original factor number	**Loading**
Honouring regular working schedule - no overtime	Not applicable

Factor 14 – Non-life-threatening project with no refactoring activities

Cronbach Alpha: 0.561	**Average inter-item correlation: 0.391**
Original factor number	**Loading**
Rigorous refactoring activities	.514
Project nature being non-life-critical	.700

Factor 15 – Cooperative organizational culture instead of hierarchical

Original factor number	**Loading**
Cooperative organizational culture instead of hierarchal	Not applicable

Factor 16 – Team members with great motivation

Original factor number	**Loading**
Team members with great motivation	Not applicable

Table 4.2.7.4 below depicts the reliability of these factors with their respectable Cronbach alphas and average inter-item correlations. All Cronbach alphas and average inter-item correlations meet the standards that were stipulated earlier in this chapter.

***Table* 4.2.7.4 – Critical success factors reliability measures**

Factor	**Cronbach Alpha**	**Average inter-item correlation**
1. Environment suitable for Agile	.884	.587
2. Strong customer involvement	.520	.266
3. Appropriate management of the Agile process with satisfactory amount of documentation	.720	.472
4. Reward system appropriate for the Agile process	n/a	n/a
5. Frequent delivery of usable software meeting frequently changing requirements	.703	.393
6. Proper planning and emphasis on face-to-face communication	.733	.436
7. Project had a clear vision and	.777	.491

was well driven by executives		
8. Adaptive manager and project team	.581	.324
9. Competent team in one location	.524	.367
10. Sufficient integration testing	.687	.524
11. Suitable team size	.557	.387
12. Good design practices and technical knowledge applied to the project	.730	.406
13. Regular schedule was honoured with no overtime	n/a	n/a
14. Non-life-threatening project with no refactoring activities	.561	.391
15. Cooperative organizational culture instead of hierarchical	n/a	n/a
16. Team members with great motivation	n/a	n/a

Table 4.2.7.5 below depicts the descriptive statistics for the various critical success factors. Both the mean and the standard deviation are documented.

Table 4.2.7.5 – Critical success factors descriptive statistics

Factor	Valid N	Mean	Standard Deviation
1. Environment suitable for Agile	129	2.87	0.802
2. Strong customer involvement	129	3.30	0.752
3. Appropriate management of the Agile process with satisfactory amount of documentation	129	3.34	0.722
4. Reward system appropriate for the Agile process	129	2.99	1.093
5. Frequent delivery of usable software meeting frequently changing requirements	129	3.39	0.565
6. Proper planning and emphasis on face-to-face communication	129	3.44	0.783
7. Project had a clear vision and was well driven by executives	129	3.51	0.732
8. Adaptive manager and project team	129	3.49	0.589
9. Competent team in one location	129	3.68	0.664
10. Sufficient integration testing	129	3.63	0.680
11. Suitable team size	129	3.29	0.755

12. Good design practices and technical knowledge applied to the project	129	3.41	0.644
13. Regular schedule was honoured with no overtime	129	2.77	1.079
14. Non-life-threatening project with no refactoring activities	129	3.33	0.523
15. Cooperative organizational culture instead of hierarchical	129	3.04	1.064
16. Team members with great motivation	129	3.71	0.661

4.2.8 Project outcome (effectiveness)

The final section of the questionnaire measured the effectiveness of the system development process, as well as the quality of the product delivered (outcome). Table 4.2.8.1 below shows the descriptive analysis on the various sizes of projects that users were involved with.

Table 4.2.8.1 – Project size

	N	**%**
Project size		
Small	8	6.2
Medium	31	24.03
Large	52	40.31
Very large	35	27.13
User did not participate in any projects	3	2.33

The *Table* above shows that the majority of projects were large or very large. Table 4.2.8.2 below shows the various outcomes of these projects.

***Table* 4.2.8.2 – Project outcome**

	N	**%**
Project outcome		
Project was cancelled/terminated before completion	20	15.5
Project was completed but not implemented	13	10.08
Project was completed and implemented but is no longer in use	9	6.98
Project was completed and implemented and is still in use	87	67.44

Almost 70 percent of projects were completed and implemented and is still in use by the organisation. There is still much room for improvement based on the fact that 15.5 percent of projects were cancelled/terminated before completion. Another 10 percent of projects were completed but not implemented.

4.2.9 Success of the process

This section of the questionnaire measured the success of the system development process. If the project with which the respondent was involved with was cancelled before completion, this section could not be completed.

A total of eight success statements were presented to the user, after which a statistical factor analysis was performed to create two factors. These factors as well as their reliability and factor loadings are depicted in the summary below:

Factor 1 – Process and Project Management Quality

Cronbach Alpha: 0.813	**Average inter-item correlation: 0.478**	
Original Statement	**Factor Loading**	**Cronbach alpha if deleted**
The project was completed on schedule	0.703	0.791
The project was completed within the budget	0.860	0.733
The developed system satisfied all the stated requirements	0.719	0.768
The speed of developing the project was high	0.493	0.811
The cost of the project is low when compared to the size and complexity of the system developed	0.712	0.773

Factor 2 – Excellence of the project

Cronbach Alpha: 0.897	**Average inter-item correlation: 0.749**	
Original Statement	**Factor Loading**	**Cronbach alpha if deleted**
The project achieved its goals	0.880	0.850
Overall, the project represents excellent work	0.893	0.864
Overall, the project was a success	0.874	0.845

Table 4.2.9.1 below depicts the reliability of these factors with their respectable Cronbach alphas and average inter-item correlations.

Table 4.2.9.1 – Process success reliability measures

Factor	Cronbach Alpha	Average inter-item correlation
1 - Process and Project Management Quality	0.813	0.478
2 - Excellence of the Project	0.897	0.749

Table 4.2.9.2 below shows the descriptive statistics for the two factors mentioned above. These factors were measured on a scale of 1 to 5 where 1 is bad and 5 is very good.

Table 4.2.9.2 – Process success descriptive statistics

Factor	Valid N	Mean	Standard Deviation
1 - Process and Project Management Quality	108	3.16	0.790
2 - Excellence of the Project	108	3.86	0.681

Table 4.2.9.2 above shows that the excellence of the projects that respondents were involved with is relatively high, with a mean of 3.86. The process and project management quality was almost equally as good, with a mean of 3.16.In the next section the correlation statistics are presented. Correlations were investigated between the variables that are presented in *Table* 4.2.9.3 below and the process success variables, namely process and project management quality, and excellence of the project.

Table 4.2.9.3 - Correlation statistics on process success

	1 - Process and Project Management Quality	2 - Excellence of the Project
Agile Methodology was applied	-0.003	0.018
PMBOK Methodology was	0.117	-0.013

applied		
PRINCE2 Methodology was applied	-0.234	-0.151
1. Environment suitable for agile	0.311	0.239
2. Strong customer involvement	0.361	0.319
3. Appropriate management of the agile process with satisfactory amount of documentation	0.550	0.249
4. Reward system appropriate for the agile process	-0.236	-0.041
5. Frequent delivery of usable software meeting frequently changing requirements	0.315	0.106
6. Proper planning and emphasis on face-to-face communication	-0.300	-0.155
7. Project had a clear vision and was well driven by executives	0.329	0.067
8. Adaptive manager and project team	0.017	0.149
9. Competent team in one location	0.163	0.217
10. Sufficient integration testing	0.234	0.149
11. Suitable team size	0.259	0.040
12. Good design practices and technical knowledge applied to the project	0.132	0.152
13. Regular schedule was honoured with no overtime	0.175	-0.090
14. Non-life-threatening project with no refactoring activities	0.164	0.036
15. Cooperative organizational culture instead of hierarchical	0.388	0.160
16. Team members with great motivation	0.039	0.163

The marked correlations in red from *Table* 4.2.9.3 are significant where the p-value, which is being reported, is smaller than 0.05. It is clear that the application of the

PRINCE2 project management methodology correlated with the process and project management quality. An environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation, a reward system appropriate for the agile process, frequent delivery of usable software meeting frequently changing requirements, proper planning and emphasis on face-to-face communication, project with a clear vision and well driven by executives, sufficient integration testing, suitable team size, and a cooperative organizational culture instead of hierarchical were all critical success factors which positively impacted the process and project management quality.

An environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation and a competent team in one location all had a positive effect on the excellence of the project.

The next section of this chapter will report the regression statistics with regard to the process success. The variables that had a meaningful impact in the correlations mentioned above were used to perform regression. Firstly a standard regression was performed, followed by forward - and backward stepwise regressions.

Table 4.2.9.4 - Standard regression - Process and Project Management Quality

N = 53	**Multiple R = 0.780 Multiple R^2 = 0.609 Adjusted R^2 = 0.504** **F(11,41) = 5.794 p < 0.00001 Standard error of estimate = 0.559**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(41)**	**p-value**
Intercept			2.254	1.306	1.725	0.092
PRINCE2	-0.108	0.136	-0.082	0.103	-0.797	0.430
1. Environment suitable for agile	-0.042	0.129	-0.050	0.151	-0.327	0.745
2. Strong customer	0.127	0.117	0.137	0.127	1.078	0.287

involvement						
3. Appropriate management of the agile process with satisfactory amount of documentation	0.669	0.148	0.946	0.209	4.520	0.00005
4. Reward system appropriate for the agile process	-0.113	0.123	-0.086	0.094	-0.918	0.364
5. Frequent delivery of usable software meeting frequently changing requirements	-0.009	0.121	-0.016	0.218	-0.073	0.942
6. Proper planning and emphasis on face-to-face communication	-0.169	0.131	-0.212	0.165	-1.286	0.206
7. Project had a clear vision and was well driven by executives	-0.056	0.133	-0.067	0.158	-0.426	0.672
10. Sufficient integration testing	-0.108	0.157	-0.141	0.205	-0.690	0.494
11. Suitable team size	-0.211	0.139	-0.242	0.160	-1.517	0.137
14. Non-life-threatening	0.042	0.127	0.070	0.213	0.327	0.745

project with no refactoring activities						

Table 4.2.9.5 - Forward stepwise regression - Process and Project Management Quality

N = 53	**Multiple R = 0.765 Multiple R^2 = 0.587 Adjusted R^2 = 0.543** **F(5,47) = 13.34 p < 0.0000 Standard error of estimate = 0.537**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(47)**	**p-value**
Intercept			1.991	0.781	2.551	0.014
3. Appropriate management of the agile process with satisfactory amount of documentation	0.677	0.109	0.959	0.154	6.239	0.000
6. Proper planning and emphasis on face-to-face communication	-0.160	0.113	-0.201	0.143	-1.408	0.166
11. Suitable team size	-0.155	0.094	-0.178	0.108	-1.644	0.107
4. Reward system appropriate for the agile process	-0.162	0.111	-0.124	0.085	-1.464	0.150
7. Project had a clear vision and	-0.110	0.107	-0.131	0.127	-1.033	0.307

was well driven by executives						

Table 4.2.9.6 - Forward stepwise regression summary - Process and Project Management Quality

Variable	Step +in/- out	Multiple R	Multiple R^2	R^2 change	F - to entr/rem	p-value	Variables included
3. Appropriate management of the agile process with satisfactory amount of documentation	1	0.692	0.479	0.479	46.832	0.000	1
6. Proper planning and emphasis on face-to-face communication	2	0.735	0.540	0.061	6.684	0.013	2
11. Suitable team size	3	0.749	0.561	0.021	2.308	0.135	3
4. Reward system appropriate for the agile process	4	0.760	0.577	0.016	1.853	0.180	4
7. Project had a clear vision and was well driven by executives	5	0.766	0.587	0.009	1.067	0.307	5

Table 4.2.9.7 - Backward stepwise regression - Process and Project Management Quality

N = 53	Multiple R = 0.692 Multiple R^2 = 0.479 Adjusted R^2 = 0.468 F(1,51) = 46.83 p < 0.0000 Standard error of estimate = 0.579					
	b*	Std. Err. of b*	b	Std. Err. of b	t(51)	p-value
Intercept			-0.197	0.468	-0.420	0.676
3. Appropriate management of the agile process with satisfactory amount of documentation	0.692	0.101	0.978	0.143	6.843	0.000

From these tables above, we can clearly see that the appropriate management of the agile process with a satisfactory amount of documentation has a great impact on the process and project management quality. An incredible 48 percent of the time, the process success can be predicted with regard to the process and project management quality by looking at this specific critical success factor.

Table 4.2.9.8 - Standard regression - Excellence of the project

N = 108	Multiple R = 0.362 Multiple R^2 = 0.131 Adjusted R^2 = 0.097 F(4,103) = 3.878 p < 0.00563 Standard error of estimate = 0.647					
	b*	Std. Err. of b*	b	Std. Err. of b	t(103)	p-value
Intercept			2.514	0.434	5.794	0.000
1. Environment suitable for agile	-0.014	0.108	-0.012	0.091	-0.127	0.899
2. Strong customer	0.254	0.107	0.229	0.096	2.373	0.019

involvement						
3. Appropriate management of the agile process with satisfactory amount of documentation	0.155	0.109	0.151	0.107	1.415	0.160
9. Competent team in one location	0.038	0.104	0.039	0.108	0.364	0.716

Table 4.2.9.9 - Forward stepwise regression - Excellence of the project

N = 108	**Multiple R = 0.360 Multiple R² = 0.130 Adjusted R² = 0.113** **F(2,105) = 7.830 p < 0.00068 Standard error of estimate = 0.641**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(105)**	**p-value**
Intercept			2.611	0.330	7.923	0.000
2. Strong customer involvement	0.253	0.105	0.228	0.094	2.424	0.017
3. Appropriate management of the agile process with satisfactory amount of documentation	0.160	0.105	0.157	0.102	1.534	0.128

Table 4.2.9.10 - Forward stepwise regression summary - Excellence of the project

Variable	Step +in/- out	Multiple R	Multiple R^2	R^2 change	F - to entr/rem	p-value	Variables included
2. Strong customer involvement	1	0.332	0.110	0.110	13.14	0.000	1
3. Appropriate management of the agile process with satisfactory amount of documentation	2	0.360	0.130	0.020	2.35	0.128	2

Table 4.2.9.11 - Backward stepwise regression - Excellence of the project

N = 53	Multiple R = 0.332 Multiple R^2 = 0.110 Adjusted R^2 = 0.102 F(1,106) = 13.14 p < 0.00045 Standard error of estimate = 0.645					
	b*	Std. Err. of b*	b	Std. Err. of b	t(106)	p-value
Intercept			2.894	0.275	10.53	0.000
2. Strong customer involvement	0.332	0.092	0.299	0.082	3.625	0.000

From these regression tables above, it is eminently clear that having a strong customer involvement is critical to the excellence of the project. In 11 percent of project situations, the process success can be predicted by looking only at the customer involvement in the project.

4.2.10 Success of the product

This section of the questionnaire measured the success of the product that was developed. If the project with which the respondent was involved with was cancelled before completion, this section could not be completed.

The user was presented with 11 statements regarding the success of the product resulting from the system development process. A statistical factor analysis was performed and once again two factors emerged. These factors as well as their reliability and factor loadings are depicted in the summary below:

Factor 1 – Product quality and user friendliness

Cronbach Alpha: 0.920	**Average inter-item correlation: 0.603**	
Original Statement	**Factor Loading**	**Cronbach alpha if deleted**
The functionality of the developed system is high	0.793	0.912
The reliability of the developed system is high	0.778	0.910
The efficiency of the developed system is high	0.481	0.915
The usability of the developed system is high	0.563	0.915
The developed system meets user needs	0.937	0.904
Overall, the quality of the developed system is high	0.770	0.907
Overall, the users are satisfied with the developed system	0.794	0.906
Overall, the developed system is a success	0.922	0.904

Factor 2 – The sustainability of the developed system

Cronbach Alpha: 0.760	**Average inter-item correlation: 0.522**	
Original statement	**Factor Loading**	**Cronbach alpha if deleted**
The maintainability of the	0.666	0.625

developed system is high		
The portability of the developed system is high	0.942	0.735
The documentation of the developed system is good	0.628	0.673

Table 4.2.10.1 below depicts the reliability of these factors with their respectable Cronbach alphas and average inter-item correlations. The values in the table below are acceptable according to the standards stated earlier in this chapter.

***Table* 4.2.10.1 – Product success reliability measures**

Factor	Cronbach Alpha	Average inter-item correlation
1 - Product quality and user-friendliness	0.920	0.603
2 - The sustainability of the developed system	0.760	0.522

Table 4.2.10.2 below shows the descriptive statistics for the two factors mentioned above. These factors were measured on a scale of 1 to 5 where 1 is "not at all" and 5 is "to great extent".

Table 4.2.10.2 – Product success descriptive statistics

Factor	Valid N	Mean	Standard Deviation
1 - Product quality and user-friendliness	108	3.76	0.59
2 - The Sustainability of the developed system	108	3.15	0.79

As seen from Table 4.2.10.2 above, the product quality and user friendliness is quite high with a mean of 3.76. The sustainability of the developed system is almost as high with a mean of 3.15.

In the next section the correlation statistics are presented. Correlations were investigated between the variables that are presented in *Table* 4.2.10.3 below and the product success variables, namely product quality and user-friendliness and the sustainability of the developed system.

Table 4.2.10.3 - Correlation statistics on product success

	1 - Product quality and user-friendliness	2 - The sustainability of the developed system
Agile Methodology was applied	-0.074	-0.059
PMBOK Methodology was applied	0.071	0.160
PRINCE2 Methodology was applied	-0.212	-0.269
1. Environment suitable for Agile	0.309	0.494
2. Strong customer involvement	0.272	0.348
3. Appropriate management of	0.305	0.488

the Agile process with satisfactory amount of documentation		
4. Reward system appropriate for the Agile process	-0.053	-0.219
5. Frequent delivery of usable software meeting frequently changing requirements	0.204	0.248
6. Proper planning and emphasis on face-to-face communication	-0.024	-0.081
7. Project had a clear vision and was well driven by executives	0.268	0.344
8. Adaptive manager and project team	0.236	0.021
9. Competent team in one location	0.118	0.249
10. Sufficient integration testing	0.353	0.172
11. Suitable team size	0.037	0.241
12. Good design practices and technical knowledge applied to the project	0.362	0.210
13. Regular schedule was honoured with no overtime	-0.087	0.191
14. Non-life-threatening project with no refactoring activities	0.230	0.114
15. Cooperative organizational culture instead of hierarchical	0.174	0.419
16. Team members with great motivation	0.057	0.110

The correlations marked in red in Table 4.2.10.3 are significant where the p-value, which is being reported, is smaller than 0.05. It is clear from the data presented that the use of PRINCE2 correlated with the product quality and user friendliness and the

sustainability of the developed system. An environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation, frequent delivery of usable software meeting frequently changing requirements, project with a clear vision and well driven by executives, and the application of good design practices and technical knowledge were all critical success factors which positively impacted the product quality and user friendliness as well as the sustainability of the developed system.

The presence of an adaptive manager and project team, sufficient integration testing, and a non-life-threatening project with no refactoring activities had a positive effect on the product quality and user friendliness.

Having a competent team in one location, a suitable team size, regular schedule with no overtime, and a cooperative organizational culture instead of hierarchical had a very good effect on the sustainability of the developed system.

The next section of this chapter will report the regression statistics with regard to the product success. The variables that had a meaningful impact in the correlations mentioned above were used to perform regression. Firstly a standard regression was performed, followed by forward - and backward stepwise regressions.

Table 4.2.10.4 - Standard regression - Product quality and user-friendliness

N = 53	**Multiple R = 0.626 Multiple R^2 = 0.392 Adjusted R^2 = 0.247** **F(10,42) = 2.703 p < 0.01180 Standard error of estimate = 0.414**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(42)**	**p-value**
Intercept			1.888	0.727	2.597	0.013
1. Environment suitable for agile	-0.041	0.163	-0.029	0.115	-0.250	0.804
2. Strong customer involvement	-0.032	0.158	-0.021	0.103	-0.202	0.841

3. Appropriate management of the agile process with satisfactory amount of documentation	0.162	0.179	0.138	0.152	0.906	0.370
PRINCE2 Methodology was applied	-0.181	0.176	-0.082	0.080	-1.028	0.310
5. Frequent delivery of usable software meeting frequently changing requirements	-0.073	0.146	-0.079	0.158	-0.501	0.619
7. Project had a clear vision and was well driven by executives	0.076	0.154	0.055	0.111	0.493	0.624
8. Adaptive manager and project team	-0.118	0.160	-0.114	0.154	-0.740	0.463
10. Sufficient integration testing	0.151	0.152	0.118	0.119	0.996	0.325
12. Good design practices and technical knowledge applied to the project	0.459	0.172	0.514	0.192	2.677	0.011
15. Cooperative organizational culture instead	-0.011	0.150	-0.006	0.075	-0.076	0.940

of hierarchical						

Table 4.2.10.5 - Forward stepwise regression - Product quality and user-friendliness

N = 53	**Multiple R = 0.613 Multiple R^2 = 0.376 Adjusted R^2 = 0.324** **F(4,48) = 7.237 p < 0.00012 Standard error of estimate = 0.392**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(48)**	**p-value**
Intercept			1.502	0.547	2.745	0.008
12. Good design practices and technical knowledge applied to the project	0.394	0.134	0.442	0.150	2.939	0.005
10. Sufficient integration testing	0.148	0.130	0.116	0.102	1.140	0.260
PRINCE2 Methodology was applied	-0.214	0.131	-0.098	0.060	-1.638	0.108
3. Appropriate management of the agile process with satisfactory amount of documentation	0.182	0.140	0.155	0.119	1.302	0.199

Table 4.2.10.6 - Forward stepwise regression summary - Product quality and user-friendliness

Variable	Step +in/- out	Multiple R	Multiple R^2	R^2 change	F - to entr/rem	p-value	Variables included
12. Good design practices and technical knowledge applied to the project	1	0.547	0.300	0.300	21.82	0.000	1
10. Sufficient integration testing	2	0.580	0.336	0.037	2.76	0.103	2
PRINCE2 Methodology was applied	3	0.595	0.354	0.018	1.36	0.249	3
3. Appropriate management of the agile process with satisfactory amount of documentation	4	0.613	0.376	0.022	1.69	0.200	4

Table 4.2.10.7 - Backward stepwise regression - Product quality and user-friendliness

N = 53	Multiple R = 0.547 Multiple R^2 = 0.300 Adjusted R^2 = 0.286 F(1,51) = 21.816 p < 0.00002 Standard error of estimate = 0.403					
	b*	Std. Err.	b	Std. Err.	t(51)	p-value

		of b*		of b		
Intercept			1.496	0.466	3.210	0.002
12. Good design practices and technical knowledge applied to the project	0.547	0.117	0.613	0.131	4.671	0.000

From the tables above, it is clear that the product quality and user-friendliness can be predicted by looking at the design practices and technical knowledge that was applied to the project.

Table 4.2.10.8 - Standard regression – Sustainability of the developed system

N = 108	**Multiple R = 0.675 Multiple R² = 0.455 Adjusted R² = 0.386** **F(12,95) = 6.613 p < 0.00000 Standard error of estimate = 0.622**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(95)**	**p-value**
Intercept			0.457	0.645	0.710	0.480
PRINCE2 Methodology was applied	-0.208	0.087	-0.328	0.138	-2.383	0.019
1. Environment suitable for agile	0.251	0.104	0.247	0.103	2.402	0.018
2. Strong customer involvement	-0.032	0.095	-0.034	0.099	-0.342	0.733
3. Appropriate management of the agile process with satisfactory amount of	0.189	0.109	0.215	0.124	1.736	0.086

documentation						
4. Reward system appropriate for the Agile process	-0.213	0.097	-0.154	0.070	-2.197	0.030
5. Frequent delivery of usable software meeting frequently changing requirements	-0.069	0.097	-0.102	0.144	-0.710	0.479
7. Project had a clear vision and was well driven by executives	0.060	0.095	0.063	0.099	0.638	0.525
9. Competent team in one location	0.050	0.095	0.061	0.116	0.524	0.601
11. Suitable team size	-0.124	0.091	-0.128	0.093	-1.371	0.173
12. Good design practices and technical knowledge applied to the project	0.123	0.102	0.160	0.133	1.204	0.232
13. Regular schedule was honoured with no overtime	0.091	0.082	0.066	0.060	1.115	0.268

Table 4.2.10.9 - Forward stepwise regression - Sustainability of the developed system

N = 108	Multiple R = 0.668 Multiple R^2 = 0.446 Adjusted R^2 = 0.401 F(8,99) = 9.953 p < 0.00000 Standard error of estimate = 0.614					
	b*	Std. Err. of b*	b	Std. Err. of b	t(99)	p-value
Intercept			0.434	0.517	0.839	0.403
15. Cooperative organizational culture instead of hierarchical	0.277	0.092	0.443	0.147	3.025	0.003
3. Appropriate management of the agile process with satisfactory amount of documentation	0.196	0.095	0.223	0.108	2.065	0.042
PRINCE2 Methodology was applied	-0.203	0.085	-0.321	0.134	-2.387	0.019
1. Environment suitable for agile	0.265	0.093	0.261	0.092	2.852	0.005
4. Reward system appropriate for the Agile process	-0.197	0.087	-0.142	0.063	-2.248	0.027
12. Good design practices and technical knowledge applied to the	0.125	0.097	0.163	0.126	1.293	0.200

project						
11. Suitable team size	-0.118	0.088	-0.121	0.091	-1.333	0.186
13. Regular schedule was honoured with no overtime	0.102	0.080	0.074	0.058	1.278	0.204

Table 4.2.10.10 - Forward stepwise regression summary - Sustainability of the developed system

Variable	Step +in/- out	Multiple R	Multiple R^2	R^2 change	F - to entr/rem	p-value	Variables included
15. Cooperative organizational culture instead of hierarchical	1	0.479	0.229	0.229	31.51	0.000	1
3. Appropriate management of the agile process with satisfactory amount of documentation	2	0.564	0.318	0.089	13.76	0.000	2
PRINCE2 Methodology was applied	3	0.610	0.373	0.054	8.97	0.003	3
1. Environment suitable for agile	4	0.631	0.399	0.026	4.45	0.037	4
4. Reward system	5	0.648	0.420	0.021	3.70	0.057	5

appropriate for the Agile process							
12. Good design practices and technical knowledge applied to the project	6	0.656	0.430	0.010	1.84	0.178	6
11. Suitable team size	7	0.661	0.437	0.007	1.18	0.281	7
13. Regular schedule was honoured with no overtime	8	0.668	0.446	0.009	1.63	0.204	8

Table 4.2.10.11 - Backward stepwise regression - Sustainability of the developed system

N = 108	**Multiple R = 0.564 Multiple R² = 0.318 Adjusted R² = 0.305** **F(2,105) = 24.534 p < 0.00000 Standard error of estimate = 0.662**					
	b*	**Std. Err. of b***	**b**	**Std. Err. of b**	**t(105)**	**p-value**
Intercept			0.031	0.459	0.067	0.946
3. Appropriate management of the agile process with satisfactory amount of documentation	0.320	0.086	0.365	0.098	3.710	0.000
15. Cooperative organizational culture instead	0.365	0.086	0.583	0.138	4.231	0.000

of hierarchical						

It is imminent from the regression tables above that the appropriate management of the agile process with a satisfactory amount of documentation and a cooperative organizational culture instead of hierarchical are crucial to the sustainability of the developed system.

Statistical analysis was also performed to find correlations between the time of use, horizontal use and intensity of use of SDM and PMM against the success of the process as well as the success of the product. These results yielded no statistically meaningful relationships.

4.3 Summary

In this chapter the research method was discussed and the results presented. We also discussed the measurement and the reliability of our research variables regarding the critical success factors as well as the project outcome variables. To estimate the reliability of these variables, we used a value of at least 0,5 for Cronbach's alpha in conjunction with an average inter-item correlation between the range of 0,15 and 0,55. All measurements for the research variables adhere to this standard, which increases our confidence in the measurement of these variables quite substantially.

Chapter 5
Conclusion and recommendations

5.1 Introduction

In this chapter, the results and limitations of this research are summarised and discussed. The contributions made by this research constitute an important issue, which is addressed by answering the questions as asked in chapter 1 in terms of the research aims and objectives. Since research should lead to practical implementations and various improvements as supported by the data in this study, the author makes various recommendations. Some of the many different topics that could be researched in order to improve knowledge concerning the use of ASDMs and critical success factors in information technology projects, are also mentioned.

5.2 Contributions of this research

The problem statement and research objectives mentioned in chapter 1 have been addressed and are supported by the data as described in chapter 4.

5.2.1 Determine critical success factors in agile systems development projects

There are many success factors that exist in the IT industry today. After an in-depth study, 43 critical success factors were selected and measured by means of a questionnaire. After factor analysis, these critical success factors could be reduced to 16 critical success factors that directly impact the success of IT projects. These critical success factors are depicted in *Table* 5.2.1 below:

***Table* 5.2.1 - The critical success factors in IT projects**

Factor
1. An environment suitable for Agile
2. Strong customer involvement
3. Appropriate management of the agile process with a satisfactory amount of documentation
4. Reward system appropriate for the agile process
5. Frequent delivery of usable software meeting frequently changing requirements
6. Proper planning and emphasis on face-to-face communication
7. A project with a clear vision and well driven by executives
8. An adaptive manager and project team
9. A competent team in one location
10. Sufficient integration testing
11. Suitable team size
12. Good design practices and technical knowledge applied to the project
13. Honouring a regular schedule with no overtime
14. Non-life-threatening project with no refactoring activities
15. Cooperative organizational culture instead of hierarchical
16. Team members with great motivation

These critical success factors correlate closely with the work of Chow and Cao (2007). Their study found the following factors to be most critical to the success of an agile systems development project: The team environment, the capability of the project team, the level of customer involvement, and the project management process.

The following critical success factors correlated with the process success: an environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation, a reward system appropriate for the agile process, frequent delivery of usable software meeting frequently changing requirements, proper planning and emphasis on face-to-face communication, project with a clear vision and well driven by executives,

sufficient integration testing, suitable team size, and a cooperative organizational culture instead of hierarchical were all critical success factors which correlated with the process and project management quality.

An environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation and a competent team in one location all had correlations with the excellence of the project.

The following critical success factors correlated with the product success: an environment suitable for agile, strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation, frequent delivery of usable software meeting frequently changing requirements, project with a clear vision and well driven by executives, and the application of good design practices and technical knowledge were all critical success factors which correlated with the product quality and user friendliness as well as the sustainability of the developed system.

The presence of an adaptive manager and project team, sufficient integration testing, and a non-life-threatening project with no refactoring activities had correlations with the product quality and user friendliness.

Having a reward system appropriate for agile, a competent team in one location, a suitable team size, regular schedule with no overtime, and a cooperative organizational culture instead of hierarchical had correlations with the sustainability of the developed system.

5.2.2 Evaluate ASDMs against the various critical success factors

According to Table 2.13.4 in chapter 2, there were many critical success factors addressed by the various agile systems development methodologies. If a specific ASDM addressed a critical success factor, a "yes" was recorded. If the ASDM only partially addressed the critical success factor, a "partially" was recorded. In the cases where the ASDM did not address the critical success factor, a "no" was

recorded. There were very few critical success factors not addressed by the various ASDMs. These unaddressed critical success factors are listed in *Table* 5.2.2 below:

***Table* 5.2.2 - The critical success factors not addressed by ASDMs**

Critical success factor	**Level of address**
Oral culture placing high value on face-to-face communication	Partially
Honouring regular working schedule - no overtime	Partially
Customer having full authority	Partially/No
Well-defined coding standards up front	Partially/No
Right amount of documentation	Partially/No
Sufficient testing was done during the project	Partially/No

These listed critical success factors had a majority of "partially" or "no" scores, implying that the ASDMs did not address the particular critical success factor.

All other critical success factors were addressed by the majority of the ASDMs investigated in this study. Only the six critical success factors in Table 5.2.2 above, weren't addressed. This means that ASDMs addressed 37 of the 43 critical success factors, which results in a percentage rating of 86 percent, which is very high.

5.2.3 Evaluate PMMs against the various critical success factors

According to Table 2.13.5 in chapter 2, there were many critical success factors addressed by the various project management methodologies. If a specific PMM addressed a critical success factor, a "yes" was recorded. If the PMM only partially addressed the critical success factor, a "partially" was recorded. In the cases where the PMM did not address the critical success factor, a "no" was recorded. There were very few critical success factors not addressed by the various PMMs. These unaddressed critical success factors are listed in Table 5.2.3 below:

***Table* 5.2.3 - The critical success factors not addressed by PMMs**

Critical success factor	Level of address
Cooperative organizational culture instead of hierarchical	Partially
Oral culture placing high value on face-to-face communication	Partially
Organizations where agile methodology is universally accepted	Partially
Facility with proper agile-style work environment	Partially/No
Reward system appropriate for agile	Partially
Following agile-oriented requirement management process	Partially
Strong communication focus with daily face-to-face meetings	No
Honouring regular working schedule - no overtime	Partially
Strong customer commitment and presence	Partially
Customer having full authority	No
Rigorous refactoring activities	No
Projects with dynamic, accelerated schedule	Partially/No
Projects with small team	Partially
Projects with no multiple independent teams	Partially
The project could handle unexpected crises and deviations from plan	Partially

These listed critical success factors listed above had a majority of "partially" or "no" scores. In cases where at least one of the PMMs addressed the specific critical success factor, that critical success factor was not taken into account for Table 5.2.3 above. All critical success factors not listed in Table 5.2.3 above were addressed by

at least one or both of the PMMs that form part of this study. This implies that the PMMs addressed 28 of the 43 critical success factors, resulting in a percentage rating of 65%, which is satisfactory.

5.2.4 Study the use/effectiveness of agile systems development methodologies in the industry

The results show that 42,58% of the respondents' organizations are implementing agile systems development methodologies. Almost 60% of these organizations are applying their ASDM knowledge across all departments and for every project. In 86% of these organizations, the ASDM is either used as a guideline or adapted on a project-to-project basis, although 13% of these organizations follow their respective ASDM rigorously.

Over 65% of the respondents' organizations have been using ASDMs for more than 3 years.

5.2.5 Study the use/effectiveness of project management methodologies in the industry

According to the data, 69% of the respondents' organizations are implementing project management methodologies. Of these organizations, almost 75% are implementing PRINCE2 as opposed to just over 25% which are using PMBOK. The respondents' organizations implementing PMMs, are applying their methodology knowledge across all departments and projects more than 56% of the time, as opposed to only 13% of organizations which are not implementing PMM knowledge. Almost 70% of the respondent organizations are applying their PRINCE2 and PMBOK methodology knowledge as a guideline or adapting it on a project-to-project basis and the remaining 30% follow their respective project management methodology rigorously.

More than 57% of organizations using PMBOK, have been using it for more than 5 years whereas 62% of organizations using PRINCE2, have been using it for more than 5 years.

5.3 Practical implications

There are a vast number of practical implications that can be found in the data and the results of this research. These recommendations are based on the data collected during this study and can help an organisation to improve the development of information systems as well as the maintenance of these systems.

The first recommendation is based on the findings that strong customer involvement and the appropriate management of the agile process with a satisfactory amount of documentation resulted in greater process success. Therefore, organisations should encourage these critical success factors when implementing an ASDM as this has a positive effect on the quality of the development process. The result regarding strong customer involvement can be expected, since it is one of the basic principles of agile systems development. However, the fact that a satisfactory amount of documentation is required may come as a surprise. Agile systems development methodologies were developed as a reaction to traditional systems development methodologies that required extensive amounts of documentation. This is an indication that even agile systems development projects require some documentation, and not no documentation as some may believe.

The appropriate management of the agile process with a satisfactory amount of documentation, the application of good design practices and technical knowledge to a project, and a cooperative organizational culture instead of hierarchical are three of the key factors when it comes to project outcome. By addressing these factors, the success of the entire project can be predicted with relative confidence. The fact that a cooperative organizational culture instead of a hierarchical organizational culture is positively related to product success can be expected. Agile projects differ from traditional projects in the sense that management is more cooperative and flexible, and the manager acts as a facilitator. Although the management style may be more relaxed, good design practices are still a requirement for project success.

The regression results indicated a negative relationship between the use of PRINCE2 and the sustainability of a project. Although this may seem strange, one should take into account that the evaluation of the PMM against the critical success factors, indicated that a large number of these critical success factors are not

addressed by the PMM. This is an indication that PMM may need to be extended or other measures need to be taken to address these shortcomings.

5.4 Limitations and recommendations for future research

During this study, certain limitations were encountered which will be discussed in this section. Through the discussion of these limitations, a positive contribution can be made to future research in this field of study.

The first limitation is that only 129 responses were received. By limiting the number of questions or encouraging more organisations to get involved with the research, a higher number of responses could be obtained.. More responses could improve the results of the study. By gathering more data, the accuracy and value of existing relationships could be improved.

This study was only carried out in one country, which limits the applicability thereof across organisations in countries other than the one where the research was carried out.

Another limitation is the time frame in which the research took place. Due to the fact that ASDMs and PMMs are still a relatively new concept in the particular country where the research took place, not many organisations were able to take part in the study. By implementing a similar study across more countries, this limitation could be overcome.

Future research could investigate the critical success factors in agile systems development projects across a broader spectrum which includes a greater number of projects. By carrying out this same study across more countries, one could determine more relationships and identify certain anomalies.

The critical success factors should also be investigated across multiple countries to enable a comparison between different regions of the world. Similarities and relationships could then be researched in different organisations around the world.

5.5 Conclusion

In conclusion it is clear that there are many critical success factors that greatly impact the success of the developed product as well as the success of the development process. Strong customer involvement, the appropriate management of the agile process with a satisfactory amount of documentation and the application of good design practices and technical knowledge to a project are some of the key success factors that were identified. By focussing on these critical success factors, the success of the entire project can be predicted.

The use of agile systems development methodologies has been increasing over the past few years, as can be seen from the data in chapter 4, implying that more organisations are starting to implement ASDMs. These are just some of the relationships identified in this study, and more research in the field could yield even more existing relationships. Much value can be added by further academic studies regarding the critical success factors in agile systems development projects.

References

ABRAHAMSSON, P., & KOSKELA, J. 2004. *Extreme Programming: A Survey of Empirical Data from a Controlled Case Study.* Oulu: Computer Society.

ABRAHAMSSON, P., SALO, O., RONKAINEN, J. & WARSTA, J. 2002. Agile software development methods: Review and analysis. *VTT Publications* 478. 107 p.

ANDERSON, P.F. 1983. Marketing, Scientific Progress, and Scientific Method. *Journal of Marketing*, p. 18-31.

ATKINSON, R. 1999. Project Management: cost, time and quality, two best guesses and a phenomenon, it is time to accept other success criteria. *International Journal of Project Management*, 17(6):337-342.

BECK, K., *et al. (*2001). Manifesto for Agile Software Development. http://www.agilemanifesto.org Date of access: 28 March. 2010.

BECK, K. (1999). *Extreme Programming Explained: Embrace change.* Reading, MA:Addison-Wesley.

BHARADWAJ, A. 2000. Resource-based Perspective on Information technology Capability and Firm Performance: An Empirical Investigation. *MIS Quarterly*, 24(1): 169-196.

BLUM, B. (1996). *Beyond Programming: To a New Era of Design*: Oxford University Press.

BOEHM, B. & TURNER, R. (2004). *Balancing Agility and Discipline: A Guide for the Perplexed.* Boston, MA:Addison-Wesley.

BOEHM, B. & TURNER, R. (2005). Management challenges to implement agile processes in traditional development organizations. *IEEE Software* 22 (5), 30–39.

BOURNE, L. & WALKER, D.H.T. (2004). Advancing project management in learning organizations*. Learn Organizat,* 11(3):26–43.

BROWN, H.I. 1977. *Perception Theory and Commitment*. University of Chicago Press, Chicago, Illinois.

BUCKLE, P. & THOMAS, J. (2003). Deconstructing project management: a gender analysis of project management guidelines. *International Journal of Project Management,* 21(6):41-43.

CASEMaker, (2000). What is Rapid Application Development? http://www.casemaker.com/download/products/totem/rad_wp.pdf Date of Access 19 February. 2010.

CESCHI, M., SILLITTI, A., SUCCI, G. & DE PANFILIS, S. (2005). Project management in plan-based and agile companies. *IEEE Software*, 22(3):21-27, May-June.

CHAN, F.K.Y. & THONG, J.Y.L. (2009). Acceptance of agile methodologies: A critical review and conceptual framework. *Decision Support Systems*, 46:803-814.

CHAOS Summary. (2009). http://www1.standishgroup.com/newsroom/chaos_2009.php Date of Access: 8 June 2011.

CHOW, T. & CAO, D. (2007). A survey study of critical success factors in agile software projects. *The Journal of Systems and Software*, 81:961-971, August.

CLARK, L.A. & WATSON, D. (1995). Constructing validity: Basic issues in objective scale development. Psychological Assessment, 7, 309-319.

CLELAND, D. I. 1995. Leadership and the project management body of knowledge. *International Journal of Project Management*, 13(2):83-88.

COCKBURN, A. (2002). *Agile Software Development*. Addison-Wesley. Boston.

COCKBURN, A. & HIGHSMITH, J. (2001). Agile software development: The business of innovation. *IEEE Computer*, 120–122, Sept.

COHEN, D., LINDVALL, M., & COSTA, P. (2004). An Introduction to Agile Methods. p16-22. Advances in Computers.

COOK, T.D. & CAMPBELL, D.T. 1979. *Quasi-Experimentation: Design and Analysis Issues for Field Settings*. Houghton Mifflin, Boston, MA.

DEANE, R. H., CLARK, T. B., YOUNG, A. P. 1997. Creating a learning project environment: aligning project outcomes with customer needs. *Information Systems Management,* p54-60.

DRUCKER, P. (1999*). Management Challenges for the 21st Century*. Harper Business. p. 33.

DYBA, T. & DINGSOYR, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50:833-859.

GABLE, G.G. 1994. Integrating case study and survey research methods: an example in information systems. *European Journal of Information Systems*, 3(2):112-126.

GERMAIN, E. & ROBILLARD, P. N. (2005). Engineering based processes and agile methodologies for software development: a comparative case study. *Journal of Systems and Software,75*(1–2):17–27.

GUTEK, B.A. 1991. Survey and Other Methodologies Applied to IT Impact Research: Experiences From a Comparative Study of Business Computing. In The Information Systems Research Challenge: *Survey Research Methods*, 3:317-322.

HIGHSMITH, J. 1999. *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. New York, NY: Dorset House Publishing.

HIGHSMITH, J. 2001. History: The Agile Manifesto. http://www.agilemanifesto.org/history.html Date of access 28 March. 2010.

HIGHSMITH, J. 2002. *Agile Software Development Ecosystems*. Addison-Wesley, Boston, Massachusetts.

HIRSCHHEIM, R. & KLEIN, H.K. 1989. Four Paradigms of Information Systems Development. Communications of the ACM, 32(10):1199-1216, October.

HOWARD, A. 1997. Industrial Management and Data Systems: *A New RAD-based approach to commercial information systems development – the dynamic system development method.* www.emeraldinsight.com

HUISMAN, H.M. (2000). "The deployment of systems development methodologies: A South African Experience". PhD Thesis, Potchefstroomse Universiteit vir CHO, Potchefstroom, South Africa, 242 pages.

JAMES, M. 2009. Scrum. www.scrummethodology.com/Scrum_Refcard Date of Access 19 February. 2010.

KARLSTROM, D. & RUNESON, P. 2005. Combining agile methods with Star-Gate project management. *IEEE Software* 22 (3), 43–49.

KARLSTRöM, D. & RUNESON, P. 2006. Integrating agile software development into stage-gate managed product development. *Empirical Software Engineering*, 11(2):203-225.

KERZNER, H. (2009). *Project Management: A Systems Approach to Planning, Scheduling and Controlling*. New Jersey. Wiley & Sons.

KETTER, W., BANJANIN, M., GUIKERS, R. & KAYSER, A. 2009. Introducing an Agile Method for Enterprise Mash-Up Component Development. *IEEE Conference on Commerce and Enterprise Computing*. p. 293-300.

KHAZANCHI, D. & MUNKVOLD, B.E. 2002. On the Rhetoric and Relevance of IS Research Paradigms: A Conceptual Framework and Some Propositions. 36th Hawaii International Conference on System Sciences.

KLEIN, H.K., & LYYTINEN, K. 1984. The Poverty of Scientism in Information Systems in Research methods in Information Systems. Proceedings of the IFIP WG Colloquium, p. 131-162, Sept.

KOCH, A.S. 2005. *Agile Software Development: Evaluating the Methods for Your Organizations.* Artech House, Northwood, Massachusetts.

KRUTCHEN, P. 2001. Agility with the RUP. *Cutter IT Journal, 14*(12):27–33.

KUMAR, D. R. 2005. Lean Software Development. *The Project Perfect White Paper Collection*, June, www.projectperfect.com.au Date of Access 8 March. 2010.

LINDVALL, M., *et al.* 2002. *Empirical Findings in Agile Methods*. (*In* Wells, D. & Williams, L.A., *eds.* Proceedings of the Second XP Universe and First Agile Universe Conference on Extreme Programming and Agile Methods: XP/Agile Universe 2002. London, UK. Springer-Verlag. **p. 197-207.**)

LEFFINGWELL, D. 2007. *Scaling Software Agility: Best Practices for Large Enterprises*. Pearson Publication, MA, USA.

LEONG, S.M. 1985. Metatheory and Metamethodology in Marketing: A Lakatosian Reconstruction. *Journal of Marketing*, 49:23-40.

MACCORMACK, A., VERGANTI, R. & IANSITI, M. 2001. Developing Products on “Internet Time”: The Anatomy of a Flexible Development Process. *Management Science, 47*(1):133–150.

MESO, J. & JAIN, R. 2006. *Information Systems Management*. 23(3):19-30.

MEYER, C. 1994. How the right measures help teams excel. *Harvard Business Review.* p95-103.

MISRA, S.C., KUMAR, V. & KUMAR U. (2009. Identifying some important success factors in adopting agile software development practices. *The Journal of Systems and Software,* 82:1869–1890.

MORRIS, P. W. G., PATEL, M. B. & WEARNE, S. H. 2000. Research into revising the APM project management body of knowledge. *International Journal of Project Management*, 18:155-164.

NERUR, S., MAHAPATRA, R.K., MANGALARAJ, G. 2005. Challenges of migrating to agile methodologies. *Communications of the ACM* 48 (5), 72–78.

NUNALLY, J. 1967. *Psychometric Theory*. McGraw Hill, New York.

OGC - Office of Government Commerce. (2009. Managing successful project with PRINCE2. The Stationery Office.

OISEN, R. P. 1971. Can project management be defined? *Project Management Quarterly*, 2(1):12-14.

PALMER, S. R. & FELSING, J. M. (2002. *A Practical Guide to Feature-Driven Development.* Upper Saddle River, NJ, Prentice-Hall.

PARTHASARTHY, R. & RANGARAJAN, A. 2008. A Framework for Real Time Seamless Integration of Business Process Development with Agile Software Development. 6(4):7-16.

PMI - Project Management Institute. 2008. *A Guide to the Project Management Body of Knowledge* (PMBOK Guide), 4th Edition. Pennsylvania, USA.

POPPENDIECK, M. & POPPENDIECK, T. 2006. *Implementing Lean Software Development: From Concept to Cash*. Addison Wesley. p23–40.

REEL, J.S. 1999. Critical success factors in software projects. *IEEE Software* 16 (3), 18–23.

REILLY, J. P. & CARMEL, E. 1995. Does RAD live up to the hype? *IEEE Software* (Sept. 1995), 24–26.

SCHAAF, R. J. 2007. Agility XL. *Systems and Software Technology Conference*. Tampa, FL.

SCHATZ, B. & ABDELSHAFI, I. 2005. Primavera gets agile: A successful transition to agile development. *IEEE Software* 22 (3), 36–42.

SCHWABER, K. & BEEDLE, M. 2002. *Agile Software Development with Scrum*. Upper Saddle River, NJ, Prentice-Hall.

SCHWALBE, K. 2010. *Managing Information Technology Projects*. Sixth Edition. Cengage.

SIEGELAUB, J. M. 2006. How PRINCE2 Can Complement PMBOK and Your PMP. PMI Global Congress Proceedings. Anaheim, California.

STRAUB, DETMAR, GEFEN, D., BOUDREAU, M.C. 2004. The ISWorld Quantitative, Positivist Research Methods Website. http://dstraub.cis.gsu.edu:88/quant/default.asp Date of access: 16 Feb. 2011.

STRUCKENBRUCK, L. 1987. Who determines project success. Project management Institute Seminar/Symposium, Montreal, Canada. p85-93, Sept.

SUTHERLAND, J. & VAN DEN HEUVEL, W.J. 2002. Enterprise Application Integration and Complex Adaptive Systems. *Communications of the ACM, 45*(10):59–64.

TAIICHI, O. 1988. *Toyota Production System: Beyond Large Scale Production*, Productivity Press. p. 4.

TAKEUCHI, H. & NONAKA, I. 1986. The New Product development Game. *Harvard Business Review* Jan. /Feb.: 137.

WIDEMAN, R. M. 2002. *Comparing PRINCE2 with PMBoK.* AEW Services. Vancouver, BC, Canada.

VIDICH, A.J. & SHAPIRO, G. 1955. A Comparison of Participant-Observation and Survey Data. *American Sociological Review*, (20) pp.28-33.

VINEKAR, V., SLINKMAN, C.W. & NERUR, S. 2006. Can Agile and Traditional systems development approaches coexist? An ambidextrous view. *Information Systems Management*.

WINDHOLTZ, M. 2005. Lean Software Development. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.2561&rep=rep1&type=pdf Date of Access 7 March. 2010.

WRIGHT, J. N. 1997. Time and budget: the twin imperatives of a project sponsor. *International Journal of Project Management*, 15(3):181-186.

WYNEKOOP, J.L. & RUSSO, N.L. 1997. Studying system development methodologies: an examination of research methods. *Information Systems Journal*, 7:47-65.

YEONG, A. 2009. The Marriage Proposal of PRINCE2 and PMBOK. November 2009 Edition.

Appendix A

North-West University
Potchefstroom Campus

Private bag X6001 Potchefstroom 2520
Tel (018) 299 1111 Fax (018) 299 2799
http://www.nwu.ac.za

Computer Science and Information Systems
Tel (018) 299-2537
Fax (018) 299-2570
E-Mail 20850352@nwu.ac.za

16 August 2011

Research Questionnaire

Investigating Critical Success Factors in Agile Systems Development Projects

The primary objective of this study is to investigate the critical success factors involved when applying agile systems development methodologies and project management methodologies to an information technology project.

Your participation is of utmost importance to this study! Answers will be kept confidential and participation is voluntary. Please return this questionnaire as soon as possible. A copy of the results can be sent to you on request.

Return deadline: 7 September 2011

Yours Sincerely

Ruhan Wagener

M.Com Computer Science and Information Systems

Tel: 083 659 0529

Email: 20850352@nwu.ac.za

rpwagener@gmail.com

Section 1: Background information

1. What is your primary role in systems development?	
IS manager	1
Project manager	2
Team leader	3
Systems analyst	4
Analyst/Programmer	5
Programmer	6
Other, please specify: …… Information Administrator (Inter) …… ……	7

2. Please indicate the highest qualification you have obtained.	
Senior certificate (High school)	1
Certificate or diploma	2
University or technicon degree	3
Honours or Masters degree	4
PhD degree	5
Other, please specify …… …… ……	6

3. What is your personal experience regarding systems development?	
None	1
Less than 1 year	2
1 - 3 years	3
3 - 5 years	4

5 - 10 years	5
More than 10 years	6

4. Please indicate the sector of your organization.	
System/Software development	1
Telecommunication	2
Manufacturing	3
Mining	4
Consulting	5
Financial/Banking/Insurance	6
Government	7
Other, please specify …… …… ……	8

5. Please indicate what percentage of your personal time/work is dedicated to the following development activities:	
Systems planning, analysis and design	%
Programming	%
Testing	%
Installation	%
User Training	%
Project Management	%
Other, please specify …… …… ……	%

Section 2: The use of systems development methodologies

1. There are various types of system development methodologies, for example XP (Extreme programming) and IE (Information Engineering). Name the methodologies your organization uses, and indicate how intensively each one is used: (1 = used very infrequently, 5 = used very often)					
Name of Methodology	**Intensity of use** **Very infrequently**				**Very often**
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5

2. How long has your systems development methodology been in use in your IS department?	
Less than 1 year	1
1 - 2 years	2
3 - 5 years	3
5 - 10 years	4
More than 10 years	5
I don't know	6

3. What is the proportion of projects that are developed in your IS department by applying systems development methodology knowledge?	
None	1
1 – 25 %	2
26 – 50 %	3
51 – 75 %	4
Over 75 %	5

4. What is the proportion of people in your IS department that apply systems development methodology knowledge regularly?	
None	1
1 – 25 %	2
26 – 50 %	3
51 – 75 %	4
Over 75 %	5

5. Which of the following best describes how your IS department makes use of its systems development methodologies?	
A general guideline for all projects.	1
A standard which is followed rigorously for all projects.	2
Adapted on a project –to-project basis.	3

6. There are various types of project management practices, for example PMBOK (Project management body of knowledge) and PRINCE2 (PRoject IN Controlled Environment Version 2). Name the project management practices your organization uses, and indicate how intensively each one is used: (1 = used very infrequently, 5 = used very often)

Name of project management practice	**Intensity of use**				
	Very infrequently				**Very often**
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5
	1	2	3	4	5

7. How long has your company been using project management practices in your IS department?	
Less than 1 year	1
1 - 2 years	2
3 - 5 years	3

5 - 10 years	4
Over 10 years	5
I don't know	6

8. What is the proportion of projects that are developed in your IS department by applying project management practices?	
None	1
1 – 25 %	2
26 – 50 %	3
51 – 75 %	4
Over 75 %	5

9. What is the proportion of people in your IS department that apply project management practices knowledge regularly?	
None	1
1 – 25 %	2
26 – 50 %	3
51 – 75 %	4
Over 75 %	5

10. Which of the following best describes how your IS department makes use of its systems development methodologies?	
A general guideline for all projects.	1
A standard which is followed rigorously for all projects.	2
Adapted on a project –to-project basis.	3

Section 3: Critical Success Factors in Information Technology Projects

1. To what extent do you agree with the following statements as valid descriptions of the last project(s) you were involved with? (1 - Not at all, 5 - To great extent)					
	Not at all				**To great extent**
Strong executive support	1	2	3	4	5
Committed sponsor or manager	1	2	3	4	5
Cooperative organizational culture instead of hierarchal	1	2	3	4	5
Oral culture placing high value on face-to-face communication	1	2	3	4	5
Agile methodology is universally accepted in the organization	1	2	3	4	5
Collocation of the whole team	1	2	3	4	5
Facility with proper agile-style work environment	1	2	3	4	5
Reward system appropriate for the agile process	1	2	3	4	5
Team members with high competence and expertise	1	2	3	4	5
Team members with great motivation	1	2	3	4	5
Managers knowledgeable in the agile process	1	2	3	4	5
Managers who have light-touch or adaptive management style	1	2	3	4	5
Coherent, self-organizing teamwork	1	2	3	4	5
Good customer relationship	1	2	3	4	5
Following agile-oriented requirement management process	1	2	3	4	5
Following agile-oriented project management process	1	2	3	4	5
Following agile-oriented configuration management process	1	2	3	4	5
Strong communication focus with daily face-to-face meetings	1	2	3	4	5
Honoring regular working schedule - no overtime	1	2	3	4	5
Strong customer commitment and presence	1	2	3	4	5
Customer having full authority	1	2	3	4	5
Well-defined coding standards up front	1	2	3	4	5
Pursuing simple design	1	2	3	4	5
Rigorous refactoring activities	1	2	3	4	5

Right amount of documentation	1	2	3	4	5
Regular delivery of software	1	2	3	4	5
Delivering most important features first	1	2	3	4	5
Correct integration testing	1	2	3	4	5
Appropriate technical training to team	1	2	3	4	5
Project nature being non-life-critical	1	2	3	4	5
Project type being of variable scope with emergent requirement	1	2	3	4	5
Projects with dynamic, accelerated schedule	1	2	3	4	5
Projects with small team	1	2	3	4	5
Projects with no multiple independent teams	1	2	3	4	5
Projects with up-front cost evaluation done	1	2	3	4	5
Projects with up-front risk analysis done	1	2	3	4	5
The project could handle unexpected crises and deviations from plan	1	2	3	4	5
The project was efficiently coordinated and successfully integrated activities, documents and personnel	1	2	3	4	5
Sufficient testing was done during the project	1	2	3	4	5
The project had clearly defined business priorities	1	2	3	4	5
The project had a clear vision and objectives	1	2	3	4	5
Project management was good	1	2	3	4	5
The project applied appropriate systems development methodology knowledge.	1	2	3	4	5

Section 4: Project Outcome (Effectiveness)

1. What is the size of the last information system development project you were involved with?	
Small	1
Medium	2
Large	3

Very large	4
I have not partaken in any information system development projects	5

2. Which of the following best describes the outcome of the last systems development project you were involved with?	
The project was canceled/terminated before completion.	1
The project was completed but not implemented.	2
The project was completed and implemented, but is not in use anymore.	3
The project was completed and implemented, and is in use for ……..months. (Please specify)	4

If your last project was cancelled before completion, you have completed the questionnaire.

Thank you for your cooperation!

If your last project was not cancelled before completion, please answer the following questions.

3. To what extent do you agree with the following statements about the last project you were involved with? (1 - Not at all, 5 - To great extent)					
	Not at all				**To great extent**
The project was completed on schedule	1	2	3	4	5
The project was completed within the budget	1	2	3	4	5
The developed system satisfied all the stated requirements	1	2	3	4	5
The speed of developing the project was high	1	2	3	4	5
The cost of the project is low when compared to the size and complexity of the system developed	1	2	3	4	5
The project achieved its goals	1	2	3	4	5
Overall, the project represents excellent work	1	2	3	4	5
Overall, the project was a success	1	2	3	4	5

4. To what extent do you agree with the following statements about the last project you were involved with? (1 - Not at all, 5 - To great extent)					
	Not at all				**To great extent**
The functionality of the developed system is high	1	2	3	4	5
The reliability of the developed system is high	1	2	3	4	5
The maintainability of the developed system is high	1	2	3	4	5
The portability of the developed system is high	1	2	3	4	5
The efficiency of the developed system is high	1	2	3	4	5
The usability of the developed system is high	1	2	3	4	5
The developed system meets user needs	1	2	3	4	5
The documentation of the developed system is good	1	2	3	4	5
Overall, the quality of the developed system is high	1	2	3	4	5
Overall, the users are satisfied with the developed system	1	2	3	4	5
Overall, the developed system is a success	1	2	3	4	5

Thank you for taking part in the survey!