The development of a hybrid Agile Software Development Methodology through the integration of Agile Software Development Methodologies with Project Management Methodologies

F Swanepoel

orcid.org/0000-0003-4179-4436

Dissertation accepted in partial fulfilment of the requirements for the degree *Master of Science in Computer Science* at the North-West University

Supervisor: Prof HM Huisman

Graduation May 2021

31469698

ACKNOWLEDGEMENTS

It gives me great pleasure to thank all the people who made this journey possible, turning my inspiration into a reality. First of all, I would like to express my sincere gratitude to my Supervisor, Prof Magda Huisman, not only for her invaluable guidance and encouragement, but also for her insight and immense knowledge she shared during this whole process. I further want to thank her for her kindness and inspiration that made this a memorable experience.

I would also like to thank my lovely wife and wonderful children for their support and constant encouragement throughout my research.

A special thanks to my colleagues who assisted and participated in this process, and specifically Mrs Ronel Roets for her immense perseverance and enthusiasm and the contribution she made along the way.

Most importantly, I would like to thank and praise my heavenly Father who enabled me to complete this with such devotion that made me realise again that you can do all things through Christ who strengthens you (Philippians 4:13).

A final thought which helped and motivated me during this journey is from Tony Robbins who said that “the only impossible journey is the one you never begin”.

ABSTRACT

Implementing software development projects successfully remains a challenge for many organisations across all industries. Organisations today have to adapt to complex business environments where continuous change and transformation are important factors to project success. The use of traditional software models and methodologies for software development projects has shifted to the adoption of different agile software development methodologies (ASDMs) and project management methodologies (PMMs). These methodologies find it difficult to address all challenges related to software development projects and many organisations end up only adopting one of the two methodologies to implement software development projects. Because organisations differ in so many ways, there is a need to develop a hybrid ASDM based on a combination of ASDMs and PMMs to deliver a suitable and flexible methodology that offers the best of both worlds to organisations according to their unique operating environments. The aim of this study is to develop a hybrid ASDM to enable organisations to execute software development projects effectively.

Literature on the most popular PMMs and ASDMs where each methodology was defined and described in detail was investigated. The advantages and disadvantages of each were studied, including areas regarding where and how they are applied. To achieve the objective of the study, PMBOK was selected as the preferred PMM which was integrated with selected ASDMs. The integration delivered a hybrid ASDM that can be customised and used for different organisations and industries based on their unique environments.

Different research paradigms and methods were studied to identify and select the most suitable method for the study. A critical social theory paradigm was followed, and action research was selected as the research method. Three cycles of diagnosing, action planning, action taking, evaluating and specifying learning were applied during this study to test and improve the hybrid ASDMs that were selected. Interviews were conducted with the project teams to ensure that a collaborative approach was followed. Feedback from the various teams was incorporated in the study.

As part of the action research process, historical projects for a particular organisation were evaluated and used to diagnose the problem areas the organisation experienced in executing software development projects. Three projects were identified to which different hybrid ASDMs were applied. What was learnt from previous cycles was adopted in the following action research cycles to further improve the situation. Each project incorporated the specific organisation’s characteristics after which the delivered projects were evaluated by the project

teams. The selected hybrid ASMDs delivered positive results and all three projects were delivered with better results compared to the historical projects. The outcome of the study proposes a hybrid ASDM that can be adapted to various organisations and industries based on their unique characteristics and environments.

Keywords: Project management, Software development methodology, SDM, Agile, Integration, Action research

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: PROBLEM STATEMENT

1.1 Introduction

In this chapter, the study is introduced. The problem statement is discussed in section 1.2. In section 1.3, previous studies done on agile software development methodologies (ASDMs) and project management methodologies (PMMs) integration are discussed. The shortcomings and gaps related to research that has already been done are discussed in section 1.4 and the importance of the research is covered in section 1.5. The research aims and objectives are discussed in section 1.6 and will be used as guidelines in the process of investigating the problem statement. The method of investigation is discussed in section 1.7 and in section 1.8, a summary and conclusion to Chapter 1 are provided.

1.2 Problem statement

One of the most common challenges that companies are facing today is their ability to successfully execute software projects (Ebad, 2016:1145; Lehtinen *et al*., 2014:641; Hussain *et al*., 2016:307). Inadequate resources, unrealistic timelines, underestimated costs, overlooked requirements, poor governance and substandard coding are all influencing factors that can lead to project failure (Ebad, 2016:1150). Organisations are consequently focussing their efforts on the project team's ability to select appropriate resources, techniques and methods to meet the needs of customers and to execute projects successfully (Rebaiaia and Vieira, 2014:40).

For software projects to be successful, the primary objective is to develop and deliver software solutions with agreed functionality, on time, within budget and of good quality (Alami, 2016:63; Mandal and Pal, 2015:118). Many projects in the past have failed, not only because of time and cost overruns, but also because of not delivering all the required features (Mandal and Pal, 2015:119). Poor project management seems to be the root cause of many software project failures with most of the challenges being directly related to how projects are managed through all phases of their life cycles (Arcidiacono, 2017:1,9). According to Rebaiaia and Vieira (2014:41), more companies are progressing toward the exploitation of new tools and methodologies to improve their project management process and execution. PMMs offer a large selection of the tools, tips, techniques, and tactics that are available to guide project managers to the successful completion of their projects. A project management methodology is defined by the Project Management Institute as a set of methods, techniques, procedures, rules, templates, and best practices used on a project (PMBOK Guide, 2013). Charvat (2003) defines PMM as a set of guidelines and principles that can be tailored and applied to a specific situation, where guidelines could be a task list or a specific approach to a project with defined

tools and techniques. Similarly, Introna and Whitley (1997) define a project management methodology as a structured set of techniques and tools used for solving a specific problem.

Another method to prevent previously mentioned project failures is by means of agile software development methodologies (ASDMs), which is an approach to develop software. Although ASDMs appear practical and attractive in achieving results quickly, there is often a question as to whether ASDMs can deliver projects without following a complete project management process. Salameh (2014: 61-65) mentions ASDMs that execute responsibilities at project level, as well as iteration level. According to Rebaiaia and Vieira (2014:40), project management and lean and agile methodologies are complementary and can be beneficial in managing projects effectively. Larmen (2003:25) states that it is not possible to exactly define ASDMs due to the specific practices that vary, although similarities exist in terms of short iterations where plans and goals are regularly adapted. They also promote practices and principles that follow simplicity, lightness, communication, self-directed teams, and programming over documenting which encourage fast and flexible response to change. Yadav (2015:13), however, defines ASDM as a methodology that addresses the need for flexibility and incorporates a level of pragmatism in delivering the final product by focussing on simplified coding, regular testing and releasing functional bits of the application as soon as they are ready. The objective is to deliver small client-approved parts throughout the project process instead of one large application at the end of the project.

There are various software development methodologies, such as Evo (May and Zimmer, 1996:1-2), unified process family (Pressman, 2014:56-58), lean development (Despa, 2014:46-47) and also the agile family, consisting of various methodologies, for example Scrum (Mahalakshmi and Sundararajan, 2013:192), dynamic systems development model (Despa, 2014:46), rapid application development (Despa, 2014:43), extreme programming (Despa, 2014:43-44), feature-driven development (Despa, 2014:48) and various others. For the purpose of this study, a selected PMM will be compared and integrated with some of the most commonly used ASDMs. The following ASDMs will be used within this study:

- Scrum;
- Extreme programming (XP);
- Dynamic systems development method; and
- Feature-driven development (FDD).

These ASDMs are selected because they are amongst the well-established methodologies for agile software development as described in the agile software development manifesto (Beck *et al*., 2001).

livari and Huisman (2007) studied the relationship between organisational culture and the deployment of systems development methodologies, and one of their findings was that ASDMs should be customised to fit the specific needs of an adopting organisation (livari and Huisman 2007:48). It is not easy to select the most suitable ASDM for a particular project as there are various factors that can have an impact on the success of a project. According to Alqudah and Razali (2017:533), the nature of the project, skills of the development team, project constraints, culture of the organisation and customer involvement are the main factors that will determine the success of a project when selecting an ASDM. Due to the dynamic business environment, requirements change continuously and unexpectedly. Consequently, there is a need for system development methodologies and project management methodologies with the ability to adapt to a changing environment.

Due to ever-changing and complex business environments, organisations need to be able to adapt themselves (Stoica *et al*., 2013:64). Because organisations differ in so many ways, based on industry, culture, maturity, and complexity, the integration of ASDMs with formal project management methodologies might be required to identify suitable methodologies for specific organisations. One of the reasons why many software projects do not meet user requirements, are behind schedule and are over budget is due to the lack of following a PMM or ASDM. Consequently, the use of following only a traditional PMM or only an ASDM is not effective. A combination of both methodologies may be required to improve project success (Abdelghany *et al*., 2017:12).

Currently, ASDM and PMM find it difficult to address the challenges related to an ever-changing environment and many organisations adopt one of the two to implement software development projects (Ziolkowski, 2014:65). SDMs are not always a perfect fit for specific projects and should be customised to fit the needs of the organisation (Despa, 2014:55). Therefore, a need exists to develop a hybrid software development methodology based on a combination of ASDMs and PMMs to deliver a suitable and flexible methodology that offers the best of both worlds to organisations according to their unique operating environments (Ziolkowski, 2014:65).

The conclusion can therefore be made that there is no single methodology that can work for all organisations across all industries and using a hybrid methodology between ASDMs and PMMs would be ideal (Ziolkowski, 2014:65). Although there have been limited studies on combining

ASDMs with PMMs, the results of these studies have not been applied or implemented for organisations. One of the focus points of this study will include the application and implementation of a hybrid methodology for a specific organisation.

1.3 Previous studies

Various studies have been done in the field of combining and/or integrating individual ASDMs with a PMM. Table 1-1 illustrates the researcher's own summary of similar studies that have been done on the integration of ASDMs with PMMs, including studies on agile and traditional project management. The integration of ASDMs with PMMs will be discussed in detail in the chapters to follow. The table below indicates the study title, findings of the study and the author who has conducted the study. The studies are based on research that was carried out between 2010 and 2019.

Table 1-1: Previous studies on agile and traditional project management

Title	***Description***	***Author(s)***
Integrating PMBOK standards, lean and agile methods in project management activities	The study compares the effect of the project management body of knowledge (PMBOK) with lean and agile methods on the project management processes. The outcome of the study indicates that PMBOK, lean and agile strategies are complementary and can improve the execution of complex projects.	Rebaiaia and Vieira (2014:40-46)
Software development: agile vs. traditional	Organisations require more agility to be more adaptable to market changes which is key for gaining a strategic advantage. This requires organisations to adopt various techniques, tools and methods to react timeously on continuous changing requirements. Transforming from traditional PMMs to ASDMs is discussed.	Stoica *et al*. (2013:64)
Mixed agile/traditional project management methodology – reality or illusion?	An overview of project management approaches is explained. The need to combine PMM and ASDM is discussed and the study further elaborates on how methodologies could be designed for specific projects.	Spundak (2014)

Table 1-1: Previous studies on agile and traditional project management (continued)

What, when, why, and how? A comparison between agile project management and traditional project management methods	Agile project management (APM) delivers projects faster with a reduction in project costs with continuous customer involvement following an iterative process. APM is compared with traditional project management (TPM) by using the PMBOK process groups and knowledge areas related to leadership style, communication, change, scope, and risk management.	Salameh (2014)
Agile software development methodology	The study describes the various characteristics and benefits of agile software development methodologies and compares it briefly to the waterfall software development methodology.	Edeki (2015)
Relationships between a project management methodology and project success in different project governance contexts	More experienced project management offices are using combinations of ASDM that allow flexibility in the processes and methodologies.	Aubry (2010:775)
Agile preparation within a traditional project management course	This study explains how ASDM can be taught in conjunction with a traditional PMM as part of a project management course. ASDMs are integrated with PMM.	Landry and McDaniel (2016)
Popular agile methods in software development: review and analysis	The objective of the study is to assist project managers in selecting the most suitable ASDMs that fit their projects, based on their needs and requirements.	Anand and Dinakaran (2016:3433)
Traditional SDLC vs Scrum methodology – A comparative study	Traditional methodologies are not able to meet the requirements of the market. This study explains the Scrum software development methodology, and describes it and the difference between traditional SDLC (waterfall) and Scrum.	Mahalakshmi and Sundararajan (2013)

Based on the previous studies, summarised and listed in Table 1-1, it is evident that research about the integration of agile software development methodologies with project management is limited. Previous studies focused more on comparisons and the adoption of agile methodologies rather than the integration between these methodologies and project management methodologies.

Based on the previous studies, it is clear that considerable research has been done on the topics of ASDMs and PMMs, but these studies were carried out in isolation. Research on how these methodologies can be used in conjunction with one another is still in the early stages.

1.4 Importance of the research

Project success is traditionally measured by delivering a project of acceptable quality within the agreed time plan and budget (Kerzner, 2009:7). However, project success goes beyond the milestone where the project is delivered to the customer and when project management ends. The impact on the organisation once the project is implemented and in use must also form part of the measurement (Serrador and Pinto, 2015:1043). According to the PMBOK Guide (2013:35), project success should be measured by means of delivering the project within the boundaries of the project scope, time, cost, quality, resources, and risk according to the agreement between the project managers and senior management. Project success will be based on the last agreed baselines that were approved by the authorised stakeholders. For the purpose of this study, project success will be based on the following key success factors:

- *Project Scope.* This involves the management and incorporation of scope changes which are management by means of additional enhancements required on the project.
- *Time.* Refers to when the project is delivered and accepted by the key stakeholders against an approved and based-lined project plan.
- *Quality.* Quality is measured in terms of the amount of rework required on the project, including customer satisfaction.
- *Cost*. Involves the management of the project costs ensuring that it stays within the agreed and approved budget.
- *Resources:* This involves the effective utilisation of manpower involved in the project. Some resources are shared between projects which requires effect management and planning of resources. Resource unavailability can have a direct impact on the project plan.
- *Risk:* All factors that may affect any of the key success factors of the project need to be managed and mitigated accordingly.
- *Project methodology:* This includes feedback from the project team by means of interviews regarding project process, methodology and success.

This research aims to assist organisations to apply a hybrid ASDM according to their own environment and culture that will enable them to execute and implement software development projects to the extent that the organisation benefits from it operationally while delivering projects on time and within budget.

Some of the first advantages a company will experience in following an ASDM include financial savings and delivering projects in a shorter period of time (Shankarmani *et al.*, 2016:32). It also results in increased customer satisfaction due to strong customer feedback mechanisms and real-time updates on the status of development (Shankarmani *et al.*, 2016:32). Altameem (2015:9) states that ASDMs empower development teams throughout the challenges of continuous changing user requirements. This makes the development process more flexible which allows for the developers to work more efficiently, thereby creating a pleasant working environment for developers. It further has a positive impact on developers, as the methods and techniques of ASDM can be incorporated into their existing knowledge (Altameem, 2015:13). The management style during the development process motivates the developers to work more effectively in delivering their development tasks and increases their creativity and innovation. This enables developers to contribute effectively to the resolution of project problems to eventually deliver quality projects (Altameem, 2015:13).

This study will contribute to the following areas:

- research in the integration of ASDMs and PMMs;
- improve knowledge regarding the use of ASDMs and PMMs;
- provide organisations with a flexible framework that can be applied in the execution of software development projects thereby improving the success rate of such projects; and
- improve the working conditions of software development teams.

This study will therefore focus on the integration of ASDMs with a PMM to determine if this can improve project success.

In order to find an answer to the problem statement stated in section 1.2, and to ensure that the study will contribute to the above-mentioned areas, secondary aims and objectives are set. These aims and objectives are discussed in the next section.

1.5 Research aims and objectives

This research aims to assist organisations to apply a hybrid ASDM according to its own environment and culture that will enable them to execute software development projects effectively. To accomplish this objective, the researcher will have to achieve the following secondary objectives:

- Study literature on the execution of software development projects within companies;

- Study different ASDMs that are most commonly used by organisations;
- Study PMMs that are most commonly used by organisations;
- Study literature on previous studies where ASDMs and PMMs were applied in conjunction to deliver software projects;
- Gather information on previously software development projects for a particular organisation and evaluate the execution against predefined success criteria in terms of ASDMs and PMMs that were used;
- Develop an ASDM and PMM matrix that can be applied to various organisations and industries.
- Identify an applicable ASDM and PMM framework for a particular organisation and apply the integration on software development projects; and
- Evaluate the effectiveness of the applied ASDM and PMM integration framework.

1.6 Method of investigation

The research paradigm that will be followed will be a critical social research method. The researcher will make use of action research as a research method. The focus is to identify the social problem, namely software development project failures, with the aim of improving the situation. Therefore, action research will be applied to better understand the situation and to possibly change and improve it.

Information will be gathered on historical projects and evaluated against predefined success criteria. The success criteria will be based on project success factors as per industry standards. The researcher will work with a project team where future projects will be executed to test the proposed solution. This includes the collection of data based on industry standard metrics and interviews with developers regarding the project process and how they perceived the outcome of the project. Data analysis will include metrics with comparative statistics and interviews with content analysis.

1.7 Outline of this study

The chapters of this dissertation are divided as follows and a brief description of each chapter is provided. Figure 1-1 presents the layout of this dissertation.

- Chapter 1 – Problem statement: In this chapter, the problem statement and previous studies related to this study are discussed. The importance of the research is described and the aims and objectives are defined. The research methodology, data collection method and data analysis methods are discussed briefly. The aim of this chapter is to serve as an

introduction for the study and to give more background regarding research that will be conducted.

- Chapter 2 – Project management methodologies: In this chapter, literature regarding project management methodologies is discussed. The aim of this chapter is to improve knowledge regarding the use of project management methodologies. Three PMMs are discussed, namely Project Management Body of Knowledge (PMBOK), PRoject IN Controlled Environment (PRINCE2) and Agile Project Management (APM).
- Chapter 3 – Agile software development methodologies: In this chapter, literature regarding agile software development methodologies is discussed. The aim of this chapter is to improve knowledge regarding the use of agile software development methodologies. Four ASDMs are discussed, namely Scrum, extreme programming (XP), dynamic systems development method and feature-driven development.
- Chapter 4 – Research method: The manner in which the research is conducted is discussed in this chapter. The research paradigm, research method, data collection method and the data analyses method will be expanded upon. The aim of this chapter is to indicate the research process followed during this study.
- Chapter 5 – Research results: The results of the research are analysed by discussing the processes followed during the action research cycle.
- Chapter 6 – Results discussion and conclusion: In this chapter, the results are discussed. The purpose of this chapter is to determine whether the aims and objectives set to address the problem statement discussed in section 1.2 were achieved.
- Bibliography: In this section, a list of authors used as references for this study is provided. The reference style used throughout the dissertation is the Harvard style.

The development of a hybrid Agile Software Development Methodology through the integration of Agile Software Development Methodologies with Project Management Methodologies

Chapter 1: Problem statement

1.1 Introduction
1.2 Problem statement
1.3 Previous studies
1.4 Importance of the research
1.5 Research aims and objectives
1.6 Method of investigation
1.7 Outline of this study
1.8 Summary / conclusion

Chapter 2: Project management methodologies

2.1 Introduction
2.2 Project management methodologies
2.2.1 Definition of project management methodology
2.2.2 Project management body of knowledge (PMBOK)

Figure 1-1: Layout of dissertation

2.2.3 PRoject IN Controlled Environment (PRINCE2)
2.2.4 Agile project management (APM)
2.2.5 PMBOK, PRINCE2 and APM comparison

Chapter 3: Agile software development methodologies

3.1 Definition of software development methodology
3.2 Agile software development methodology
3.2.1 Scrum
3.2.2 Extreme programming (XP)
3.2.3 Dynamic systems development method (DSDM)
3.2.4 Feature-driven development (FDD)
3.2.5 ASDM comparison
3.3 ASDM and PMM integration
3.4 Summary

Chapter 4: Research method

4.1 Research paradigms
4.1.1 Positivist research paradigm
4.1.2 Interpretive research paradigm
4.1.3 Critical social theory
4.1.4 Research paradigm used for this study
4.2 Research methods associated with critical research
4.3 Research method used for this study
4.3.1 Action research
4.3.2 Action research characteristics
4.3.3 Action research principles
4.3.4 Action research advantages
4.3.5 Action research disadvantages
4.4 Action research cycle

Chapter 5: Contextual background for the study

5.1 Organisational environment characteristics
5.2 Evaluation criteria
5.3 Previously executed projects with no methodology
5.3.1 Capital expenditure application system (big project)
5.3.2 Fleet management system (big project)
5.3.3 Dealsheet management system (medium project)
5.3.4 Frontend – Integrated business planning (medium project)
5.3.5 Mobile device ordering system (small project)
5.3.6 Product configuration management (small project)
5.3.7 Summary
5.4 Project selected for this study

Chapter 6: Action research cycle 1 – small project

6.1 Quality concessions (small project)
6.2 Diagnose
6.3 Action planning
6.4 Action taking
6.5 Evaluating
6.6 Specifying learning

Chapter 7: Action research cycle 2 – medium project

7.1 Overseas travel control system (medium project)
7.2 Diagnose
7.3 Action planning
7.4 Action taking
7.5 Evaluating
7.6 Specifying learning

Figure 1-1: Layout of dissertation (continued)

Chapter 8: Action research cycle 3 – big project

8.1 Cashbook (big project)
8.2 Diagnose
8.3 Action planning
8.4 Action taking
8.5 Evaluating
8.6 Specifying learning

Chapter 9: Results discussion and conclusion

9.1 Introduction
9.2 Research objectives
9.2.1 Study literature on the execution of software development projects within companies
9.2.2 Study different ASDMs that are most commonly used by organisations
9.2.3 Study PMMs that are most commonly used by organisations
9.2.4 Study literature on previous studies where ASDMs and PMMs were applied in conjunction to deliver software projects
9.2.5 Gather information on previous software development projects for a particular organisation and evaluate the execution against a predefined success criteria in terms of ASDMs and PMMs that were used
9.2.6 Develop an ASDM and PMM matrix that can be applied to various organisations and industries
9.2.7 Identify an applicable ASDM and PMM framework for a particular organisation and apply the integration on software development projects
9.2.8 Evaluate the effectiveness of the applied ASDM and PMM integration framework
9.3 Discussion of results
9.4 Contribution of this study
9.5 Limitations of this study
9.6 Future work
9.7 Conclusion

Figure 1-1: Layout of dissertation (continued)

1.8 Summary / Conclusion

In this chapter, the introduction to the dissertation and the research which will be conducted were outlined. It provided the problem statement that explains the challenges organisations are facing in the execution of software development projects and the different methodologies with which companies are experimenting in an attempt to improve their project success rates.

Previous studies have shown that companies are using various ASDMs and PMMs to implement software development projects. Some authors claim that these methodologies can be used together to achieve better results, but this has not been applied and tested on organisations which is what the focus of this study will be.

Three of the most well-established PMMs will be discussed in the next chapter and these methodologies will be defined and evaluated.

CHAPTER 2: PROJECT MANAGEMENT METHODOLOGIES

2.1 Introduction

In this chapter, ASDMs and PMMs will be discussed. The definitions of ASDMs and PMMs will be investigated, including the timeline of how these methodologies matured over time. In addition, the advantages and disadvantages for the above-mentioned methodologies will be evaluated, including how they have been adopted in the industry. Lastly, a preferred PMM will be selected which will be the basis for the integration of ASDMs with the selected PMM.

2.2 Project management methodologies

Project management as a term will first be defined whereafter three of the most commonly used PMM's will be discussed in detail. One of the PMMs will then be selected for the purpose of integration with ASDMs. The three PMMs that will be investigated are:

- PMBOK (project management body of knowledge)
- PRINCE2 (project in controlled environment)
- APM (agile project management)

These three PMMs were selected as they are all commonly known as approved methodologies in terms of managing and delivering all types of project (Priyanka, 2016; Ghosh *et al*., 2015; Wideman, 2005; Matos and Lopes, 2013:788). The focus of this study is in the software development space which means that the selected PMM must be commonly known in the execution of information technology projects.

2.2.1 Definition of a project management methodology

To understand the definition of a project management methodology, the term project management must first be defined and explained. Project management on its own is also a very broad field and it is therefore important to understand and first define a project. Table 2-1 lists various definitions of the term "project".

Table 2-1: Definitions of "Project"

Definition	Source
An individual or collaborative enterprise that is carefully planned to achieve a particular aim.	Oxford University Press (2019)
A project is a temporary endeavour undertaken to create a unique product, service, or result.	PMBOK Guide (2013: 3)

Table 2-1: Definitions of "Project" (continued)

A project is a temporary organization that is created for the purpose of delivering one or more business products according to an agreed Business Case.	Office of Government commerce (2009:31)
A project is a time- and cost-constrained operation to realize a set of defined deliverables (the scope to fulfil the project's objectives) up to quality standards and requirements.	International project management association (2006:13)
A project refers to a value creation undertaking based on a specific, which is completed in a given or agreed timeframe and under constraints, including resources and external circumstances.	Project management association of Japan (2005:15)
Project is a unique process consisting of a set of coordinated and controlled activities with start and finish dates, undertaken to achieve an objective conforming to specific requirements, including constraints of time, cost and resources.	ISO 10006:2017 (2017:2)
A project is a managed set of interrelated resources that delivers one or more products to a customer or end user. The set of resources has a definite beginning and end and operates according to a plan.	Software Engineering Institute (2001:24)
A project can be considered to be any series of activities and tasks that: • Have a specific objective to be completed within certain specifications • Have defined start and end dates • Have funding limits (if applicable) • Consume human and nonhuman resources (i.e., money, people, equipment) • Are multifunctional (i.e., cut across several functional lines)	Kerzner (2009:2)
Projects are the building blocks in the design and execution of strategies for an organization. Projects provide an organizational focus for conceptualizing, designing, and creating new or improved products, services, and organizational processes.	Cleland (2004:3)
A project is "a temporary endeavour undertaken to create a unique product, service, or result."	Schwalbe (2017:4)

Based on the above-mentioned definitions it is evident that there are many similarities which can therefore be summarised in the following definition, which is the one adopted in this study: *"A project is a set of activities and tasks that are executed within time and cost constrains to achieve a specific objective."*

Now that the term project has been clarified, the term project management can be defined. As with the term "project", there are many definitions of "project management" which are listed in Table 2-2.

Table 2-2: Definitions of "project management"

Definition	**Source**
Project management is "the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements".	PMBOK Guide (2013: 5)
Project management "is the planning, organizing, directing, and controlling of company resources for a relatively short-term objective that has been established to complete specific goals and objectives".	Kerzner (2013:4)
Project management is the process of managing, allocating, and timing resources to achieve a given goal in an efficient and expeditious manner.	Badiru and Osisanya 2013:50
Project management "is the art of managing all the aspects of a project from inception to closure using a scientific and structured methodology".	Management Study Guide (2008)
Project management "is the planning, delegating, monitoring and control of all aspects of the project, and the motivation of those involved, to achieve the project objectives within the expected performance targets for time, cost, quality, scope, benefits and risks".	Office of government commerce (2009:4)
"From a functional point of view, project management can be defined as the process of guiding a project from the beginning through its complete duration to its end using a set of management models".	Bruegge and Dutoit (2010:578)
Project management is "the application of knowledge, skills, tools and techniques to project activities to meet the project requirements".	Schwalbe (2017:8)

By taking the definition of a project into consideration and evaluating the definitions of project management as per Table 2-2, project management can be summarised and defined as follows, which is the one adopted in this study:
Project management is the coordination of resources and activities to achieve agreed milestones in order to deliver a defined goal within an agreed timeframe.

As this study is about the integration of ASDMs and PMMs, project management methodologies will be defined and discussed by taking the definitions of a project and project management into consideration. To define a project management methodology, one must first define what a methodology is. Table 2-3 lists various definitions of a methodology.
Firstly, the definition of a methodology will be discussed whereafter some of the most commonly used PMMs will be evaluated in detail. Table 2-3 lists many definitions of the term "methodology".

Table 2-3: Definitions of a "methodology"

Definition	**Source**
"A methodology is a system of practices, techniques, procedures and rules used by those who work in a discipline".	PMBOK Guide (2013:546)
"A system of methods used in a particular area of study or activity".	Oxford University Press (2019)
"A methodology describes how things should be done".	Schwalbe (2014:89)
"A methodology is a collection of methods for solving a class of problems and specifies how and when each method should be used".	Bruegge and Dutoit (2010:15)
"A methodology is a set of guidelines or principles that can be tailored and applied to a specific situation. In a project environment, these guidelines might be a list of things to do. A methodology could also be a specific approach, templates, forms, and even checklists used over the project life cycle".	Charvat (2003:3)
"A software development methodology is a set of rules and guidelines that are used in the process of researching, planning, designing, developing, testing, setup and maintaining a software product".	Despa (2014:41)

Based on the above-mentioned definitions, the definition of a methodology in a software development context that will be adopted in this study is as follows:
A methodology is a set of tools and methods used to develop, deliver and maintain software solutions throughout the software development life cycle.

A project management methodology can be defined as a "conceptual framework for program and project management" that contains a "set of guidelines, standards and processes. It is a communication mechanism used to share lessons learned and best practices for program and project management (ITS Project Management Group, 2014:10). The PM² Project Management Methodology Guide (2016:11) defines a PMM as a methodology that is built on project management best practices which is supported by four pillars namely a project governance model (i.e. Roles & Responsibilities), a project life cycle (i.e. Project Phases), a set of processes (i.e. Project Management Activities) and a set of project artefacts (i.e. templates and guidelines). A PMM is further defined as "a comprehensive set of best practices, tools and techniques that are dynamic, flexible, adaptive and customisable to suit different projects within a specific environment" (Chin and Spowage, 2012:108). Therefore, based on the adopted definitions of a project, project management and methodology, including the definitions of a PMM, the definition of a PMM that will be adopted in this study is as follows:
Project Management Methodology is a combination of practices, methods and processes that determine how best to plan, develop, control and deliver a project throughout the software development life cycle. It is a scientifically-proven, systematic and disciplined approach to project design, execution and completion.

Three PMMs, namely PMBOK, PRINCE2 and APM will be discussed in the following sections.

2.2.2 Project management body of knowledge (PMBOK)

PMBOK is a guide to project management, designed and published by the project management Institute. Nasir and Sahibuddin (2015:1283) state that the most prominent and internationally recognised project management framework is the project management body of knowledge.

The PMBOK has been extensively revised and expanded since 1983, and in 1996 the first edition of "A guide to the project management body of knowledge," known as the PMBOK Guide, was released (Nasir and Sahibuddin, 2015:1283). The second edition was released in 2000 and the third edition in 2004 with the fourth addition in 2009 and the fifth edition in 2013. The sixth and latest edition was released in late 2017 and incorporated new processes, a new chapter on the role of the project manager, reduced tools and techniques, new sections in the knowledge areas, an introduction to agile, the usage of notations and lastly, additional language

translations. With the incorporation of these changes, the overall framework remains the same as that in the fifth edition (Edureka, 2019). Because the sixth edition is still fairly new, the fifth edition of the PMBOK Guide will be used for this study to explain the PMBOK methodology.

PMBOK defines a total of forty-seven project management processes which describe activities throughout a project's life cycle (PMBOK Guide, 2013:4-5). These processes are arranged into a table consisting of five project management process groups and ten knowledge areas.

2.2.2.1 Project management process groups

Project management takes place through processes, using the knowledge, skills, tools and project management techniques. These processes are combined into five groups called project management process groups. The PMBOK guide defines how the five process groups are integrated as they follow a sequential process of initiation, planning, executing, monitoring and controlling, and closing (PMBOK Guide, 2013:48-49) illustrated in Figure 2-1.

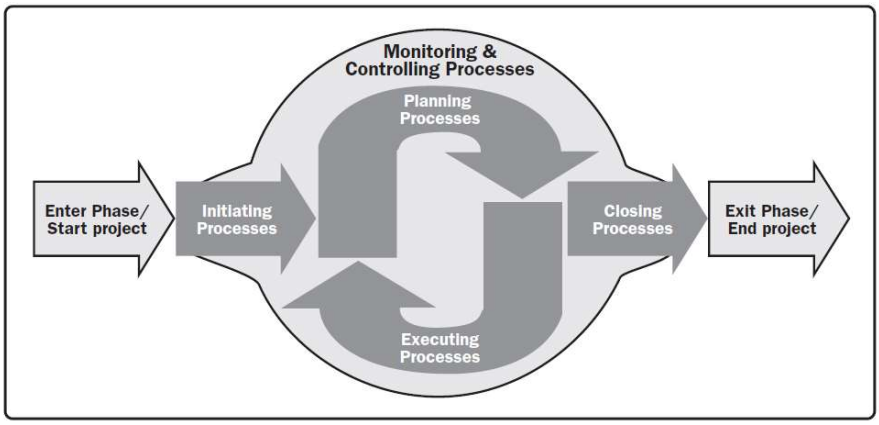

Figure 2-1: Project management process groups (PMBOK Guide, 2013:50)

The process groups are overlapping events that happen throughout the project and consist of various activities that must be completed which will be covered in the following sections.

2.2.2.1.1 Initiating process group

The initiating process group involves the processes, activities, and skills needed to effectively define the beginning of a project. This is where the necessary authorisations are obtained. Most of the time the project manager is involved in initiating a project, but in many cases the project manager is only selected after many initiation activities have taken place (Schwalbe, 2014:91).

During this phase, the project scope is defined, the required resources are obtained, and all stakeholders are identified. A project charter is drafted, containing the project title, project start date, budget information, success criteria and roles and responsibilities. Other tasks that also form part of this phase involve determining the scope, time and cost constraints for the project, including the selection of a project sponsor and compiling a business case.

Initiating processes occur during each phase of a project. Initiating processes are also required to close out a project where activities are initiated to ensure that the project team completes all work and documentation required to close a project (Schwalbe, 2014:83).

2.2.2.1.2 Planning process group

Upon authorisation of the project, the project must be planned. This phase produces a document called a project management plan which is the master planning document that establishes stakeholder expectations and makes it clear how the project will be managed (Schwalbe, 2014:96). This process group also addresses a narrower clarification of all project goals and expectations and puts in place the project infrastructure necessary to achieve those goals according to the timeline and budgetary constraints. The planning process outlines the project's scope, cost, deadlines, milestones, communication needs, and anything else that shows the stakeholders how the project will be managed. The following tasks and activities can be executed during this process (PMBOK Guide, 2013:61):

- project management plan development;
- requirements collection;
- scope definition;
- work breakdown structure creation;
- activities definition;
- activities sequencing;
- activity resources estimation;
- schedule development;
- cost estimation;
- budget determination;
- quality planning;
- human resource plan development;
- communications planning;
- risk management planning;
- risk identification;
- qualitative risk analysis;

- quantitative risk analysis;
- risk response planning; and
- procurement planning.

This phase is extremely important as a lack of planning can result in cost and schedule overruns as well as other project changes that can have a negative impact on the project.

2.2.2.1.3 Execution process group

During this phase, the project plan is implemented and it includes all work required to deliver the final product or service (Cleland, 2004:44). The execution of the project requires coordination of human resources, managing stakeholder expectations, and dealing with project changes. The following tasks and activities are executed during this process (PMBOK Guide, 2013:61):

- directing and managing project execution;
- performing quality assurance;
- acquiring the project team;
- managing the project team;
- distributing information;
- managing stake holder expectations; and
- conducting procurements.

This phase is the most resource intense and delivers the project deliverable to the customer (Schwalbe, 2014:106).

2.2.2.1.4 Monitoring and controlling process group

This process monitors and controls the project to ensure that project deliverables are on time, on budget and of acceptable quality. The project manager must monitor and control the project work and ensure that corrective action is taken to address any deviations from the plan (Schwalbe, 2014:111). Monitoring and controlling are done throughout the life of a project and involve the following activities (PMBOK Guide, 2013:61):

- monitoring and controlling the project work;
- performing integrated change control;
- scope verification;
- controlling the scope;
- controlling the schedule;
- controlling the cost;
- performing quality control;

- reporting performance;
- monitoring and controlling risks; and
- administering procurements.

If changes are required to any part of the project as documented in the project management plan, they need to be documented and result in an updated plan. This includes changes to deadlines, costs, deliverables, and any other change to the project as envisioned.

2.2.2.1.5 Closing process group

This phase includes the formal acceptance of the project and the ending thereof. Administrative activities include the archiving of the files and documenting lessons learned. The result of this phase is a report to the approving authority for the project that all actions have been completed and all resources accounted for (Schwalbe, 2014:114-115).

2.2.2.2 Knowledge areas

PMBOK defines the following ten knowledge areas that will be discussed in detail in the later sections to follow:

- Project integration management
- Project scope management
- Project time management
- Project cost management
- Project quality management
- Project human resource management
- Project communications management
- Project risk management
- Project procurement management
- Project stakeholder management

As supporting elements, the knowledge areas provide a detailed description of the process inputs and outputs along with a descriptive explanation of tools and techniques most frequently used within the project management processes to produce each outcome (PMBOK Guide, 2013:59). Monitoring and controlling processes happen continuously within the other process groups covered in the previous sections.

PMBOK outlines quality management principles and practices as they relate to the management of projects. It provides the guidance on quality issues that impact projects. It is applicable to projects of varying complexity, size and length. The guidelines can be applied to

projects managed by an individual or by a team or for a portfolio of projects. The guide therefore aims rather to stabilise and structure the knowledge needed to drive a current project in the best conditions. PMBOK is also process-oriented and each process is described in terms of inputs, outputs and activities (Rebaiaia and Vieira, 2014:42).

Table 2-4 describes the forty-seven project management processes within the five project management process groups and ten knowledge areas (PMBOK Guide, 2013:59-60).

Table 2-4: Project management process group and knowledge area mapping (PMBOK Guide, 2013:61)

Knowledge Areas	Project Management Process Groups				
	Initiating Process	**Planning Process**	**Executing Process**	**Monitoring and Controlling**	**Closing Process**
Project Integration Management	Develop Project Charter	Develop Project Management Plan	Direct and Manage Project Work	Monitor and Control Project Work Perform Integrated Change Control	Close Project or Phase
Project Scope Management		Plan Scope Management Collect Requirements Define Scope Create WBS		Validate Scope Control Scope	
Project Time Management		Plan Schedule Management Define Activities Sequence Activities Estimate Activity Resources Estimate Activity Durations Develop Schedule		Control Schedule	
Project Cost Management		Plan Cost Management Estimate Costs Determine Budget		Control Costs	
Project Quality Management		Plan Quality Management	Perform Quality Assurance	Control Quality	
Project Human Resource Management		Plan Human Resource Management	Acquire Project Team Develop Project Team Manage Project Team		
Project Communications Management		Plan Communications Management	Manage Communications	Control Communications	

Table 2-4: Project management process group and knowledge area mapping (PMBOK Guide, 2013:61) (continued)

Project Risk Management		Plan Risk Management Identify Risks Perform Qualitative Risk Analysis Perform Quantitative Risk Analysis Plan Risk Responses		Control Risks	
Project Procurement Management		Plan Procurement Management	Conduct Procurements	Control Procurements	Close Procurements
Project Stakeholder Management	13.1 Identify Stakeholders	Plan Stakeholder Management	Manage Stakeholder Engagement	Control Stakeholder Engagement	

2.2.2.2.1 Project integration management

Project integration management describes the processes and activities that integrate the various processes and project management activities within the project management process groups. The deliverables of this knowledge area include the project charter, project management plan, change control and requests, project management plan updates, and the final results of the project as it reaches closure (PMBOK Guide, 2013:63-64). According to Schwalbe (2014:130-131) and the PMBOK Guide (2013:63-64), the following processes are involved in project integration management:

- *Developing the project charter:* The project charter is what gives the authority to initiate the project. A project charter contains the preliminary roles and responsibilities of the project, including the goals and objectives, and the appointing of a project manager. It is used as a reference document as the project moves forward.
- *Developing the project management plan:* The creation of a project management plan defines how the various processes in the project can work together for greater efficiency and productivity. This is a formal document to help, guide, control and execute the project.
- *Directing and managing project execution:* This is the process to manage the technical and organisational parts of the project. It serves to deliver a smooth execution of project work. The outputs of this process are deliverables, requested changes, work performance information, project plan and project documentation updates.
- *Monitoring and controlling project work:* This involves tracking the progress of the project tasks and activities. Deliverables of this process include change requests and updated project plans and documentation.
- *Performing integrated change control:* This is the process where change management takes place to manage change throughout the course of the project.

- *Closing the project or phase:* This process involves formally completing the project and handing over all completed operations to the responsible parties. The outputs to this process are the delivery of the final product or service and all completed, up to date project documentation.

2.2.2.2.2 Project scope management

Project scope management includes processes required to ensure that the project includes all the work required to complete the project successfully. According to Jainendrakumar (2015:1), project scope management is the process to ensure that everything that the project requires to be successful is included in the project and nothing more. Project scope management ensures that all stakeholders of the project, including the project team are all in agreement regarding what the project will deliver and what processes to follow to achieve the end deliverable (Schwalbe, 2014:178). It involves five main processes (Schwalbe, 2014:178-179; PMBOK Guide, 2013:105-140):

- *Collecting requirements:* This process involves documenting the requirements that will produce the project deliverable according to stakeholder expectation. Outputs of the process are stakeholder requirements documentation, a requirements management plan and a requirements traceability matrix.
- *Defining scope:* During this process, a scope statement is created based on the project charter, requirements documents and organisational process.
- *Creating the work breakdown structure (WBS):* This process involves breaking down the main project deliverables into smaller components that are more manageable. Documents produced involve a work breakdown structure, a WBS dictionary, a scope baseline, and general updates to project documents.
- *Verifying scope:* During this process the project deliverables are formally accepted and signed off by the key stakeholders. Change requests are raised where deliverables are not acceptable.
- *Controlling scope:* Involves the management of project scope changes during the project life cycle. Changes mostly have an impact on time and cost which is why change requests are closely evaluated by the project manager. This process generates change requests, work performance measurements, and updates to project-related documents.

2.2.2.2.3 Project time management

Project time management is the process of ensuring that the project tasks and activities are completed according to the agreed project plan to deliver the project on time (Schwalbe,

2014:213; PMBOK Guide, 2013:141). There are six processes involved in project time management:

- *Defining activities:* During this process, the activities required to deliver the project are listed. Activities in this context mostly have start and end dates and require project resources to be executed and completed (Schwalbe, 2014:213).
- *Sequencing activities:* This process involves the identification of the relationships between the activities and generates project schedule network diagrams and project document updates (Schwalbe, 2014:213).
- *Estimating activity resources:* During this process it is estimated how many resources (people, materials, equipment) are required to deliver the project. This process generates activity resource requirements, a resource breakdown structure, and project document updates (Schwalbe, 2014:213; PMBOK Guide, 2013:141).
- *Estimating activity durations:* This process involves the estimation of the number of work periods required to complete the individual activities with the available resources (Schwalbe, 2014:213; PMBOK Guide, 2013:141).
- *Developing the schedule:* This process involves the analysis of the activity sequences, resource estimates per activity, and estimates on the duration of the activities to compile the project schedule (Schwalbe, 2014:213; PMBOK Guide, 2013:141).
- *Controlling the schedule:* This process involves the management and controlling of project schedule changes to ensure that the project stays within the project plan. It generates documents, such as work performance measurements, change requests, project management plan updates and project document updates (Schwalbe, 2014:213; PMBOK Guide, 2013:141).

2.2.2.2.4 Project cost management

Project cost management includes activities to manage cost around planning, budgeting and financing in order to control costs to ensure the project is completed within the approved budget (PMBOK Guide, 2013:193). Three processes are involved in project cost management:

- *Estimating costs:* This process involves the estimation of the resource costs required to complete the project. A cost management plan is created as part of the project management plan under integration management. Outputs of this process are activity cost estimates, basis of estimate and project document updates (Schwalbe, 2014:256; PMBOK Guide, 2013:200).
- *Determining the budget:* This process involves the establishment of a baseline for measuring performance by allocating the total cost estimate to individual project activities.

The outputs are a cost baseline, project funding requirements and project document updates (Schwalbe, 2014:256; PMBOK Guide, 2013:208).

- *Controlling costs:* During this process, changes to the project budget are monitored and controlled. The outputs are work performance information, budget forecasts, change requests, project management plan updates, project document updates and organisational process asset updates (Schwalbe, 2014:256; PMBOK Guide, 2013:215).

2.2.2.2.5 Project quality management

Project quality management describes the process required to achieve the project objectives and to ensure that the client requirements are satisfied (Schwalbe, 2014:295). Quality has the same importance as the project scope, time and cost, as it forms part of meeting the requirements of the client. Three main processes are involved in project quality management:

- *Planning quality:* During this process, the relevant quality standards are identified and how those standards can be met is established. The outputs of this process are a quality management plan, quality metrics, quality checklists, a process improvement plan, and project document updates (Schwalbe, 2014:295)
- *Performing quality assurance:* During this process, the project is assessed to ensure that the quality standards of the project are adhered to which is done taking responsibility for quality during the course of the project. This process generates organisational process asset updates, change requests, project management plan updates, and project document updates (Schwalbe, 2014:295).
- *Performing quality control:* With the purpose of continuously improving the overall quality, this process involves the continuous monitoring of certain project results thereby ensuring that they comply with the quality standards (Schwalbe, 2014:295).

2.2.2.2.6 Project human resource management

Project human resource management includes the processes required to organise and manage the project team and to effectively utilise all manpower involved with a project (Schwalbe, 2014:342). The following processes are involved (Schwalbe, 2014:343; PMBOK Guide, 2013:255):

- *Developing the human resource plan:* This process involves the development of a project management plan, indicating the activity resource requirements and stipulating the project roles, responsibilities and reporting relationships. The output of this process is a human resource management plan.

- *Acquiring the project team:* This process involves the allocation of the required human resources to the project and ensuring that they dedicate time to the project. The outputs are project staff assignments, resource calendars, and project management plan updates.
- *Developing the project team:* This involves growing the skills of the project human resources to improve project performance. This process generates team performance assessments and enterprise environmental factors updates.
- *Managing the project team:* This process involves monitoring and tracking human resource performance, resolving conflict, communicating project-related matters to the project team, managing changes and improving project performance. Outputs of this process are enterprise environmental factors updates, organisational process assets updates, change requests, and project management plan updates.

2.2.2.2.7 Project communications management

Project communications management includes the processes around communication to ensure the project is executed successfully throughout its life cycle from planning to implementation and sign-off. The objective is to ensure that accurate project information is generated, collected, disseminated, stored and distributed to different stakeholders (Schwalbe, 2014:383). There are mainly five processes in project communications management:

- *Identifying stakeholders:* This process involves identifying everyone involved in or affected by the project and establishing the best ways to manage relationships with them. The outputs of this process are a stakeholder register and stakeholder management strategy (Schwalbe, 2014:384).
- *Planning communications:* During this process, the stakeholder information requirements are established in terms of what information they need by when and how it will be distributed to them. A communications management plan and project document updates are outputs of this process (Schwalbe, 2014:384).
- *Distributing information:* This involves the process of sharing information timeously with the stakeholders. This process mainly generates organisational process assets updates (Schwalbe, 2014:384).
- *Managing stakeholder expectations:* During this process, the communications are managed to comply with the project stakeholder expectations. Organisational process assets updates, change requests, project management plan updates, and project document updates are outputs of this process (Schwalbe, 2014:384).
- *Reporting performance:* This process involves the collection and distribution of project performance information, status reports, progress measurements, and forecasts. Outputs of

this process are performance reports, organisational process assets updates, and change requests (Schwalbe, 2014:384).

2.2.2.2.8 Project risk management

Project risk management includes activities that cover the management and mitigation of project-related risks. It “is the use of risk identification, risk quantification, risk response development, and risk response control in order to analyse, prepare for, and respond appropriately to project risk” (Cleland, 2004:85). There are six main processes involved in project risk management:

- *Planning risk management:* During this process, a decision needs to be made on how to approach and plan the risk management activities. Evaluating all the project documentation enables the project team to identify and analyse the risk management activities. This process mainly generates a risk management plan (Schwalbe, 2014:427).
- *Identifying risks:* This involves identifying the risks that are likely to have an impact on the project and documenting more detail around the risks. The output of this process is the opening of a risk register (Schwalbe, 2014:427).
- *Performing qualitative risk analysis:* This involves the prioritisation of risks according to the effect on the project and the probability of occurring. This process generates updates to the risk register (Schwalbe, 2014:427).
- *Performing quantitative risk analysis:* During this process it is numerically estimated what effect the risks have on the objectives of the project. Further risk register updates are generated by this process (Schwalbe, 2014:427).
- *Planning risk responses:* During this process, steps are taken to increase and improve the opportunities and to minimise the threats in order to meet the project objectives (Schwalbe, 2014:427).
- *Monitoring and controlling risk:* During this process, the risk management processes are performed, in which identified and residual risks are monitored, new risks are identified, risk response plans are carried out, and the effectiveness of the risk strategies is evaluated through the project’s life cycle. The outputs of this process are change requests and updates to the risk register, organisational process assets, project management plan, and project documents (Schwalbe, 2014:427).

2.2.2.2.9 Project procurement management

Project procurement management includes the processes required to purchase products or services needed to complete the project successfully (Schwalbe, 2014:465). There are four main processes involved:

- *Planning procurements:* During this process, it is determined what to procure, by when and how. Decisions regarding outsourcing and types of agreement must be made and the work must be explained to suppliers, contractors and/or service providers. This process generates a procurement management plan, statements of work, make-or-buy decisions, procurement documents, source selection criteria, and change requests (Schwalbe, 2014:465).
- *Conducting procurements:* During this process, sellers, being suppliers, vendors, contractors, and subcontractors are selected, and contracts are signed for their services. Outputs related to this process are selected sellers, procurement contract awards, resource calendars, change requests, and updates to the project management plan and other project documents (Schwalbe, 2014:465; PMBOK Guide, 2013:355).
- *Administering procurements:* This process involves relationship management with sellers, and verifying and monitoring supplier performance and related change management. The main outputs that this process generates are procurement documentation, organisational process asset updates, change requests, and project management plan updates (Schwalbe, 2014:465).
- *Closing procurements:* During this process, agreements are concluded, and all outstanding items are resolved. Outputs of this process are closed procurements and organizational process asset updates.

2.2.2.2.10 Project stakeholder management

Project stakeholder management describes the processes required to manage people, teams or organisations that could be impacted by the project (PMBOK Guide, 2013:391). The following process are involved in project stakeholder management:

- *Identify Stakeholders:* During this process, all entities that can have an impact on or be impacted by the project are identified. Relevant information regarding their interests, project involvement, influence, interdependencies and impact on project success are analysed and documented. Outputs of this process involve a stakeholder register (PMBOK Guide, 2013:393).
- *Plan Stakeholder Management:* During this process, management strategies are developed to communicate with the stakeholders throughout the project according to their

requirements, interests and impact of the success of the project. Outputs involved in this process are stakeholder management plans and project document updates (PMBOK Guide, 2013:399).

- *Manage Stakeholder Engagement:* This process involves engaging with stakeholders to satisfy their requirements, resolve problems as they occur, and to encourage engagement throughout the project. Outputs involve issue logs, change requests, project management plan updates, project document updates and organisational process assets updates (PMBOK Guide, 2013:404).
- *Control Stakeholder Engagement:* During this process, the relationships with stakeholders are monitored with strategies and plans being adjusted for engaging with stakeholders. The outputs of this process are work performance information, change requests, project management plan updates, project document updates and process assets updates (PMBOK Guide, 2013:410).

With the process groups and knowledge areas of PMBOK explained, the advantages and disadvantages will be discussed, including the application of PMBOK in industries.

2.2.2.3 Advantages of PMBOK

The purpose of PMBOK is to guide project managers to complete projects successfully and also to provide knowledge and best practices on how to manage a project (Matos and Lopes, 2013:788). According to Wideman (2005:3) PMBOK is a comprehensive knowledge-based project management guide, covering widely proven practices.

Goncalves *et al.* (2016:575-576) list some of the advantages that PMBOK offers:

- PMBOK delivers detailed planning and control processes;
- Communication is improved between team members and stakeholders;
- PMBOK ensures that more detailed documentation is generated throughout the project life cycle;

Furthermore, Obrutsky (2016:3-4) explains more advantages that PMBOK delivers:

- It is a standard that is globally recognised;
- It offers years of good practices from thousands of project managers;
- It is process-based;
- Projects from different industries can be managed with PMBOK;
- Every process is totally defined with inputs, tools, techniques and outputs.

2.2.2.4 Disadvantages of PMBOK

Despite PMBOK's advantages, shortcomings have also been identified with many of the shortcomings relating to its application in practice (Chin, 2011:36). There is a general false impression that by following the PMBOK processes the project will be a success (Chin, 2011:36). PMBOK does not include any templates or checklists required to compile a project plan (Yeong, 2007:8). Processes are bureaucratic and can have an impact on the project manager's creativity.

PMBOK requires copious documentation and paperwork as the primary communication tool and this creates a heavy administration load on the project manager and project team that could lead to resistance from project team members.

Further disadvantages and shortcomings that are associated with PMBOK are:

- It is too complex for small projects (De Jaeger, 2019)
- According to size, scope, time, budget and quality constraints, the standard has to be modified (De Jaeger, 2019)
- It can be difficult to keep the team connected (De Jaeger, 2019).

In the area of knowledge management, Gasik (2015:1) explains that the biggest shortcoming of PMBOK is absence of the definition of the concept of knowledge. The knowledge contained by an organisation's personnel and project team must be considered, including documented knowledge of an organizational knowledge base (Gasik, 2015:11). Chaves *et al. (*2015:28) also state that both knowledge management and lessons learned are neglected by PMBOK. Project management theory should be prescriptive and indicate how goals can be achieved by means of what actions need to be taken. PMBOK therefore lacks a prescriptive approach.

2.2.2.5 Application of PMBOK

Rebaiaia and Vieira (2014:42) state that much about PMBOK is unique to project management and with quality management PMBOK recommendations are simple to understand and easy to follow. It aims to follow established quality policies and standards to ensure that the project is managed in a quality manner. PMBOK prioritises customers and it does a good job of describing the need to identify the customer, and to determine and manage his/her requirements. PMBOK covers the requirement to engage with all stakeholders, including management, about the project's progress and dictates the process for identifying and acquiring resources for the project. It covers a description of how to review resource requirements, rather than the process of acquiring and managing the resources.

The PMBOK principles as per the PMBOK guide can be used to manage projects across various industries (Tavan, 2016:28). It is recognised as an essential methodology for practitioners in all organisations of different industries and regions of the world (Errihani, *et al*. 2015:201).

PMBOK is a modern PMM and one of the most well-known and established PMMs and can be successfully applied from information technology projects to construction projects. PMBOK is a general methodology that can be applied to any project and is not governed to a particular industry or project size. However, PMBOK is very intense regarding documentation which makes it less suitable for smaller projects (Karaman, 2015:578).

Carton, Adam and Sammon (2008) explain in a case study how PMBOK was applied to implement an ERP system successfully for a pharmaceutical company. Project governance was one of the PMBOK components that assisted with the successful implementation of the project, as it ensured that the ERP project stayed on track, minimised project delays and re-work due to timely problem resolution and decision making. The human resource knowledge area ensured the proper selection of team members which enabled the project to make use of specific local skills at various points in the project; this played an important role in the success of the implementation.

Another example where PMBOK has been applied successfully is described by Jaferi *et al*. (2014) where the efficient use of limited resources was required in a project-oriented organisation. Communications management plays a key role in project-oriented organisations and the communication management field of PMBOK assisted in creating efficient and effective communication channels. This included the collection of data by means on interviews with related personnel, process identification and processes analysis according to the communications management field of PMBOK. This resulted in the identification and removal of additional and unnecessary information in the relevant processes which in turn eliminated many project delays. It also showed an improvement in all the knowledge areas of PMBOK that resulted in time and savings, efficient resource utilisation and increased efficiency and viability of the organisation.

Chin (2012:52-60) did research on projects executed in three different sectors in the United States of America using the PMBOK methodology. These sectors included academic institutions, industrial organisations in managing IT-related projects and the government sector.

This is an indication that PMBOK is used by various organisations from various sectors in the United States of America.

Different organisations have different project processes and it is therefore common for organisations to customise PMBOK processes by only adopting some of the processes that work best for them (Chin, 2012:57). This can be an indication that PMBOK on its own is not a suitable methodology which may be the reason why organisations customise its processes. This is one of the compelling reasons for this study which will propose hybrid methodologies between ASDMs and PMMs in an attempt to improve project success. Based on Chin's (2012) literature, it is not clear if the customisation improved project success, therefore this will also be tested as part of this study by applying hybrid methodologies to various projects.

In the next section, PRoject IN Controlled Environment (PRINCE2) will be described in detail.

2.2.3 PRoject IN Controlled Environment (PRINCE2)

During 1975, Simpact Systems Ltd. developed the first project management method to address the concerns related to project management called PROMPT which is the abbreviation for project resource organisation management and planning techniques (Jaziri *et al.*, 2018:116). In 1979, CCTA (the Central Computer and Telecommunications Agency) adopted PROMPT as a standard and during 1989, PRoject IN Controlled Environment or formally known as PRINCE replaced PROMPT (Matos and Lopes, 2013:789). The Office of Government Commerce (OGC) launched PRINCE2 as the second version in 1996 where both versions were registered as trademarks of the British Government (Jaziri *et al.*, 2018:116).

In 2009, OCG released a major revision where PRINCE2 was made simpler and more customisable. This version also introduced the seven principles that will be covered in the sections to follow. In 2017, the latest version was released by Axelos who took ownership of PRINCE2 in 2013. Although it was regarded as a major version change, the core principles remained the same (ILX Marketing Team, 2017). For this study, the 2017 edition will be used to explain the PRINCE2 methodology.

PRINCE2 explains project management as the planning, delegating, monitoring and controlling of all aspects of the project to deliver projects within the agreed time, cost, quality, scope, benefits and risk (Pawar and Mahajan, 2017:190). The project team's roles and responsibilities are clearly defined, and emphasis is placed on the product that the project must deliver. PRINCE2 is a flexible method that divides the project clearly into phases of the beginning, middle and end, where the final product is delivered at the end of the project.

2.2.3.1 The Structure of PRINCE2

PRINCE2 is a methodology that offers an integrated framework of seven principles, seven themes, seven processes, and a project environment aimed to improve the efficiency of the deployment of a project (Jaziri *et al*., 2018:116; Office of Government Commerce, 2009:5). This structure is demonstrated in Figure 2-2.

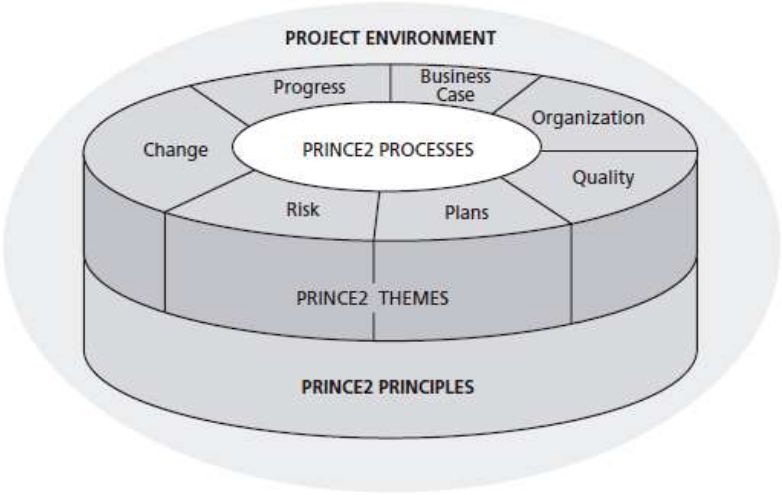

Figure 2-2: Project management process groups (Axelos, 2017:3)

2.2.3.2 PRINCE2 principles

These are principles that guide the project to be delivered successfully within the agreed project objectives. All the principles must be applied to conform to PRINCE2 methodology Axelos, 2017:3). The principles can be defined as follows (Jaziri *et al*., 2018:116-117; Pawar and Mahajan, 2017:191):

- **Continued business justification:** This principle ensures that the project remains in line with the benefits and business objectives which are stipulated in a business case. The business case is reviewed at the end of each project stage to establish if the project is still feasible and if it should continue.
- **Learn from experience:** During this process, previous projects are assessed for any lessons learned that can be applied to the current project. It typically considers where things went wrong and where things went well from previous projects. Lessons learned are taken into consideration during the planning and development stages of a new project.

- **Defined roles and responsibilities:** To ensure project success, all project team members have clearly defined roles and responsibilities to ensure they understand what is required from them.
- **Manage by stages:** Projects are planned, executed and controlled by breaking them down into stages. At the end of each stage, the business case and project plan are reviewed to ensure that it is still feasible to continue with the project.
- **Manage by exception:** Tolerances for all project deliverables are defined to determine limits of delegated authority. When tolerances are exceeded, they are viewed as exceptions and must be escalated to senior management for decision making.
- **Focus on products:** Projects focus on the delivering of products within the defined objectives of scope, time, cost and quality constraints to ensure that the project stakeholder expectations are met.
- **Tailor to suit the project environment:** PRINCE2 offers flexibility in environmental constraints, such as project size, complexity, priority, capability and risk that can be adjusted when necessary. The methodology is therefore adaptable to the project environment.

2.2.3.3 PRINCE2 themes

The PRINCE2 themes describe the aspects of project management that must be addressed continually and integrally throughout the project and provide insight into how the project should be managed (Pawar and Mahajan, 2017:191). These themes can be defined as follows:

1. **Business Case:** The business case explains how the idea is developed into an investment proposition for the organisation and whether the project is desirable, achievable and feasible.
2. **Organisation:** The organisation defines the structure and responsibilities of the project team and project stakeholders to manage the project effectively.
3. **Quality:** Quality demonstrates that the required specifications of a defined product, system or service have been delivered. The goal is to deliver the project according to stakeholder expectation without any complications.
4. **Plans:** Plans indicate the methods of management to deliver the project, system or service. This theme further describes the actions and tasks that are required to develop plans, and it focusses on the project benefits.
5. **Risk:** Risk defines the methodology for identifying, evaluating and mitigating the amount of risk. It describes how uncertainties in plans should be managed.
6. **Change:** This theme defines how changes are identified, evaluated and managed effectively. It further describes how management deals with issues that have a potential impact on the project objectives.

7. **Progress:** This theme describes the monitoring and comparing of actual achievements against planned ones. It further assesses the ongoing viability of plans and ensures that the project is executed according to plan.

2.2.3.4 PRINCE2 processes

PRINCE2 is a processed-based PMM involving seven processes to assist project managers in managing projects successfully. Figure 2-3 is an illustration of how these processes interact to create the processed-based methodology of PRINCE2 (Axelos, 2017:158-159). They describe who is responsible for decision making and by when. The processes form a description of how the seven PRINCE2 themes are applied both before and during the project.

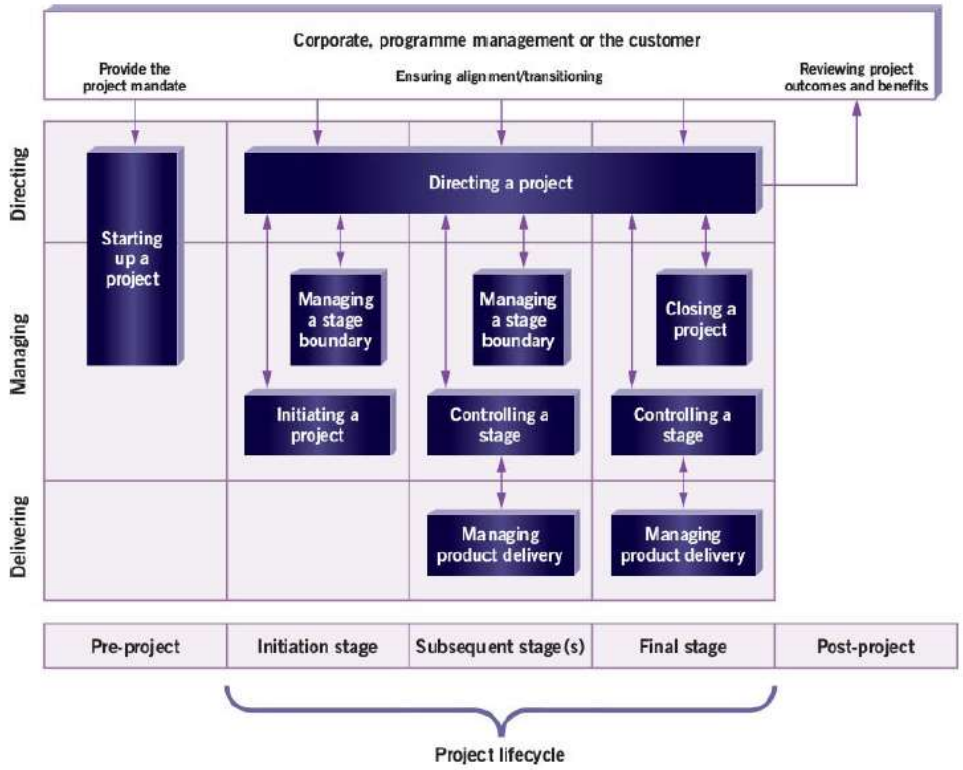

Figure 2-3: The PRINCE2 processes (Axelos, 2017:159)

The seven processes of PRINCE2 and their related objectives will be described in the sections that follow (Axelos, 2017:165-268; Office of Government Commerce, 2009:119-205; Hedeman and Seegers, 2009:81-115).

2.2.3.4.1 Starting up a project

This first process in PRINCE2 is a pre-project process which verifies if a project is viable. The main input to this process is a project mandate that defines the motivation for the project and the expected outcome. The objective of this process is to ensure that (Axelos, 2017:165-166; Office of Government Commerce, 2009:121-131; Hedeman and Seegers, 2009:81-82):

- there is a business motivation and reason for starting the project;
- the necessary authority is given to initiate the project;
- the scope of the project can be defined by means of sufficient information;
- the preferred approach to deliver the project is identified and selected;
- individuals are appointed in project management roles who will commit to perform the responsibilities required in the initiation phase;
- the requirements to initiate the project are planned; and
- a project is not initiated based on poor assumptions thereby wasting unnecessary time and effort.

A project brief with sufficient information is shared with the project board to enable them to decide if a project should be initiated or not. The outline business case is also prepared during this process and together with the project brief requires continuous interaction between all project stakeholders. The activities included in this process are (Office of Government Commerce, 2009:122-123; Axelos, 2017:168-178; Hedeman and Seegers, 2009:82-86):

- *Appoint the executive and the project manager:* The appointment of an executive and project manager is one of first steps to ensure that the project is justified and managed on a day-to-day basis.
- *Capture previous lessons:* Lessons learned from previous projects are recorded during this process and taken into consideration in the design of the project team, outline business case, the project brief and stage plan.
- *Design and appoint the project management team:* This process involves the appointment of the correct individuals to the project team and ensuring that they understand their responsibilities, and accountabilities, including the reporting and communication lines.
- *Prepare the outline business case:* During this process, the outline business case is compiled that indicates the high-level reasons for executing the project and the benefits it will bring to the organisation.
- *Select the project approach and assemble the project brief:* During this process it is decided how the project will be approached. The client's standards, practices and guidelines will dictate how the work will be done.

- *Plan the initiation stage:* This process is time and resource consuming as it requires thorough planning and authorisation before the project can be constructively initiated.

2.2.3.4.2 Directing a project

This is an ongoing process from the beginning to the end of a project. This process is aimed at the project board that is responsible for the success of the project. The project board delegates the management of the project to the project managers who generate reports to the board, thereby enabling them to monitor and control the project (Axelos, 2017:179; Government Commerce, 2009:135-136; Hedeman and Seegers, 2009:87-88). According to Axelos (2017:180), this process must ensure that:

- the necessary authority has been obtained to initiate the project;
- the necessary authority has been obtained to deliver the project's products;
- the project receives the required management direction and control during its life cycle;
- the project remains realistic and feasible;
- the customer or programme management has an interface to the project;
- authority has been obtained to close the project if needed; and
- the plans related to the realisation of the post-project benefits are managed and reviewed.

This process is focussed on the activities of the project board that manages the project by exception. It manages by making use of project reports and directs the project by decision making. The project board should continuously stay in communication with the project stakeholders. A key function of the project board is to give guidance to the project manager and to ensure that the project continues to add business value. The following activities are involved in this process (Axelos, 2017:181-192; Hedeman and Seegers, 2009:88-90):

- *Authorise initiation:* This process involves a decision of allowing the project to be initiated. The project board needs to ensure that all stakeholders are in agreement and that the project manager receives a documented instruction from the executive to continue with project initiation.
- *Authorise the project:* During this process, a decision is made to proceed with the project or not. Confirmation is required that a viable business case exists, the project can be delivered within an acceptable time frame and that suitable measurements are in place that will continuously measure and review the project benefits.
- *Authorise a stage or exception plan:* During this process, the performance of the existing stage is evaluated and the stage plan for the next stage is approved. The project board may request an exception plan for approval if any exceptions occurred during the stage.

- *Give ad hoc direction:* Project board members can give guidance and respond to requests, reports, external influences, concerns, or changes at any point in time.
- *Authorise project closure:* This activity involves the authorisation that the project can be formally closed. During this process, the project documentation is evaluated to determine whether the project objectives have been delivered and if it there were any deviations from the original plan.

2.2.3.4.3 Initiating a project

The reason for this process is to explain to the organisation what is required to deliver the products successfully before spending a significant amount of the budget (Axelos, 2017:195). The objective of initiating a project process is to ensure that there is a general understanding of (Axelos, 2017:196-197; Government Commerce, 2009:149; Hedeman and Seegers, 2009:91-92):

- why the project is executed and what the related advantages and risks are;
- what the project scope of the required work is, including the project deliverable;
- how and during what time frame the final product will be delivered and at what cost;
- who the decision makers are;
- the way in which the required quality will be delivered;
- the manner in which baselines will be determined and controlled;
- the manner in which risks, problems and changes will be determined, investigated and controlled;
- the manner in which progress will be managed;
- the individuals who need information, the formats and time frames; and
- adjusting the corporate, programme management or customer method to be suitable for the project.

The key activities involved in the process are (Axelos, 2017:197-202; Government Commerce, 2009:150-164; Hedeman and Seegers, 2009:92-96):

- *Agree the tailoring requirements:* The manner in which the project is managed and directed may be changed by the project manager in order to identify external and internal factors that have an impact on project delivery. Amendments from the organisations project management approach need to be authorised and documented.
- *Prepare the risk management approach:* This process explains the risk management objectives, the procedures, roles and responsibilities, risk tolerance, methods to be applied and reporting needs.

- *Prepare the change control approach:* The process is necessary to keep control over the management and specialist products. The amount of control required is established by breaking down the products to their smallest form where they can be individually installed, replaced or changed.
- *Prepare the quality management approach:* This approach involves the compilation and maintenance of agreements required to deliver the project according to the agreed stakeholder expectations, including the measurement thereof.
- *Prepare the communication management approach:* This process involves the internal and external communications with regard to how information is sent and received within the organisation.
- *Set up the project controls:* During this process, the extent of the board's control has to be agreed to, including the extent of control required by the project manager related to the required work that has to be performed on the project.
- *Create the project plan:* The schedule indicating the activities that will be performed in agreed time frames are drawn up during this process in the form of a project plan. This is required to refine the business case and to enable the board to control the project.

The recommendation is that the communication management approach is done last, as it must include all communications of the other approaches. The other activities can be executed simultaneously (Axelos, 2017:197).

2.2.3.4.4 Controlling a stage

The purpose of the controlling stage is to assign tasks to be performed, monitor the tasks, address and resolve problems, generate reports to the project board, and take the required corrective actions to deliver the project successfully (Axelos, 2017:215; Government Commerce, 2009:167-168; Hedeman and Seegers, 2009:97).

The objectives of this process are to ensure that (Axelos, 2017:216; Government Commerce, 2009:167; Hedeman and Seegers, 2009:97):

- the delivery of the products is the main focus. Any deviation from the products agreed to at the beginning of the stage is monitored to prevent unplanned changes that can have a negative effect on the project;
- risks and issues are controlled;
- the business is reviewed;
- the required products are delivered according to the committed quality, cost and time; and

- the project team delivers within the agreed boundaries.

The daily management activities of the management stage that the project manager must carry out every day are covered in the controlling a stage process, and apply to each delivery stage of a project during the last phases of each management stage, except the final one. This process is mostly used after project board authorisation, but can also be applied during the initiation stage (Axelos, 2017:216).

The controlling a stage activity includes the following (Axelos, 2017:218-230; Government Commerce, 2009:168-182; Hedeman and Seegers, 2009:98-102):

- *Work package authorisation:* Work performed on the project must always be approved by the project manager. A work package is used for this purpose and normally contain sections from the project plan or stage plan to produce one or more products.
- *Work package status review:* During this process, the status of the work package is frequently assessed.
- *Completed work package receipt:* Teams have to confirm when assigned work has been completed and authorised. Changes must follow an applicable change control process.
- *Management stage status review:* Project events and activities that have already occurred must be continuously evaluated in respect of what was planned and what needs to be done next. A complete view of all activities and tasks regarding progress is therefore very important.
- *Reporting highlights:* A status update is distributed to the project board and other project-related information is shared with project stakeholders at agreed intervals as per the communication management approach.
- *Capture and examine issues and risks:* Problems and risks are registered as they occur and are evaluated before any decision is made related to the project.
- *Escalate issues and risks:* These are problems and events on the project that prevent the project manager from delivering a management stage within the agreed objectives. The project board is promptly notified of such events while the necessary information is gathered to compile an exception report.
- *Take corrective action:* This process involves the implementation of tasks to prevent the project from deviating from the agreed plan and related project objectives.

2.2.3.4.5 Managing product delivery

This process controls the communication between the project manager and the team managers. It controls the flow of work packages from assignment to delivery (Axelos, 2017:235). The

objective of this process is to ensure that (Axelos, 2017:235; Government Commerce, 2009:185; Hedeman and Seegers, 2009:103):

- authorisation is obtained for all work performed assigned to the project team;
- the project objectives are thoroughly communicated and explained to the project team, including team managers and suppliers;
- the final products are delivered according to stakeholder requirements without deviating from the project objectives; and
- updated information is reported to the project manager as agreed to comply with the project expectations.

The following activities are involved in managing the product delivery process (Axelos, 2017:237-240; Government Commerce, 2009:186-190; Hedeman and Seegers, 2009:104-106):

- *Work package acceptance:* The project manager and team manager agree on the deliverable prior to assigning the work package to the project team. This includes the reporting requirements, constraints, procedures and the reasonability of the work package requirements.
- *Work package execution:* This process involves the execution and monitoring of the approved work package according to agreed requirements. The agreed tolerances should not be exceeded, as agreed with the project manager.
- *Work package delivery:* The project manager must be notified when the work package is completed.

2.2.3.4.6 Managing a stage boundary

This process provides the project board with key decision points on whether to continue with the project or not. This process tells what should be done for a stage that has gone outside its tolerance levels (Axelos, 2017:245). The objectives of this process are (Axelos, 2017:245-246; Government Commerce, 2009:194; Hedeman and Seegers, 2009:108):

- provide the assurance to the project board that the products as per the current stage plan have been completed and authorised;
- stage plan preparation to be ready for the next management stage;
- the business case, project plan, project approaches, project management team structure and team role descriptions are reviewed;
- information is shared to assist the project board to establish if the project is still viable;
- all information that can further assist management stages is recorded; and
- approval to initiate the next management stage is requested.

The following activities are involved in this process (Axelos, 2017:248-255; Government Commerce, 2009:194-202; Hedeman and Seegers, 2009:108-112):

- *Next management stage planning:* Close to the end of the existing management stage, the stage plan for the next management plan is planned. Planning is done in consultation with the project board, project assurance, team managers and other relevant stakeholders to ensure that a viable plan is created.
- *Project plan update:* As the project plan is used to measure and track project progress the project plan is updated and should include the actual progress from the current management stage that is coming to an end. The anticipated time frame and cost from the exception plan for the next management stage are also forecasted in this process.
- *Business case update:* Projects and the internal and external factors influencing them change regularly. These changes must be incorporated in the business case and reviewed to ensure that the project remains feasible.
- *Management stage end report:* The project board should be updated on the project at all times which includes updates on the results of a management stage that indicates progress to the project management team.
- *Exception plan production:* Deviation from the agreed project tolerances must get separate approval from the project board. This is obtained by means of exception reports. The board requires exception plans that explain the activities that caused the deviation with the actions required to get the project back on track.

2.2.3.4.7 Closing a project

The purpose of this process is to identify the stage in a project when the project objectives have been met and the deliverables accepted by the client and project stakeholders (Axelos, 2017:259). The objective of closing a project is to (Axelos, 2017:259; Government Commerce, 2009:205; Hedeman and Seegers, 2009:114):

- confirm that the delivered products are accepted by the client or end user;
- ensure that the delivered project can by supported by the host site when the project is terminated;
- measure the performance of the project against the project baselines;
- evaluate all benefits that the project has already delivered and ensure that the benefits management approach includes all post-project benefit reviews; and
- ensure that all open tasks and risks will be addressed with follow-up actions and recommendations.

The following activities are involved in closing a project (Axelos, 2017:262-267; Government Commerce, 2009:206-212; Hedeman and Seegers, 2009:115-118):

- *Planned closure preparation:* The project manager has to ensure that the agreed project objectives have been delivered before the project closure process can be recommended.
- *Premature closure preparation:* Sometimes the project manager has to close the project prematurely as instructed by the project board. When the instruction is received, the project manager must ensure that all valuable work in progress is completed and the remaining outstanding items are reported to corporate programme management or the customer.
- *Products handover:* Before the project is closed, the final products must be transferred to an operational and maintenance environment where the products are handed over to the client, either in various releases or in one final release.
- *The project evaluation:* This process evaluates if the project was delivered successfully or not. The output of this process can be used as inputs in future projects for more accurate estimations.
- *Project closure recommendation:* Once project closure is confirmed by the project manager, a closure recommendation is submitted to the project board, indicating that the project can be closed.

In the next section the advantages of PRINCE2 will be discussed.

2.2.3.5 Advantages of PRINCE2

According to Pawar and Mahajan (2017:194) there are several benefits in implementing PRINCE2:

- The PRINCE2 approach makes it easy to adapt to a project from any stage in a project as it offers a constant, reliable and organised approach. The project stages simplify managing the project and help the project team to keep the workforce motivated and focussed in delivering the project objectives. Management stages and technical stages are distinguished from one another that further simplifies the management of the project (Pawar and Mahajan, 2017:194).
- Communication between project team members and stakeholders are improved which enables the project manager to better control the project and encourage the team members to work efficiently and without conflict. It enables stakeholders to participate and contribute to the project by improved decision making as they are always up to date with the project by means of reports and regular meetings. Good communication channels are established between team members by means of clearly defined roles and responsibilities and

resources utilisation is optimised as only the relevant resources are involved in meetings (Pawar and Mahajan, 2017:194; Hedeman and Seegers, 2009:11).

- The project stages are well documented that simplifies knowledge transfers at various levels of the project team (Pawar and Mahajan, 2017:194).
- The first step in all project stages involves planning, which makes PRINCE2 plan-oriented and delivers plans according to the requirements of various levels of management (Pawar and Mahajan, 2017:194).
- The PRINCE2 is a proven methodology that is globally recognised and is applicable to any project. It is output-driven with the main purpose of product delivery according to the stakeholder requirements of time, cost and quality (Pawar and Mahajan, 2017:194; Hedeman and Seegers, 2009:11).
- PRINCE2 offers clearly defined roles and responsibilities (Pawar and Mahajan, 2017:194; Hedeman and Seegers, 2009:11).
- The methodology empowers project managers to manage the project with more confidence (Pawar and Mahajan, 2017:194).
- Senior management's time is effectively managed as the project board only manages the project by exception (Pawar and Mahajan, 2017:194).
- Offers a general vocabulary and approach (Hedeman and Seegers, 2009:11).
- Industry standards are easily integrated with PRINCE2 (Hedeman and Seegers, 2009:11).
- Reports are thorough (Hedeman and Seegers, 2009:11).
- Encourages continuous improvement and learning (Hedeman and Seegers, 2009:11).
- Offers the re-use of assets and facilitates staff mobility (Hedeman and Seegers, 2009:11).
- Makes accredited training organisations available (Hedeman and Seegers, 2009:11).
- The viability of the project is continuously monitored (Hedeman and Seegers, 2009:11).
- Stakeholders are involved in planning and decision making (Hedeman and Seegers, 2009:11).
- Quality is managed and controlled throughout the project life cycle (Hedeman and Seegers, 2009:11).

In the next section the disadvantages of PRINCE2 will be discussed.

2.2.3.6 Disadvantages of PRINCE2

There are also several issues in managing a project by means of PRINCE2:

- As PRINCE2 is procedure-oriented it makes it difficult to approve a project change in the middle of project development. The change management process in the PRINCE2 methodology is complex compared to other PMMs (Pawar and Mahajan, 2017:194).

- PRINCE2 requires large quantities of documentation during the whole project life cycle that is very time-and resource-consuming (Pawar and Mahajan, 2017:194; Priyanka, 2016:18704).
- The methodology has limited guidance on problem resolution which can have negative consequences on the project, thereby increasing project risks (Pawar and Mahajan, 2017:194-195).
- PRINCE2 does not handle interpersonal aspects of the project manager directly, which can lead to a demotivated team should the project manager not understand the importance of team motivation (Pawar and Mahajan, 2017:195).
- The deliverables do not have clearly defined periods as it depends on the scope (Priyanka, 2016:18704).
- PRINCE2 is a dominant methodology in a project (Priyanka, 2016:18704).

Ghosh *et al.* (2015:14) listed the following gaps in PRINCE2 compared to PMBOK:

- Procurement is not mentioned by PRINCE2;
- PRINCE2 does not stipulate detailed information on the best techniques that should be chosen for a project;
- There is no information about human resources or human resource management;
- PRINCE2 lacks detail around leadership capabilities or soft skills. It briefly lists that the best training programs should be used for the relevant environment.

The following section will explain the application of PRINCE2.

2.2.3.7 Application of PRINCE2

PRINCE2 can be applied to any type of project in any environment, as it consists of a complete set of concepts and project management processes to ensure projects are executed successfully (Office of Government Commerce, 2009:7). According to PRINCE2, a project is temporary by nature and is created to achieve a specified business benefit or objective. When the work has been completed, the project is disbanded (Office of Government Commerce, 2009:24).

PRINCE2 has been applied in many industries, ranging from government and commercial IT to building projects in the United Kingdom (Gist and Langley, 2007:51).

PRINCE2 was implemented at the University of Western Australia library which resulted in the better management of library projects, aided staff with new skills and knowledge and improved

how the library staff worked together. The methodology helped in establishing a more cooperative and collaborative work environment and improved project reporting that increased the level of understanding of the projects (APMG-Australasia, 2005).

In the United Kingdom, it became possible to transfer money between banks in real time by means of a project that was successfully implemented following the PRINCE2 methodology. A payment services organisation used PRINCE2 to plan its project delivery. PRINCE2 assisted to make the organisation's processes more robust and was used to implement the faster payments service project successfully (APMG-Australasia, 2008).

PRINCE2 enabled a global trade credit insurance organisation to reduce their risk, increased their efficiency in their internal processes and improved the desired outcomes of their projects. The roll out of the methodology furthermore has improved team spirit within the organisation (ILX, 2014).

In the next section, agile project management will be described in more detail.

2.2.4 Agile project management (APM)

Aguanno (2004) states that APM can be traced back to the 1980s when the Japanese automobile manufacturers applied them in their product development projects, known initially as light weight methods before it was referred to as agile that pointed to projects with high levels of change. Sutherland (2001) combined the Scrum ASDM with other agile processes in 1993 and applied it to various organisations (Owen and Koskela, 2006:23). Owen and Koskela (2006:23) further state that in 2001 the term "agile" became the adopted term for advanced software development methodologies which originated in the early 1990s. From early 2003, the agile movement started taking off within the information systems industry and APM became more popular.

Agile project management (APM) is an iterative approach to execute projects (Madampe, 2017:27). According to Gustavsson (2016:2), APM is a lightweight and flexible way of managing software development projects through short iterative cycles. As opposed to the traditional methodologies, APM has been introduced as a flexible methodology to respond faster to requirements and successfully deliver a product giving organisations a competitive advantage (Madampe, 2017:28).

APM often gets confused with agile methods, such as Scrum and Kanban due to overlapping concepts, such as executing tasks in iterative cycles (Weinreich *et al.*, 2015). APM is still in its early stages of development and this can be one of the reasons for the misconception.

Agile project management consists out of four focal points (Highsmith, 2010:1):

- the development of the agile era and effect it had on software development projects;
- the values and principles that determined agile project management;
- the specific practices that embraced the principles; and
- practices to assist organisations to accept and adopt agility.

2.2.4.1 Agile project management values

The Declaration of Interdependence founded by the agile project leadership network and the manifesto for agile software development as per Beck *et al.* (2001) are the two main sources for agile values (Highsmith, 2010:15). With the focus on project leaders, the Declaration of Interdependence was developed, whereas the manifesto for agile software development was developed with the focus on software development (Highsmith, 2010:15). Highsmith (2010:17) lists three values that summarise the values in the agile manifesto and the Declaration of Interdependence:

1. Delivering value over meeting constraints
2. Leading the team over managing tasks
3. Adapting to change over conforming to plans

2.2.4.2 Principles of agile project management

APM moves the focus from planning up-front to project execution and does not follow well-defined activities in a linear sequence like traditional project management. It shifts to shared decision-making, self-management and team-learning activities to address problems throughout the project life cycle (Ruhe and Wohlin, 2014:293). APM offers the following principles (Ruhe and Wohlin, 2014:293):

- *Minimum critical specification:* Only what is absolutely necessary and critical to the project success are specified.
- *Autonomous teams:* Autonomous teams manage and monitor their own processes and execute tasks accordingly.
- *Redundancy:* This involves multi-skilling in various functions.
- *Feedback and learning:* This activity is very important for project execution, including interaction activities with the environment.

2.2.4.3 Agile project management delivery framework

The APM delivery framework supports the organisation's business objectives and is organic, flexible and easily adaptable (Highsmith, 2010:81). It also needs to (Highsmith, 2010:81):

- support an envision, explore and an adaptable culture;
- support teams that are self-organising and self-disciplined;
- promote reliability and consistency as far as possible given the level of project uncertainty;
- be flexible and easily adaptable;
- support transparency and visibility into the process;
- include learning;
- include practices that support each phase; and
- include management checkpoints for review.

The delivery framework is illustrated in Figure 2-4 and emphasises delivery (execution) and adaption which is exploratory rather than deterministic (Highsmith, 2010:82).

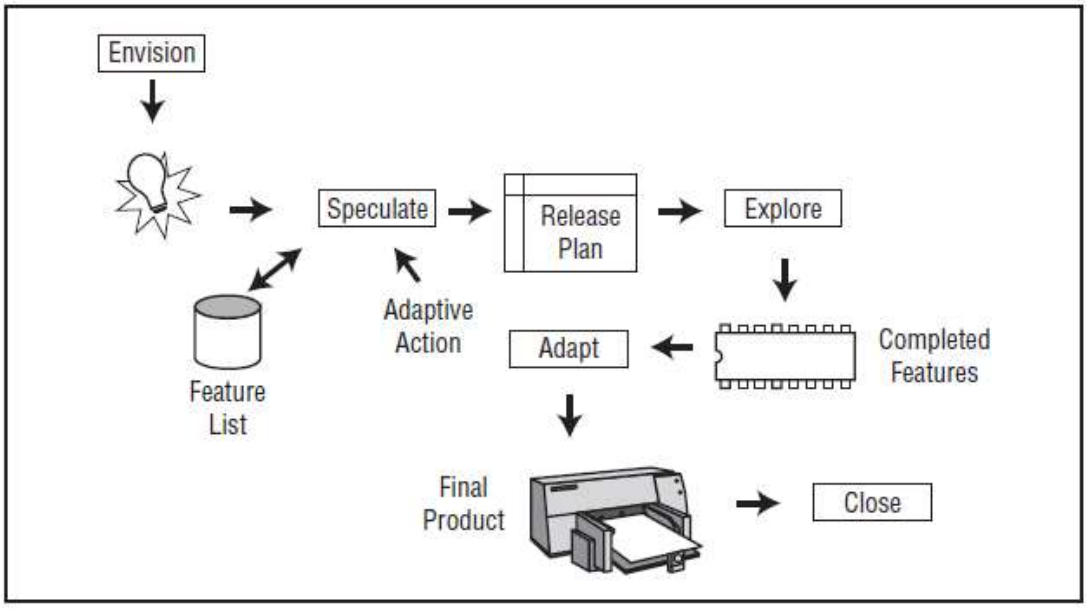

Figure 2-4: APM Delivery Framework (Highsmith 2010:81)

The envision phase replaces the traditional initiate phase to emphasise the criticality of the vision. The speculate phase replaces the traditional plan phase which relates to uncertainty, although it is endeavoured to remove the uncertainty with more planning activities. The focus is to speculate and to adapt compared to plan and build (Highsmith, 2010:82). The traditional manage phase is replaced with the explore phase that focuses on the delivery of iterations that

is a non-linear, concurrent, non-waterfall model (Highsmith, 2010:82). The adapt phase concentrates on the vision, information monitoring and adapting to existing conditions. The last phase is the close phase where knowledge transfer and project closeout take place (Highsmith, 2010:82). A summary of the APM phases is as follows (Highsmith, 2010:82):

- **Envision:** During this phase the product vision, project scope and the project community is determined, including how the team will be working together.
- **Speculate:** This phase involves the development of a feature-based release, milestones, and iteration plan that will be delivering on the vision.
- **Explore:** Deliver tested features in a short time frame, constantly seeking to reduce the risk and uncertainty of the project.
- **Adapt:** Review the delivered results, the current situation, and the team's performance, and adapt as necessary.
- **Close:** Conclude the project, pass along key learnings, and celebrate.

2.2.4.4 Advantages of APM

Changing priorities, team productivity, customer satisfaction, and effective risk management are all important and desirable aspects in managing projects across all industries (Ciric and Gracanin, 2017:332). There are various advantages that APM has across different domains:

- APM offers more flexibility with reduced risk, cost and project time (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:28).
- Deliver what the customer expects, especially in the innovation management and product development domains (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:27).
- Project plans are frequently updated (Ciric and Gracanin, 2017:335).
- Shortcomings are identified early throughout the iterations and incremental testing (Ciric and Gracanin, 2017:335).
- Project plans are compiled in a collaborative manner with shared responsibility and self-organising small teams (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:27).
- Planning time is reduced with improved communication and better project control (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:27).
- There is a better understanding of requirements with improved implementation that makes the project more adaptive to a changing environment (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:28).
- Better team and project effectiveness with higher team morale and flexible team structures (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:27).

- Reduced re-work and more relevant metrics (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:27).
- Increased productivity with improved quality and performance visibility (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:26-27).
- There is more project predictability, transparency and openness of the project (Ciric and Gracanin, 2017:335; Bunsiri and Kumprom, 2016:28).
- Decisions are based on facts and evidence (Ciric and Gracanin, 2017:335).
- Development is iterative and incremental (Ciric and Gracanin, 2017:335).
- Project feedback is continuous (Ciric and Gracanin, 2017:335).

APM is focused on improving customer satisfaction by following an iterative approach and involving the user throughout the whole process. The customer requirements are tested during each iteration and can be adjusted at any time during the development process (Bunsiri and Kumprom, 2016:29).

The agile methodology follows a train-the-trainer approach and promotes collaboration and ownership between the project team, product owner and Scrum master (Bunsiri and Kumprom, 2016:29).

2.2.4.5 Disadvantages of APM

APM has been adopted slowly in the public sector which is evident in the available literature as there are limited studies on the APM adoption in the public sector (Nuottila *et al*., 2016:65). Challenges that organisations face in adopting APM are as follows (Nuottila *et al*., 2016:67-68):

- Agile development focusses on talent and skills of individuals who must be committed to working according to the agile approach that requires different roles and self-discipline.
- Some developers have a fear that their shortcomings will be exposed in an agile team and this causes negativity to APM adoption.
- There is an increased dependency on social skills and teamwork that can affect some individuals negatively.
- Individuals may find it difficult to work according to agile methods if they are used to more traditional methodologies.
- Organisations should be ready for APM; it requires management involvement throughout the project life cycle.
- Developers can lack business-related knowledge required to develop a system according to the business requirements that can become a major problem if the business owner and product owner are not working closely with the agile team.

- For smaller project teams, developers have to understand various technologies and need to have a good understanding of the business; this requires more senior and experienced developers.
- All parties must have a good understanding of how agile methods work.
- Knowledge transfer can be problematic due to the limitation of documentation when team members are replaced.
- The lack of agile-developed recruitment policies complicates the evaluation and appointment process.
- Decentralised teams complicate communication and require effective communication tools.
- Customers need to be actively and continuously involved throughout the process and their availability might not always be possible.
- The integration of agile practices with legacy systems and processes can be problematic and complex.

To summarise, the main disadvantages of APM relate to documentation, personnel skills, experience and commitment, communication and involvement, agile roles and team locations (Nuottila *et al*., 2016:82).

2.2.4.6 Application of APM

APM has the ability of dealing with a continuous changing environment, resource productivity, customer satisfaction and effectively addressing risks that are drawing many project managers across other industries to the adoption of APM methodology (Ciric and Gracanin, 2017:332).

There is currently some confusion in the application of APM, as various literature studies describe the application of APM by means of an ASDM. Azanha *et al.* (2017), for example, did a case study on a Brazilian pharmaceutical company applying Scrum as APM. Similarly, Schwaber (2004) and Mahnic (2005) published literature on using Scrum for APM. In these literature examples, Scrum is referred to as APM, but based on the explanation and definition of an ASDM in section 3.2 and Scrum described in section 3.2.1, it is categorised as an ASDM instead of APM.

There are limited sources that describe the application and effectiveness of APM which is due to the fact that APM is one of the latest PMM's compared to PMBOK and PRINCE2 covered in sections 2.2.2 and 2.2.3 (Ciric and Gracanin, 2017:336). It can be concluded that although APM is a growing PMM, it has not been fully established in all industries and domains (Ciric and Gracanin, 2017:336).

In the next section PMBOK, PRINCE2 and APM will be compared in detail.

2.2.5 PMBOK, PRINCE2 and APM comparison

There is no project management methodology that works for all projects, not even within the same company, project type or industry (Alexander, 2017). Alexander (2017) indicates that the selection depends on various factors, such as industry, complexity, maturity of the organisation, culture of the organisation or the involvement of the customer throughout the project. Project management involves the application and integration of various project management processes that follow a sequential process of initiation, planning, executing, monitoring and controlling, and closing (PMBOK Guide, 2013:48-49). To support this process there are various project management methodologies, such as PMBOK, PRINCE2 and APM covered in the previous sections. There are various other project management methodologies (PMMs) to consider, such as critical chain project management (Leach, 2014:1-49), Six Sigma (Monteiro de Carvalho, 2013:40-49), and Kanban (Ingason, Gestsson and Jonasson, 2013:4-5) just to name a few. PMMs are meant to enhance project effectiveness and to increase chances of success (Vaskimo, 2011:2; Joslin and Müller, 2015:1377; Karaman and Kurt, 2015:572).

One important factor of implementing projects successfully is the use and application of a PMM during the life cycle of the project. Two of the most widely used PMMs in the world are PMBOK and PRINCE2 (Karaman and Kurt, 2015:572).

In this section, these three PMMs will be critically compared to identify the characteristics of their best practices in order to select an appropriate PMM that will be integrated with ASDMs.

Various studies have been done where PMBOK and PRINCE2 were compared from various perspectives (Karaman and Kurt 2015:574) and also where PRINCE2 or PMBOK were compared to APM (Ghosh *et al*., 2015; Salameh, 2014). A high-level comparison between PMBOK, PRINCE2 and APM is described in Table 2-5.

Table 2-5: The High-level comparison between PRINCE2, PMBOK and APM (Adapted, Karaman and Kurt, 2015:574)

Feature	PRINCE2	PMBOK	APM
Definition	Structured PM methodology	Standard and guide	Uncertain

Table 2-5: The High-level comparison between PRINCE2, PMBOK and APM (Adapted, Karaman and Kurt, 2015:574) (continued)

Practical vs. Comprehensive	Practical, focuses on critical areas	Comprehensive	Practical
Themes and Knowledge Areas	Seven Themes	10 Knowledge Areas	-
Processes and Activities	Seven Processes and 35 Activities	Five Process groups and 47 Processes	General Framework
Principles	Seven Principles	-	Four Principles
Techniques	Only PRINCE2 specific techniques are explained	Covers techniques for each process	Limited
Interpersonal Skills	Not covered	Covered	Not Covered
Focus	Business Case and Product	Customer Requirements	Customer Requirements
Role of the project board	Calls for a project Board to provide Oversight	Only suggests the role the sponsor should be playing	Only suggests project team roles
Organisational assets and Environmental factors	Partly covered	Strongly integrated with processes.	Not covered
Management Principle	Manage by exception	-	Flexibility and easy adaptability

The 10 knowledge areas of PMBOK are clearly defined as a complete set of concepts, terms and activities that make up an area of specialisation. The seven themes of PRINCE2 define PRINCE2 as various aspects of project management that must be continually addressed. There is still an uncertainty surrounding APM definition due to misconceptions that exist regarding agile methods. Both PRINCE2 and APM are practical and focus on critical areas where PMBOK is more comprehensive. APM does not promote any themes or knowledge areas, but focusses mainly on an agile approach. PMBOK and PRINCE2 both have clearly defined processes and activities whereas APM follows a general project management delivery framework. APM covers limited techniques in terms of the delivery framework where PMBOK and PRINCE2 follow specific techniques related to project management. The role of the board varies between all three PMMs where PRINCE2 requires a formal project board to oversee the

project compared to only a project sponsor in PMBOK and no board role in APM. Organisational assets are strongly integrated with the PMBOK processes that are not very prominent to non-existent in the PRINCE2 and APM. APM has very flexible management principles due to the agile approach it follows compared to PRINCE2 and PMBOK.

APM is still in its early stages of development and has become very popular in the management of Information Technology projects, but has not been fully adopted in other industries (Stare, 2014:295; Ciric and Gracanin, 2017:336). For this reason, including the misconception that exists in the industry between APM and agile system development methods, APM will not be evaluated further and not be selected as a PMM that will be used to integrate with ASDMs. The following sections will critically compare PMBOK and PRINCE2 to select a preferred PMM that will be used further in this study to integrate with ASDMs.

The themes covered by PRINCE 2 in a PMBOK context are listed in Table 2-6 and the principles covered by PMBOK in a PRINCE2 context are listed in Table 2-7 (Karaman and Kurt, 2015:575).

Table 2-6: PMBOK Coverage of PRINCE2 (Karaman and Kurt, 2015:574)

PRINCE2 Themes	PMBOK Coverage	Comments
Business Case	May be periodically reviewed for multiphase projects	PRINCE2 has stronger emphasis on continuous Business Case
Organisation	Stakeholder Management, Human Resources management	PMBOK has stronger mechanisms
Quality	Quality management	-
Plans	Scope, time, cost management	-
Risk	Risk management	-
Change	Monitor and control process group	-
Progress		

Based on the results of Table 2-6 PRINCE2's quality theme is comparable with the quality management of PMBOK. Similarly, the plans theme of PRINCE2 is comparable with PMBOK's scope, time and cost management. The risk theme also compares with PMBOK's risk management knowledge area (Siegelaub, 2004:2). Lastly, the change and progress themes can be compared to the monitor and control knowledge areas of PMBOK. PRINCE2's business

case theme covers much more detail compared to PMBOK, but the defining of roles and responsibilities and human resource management are more detailed in PMBOK than in PRINCE2.

Table 2-7: PRINCE2 Coverage of PMBOK (Karaman and Kurt, 2015:574)

PMBOK	PRINCE2	Result
Integration Management	Partly covered (Plans, change, progress)	PMBOK has a complete integration mechanism
Scope Management	Plans	-
Time Management		-
Cost Management		-
Quality Management	Quality, Configuration Management	-
Human Resource Management	Partly covered by Organisation Theme	PMBOK is stronger
Communications Management	Partly covered	PMBOK has more detailed concept
Project Risk Management	Risk	-
Procurement Management	Not covered	Only covered by PMBOK
Stakeholder Management	Partly covered (Organisation)	PMBOK is stronger

The results of Table 2-7 indicate that PMBOK's scope, time and cost management knowledge areas can be compared to PRINCE2's plans, quality and configuration management and risk themes, whereas human resource management is only covered partially by PRINCE2's organisation theme. PRINCE2 does not cover procurement management (Siegelaub, 2004:2). PMBOK's stakeholder management is more intense, as PRINCE2 only covers it partially with the organisation theme. PRINCE2 does not cover integration management completely compared to PMBOK's thorough mechanism.

In an Information Technology context, PMBOK and PRINCE2 can be compared as follows (Karaman and Kurt 2015:576-577):

Table 2-8: IT project characteristics and management frameworks (Karaman and Kurt, 2015:577)

Characteristic	PRINCE2 and PMBOK Comparison
Requirements Maturity	Comparable
Development Stability	Comparable
Project Size	Although both practices are tailorable, PRINCE2 is a methodology and it has a more practical approach for small size projects.
Risk Clearness	Comparable
Scope Clearness	Comparable
Client's Commitment	PMBOK is customer requirements focused, so provides better mechanism for projects with high client commitment.
Team Relationship	PMBOK has stronger communication management mechanism, and it also covers interpersonal skills which is essential for creating the environment for effective communication. For large size and new created teams, it can be useful to apply the PMBOK approach.
Team Size	
Method of Contracting	Independent of the contract method, PMBOK has the procurement management knowledge area and explained techniques for effectively managing contracts, and procurement processes are integrated inside process groups. For this reason, it is useful to apply the management knowledge area of PMBOK for the projects that require a high level of outsourcing and comprehensive contracts.
Outsourcing	
Stakeholders Flexibility	Flexible stakeholders may influence the project from many perspectives, for both frameworks. On the other hand, it is useful to apply the stakeholder management area and stakeholder engagement techniques of PMBOK for the projects that have a high level of stakeholder engagement.

Based on Table 2-8 it is evident that PMBOK is a more comprehensive approach with thorough tools and techniques. PRINCE2, on the other hand, also has features, such as the integration of project board activities and management by exception that are not covered in PMBOK (Karaman and Kurt, 2015:577).

The main differences between PMBOK and PRINCE2 are listed in Table 2-9 (Karaman and Kurt, 2015:577).

Table 2-9: Differences between PMBOK and PRINCE2 (Karaman and Kurt, 2015:577)

Features	PRINCE2	PMBOK
Practical Approach	✓	-
Comprehensive Approach	-	✓
Project Board Activities	✓	-
Management by Exception	✓	-
Procurement Management	-	✓
Product Focus	✓	-
Customer Requirements Focus	-	✓
Detailed Techniques	-	✓
Interpersonal Skills	-	✓

Table 2-9 shows that both PMMs have unique features that can be beneficial to organisations in executing projects. The differences between PMBOK and PRINCE2 can be summarised as follows (Karaman and Kurt, 2015:577):

1. PMBOK's mechanism for integration of processes is stronger than that of PRINCE2.
2. PMBOK processes integrate organisational process assets and environmental factors better than those of PRINCE2.
3. The communication mechanism of PMBOK is stronger than that of PRINCE2's.
4. Product delivery management activities in terms of the project team are covered by PRINCE2 which is not the case with PMBOK.

To summarise, PRINCE2 is more focussed for small size IT projects whereas PMBOK is better for IT projects that have a large client commitment, are complex, with large project teams where there is a high level of outsourcing, comprehensive contracts and where stakeholder management is more intense. Therefore, both PMMs can be beneficial to an organisation, depending on what the organisation regards as important and what fits it best. The organisation culture, national standards and bureaucracy with regard to project management all play an important role in selecting the most applicable PMM.

For the purpose of this study, the project management body of knowledge (PMBOK) methodology will be used as the basis to compare and integrate the selected ASDMs as the processes described in the PMBOK are accepted as good practice within the project

management discipline and it is also an internationally recognised standard which provides the fundamentals of project management as they apply to a wide range of projects (Jamali and Oveisi, 2016:142). Ilies, Crisan, and Muresan (2010:48) also refer to PMBOK as having well-known guidelines and an international standard. Another reason for selecting PMBOK as the integrating PMM is because it has been used to deliver all types of project, including Information Technology projects for more than 20 years (Seymour and Hussein: 2014:236).

In this chapter, details related to the PMMs that are investigated in this study were covered and a preferred PMM that will be used for the integration the selected ASDMs was identified. In the next chapter, agile software development methodologies, including details about the selected number of ASDMs will be discussed.

CHAPTER 3: AGILE SOFTWARE DEVELOPMENT METHODOLOGIES

In this section, the definition of a software development methodology (SDM) will be discussed in more detail, including the concept of agile software development. Four of the most popular ASDMs will then be discussed and compared whereafter it will be investigated how these ASDMs can be integrated with PMBOK.

3.1 Definition of a software development methodology

According to Iivari, Hirscheim and Klein (2000-2001:180-181), there is no concise definition of an SDM that is universally accepted. This is because of the different opinions around the term "methodology" that researchers claim has no place in information systems as it means a "science of methods" (Schach, 1997:23). There are challenges related to the use of the term systems development method which is categorised into two types of inconsistency, namely "scope" and "category" problems (Iivari and Maansaari, 1998:502-503). Avison and Fitzgerald (2003:20) stated that the term methodology compared to the term method is a broader concept due to various characteristics that are not covered by the term method and a methodology also includes a philosophical view. Given the complexity in defining SDM, Huisman and Iivari (2006:32) define an SDM as a combination of the following:

- *A systems development approach*: This approach involves a philosophical view to which the methodology refers. It consists of the set of goals, guiding principles, fundamental concepts, and principles of the SDM process that drives interpretations and actions (Iivari, Hirscheim and Klein, 1998:165-166).
- *A systems development process model:* This involves a representation of the sequences of stages through which the system evolves (Wynekoop and Russo, 1993:182).
- *A systems development method:* With this method, a minimum of one phase of systems development is completed, consisting of a set of guidelines, activities, techniques, and tools that are based on a specific philosophy and the target system (Wynekoop and Russo, 1993:182).
- *A systems development technique:* This technique involves a procedure with a possible prescribed notation to perform a development activity (Brinkkemper, 1996:276).

Besides the above-mentioned arguments, Gonzalez-Perez and Henderson-Sellers (2005:173) define an SDM as "the specification of the process to follow, as well as the work products to be generated, plus consideration of the people and tools involved, during a software development effort". Similarly, Young (2013:1) defines an SDM as a way of managing a software development project that addresses elements, such as selecting features for inclusion in the current version, when software will be released, who works on what, and what testing is done.

Another more recent definition of an SDM is by Saravanan (2017:27) who defines an SDM as “a standard process followed in an organization to conduct all the steps necessary to analyse, design, implement and maintain information systems”. Pressman (2014:81) defines a software development methodology (SDM) as a framework for effective modelling and documentation of software-based systems. It is “a collection of values, principles, and practices for modelling software that can be applied on a software development project in an effective and light-weight manner“ (Pressman, 2014:81).

Understanding the components and definition of an SDM is important for understanding the ASDM. The term “agile system development methodology” will be explained in the following section.

3.2 Agile software development methodology

Agile software development methodologies are not new concepts as software development following some kind of iterative approach can be tracked back as far as the 1960s with adaptive software development emerging in the 1970s (Larmen, 2003:25-26, 49). Agile software development originated in 2001, when a group of 17 software developers met to discuss rapid software development methods. Together they published the manifesto for agile development. The manifesto for agile software development can be found on the agile alliance website (Beck *et al*., 2001). To the manifesto proclaimed, they value:

- individuals and interactions over processes and tools;
- working software over comprehensive documentation;
- customer collaboration over contract negotiation; and
- responding to change over following a plan.

The manifesto for agile software development is based on twelve principles:

1. Customer satisfaction by early and continuous delivery of valuable software
2. Welcome changing requirements, even in late development
3. Working software is delivered frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done

11. Best architectures, requirements, and designs emerge from self-organising teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

One of the major benefits in agile methods is that development is done in the smallest incremental form that can be tested by users who provide immediate feedback on what is working and what is not. Therefore, the relation between developers and customers become a growing conversation in which incremental steps of two to four weeks create requirements that are tested and adjusted where necessary. It is based on an iterative and incremental development process where all the team members are involved, including the customer as shown in Figure 3-1 (Rebaiaia and Vieira, 2014:42; Al-Zewairi *et al.*, 2017:74-75). ASDMs attempt to adapt to changing requirements while minimizing costs and delivering quality software. At the end of each iteration, the portion of the system is controlled, tested, presented to the customer and main stakeholders, and can be integrated into the final project deliverable, if accepted by the customer (Rao, Naidu and Chakka, 2011:35-36).

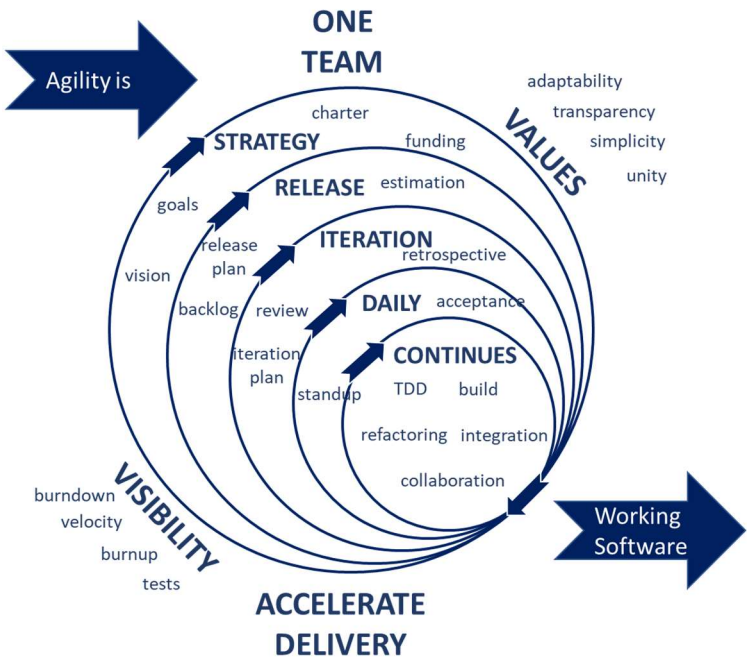

Figure 3-1: Development process of ASDM (Rebaiaia and Vieira, 2014:42)

Several methods have been highlighted in ASDM literature, including extreme programming, dynamic systems development method, Crystal Methods, Feature-Driven Development, Lean Development and Adaptive Software Development, amongst others. These methods differ in respect of specific techniques. They have several characteristics in common, including short iterative lifecycles, quick and frequent feedback from customers, and constant learning. According to Stoica *et al.*, 2013:72-73), Scrum, extreme programming (XP), dynamic systems

development method, and Feature-Driven Development (FDD) are software development methodologies that are classified amongst the main agile software development methodologies. Dingsoyr *et al*. (2012:1213) also refer to these software development methodologies as examples of agile software development methodologies, amongst others. Rebaiaia and Vieira (2014:42) state that Scrum and XP are by far the most widely adopted agile software development methodologies. Therefore, in this dissertation, only Scrum, XP, DSDM and FDD will be covered.

3.2.1 Scrum

Scrum is an SDM that was developed by Jeff Sutherland in 1993 with the goal of becoming a development and management methodology that is based on agile principles (Pressman, 2014:78; Permana, 2015:199). In 1995, Jeff Sutherland and Ken Schwaber developed and formalised the Scrum development process at the OOPSLA'95 (Object-oriented programming, systems, languages and applications) Conference (Sutherland, 2012:6). Tavares *et al*. (2017:1) define Scrum as "a structural framework used to manage complex products that allows the integration of various processes or techniques".

The term Scrum originated from the sport rugby which involves eight players from two teams, known as the pack or forward pack, binding together in three rows and interlocking with the opposing teams' forwards. At this point, the ball is fed into the gap between the two forward packs and they both compete for the ball to win possession. Scrum SDM adopted some of the rugby strategies, such as a holistic team approach, constant interaction among team members and unchanging core team members (Cho, 2008:191).

Schwaber and Beedle performed further development on the Scrum methods (Pressman, 2014:78). Scrum focusses on a strategy where the product development function is flexible with a development team that works closely together towards agreed objectives (Permana, 2015:199). Scrum focuses on project management in situations where it is difficult to plan ahead, with an importance on feedback mechanisms. Thakur and Kaur (2013:89) contend that Scrum is unique because it introduced the idea of practical experience rather than theories, known as "empirical process control".

According to Sutherland (2012:14), Scrum is an iterative approach for software development. The Scrum life cycle consists of four phases, namely planning, staging, development, and release (Larmen, 2003:113). Software requirements are captured and prioritised by the product owner and are referred to as stories, and all the stories make up the product backlog (Despa, 2014:45). The product backlog exists during the full life cycle of the project and can be referred

to as the product roadmap, containing all the customer requirements according to priority. Use cases or “user stories” are mostly used to explain the items listed in the product backlog. Story boards facilitate good communication and teamwork during meetings. Figure 3-2 shows an example of a product backlog.

					New Estimates of Effort Remaining as of Sprint...					
Item	Details (wiki URL)	Priority	Estimate of Value	Initial Estimate of Effort	1	2	3	4	5	6
As a buyer, I want to place a book in a shopping cart (see UI sketches on wiki page)	...	1	7	5						
As a buyer, I want to remove a book in a shopping cart	...	2	6	2						
Improve transaction processing performance (see target performance metrics on wiki)	...	3	6	13						
Investigate solutions for speeding up credit card validation (see target performance metrics on wiki)	...	4	6	20						
Upgrade all servers to Apache 2.2.3	...	5	5	13						
Diagnose and fix the order processing script errors (bugzilla ID 14823)	...	6	2	3						
As a shopper, I want to create and save a wish list	...	7	7	40						
As a shopper, I want to to add or delete items on my wish list	...	8	4	20						

Figure 3-2: Product backlog (Sutherland, 2012:17)

Based on priority, the items listed on the product backlog are broken down into separate tasks which are documented on a document called the sprint backlog. The sprint backlog therefore reflects the requirements seen from the point of view of the development team. It consists of all the tasks in which the user stories are broken down. Figure 3-3 displays an example of a sprint backlog.

				New Estimates of Effort Remaining as of Day...					
Product Backlog Item	Sprint Task	Volunteer	Initial Estimate of Effort	1	2	3	4	5	6
As a buyer, I want to place a book in a shopping cart	modify database		5						
	create webpage (UI)		8						
	create webpage (Javascript logic)		13						
	write automated acceptance tests		13						
	update buyer help webpage		3						
	. . .								
Improve transaction processing performance	merge DCP code and complete layer-level tests		5						
	complete machine order for pRank		8						
	change DCP and reader to use pRank http API		13						

Figure 3-3: Sprint backlog (Sutherland, 2012:20)

Scrum dictates a process where projects progress via a series of one- to four-week iterations called sprints (Thakur and Kaur, 2013:89). Sprints deliver a working version of the application

(Despa, 2014:45). The sprints end on a specific date and are never extended, even if the work has not been completed. A sprint involves a cross-functional team selecting work that is based on priorities according to the product backlog and committing to completing the work at the end of the sprint (Sutherland, 2012:14). The risk of excessive costs is limited to one calendar month of cost (Schwaber and Sutherland, 2017:9). Daily Scrum meetings of a couple of minutes are held where progress is discussed, and team efforts measured. The project team demonstrates to the stakeholders what has been built at the end of each sprint and feedback is incorporated into the next sprint. Scrum depends on artefact transparency, as decisions required to control risk and to optimise project value are based on the state of the artefacts (Schwaber and Sutherland, 2017:17). The aim of Scrum, in a software context, is to deliver shippable code that is developed, compiled and tested at the end of each sprint. These practices result in improved teamwork and delivers better quality products (Khosravi *et al*., 2017:174).

The framework of Scrum is mainly team-based with defined roles, events, artefacts and rules. Rebaiaia and Vieira (2014:42) explain that the primary roles committed to the project are the product owner, development team and Scrum master with a responsibility allocated to each of them. The product owner, for example, represents the stakeholders and the owners have to ensure that the organisation gets value from the project. The development team of about seven people has to produce a shippable deliverable with each iteration, and the Scrum master needs to buffer all distracting forces from the team, thereby making sure that the team delivers what it is supposed to deliver. He therefore must keep control and ensure that the development team remains focussed on its tasks. The Scrum master keeps track of the team activities and the project owner receives feedback at the end of each sprint (Despa, 2014:45).

Projects are delivered in an iterative and flexible manner so that at the end of every sprint of work there is a tangible deliverable to the organisation. One of the benefits of Scrum is that the quality and project risks are highlighted quickly in the project (Permana, 2015:198). The Scrum methodology is illustrated in Figure 3-4 (Thakur and Kaur, 2013:89).

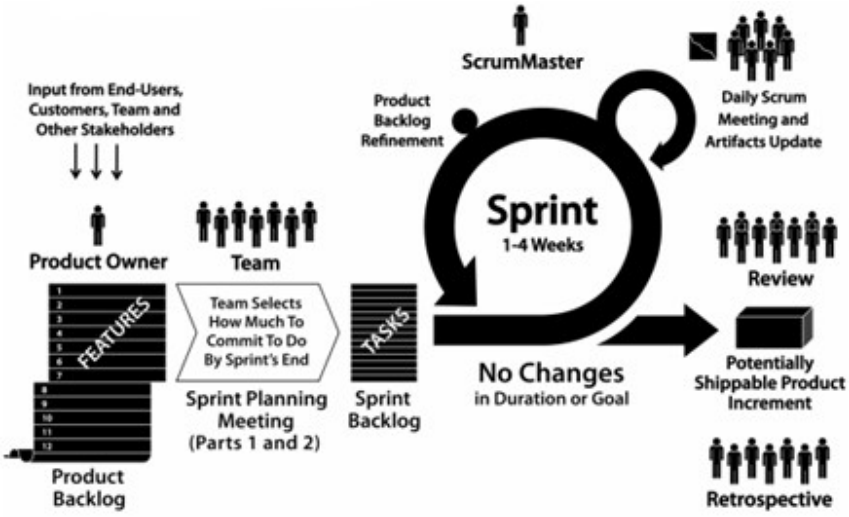

Figure 3-4: Scrum methodology (Sutherland, 2012:14)

3.2.1.1 Scrum Process

The Scrum process is carried out as follows (Permana, 2015:200):

1. *Product backlog determination*

   The product backlog is prepared by the product owner by identifying all the requirements related to the project. The Scrum master determines the features that will be built, based on the requirements and priorities. The features list becomes the product backlog.

2. *Sprint planning*

   During this process, the Scrum team discusses and evaluates the product backlog items and features required by the product owner. The team determines the number of hours required for each feature, estimates the cost and allocates the tasks to the different team members (Popli and Chauhan, 2011:147). All the tasks make up the first sprint and the team starts to work on the product backlog items according to the priorities. Changes identified during development are postponed to the next sprint.

3. *Daily stand-up meetings*

   Daily stand-up meetings are conducted to monitor the performance of the team members who are working on the features being developed. The sprint completion time gets reviewed, based on the work that has been completed. The development tasks are monitored in terms of tasks that are planned compared to the tasks that are completed and still in progress. These development tasks make up the story board related to the specific sprint and are also called the sprint board. Problems identified during development and testing are listed as part of the story board.

4. *Sprint review*

During this process, the team demonstrates the features that were developed to the product owner and client.

5. *Sprint retrospective*

   This process involves the identification and discussion of tasks that are problematic and that cannot continue to the next sprint. Tasks that are successfully completed and that can continue to the next sprint are also discussed.

The Scrum process is illustrated in Figure 3-5.

Figure 3-5: Scrum process (Permana, 2015:200)

3.2.1.2 Advantages of Scrum

- Scrum works well for small projects and is ideal for smaller teams and for projects where the tasks are of a shorter duration. This makes the project more flexible and adaptable to change (Rao, Naidu and Chakka, 2011:42).
- Requirements are prioritised in a well-structured manner (Rao, Naidu and Chakka, 2011:42).
- Scrum helps to deliver products in short cycles, so the client gets to see project deliverables early in the project (Despa, 2014:52).
- The short cycles enable quick feedback to the team and the client (Despa, 2014:52).
- Scrum makes it possible to easily and rapidly adapt to changes (Despa, 2014:52).

Khosravi *et al*. (2017:176) introduced Scrum to several talented software developers who were selected by means of a nomination process to participate as Scrum development team members. The study showed the following positive aspects regarding the adoption process:

- **Early wins**: This indicated that the technical experts could adjust themselves comfortably to the Scrum practices.
- **Positive atmosphere of review meeting**: The customer approved all sprint reviews in a positive light, although it was felt that the quality of the product increments could be improved.
- **Good retrospective meeting**: Retrospective meetings were constructive and without any complaints about the adoption process and related requirements.
- **No technical problem**: Team members adopted some of the agile project management tools without and problems or challenges.
- **No communication challenge**: As team members were working closely together in one place the communication was thorough and of good quality.
- **No resistance against change**: There was no resistance to change in the development methods they had to adopt.

3.2.1.3 Disadvantages of Scrum

- Close customer collaboration is not always possible as the customer is not on site and near the project team (Rao, Naidu and Chakka, 2011:42).
- Development teams are small which restricts improved team dynamics (Rao, Naidu and Chakka, 2011:42).
- There is a limitation of documentation (Despa, 2014:52).
- Small teams require experienced developers to be used on the project (Despa, 2014:52).
- Project estimates are restricted at the beginning of the project for large implementations which makes cost estimates inaccurate (Despa, 2014:52).

As per the previous section, the study done by Khosravi *et al*. (2017:176) also introduced several negative aspects in the study where talented software developers participated as Scrum development team members:

- **Lack of enough collaboration**: Although communication was good, there were complaints about inadequate collaboration between team members.
- **Lack of enough commitment**: There was a lack of commitment in delivering the agreed goals and meeting attendance was not great.
- **Product Owner issues**: The product owner experienced some problems in collaborating with some of the developers.

3.2.1.4 Application of Scrum

As an ASDM, Scrum plays a key role in agile software development and has been accepted largely in the software development industry (Rahman *et al*., 2018:20.1).

There are many case studies where Scrum has been applied in the successful delivery of numerous projects. Some of these case studies are as follows (http://www.scrumcasestudies.com):

- *Net Health Scales Scrum with the Nexus Framework*: Scrum was adopted by Net Health in 2014 to assist with the completion of two large software development projects. Due to Scrum's iterative approach, it streamlined the requirements phase. Following the Scrum methodology also improved timeline predictions and flexibility in scope management.
- *Discovercracow.com:* Scrum improved project delivery by optimising the tasks and removing unnecessary functionality that was not really needed in the projects.
- *European bank:* Scrum enabled the bank to resolve delivery problems, thereby achieving a stronger customer focus.
- *Major Airline:* This airline's senior management had the objective to transform the organisation to be more agile thereby responding faster to changes. A mobile applications development team adopted Scrum and shortly thereafter, more Scrum teams were formed across various business units.
- *Vodafone Turkey:* Adopting Scrum enabled Vodafone to reduce its time to market and also improved its quality of service that gave it a competitive advantage. The Scrum teams changed organisational structures and processes by creating more teamwork and transparency that led to success.
- *Google:* Agile practices that contributed to the coordination of development teams improved the process and made it more predictable. It started with two project teams which worked well and addressed many of the previous issues. Making use of spike enabled them to determine the implementation effort.
- *Schneider Electric:* Scrum enabled the development team to accelerate progress and simultaneously improve the team spirit and motivation levels. The incremental and iterative development improved cross-disciplined embedded systems development.
- *Adobe:* Adopting Scrum improved code quality, delivered better features for their customers and enabled them to work at a sustainable pace that improved their collaborations and communication. The perceptions of customers involved in new releases also improved.

In the next section, extreme programming will be discussed and explained in more detail.

3.2.2 Extreme programming (XP)

XP was created in 2000 by Kent Beck in an attempt to save a project that had been declared a failure (Vijay and Ganapathy, 2014:62; Al-Zewairi *et al.*, 2017:85). Vijay and Ganapathy, (2014:62) state that XP is a methodology that makes development more enjoyable to developers, reduces risk, is predictable, flexible, scientific and effective. Ramy Krishna *et a*l. (2011:21) refers to XP as a lightweight and informal approach for software development that is the most efficiently applied when the team environment is highly collaborative with co-located teams. The customer is part of the development team and is involved in all processes of the project (Bahrudin *et al.*, 2014:511). XP addresses the needs of small teams who need to satisfy unclear and continuously changing requirements (Mahajan and Kaur, 2010:699). XP is a frequent release development methodology where developers work in pairs for continuous code review (Al-Zewairi *et al.*, 2017:85). Teamwork is emphasised by XP, as all team members are equal partners working in a collaborative manner. It promotes a self-organising team to solve problems efficiently (Mahajan and Kaur, 2010:701). According to Bahrudin *et al.* (2014:511), XP is "an agile development method that seeks to satisfy the customer through early and continuous delivery of valuable software." Extreme programming aims to reduce cost by dictating small, iterative releases of planning, analysing, and designing throughout the entire software development process, instead of following the process once for the entire project (Despa, 2014:43-44). According to Al-Zewairi *et al*. (2017:80), it enforces pair programming where two developers use the same computer. One is writing code and the other is supervising, with them changing roles at regular intervals. There is a strong emphasis on test-driven development. Developers are required to write unit tests before writing the actual code in order to eliminate as many errors as possible. XP supports collaborative code ownership where developers are allowed to change any code sequence, even if it was not written by them (Despa, 2014:44). The project owner prioritises the tasks. Short stand-up meetings are held daily to report problems (Ramy Krishna, *et al*. 2011:23).

XP gives very robust, high quality software, at the expense of twice the development cost. As per Kent Beck (Beck, 2005:2), the XP method is based on the following values:

- **Communication**: This is one of the most important aspects contributing to the success of a project. In XP, communication is based on practices such as unit testing, pair programming, and task estimation.
- **Simplicity**: This value involves always seeking the simplest solution, with the focus of delivering a workable solution. XP also reduces the number of artefacts, i.e. user stories, plans and coding, to simplify the process even further. Simplicity and communication work

together. A simple system is easier to communicate than a complex system (Mahajan and Kaur, 2010:701).

- **Feedback**: This value involves having concrete knowledge about the current state of the system. It relates to various dimensions of system development. Feedback from the system in terms of unit tests gives developers immediate feedback on the system status. There is feedback from the customer, as the functional tests are compiled by the customer who will get thorough feedback on the current system status and feedback from the team that estimates immediately on time durations when receiving new requirements from customers (Mahajan and Kaur, 2010:702-703).
- **Courage**: This value involves admitting flaws in the system and taking immediate corrective action. It enables developers to easily make changes to their code when required and also to discard code when it is obsolete, irrespective of the time and effort that went into it (Mahajan and Kaur, 2010:703).

3.2.2.1 XP process and practices

There are five processes during the XP life cycle (Ramy Krishna, *et al*. 2011:21-22):

- **Exploration phase**: During this phase, regular meetings are held where user stories (requirements) are collected and updated. User stories are captured on user cards that are understandable by all parties involved.
- **Planning phase**: This phase involves the prioritisation of requirements that enables the project to move into the next step.
- **Iterations to release phase**: In this phase, analysis, design and testing are done by means of pair programming which makes it the most important phase.
- **Productionising phase**: During this phase, small releases are demonstrated to the customer to obtain his/her approval.
- **Maintenance phase**: In this phase, the project is updated after customer approval is obtained.
- **Death phase**: This is where the final product is released.

Figure 3-6 illustrates the different phases of extreme programming.

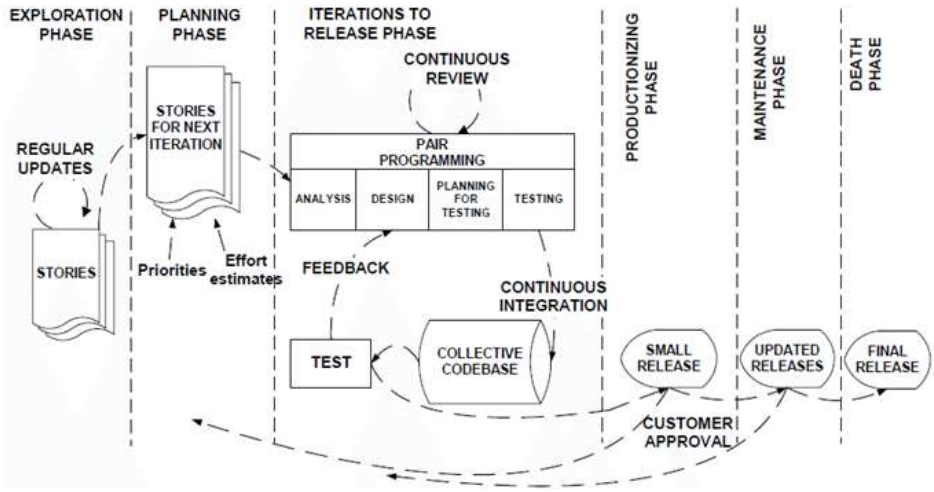

Figure 3-6: Life cycle of the XP process (Ramy Krishna *et al*. 2011:22)

XP is based on the following twelve core practices, also indicated in Figure 3-6 (Ramy Krishna *et al*., 2011:22-23):

1. **Planning**: This process involves the planning activities for the next release. During this process, the business and developers identify, estimate and prioritise the requirements for the next iteration.
2. **Small releases**: The system must be up and running by having very short cycles and quick releases. The aim is to deliver workable software as soon as possible by breaking the release into smaller increments. This enables the team to deliver more frequently and to identify risks early in the project.
3. **Metaphor**: This is a story that the project team can tell about how the system works. The naming convention of classes and methods is made as simple as possible. All development is driven by the simple story of how the whole system works.
4. **Simple design**: The simplest possible design to get the job done should be used. Complexity should be removed wherever possible. Design is a continuous process that involves release planning and quick, continuous design sessions throughout development.
5. **Testing**: Testing is a continuous activity. Programmers develop software by writing tests first, and then code that complies with the requirements as per the tests. Acceptance testing is performed by the customers to ensure that the required features are delivered. Automated regression unit testing reduces the risk of defects when changing source code (Krishna, 2011: 22).
6. **Refactoring**: Duplication of code is eliminated in the coding sessions. Automated test cases simplify this process and any changes made to the system must retain simplicity.
7. **Pair programming**: All programmers are working in pairs sitting at one computer. This ensures that all code is reviewed as it is written which produces better design, testing and

better code. It also helps with knowledge transfers and improved communication throughout the team.

8. **Collective ownership**: Anyone can change any code anywhere in the system at any time. No piece of code is owned by anyone, which encourages new ideas in all areas of the project.
9. **Continuous integration**: The system is built many times a day, as soon as a task is finished. Changes are integrated daily and the development team always work on the latest version of the software.
10. **On-site customer**: A representative of the customer organisation should be available full-time to the project team to answer questions. Ideally, a customer representative must be designated to the project team on a full-time basis.
11. **40-hour Weeks**: The maximum working hours per week for developers must not exceed 40 hours. XP dictates that programmers should not tire themselves out by working excessive hours on the project. An acceptable and comfortable working pace is determined by the team.
12. **Coding standards**: Everyone must code to the same standards that makes coding more effective.

According to Stoica, Mircea and Ghilic-Micu (2013:74), XP covers four main activities:

1. Coding, which is the main activity and takes priority over all tasks.
2. Testing, which is an ongoing activity where all modules must be tested thoroughly.
3. Listening, that promotes continuous communication between the programmer and the customer to ensure the requirements are fully understood.
4. Design, that dictates the development of an optimal system architecture which leads to an efficient system and reduces unnecessary dependencies between modules.

In the next section, the advantages of extreme programming will be discussed in more detail.

3.2.2.2 Advantages of XP

- The time for the project to reach the production environment is very short and less time is wasted on useless features. Guess work during the planning phase is reduced and programming estimates are done before implementation (Despa, 2014:52; Krishna, 2011: 22-24).
- Development is incremental and software gets delivered sooner due to small releases which offer frequent feedback to the team and the customer thereby emphasising teamwork and communication (Despa, 2014:52; Krishna, 2011: 22).

- Releases are efficiently tracked reducing the probability for the overall project to fail (Despa, 2014:52; Krishna, 2011: 22).
- The number of errors is reduced with smooth code integration and a simple design is encouraged that involves regular redesigning via refactoring and frequent testing. Testing is done more thoroughly by means of unit testing (Despa, 2014:52; Krishna, 2011: 22-24).
- Responsibility for quality is emphasised and there is a continuous measurement and monitoring that take place (Krishna, 2011: 24).
- Pair programming promotes continuous reviews with collective ownership and being two developers helps keeping them focussed. Collective ownership also enables developers to take responsibility for the whole system instead of parts of the system (Krishna, 2011: 22).
- XP promotes a 40-hour week that promotes the well-being of developers and prevents ineffectiveness that mostly starts after 40 hours (Krishna, 2011: 23).
- The customer is on-site and involved and that helps with quick and informed answers to real development questions which ensures that only what is required gets developed. Functionality, in addition, is prioritised correctly (Krishna, 2011: 23).
- Coding is done according to standards that minimise time wasted on reformatting other developer's code and internal commenting. Coding standards also promote clear and unambiguous code (Krishna, 2011: 23).

In the next section, the disadvantages of extreme programming will be discussed in more detail.

3.2.2.3 Disadvantages of XP (Despa, 2014:52)

- XP produces a small quantity of documentation. Additionally, in XP projects, the defect documentation is not always good. Lack of defect documentation may lead to the occurrence of similar bugs in the future.
- There is a resistance from developers to work in pairs, as it results in coding delays, can be frustrating and less convenient. Programmers who work or learn better alone find it difficult to work in pairs.
- XP is not ideal if programmers who are separated geographically as programmers need to work in pairs and attend regular meetings.
- Developers are reluctant to write tests prior to coding.
- XP requires frequent meetings to be conducted. The amount of time that teams spend in meetings can drain many hours by the end of a project.
- There is limited commitment to a well-defined product that can lead to a degree of reluctance on the part of the product owner.

- XP does not measure code quality assurance. It may therefore cause defects in the initial code.

3.2.2.4 Application of XP

XP is a very popular approach for simple, fast and easy-to-use software development. It is one of the most familiar and important methodologies amongst agile software development methodologies (Krishna, 2011: 21). XP encourages software development based on the values of simplicity, communication, feedback, and courage by focussing the whole team on simple practices and continuous feedback mechanisms thereby enabling them to know what the current status of the project is and to adapt the practices to their unique situation (Krishna, 2011: 24).

Vijay and Ganapathy (2017:62) summarises XP as "a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop software." Customer involvement is highly promoted, as the first priority is the customer, followed by continuous testing, continuous feedback and planning and close teamwork to deliver working software at frequent intervals (Vijay and Ganapathy, 2017:62).

Layman, Williams and Cunningham (2004) evaluated the impact of adopting XP in a case study at an airline solutions company. Two releases of the same products were compared where the first release was before the team adopted XP and the second one was after XP had been in use for approximately two years. The latter project resulted in a fifty percent productivity increase, a sixty-five percent improvement in pre-release quality and thirty-five percent improvement in post-release quality.

In the next section, the dynamic systems development method (DSDM) as an ASDM will be discussed in more detail.

3.2.3 Dynamic systems development method (DSDM)

The DSDM was developed in 1994 by project practitioners of the DSDM Consortium who aimed to improve Rapid Application Development (RAD) by creating a more advanced version of the RAD framework (Flora, 2014:3631). "Stapleton" presented first the Dynamic System Development Methodology in 1995 and in 2007 it became a generic framework for project management and the delivery of solutions (Hiwarkar *et al*. 2016:82; Anand and Dinakaran, 2016:3435).

DSDM is an agile method that sets time, quality, and cost at the beginning of the project where customer involvement is critical to setting priorities (Anand and Dinakaran, 2016:3435). It provides an organised framework for projects with tight time constraints. According to Fahad *et al*. (2017:261), it is similar to Scrum and XP, but deals with projects that have fixed time constraints. With other ASDMs, time and resources keep changing, but functionality remains the same. With DSDM the time and resources remain constant and functionality keeps changing throughout the project. DSDM also prioritises schedule and quality over functionality and uses the MoSCoW method of prioritisation, which breaks a project down into four different types of requirement, namely "must have (M)", "should have (S)", "could have (C)" and "won't have (W)" (Sani *et al*., 2013:38).

As with other SDMs, DSDM also follows an iterative approach, but has similarities to the Pareto principle (Fahad *et al*., 2017:261). This means that 80% of the functionality of the project is delivered in 20% of the time. The remaining 20% of the functionality is left for later iterations. This SDM is ideal for continuous changing requirements as all requirements are mostly not known at the beginning of a project (Fahad *et al*., 2017:261).

According to Sani *et al*. (2013:37-38), DSDM has four main phases, namely feasibility, functional model iteration, design and build iteration and implementation as shown in Figure 3-7.

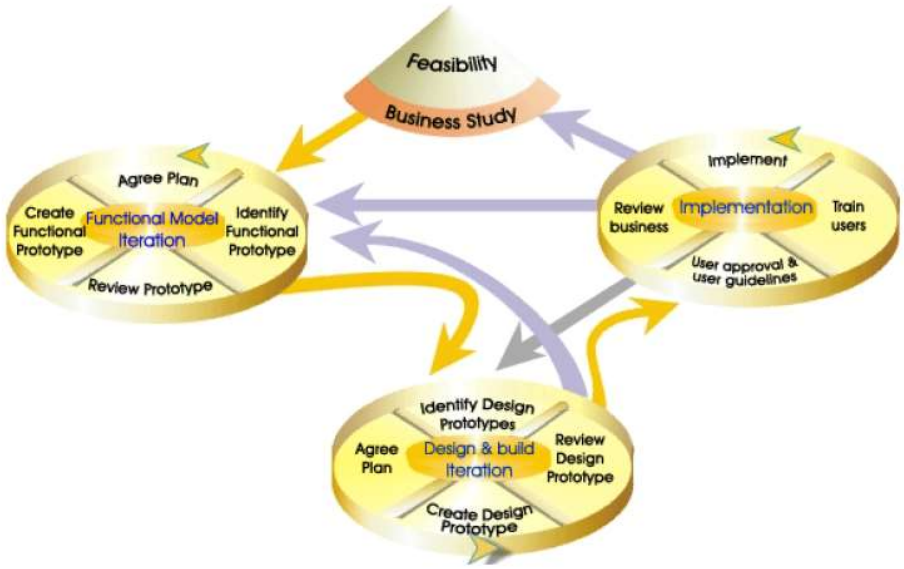

Figure 3-7: DSDM phases and sub-phases (Sani *et al*., 2013:38)

3.2.3.1 DSDM phases

DSDM can be divided into two main phases, namely the pre-project phase and post-project phase.

Pre-project phase

This phase aims to give the project an initial direction that consists of several sub-phases (Sani *et al*., 2013:38; Anwer *et al*., 2017:6). These sub-phases are:

- *Feasibility study*: During this phase, important questions are asked, for example if the project is worth doing and if it is executable from a technical perspective. The type of project, resource availability, issues related to the organisation and risks involved are all taken into consideration to determine the feasibility of the project.
- *Business study*: This phase is facilitated by workshops where experts from the customer's side sit together with the development team to identify and prioritise the system requirements. It involves the identification of business processes that must be automated, high-level functionalities and non-functional requirements, prioritisation and an outlined development plan.
- *Functional model iteration*: During this phase, software components are developed as prototypes that are based on the high-level functionalities identified in the business study.
- *System design and build iteration*: This phase involves system engineering that delivers the tested system.
- *Implementation*: During this phase, the system is taken live from the development environment to the production environment and also includes training and handover activities.

Post-project

During this phase, the post-implementation review is maintained to establish if the project was successful and delivered what it was supposed to deliver. Typically, questions, such as "Has the required benefits been delivered?" are answered.

DSDM follows various principles which are commonly found in other agile software development methodologies and are as follows (Sani *et al*., 2013:38):

1. Users are actively involved which cannot be avoided.
2. DSDM teams are empowered to make decisions.
3. The main focus is to deliver products frequently; this is more important than maximising quality.

4. The product must be suitable for the business purpose which is crucial for the approval and acceptance of the project deliverables.
5. Development is iterative and incremental.
6. All changes and modifications can be reversed during the development process.
7. Requirements are baselined at a high level.
8. Testing happens continuously during the life cycle of the project and is integrated.
9. Stakeholders must have a collaborative and cooperative approach.

One of the major benefits of DSDM is quick and in-time delivery of the project with a reduction in the cost of the project. It involves collaborative teams that are self-organised and emphasises continuous collaboration between all parties. Because of prototype development, requirements can be added after iterations. Feasibility studies are done before the project starts in order to minimise the risk of project failure (Fahad *et al*., 2017:261). Management commitment is crucial to the success of applying this SDM.

3.2.3.2 DSDM Techniques

DSDM relies on various techniques to develop and deliver applications. Some of the following techniques are used to develop applications (Anand and Dinakaran, 2016:3435; Sani *et al*., 2013:38):

- MoSCoW prioritisation – More important tasks need to be considered before less important tasks due to the minimal amount of time that is available. Functionality is therefore prioritised, based on the following criteria:
    - Must Have – the most important aspects that are crucial to the system are considered.
    - Should Have – important aspects for the business solution.
    - Could Have – useful aspects, but can be postponed for a while.
    - Won't Have – aspects that can easily be left until later.
- Prototyping – Prototypes are used to ensure that all parties involved have a thorough understanding of the various aspects of the system.
- Facilitated Workshops that offer the following benefits:
    - Creates an environment that is ideal for the formation of ideas.
    - Creates an awareness of decisions made by all parties involved, including parties that have an interest in the project.
    - Decisions can be made by a wider range of stakeholders.
    - It allows for accurate and quick decision making.
- Timeboxing

- An interval of a maximum of six weeks during which a specified set of tasks have to be completed.
- The short time span makes estimation more accurate than longer estimations.
- Several tasks can be included within time boxes which have to deliver a product.
- Timebox tasks can easily change if the priority changes during the timebox iteration which makes it ideal for a rapid response to business requirements.
- Functionality is rather compromised to deliver in time

To summarise, DSDM covers project management, estimating, prototyping, timeboxing, configuration management, prioritised requirements, implementation, testing, quality assurance, roles and responsibilities, team structures and tool environments (Anand and Dinakaran, 2016:3435; Sani *et al*. 2013:38).

In the next section, the advantages of DSDM will be discussed in more detail.

3.2.3.3 Advantages of DSDM

- DSDM delivers basic functionality quickly with more functionality being delivered at frequent intervals integrated with agile principles (Anwer *et al*., 2017:7).
- It has a framework that is easily adoptable, incorporating best practices from other approaches (Anwer *et al*., 2017:7).
- It offers good guidelines for other project aspects, for example risk management, development techniques and project management (Anwer *et al*., 2017:7).
- DSDM focusses on addressing effectively the business needs, as users are highly involved in the development process which enables them to have a proper understanding of the project (Despa, 2014:52-53).
- Post-project implementation performance assessments are done and offer an easy access by developers to end users (Despa, 2014:52-53).
- Complete documentation (Despa, 2014:52-53).
- DSDM deals effectively with project cost and time management (Nazir *et al*., 2017:4).

In the next section, the disadvantages of DSDM will be discussed in more detail.

3.2.3.4 Disadvantages of DSDM

- DSDM is costly to implement, as it requires users and developers alike to be trained to employ it effectively (Chapram, 2018:513).

- Project criticality is not considered by DSDM (Anwer *et al*., 2017:7).
- There is no specific guidance about challenges related to the size of the team and length of the iterations, as DSDM is only a framework (Anwer *et al*., 2017:7).
- DSDM requires large project teams as it has multiple roles to cover which can also create administration issues during the development process (Despa, 2014:52-53; Anwer *et al*., 2017:7).
- DSDM is a relatively new model, therefore, it is not very common and easy to understand and requires very skilled developers (Despa, 2014:52-53).

In the next section, the application of DSDM will be discussed in more detail.

3.2.3.5 Application of DSDM

DSDM requires a considerable amount of training and resource involvement which makes it costly and less feasible for smaller projects and applications (Chapram, 2018:513). In a DSDM project, the performance requirements are expressed as Must, Should, Could and Won't (MoSCoW). DSDM provides the flexibility to ensure on-time and within budget delivery of an acceptable and fit-for-purpose solution rather than a perfect one.

DSDM is often compared to other ASDMs and it is found that although the tools used in DSDM are not as powerful as in Scrum, for example, the DSDM method is more complete because of its very specific roles and principles.

DSDM provides more project management aspects than other ASDMs by using project management-related phases, such as the pre-project, implementation and post-project phases.

3.2.4 Feature-driven development (FDD)

FDD was founded by Jeff De Luca when he was working on a large project and realised that even by using all the available resources, his knowledge and traditional strategy of software development, he could not solve the problem of delivering the project in time. He consulted with Peter Coad and they came up with the concept of feature-driven development together. In 1999, they published their book titled 'Java Modelling in Colour with UML' where they announced the FDD model. In 2002, Stephen Palmer and Mac Felsing published their book, "A practical Guide to Feature-Driven Development", that presented FDD in its more generalised form (Anwer *et al*., 2017:4).

FDD combines some practices recognised in the industry into one methodology (Vijay and Ganapathy, 2017:62). FDD is an iterative and incremental methodology that divides the software into many different features, and then builds each feature separately (Al-Zewairi *et al.*, 2017: 88-89). Anand and Dinakaran (2016:3434) explain that a feature is a small valued function expressed in client-valued terms and can be completed within two weeks. Where the features take more time to complete it is broken in smaller feature sets. Development is done in the form of architectural, object models and sequence diagrams in which Unified Model Language (UML) models are used throughout (Anwer *et a*l., 2017:9). Anand and Dinakaran (2016:3434) illustrate the FDD methodology in Figure 3-8. FDD follows eight practices, namely domain object modelling, development by feature, individual class ownership, feature teams, inspection, configuration management, regular builds and progress reporting (Nawaz *et al*., 2017:53).

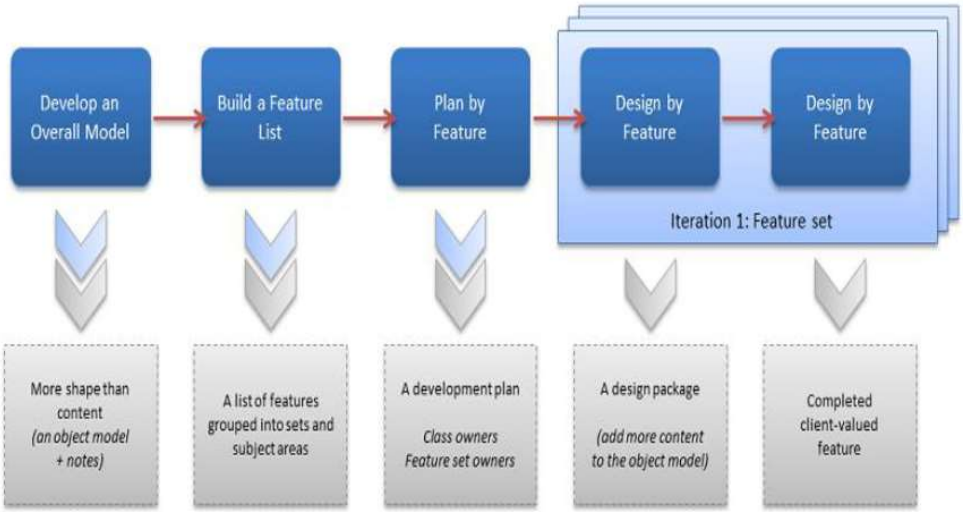

Figure 3-8: Feature-driven development methodology (Anand, 2016:3434)

3.2.4.1 FDD Processes

The FDD life cycle consists of the following five iterative processes (Anwer *et al*., 2017:4-6):

Develop the Overall Model: During this process, detailed walkthrough meetings are held by the project team to work through all the requirements and features with the objective of defining the context and scope of the project that will be delivered. The chief programmer and domain expert are the main participants who decide on the project scope. The chief programmer represents the technical team and the domain expert represents the client (Nawaz *et al*., 2017:56).

Each domain area is covered in detail, followed by the development of various object models which are then reviewed. Requirements are gathered by means of story cards where the domain expert writes the story for every feature that is required in the system. The story cards explain the system functionality without technical detail. Priority is given to the features that must be completed in the early release. The chief programmer compiles use case diagrams for the requirements stipulated in the story cards. Figure 3-9 illustrates an example of a use case diagram (Nawaz, 2017:56).

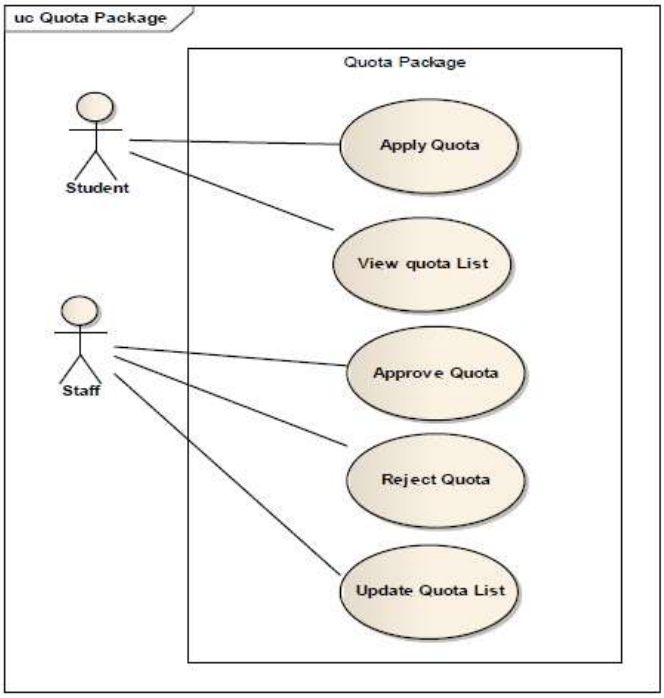

Figure 3-9: Example of a use case diagram (Firdaus *et al*., 2013:323)

Build the Feature List: The second step of FDD is to build the feature list. The development team identifies the features for each domain of the system to be developed by using the documents produced in the previous phase. Features belong to specific domains and are called feature sets. At the end of this phase, one document is generated, namely the Feature List which includes the requirements associated with each feature. Usually, each feature requires up to two weeks of development. In case the feature requires more time, then this step is decomposed into smaller steps (Nawaz, 2017:57).

Plan by feature: The third step is to plan by feature. This phase consists of project planning activities whereby the chief programmer develops a project plan and assigns responsibilities to team members by taking feature dependencies into consideration. Priorities are assigned to features while considering complexities, workloads of teams and risks (Anwer *et al*., 2017:4-5).

At the end of this phase, one document is generated, namely the project plan which includes detail regarding iterations, resources and class owners.

Design by Feature Set: The fourth step is the “Design by Feature Set”. The objective in this step is to produce the design of each feature set. Here an object model of the system is developed by the chief programmer and class owners. All design-related activities are reviewed and inspected for approval.

Build by Feature: Finally, the fifth step, “Build by feature”, involves packaging a smaller batch of features from the feature set decided in step four and developing the code for those features and unit testing them. Therefore, activities, such as coding, code inspection, unit testing and integration testing take place during this phase and are performed iteratively (Nawaz, 2017:53). The batch of features is known as a Chief Programmer Work Package (CPWP) and should be selected in such a way that it can be completed by a single feature team in less than two weeks.

FDD defines six key roles, namely the project manager, chief architect, development manager, chief programmer, class owner and domain experts (Nawaz, 2017:53). There are also a release manager, language guru, build engineer, tool smith and system administrator as supporting roles. FDD also has testers, deployers and technical writers as three additional roles. Quality is a focus point throughout the development process and is not just to test the code, but also includes things, such as coding standards, measuring audits and metrics in the code (Nawaz, 2017:53-54).

In the next section, the advantages of FDD will be discussed in more detail.

3.2.4.2 Advantages of FDD

- FFD enables multiple teams who can work simultaneously on the same project (Despa, 2014:52-53).
- FDD is an excellent solution for big and complex projects where larger teams are executing the project and especially when dealing with critical situations (Despa, 2014:52-53).
- Tracking of project progress is performed well with good reporting capabilities that keep project members up to date regarding overall status of the project (Despa, 2014:52-53).
- FDD is a methodology which helps new team members become familiar with the system in a short period of time that is easily adoptable and simple to understand (Despa, 2014:52-53).

- FDD is a highly adoptive development model with great emphasis on designing and modelling aspects of the project (Anwer, *et al*. 2017:5).
- FDD teams focus especially on quality throughout the development phases (Anwer *et al*., 2017:5).
- One- to four-week iterations help to get quick feedback about developed product (Anwer *et al*., 2017:5).

In the next section, the disadvantages of FDD will be discussed in more detail.

3.2.4.3 Disadvantages of FDD

- Requires large project teams, as it has multiple roles to cover (Despa, 2014:52-53).
- Requires very skilled developers (Despa, 2014:52-53). FDD requires highly skilled experts in designing and modelling (Anwer *et al*., 2017:5).
- The FDD prescription does not specifically address project criticality (Flora, 2014:3631).
- FDD does not provide any guidance about requirement gathering, requirement analysis and risk management therefore there is a need for some supporting methods (Anwer *et al*., 2017:6).
- FDD does not address the issues related to project criticality (Anwer *et al*., 2017:6).

3.2.4.4 Application of FDD

According to Nawaz *et al*. (2017:53), FDD delivers software according to client-valued features and follows a process-oriented and agile client-centric approach. It is adaptive and incremental by nature that delivers software development projects in short iterations. FDD has a main focus on quality, but is less responsive to changing requirements, has a reliance on experienced personnel and is not ideal for smaller projects.

3.2.5 ASDM comparison

The four SDMs described in this study will be compared by evaluating their different characteristics, strengths and weaknesses which will be integrated with the selected PMM. The ASDM comparison is described by in Table 3-1 (Adapted, Despa, 2014:51-54).

Table 3-1: ASDM comparison (Adapted, Despa, 2014:51-54)

Methodology	Characteristics	Strengths	Weaknesses
Scrum	• Iterative development • Timebox approach, known as Sprints • Daily meetings to assess progress, known as Daily Scrum • Self-organising development team • Tasks are managed using backlogs; product backlog and sprint backlog	• Delivers products in short cycles • Enables fast feedback • Rapid adaptation to change	• Lack of documentation • Requires experienced developers • Difficult to estimate the overall effort required to implement at the beginning • Large projects; thus, cost estimates are not very precise
Extreme programming	• Pair programming • Unit testing • Fast consecutive releases • Collective ownership • On-site project owner • Open workspace • Project owner decides the priority of tasks	• Application gets very fast in the production environment • Frequent releases of working code • Reduced number of bugs • Smooth code integration • Continuous feedback from the project owner	• Lack of documentation • Developers reluctance to pair programming • Developers reluctance to write tests first and code later • Requires frequent meetings • Lack of commitment to a well-defined product leads to project owner reluctance
Dynamic systems development method	• Iterative development • Prioritisation of tasks • Timebox approach • Non-negotiable deadlines • Strict quality standards set at the beginning of the project • Project team and project owner share a workplace (physical or virtual) • Test early and continually	• Focusses on effectively addressing the business needs • Post project implementation • Performance assessment • Complete documentation • Active user involvement	• Requires large project teams and it has multiple roles to cover • Requires very skilled developers
Feature-Driven	• Iterative development	• Multiple teams can work simultaneously	• Individual code ownership

Development	• Application is broken down into features • No feature should take longer than two weeks to implement • Uses milestones to evaluate progress	on the project • Scales well to large teams • Good progress tracking and reporting capabilities • Easy to understand and adopt	• Iterations are not well defined

In the next section, ASDMs and the selected PMM, namely PMBOK, will be integrated to compile and develop a hybrid ASDM.

3.3 ASDM and PMM integration

Although agile software development methodologies look practical and attractive in achieving results quickly, there is often the question of whether agile software development methodologies can deliver projects without following a complete project management process. Salameh (2014: 61-65) makes mention of agile software development methodologies that execute responsibilities at project level, as well as iteration level, meaning that ASDMs can follow a combination of ASDMs and project processes.

Although agile software development methodologies are mostly preferred over traditional ones in respect of several aspects, there are some shortcomings in terms of documentation that can lead to increased development costs where beginner developers or new members of the team cannot fully understand the project and tend to delay some iterations (Stoica, Mircea and Ghilic-Micu, 2013:70-71).

The goal of a project management methodology is to increase the probability of successful project delivery (Spundak, 2014:945; Kerzner, 2009:63). Furthermore, other benefits of a project management methodology include better project planning, avoiding mistakes, reducing cost, reducing risk, meeting project schedules, identifying and correcting errors early and avoiding excessive documentation (Charvat, J. 2003:18). However, project management methodology in itself is not always an adequate or even a necessary precondition for project success (Introna and Whitley, 1997:31-32). According to Wells (2012:57), project management methodologies should enable project managers to achieve higher project success rates, but the extent to which this is achieved is not known as projects still fail and the impact of project management methodologies on project success is unknown. Introna and Whitley (1997:31-32) continue that if the organisation and project team do not fully understand the project scope, context and background knowledge, there are no tools or techniques within any project

management methodology that will guarantee project success and without background knowledge, project management methodology will be unsuccessful. One of the preconditions for the successful methodology usage is coherence with other company processes (Charvat, 2003:97-98), which is the reason why many organisations developed their own project management methodology. The conclusion is that one project management methodology is not always enough and several possible project management methodologies can exist within the organisational context, or at least the possibility of adapting a project management methodology to specific projects (Introna and Whitley, 1997:37).

Based on the above, it is important to investigate the different characteristics of ASDMs and PMMs to identify integration points that can be applied in order to increase project success. In Table 3-2, Spundak (2014:945), adapted, the following differences are noted by looking at each approach in more detail:

Table 3-2: Difference between traditional and agile approach (Adapted, Spundak, 2014:945)

Characteristic	Traditional Approach	Agile Approach
Requirements	Clear initial requirements; Minimal changes	Creative, innovative; requirements unclear
Users	Not continuously involved	Close and frequent collaboration
Documentation	Formal documentation required	Knowledge and technique based
Project size	Bigger projects	Smaller projects
Organizational support	Use existing processes	Prepared to follow an agile approach
Team members	Fluctuation expected; distributed team	Dedicated team; smaller team
System criticality	System failure consequences serious	Less critical systems
Project plan	Linear	Complex; iterative

In Table 3-3, Landry and McDaniel (2016:28-29) listed key agile topics by PMBOK area in order to teach agile alongside traditional project management. This table can be used as the foundation in integrating PMBOK with agile software development methodologies.

Table 3-3: Agile topics by PMBOK area (Landry and McDaniel, 2016:29)

PMBOK area	Key agile topics
Integration	Agile manifesto values and planning game
Scope	User stories/backlog
Time	Timeboxing, sprints, planning poker, burndown chart, Kanban board
Cost	Financial evaluation methods (NPV, ROI)
Quality	Acceptance testing, definition of done, escaped defects
Human Resources	Emotional intelligence, Scrum team
Communication	Co-location, daily Scrum, empathic
Risk	Progressive elaboration, risk-adjusted backlog
Procurement	Agile contracting methods
Stakeholder	Scrum master, product owner, servant leadership

Table 3-3 shows that ASDMs do not cover all areas of project management, which is why the integration with a PMM may be necessary. In addition, ASDMs focus more on scope management, human resource management and quality management and lack addressing areas, such as risk management, cost management and procurement management. Integrating these methodologies may increase project success and will be practically tested in Chapters 5 and 6.

In addition, and as mentioned previously, the PMBOK structure will be used as the basis to integrate the selected ASDMs with a PMM because of its project management principles and practices that apply. The result of the comparison is presented in Table 3-4. In the following section, how the selected ASDMs can be integrated with PMBOK in terms of the PMBOK's knowledge areas is described.

Project Integration Management

In Section 2.2.2.2.1, what processes are involved in PMBOK's project Integration Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1.1, the detail processes involved in Scrum is stated and compared to project Integration Management as follows: A product backlog is formulated that lists and prioritises all requirements of the project in the form of stories and indicates what features will be built into the solution. The sprint planning process allocates time frames to the required features and combines the tasks as sprints. Daily stand-up meetings are held to monitor progress and performance. Changes identified during development are postponed to the next sprint. The product and sprint backlogs dictate strong change management, as they indicate clearly what will be implemented by means of user stories.

Extreme programming (XP): In Section 3.2.1.1, XP process and practices during the XP life cycle are explained. From a project Integration Management point of view and in terms of PMBOK, it can be compared as follows: The exploration phase collects user requirements in the form of stories which are prioritised and broken down into iterations. The time it will take and the cost to develop the necessary functionality that would satisfy the user stories are estimated by the business and development team. Coding is the main activity with testing, communication and design ensuring that all iterations are completed successfully.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Integration Management knowledge area are described. A feasibility study and business study form part of the planning stage which determines if the project is worthwhile doing, is technically possible, which business processes can be automated, outlines the development plan and prioritises functions. Various techniques are applied to deliver the project. Prototyping enables frequent delivery and incremental development. There is a strong focus on customer delivery where time and resource commitments remain unchanged. DSDM requires testing early in the development process.

Feature driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Integration Management knowledge area as follows. An overall model is developed that defines the project context and scope, based on the customer's requirements and features of the system. Requirements are gathered by means of story cards, based on the required features and functionality in the form of use case diagrams. Features are prioritised accordingly.

Project Scope Management

In Section 2.2.2.2.2, what processes are involved in PMBOK's project Scope Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1.1, the detail processes involved in Scrum are stated and compared to project Scope Management as follows: A product backlog forms the scope of the project, as it

contains all requirements that will be developed to deliver the project. Requirements are gathered and prioritised in the form of stories that make up the product backlog. The sprint review process ensures that all development gets reviewed continuously and conforms to the requirements.

Extreme programming (XP): In Section 3.2.2.1, XP process and practices during the XP life cycle are explained. From a project Scope Management point of view and in terms of PMBOK, it can be compared as follows: The exploration phase collects user requirements in the form of stories which are prioritised and defines the scope of the project. There is a continuous communication between the programmer and the customer to ensure the requirements are fully understood. Software is delivered as soon as possible by breaking the release into smaller increments.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Scope Management knowledge area are described. The business study determines the scope of the project as it contains the business processes that will be automated, high-level functionalities, non-functional requirements, and an outlined development plan. Functionality keeps changing as customer requirements change due the focus that is on the customer. The customer is continuously involved throughout project. The aim is to deliver products frequently.

Feature driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Scope Management knowledge area as follows. An overall model is developed that defines the project context and scope, based on the customer's requirements and features of the system. The requirements determine the system features and functionality which are prioritised accordingly.

Project Time Management

In Section 2.2.2.2.3, what processes are involved in PMBOK's project Time Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1.1, the detail processes involved in Scrum are stated and compared to project Time Management as follows: During the sprint planning process, the product backlog dictates the number of hours required for each feature to be developed. The tasks as per the product backlog make up the sprints. Changes identified during development are postponed to the next sprint. Sprints can take between one to four weeks and are reviewed regularly. Tasks are tracked via story board and progress is discussed during daily stand-up meetings.

Extreme programming (XP): In Section 3.2.2.1, XP process and practices during the XP life cycle are explained. Planning is done separately for each release where requirements for the next iteration are identified and prioritised.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Time Management knowledge area are described. Time and resources remain constant and functionality keeps changing throughout the project. The project schedule and quality take preference over functionality where the aim is quick and in time delivery. DSDM gives priority to time, quality, and cost because customer involvement is critical to setting priorities. It focuses on frequent delivery and follows a pareto like principle that delivers 80% of the functionality in 20% of the time.

Feature driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Time Management knowledge area as follows. Tasks and responsibilities are assigned by taking feature dependencies into consideration. Each feature requires up to two weeks of development and is broken down into smaller steps should more time be required. The plan by feature phase generates a project plan regarding detail iterations, resources and class owners.

Project Cost Management

In Section 2.2.2.2.4, what processes are involved in PMBOK's project Cost Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1.1, the detail processes involved in Scrum are stated and compared to project Cost Management as follows. During the sprint planning process, the Scrum team discusses and evaluates the product backlog items and features required by the product owner and estimates the development time, effort and cost required.

Extreme programming (XP): In Section 3.2.2, XP is explained in more detail. The programming team prepares the plan, time, and costs of carrying out the iterations, and individual developers sign up for iterations. XP aims to minimise the cost through small iterative releases throughout the entire software development process instead of following the process once for the entire project. Development is twice the cost due to pair programming.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Cost Management knowledge area are described. DSDM gives priority to time, quality, and cost and customer involvement is critical to setting priorities. One of the major benefits of DSDM is quick and in time delivery of the project with a reduction in the cost of the project. Although DSDM deals effectively with project cost, it is costly to implement, as it requires users and developers alike to be trained to employ it effectively.

Feature-driven development (FDD): The time intervals between analysis and testing are short which helps with discovering errors early thereby reducing the cost of fixing the errors. FDD has no further focus on costs.

Project Quality Management

In Section 2.2.2.2.5, what processes are involved in PMBOK's project Quality Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1.1, the detail processes involved in Scrum are stated and compared to project Quality Management as follows. Scrum's practices, such as daily stand-up meetings, review and retrospective meetings, and planning, improve the development process and result in better quality of team working, and deliver better quality products. Customer involvement throughout all iterations ensures that the projects deliver according to their expectation.

Extreme programming (XP): In Section 3.2.2.1, XP process and practices during the XP life cycle are explained. Pair programming delivers very robust, high quality software, and produces better designs and better code. Testing is continuous and programmers first write the tests with which the code has to comply. The XP life cycle promotes continual communication with the customer and amongst the team with frequent feedback through unit and acceptance testing. Problems are taken on proactively with integrated testing and changes in the development phase. Designs are simple and complexity is removed as far as possible. Everyone must code to the same standards to make coding more effective.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Quality Management knowledge area are described. DSDM sets time, quality, and cost at the beginning of the project; customer involvement is critical to setting priorities. DSDM also prioritises schedule and quality over functionality. Feasibility studies are done before the project starts as this minimises the risk of project failure. Frequent delivery of releases is more important than maximising quality and testing occurs throughout the life cycle of the project and all stakeholders must cooperate and communicate.

Feature-driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Quality Management knowledge area as follows. FDD focuses on the design and building phases, emphasises quality aspects throughout the process and includes frequent and tangible deliveries, along with accurate monitoring of the progress of the project. Quality is a focus point throughout the development process and it is not just to test the code, but also includes things, such as coding standards, measuring audits and metrics in the code.

Project Human Resource Management

In Section 2.2.2.2.6, what processes are involved in PMBOK's project Human Resource Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1, Scrum is described in more detail and compared to project Human Resource Management as follows. Sprints involve a cross-functional team that selects work based on priorities as per the product backlog. Daily Scrum meetings of a couple of minutes are held where progress is discussed, and team efforts measured. Scrum is mainly team-based with defined roles. The development team of about seven people produces a shippable deliverable with each iteration,

Extreme programming (XP): In Section 3.2.2, XP process and practices during the XP life cycle are explained. XP enforces pair programming with two developers using the same computer. One is writing code and the other is supervising and they change roles at regular intervals. The customer is part of the development team and is involved in all processes of the project. XP addresses the needs of small teams who need to satisfy unclear and continuously changing requirements. Overtime is avoided as far as possible as the maximum working hours per week for developers must not exceed 40 hours. Teamwork is emphasised by XP, as all team members are equal partners working in a collaborative manner. It promotes a self-organising team to solve problems efficiently.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Human Resource Management knowledge area are described. DSDM teams are empowered to make decisions. Workshops are facilitated to the extent that experts from the customer's side sit together with the development team to identify and prioritise the system requirements.

Feature-driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Human Resource Management knowledge area as follows. The client and development team develop an overall model where detailed walkthrough meetings are held to work through all the requirements and features with the objective to define the context and scope of the project that will be delivered. FDD follows eight practices, namely domain object modelling, development by feature, individual class ownership, feature teams, inspection, configuration management, regular builds and progress reporting. FDD requires team members who are highly skilled experts in designing and modelling.

Project Communications Management

In Section 2.2.2.2.7, what processes are involved in PMBOK's project Communications Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1, Scrum is described in more detail and compared to project Communications Management as follows. There is continuous communication with strong feedback mechanisms between all stakeholders throughout all processes during the development life cycle. Daily stand up meetings are conducted to monitor performance of the team members who are working on the features being developed. Teams communicate collaboratively via the product backlog and sprint backlogs. Story boards facilitate communication and teamwork during meetings.

Extreme programming (XP): In Section 3.2.2, XP process and practices during the XP life cycle are explained. XP is a lightweight and informal approach for software development. Communication is based on practices, such as unit testing, pair programming, and task estimation. It promotes a self-organising team to solve problems efficiently. Teamwork is emphasised, as all team members are equal partners working in a collaborative manner. XP enforces pair programming with two developers using the same computer; one is writing code and the other is supervising and they change roles at regular intervals. Short stand-up meetings are held daily to report problems. XP is most efficiently applied when the team environment is highly collaborative with co-located teams.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Communications Management knowledge area are described. DSDM is facilitated by workshops during which experts from the customer's side sit together with the development team to identify and prioritise the system requirements. It creates an environment that is ideal for the formation of ideas, including an awareness of decisions made by all parties involved. DSDM allows for accurate and quick decision making.

Feature-driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Communications Management knowledge area as follows. Walkthrough meetings are held by the project team to work through the requirements and features with the objective to define the context and scope of the project that will be delivered. Domain areas and object models are reviewed regularly. Development is done by feature which is delivered by feature teams.

Project Risk Management

In Section 2.2.2.2.8, what processes are involved in PMBOK's project Risk Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

Scrum: In Section 3.2.1, Scrum is described in more detail and compared to project Risk Management as follows. Projects are delivered in an iterative and flexible manner where at the end of every sprint of work there is a tangible deliverable to the organisation. Every sprint is

followed by testing and risk analysis, thus reducing the overall project risk. Sprints can take between one to four weeks and are reviewed regularly which reduces risk within that time frame. The risk of excessive costs is therefore limited to one calendar month of cost. The project risks are detected sooner in the project. Scrum depends on artefact transparency, as decisions required to control risk and to optimise project value are based on the state of the artefacts. Daily stand-up meetings are held to monitor progress and performance.

Extreme programming (XP): In Section 3.2.2, XP process and practices during the XP life cycle are explained. Risks are identified early in the project. It is a frequent release development methodology in which developers work in pairs for continuous code review. The customer is part of the development team and is involved in all processes of the project. Automated regression unit testing reduces the risk of defects through source code changes. XP welcomes changing requirements through the whole development process by being flexible with its short iterations that allow rapid feedback and tools, such as refactoring and unit testing that allows for simplistic coding. Together, this helps guide the project in the right direction, which in turn decreases the risk of failure. Collective ownership allows for all programmers to have a full understanding of all the code that mitigates the risk of only one programmer knowing and understanding the code and system.

Dynamic systems development method (DSDM): In Section 3.2.3.1, the DSDM phases where some of the pre-project sub-phases can be compared with PMBOK's project Risk Management knowledge area are described. All changes and modifications can be reversed during the development process. DSDM offers good guidelines for project aspects, such as risk management. Development is iterative and incremental which reduces risk.

Feature-driven development (FDD): The FDD processes are explained in section 3.2.4.1 which can be compared to PMBOK's project Risk Management knowledge area as follows. FDD requires highly skilled experts in designing and modelling which ensures that good quality code is produced. In the plan by feature phase, priorities are assigned to features while considering feature dependencies, complexities, workloads of teams and risks.

Project Procurement Management

In Section 2.2.2.2.9, what processes are involved in PMBOK's project Procurement Management are described in detail. To integrate this with ASDMs, each ASDM needs to be evaluated in terms of this process.

In all ASDMs, no distinction is made in procurement management.

Table 3-4: Integration of PMBOK and agile software development methodologies

PMBOK	Software development methodologies			
	SCRUM	**XP**	**DSDM**	**FDD**
Project Integration Management				
• Develop project charter • Develop project management plan • Direct and manage project work • Monitor and control project work • Integrated change control • Close project	• Software requirements are formulated and prioritised by the product owner as stories • Sprint planning events • Strong change management procedure with product and sprint backlog • Retrospective and next sprint planning	• User stories or requirements are created by customers and the development team • Stories are converted into iterations • Programming team and business prepares the plan, time, and costs of carrying out the iterations • Coding takes priority over all tasks • Continuous design-coding-testing-listening cycles	• Feasibility and business study as evaluation and planning mechanisms • Prioritisation is essential • Baseline of requirements and functionality is at a high level • Leans heavily on techniques to develop an application • Uses prototypes • Good for projects with tight time constraints • Testing integrated throughout the life cycle	• Developing an overall model • Building a feature list • Planning and designing by feature • Prioritise based on features
Project Scope Management				
• Scope planning • Collect requirements • Scope definition • Create work Breakdown structure • Scope validation • Scope control	• Product backlog creation • Sprint planning and sprint backlog creation • User stories are defined and prioritised • Review continuously to improve development process	• Customer defines user stories • Release planning and prioritisation by customer • Continuous feedback from customer • Small releases	• Requirements list as per the business study defines the scope • Focusing on the customer needs • Active user involvement • Frequent releases	• Developing an overall model • Develop feature list
Project Time Management				
• Plan schedule • Define activities • Sequence activities • Estimate activity resources • Estimate activity durations • Develop schedule • Control schedule	• Plan sprint durations • Sprint lasts± 2 weeks • Task board for tracking progress • Daily stand-up meetings to discuss progress	• Iterations time planning • Release planning	• Time and resources are not adjusted • Deliver the projects as early as possible without affecting the quality • Good control over quality of product, cost and time • Frequent delivery of products	• Plan by feature • Plan order of features, based on dependencies • Determine development sequence
Project Cost Management				
• Cost estimating, • Cost budgeting, and • Cost control project	• Scrum team defines cost estimates related to stories during sprint planning	• Programming team prepares the costs of carrying out iterations • Double development cost • Reduce cost via small iterations	• Deals effectively with project cost estimates • Gives priority to time, quality, and cost • Costly to implement	• No primary focus

Table 3-4: Integration of PMBOK and agile software development methodologies (continued)

Project Quality Management				
• Quality planning • Perform quality assurance • Perform quality control	• Quality goals sustained during sprints • Retrospective and next sprint planning to improve development process • Regular Scrum meetings • Customer inputs for further improvements during iterations	• Pair programming for higher quality code • Extreme testing in development phase • Continuous customer involvement and testing • Simple design • Coding standards	• Good control over quality of product, risk, cost and time • Delivering the projects without affecting the quality of the project • Focus on frequent releases rather than quality	• Continuous testing of the code • Coding standards • Measuring audits and metrics in the code
Project Human Resource Management				
• Human Resource Planning • Acquire project team • Develop project team • Manage project team	• Scrum teams self-organised and cross functional • Daily Scrum meetings • Small teams	• Pair programming • Small teams • Customer part of development team • Team-oriented methodology • Avoid working overtime	• Teams empowered to make decisions • Facilitated Workshops	• Client and development team develop overall model • Highly skilful teams • Feature teams
• **Project Communications Management**				
• Communications planning • Information distribution • Performance reporting • Manage stakeholders	• Permanent communication and close cooperation between the stakeholders at each step • Daily Scrum Meetings • Group is self-organising and collaboratively managed	• Continual communication with the customer and amongst the team • Collaborative workspaces • Co-location of development and business space • Paired development • Frequently changing pair partners • Short stand-up meetings • Unit tests and verbal communication	• Facilitated workshops • Environment ideal for the formation of ideas • Accurate and quick decision making	• Continuous review meetings • Feature teams deliver features
Project Risk Management				
• Risk Management Planning • Risk Identification • Qualitative Risk Analysis • Quantitative Risk Analysis • Risk Response Planning • Risk Monitoring and Control	• Iterative and incremental approach to control risk • Sprints limit risk to one calendar month of cost • Project risks detected sooner • Decisions to control risk are made based on the perceived state of the artefacts • Meetings to review risks	• Identify risks early in the project • Frequent releases • Unit testing • Customer part of development team • Collective ownership reduces risks of programmer dependency	• Changes are reversible during development • Feasibility study identifies risks involved • Guidelines for risk management • Reduce risk through incrementally delivering the solution	• Guidance of skilled, experienced developers • Feature planning takes risks into account
Project Procurement Management				
• Plan Purchases and Acquisitions • Plan Contracting • Request Seller Responses • Select Sellers • Contract Administration • Contract Closure	• No information	• No information	• No information	• No information

3.4 Summary

In this chapter, the term “software development methodology” was defined, after which four of the most popular ASDMs were investigated and explained (Stoica *et al.*, 2013:72-73). Each

ASDM was evaluated in relation to their processes, advantages, disadvantages, and area of application. The similarities and differences of these ASDMs were further investigated in terms of their characteristics, strengths and weaknesses. During the explanation of the four ASDMs, it became clear that ASDMs have various weaknesses related to project management (Despa, 2014:51-54).

Agile topics were further compared by each PMBOK knowledge area which further identified project management areas that are not covered in ASDMs (McDaniel, 2016:28-29). Although ASDMs can deliver, their disadvantages emphasise that ASDMs alone are not sufficient to solve project management problems, which could cause a project to fail. For this reason, this study sought to determine whether ASDMs and PMMs could be integrated to deliver software development projects more successfully. This necessitated a need to integrate each of the ASDMs with PMBOK's knowledge areas which forms the basis of this study.

The research methodology followed in the integration of ASDMs with PMMs will form the main focus of Chapter 4.

CHAPTER 4: RESEARCH METHOD

In this chapter, various research paradigms and methods will be discussed with the aim to identify and select the most suitable research method that will be applied to this study. Three different paradigms will be discussed, namely the positivist, interpretivist and critical research paradigms.

In this chapter, the different research methods that are related to the selected research paradigm will be described. The preferred research method, based on the research paradigm, will then be discussed, including the application of the selected method.

4.1 Research paradigms

According to Hirschheim and Klein (1989:1201), a research paradigm is "the most fundamental set of assumptions adopted by a professional community that allows its members to share similar perceptions and engage in commonly shared practices".

Kivunja and Kuyini (2017:26) state that the word paradigm, meaning a philosophical way of thinking, was first used in 1962 by an American philosopher, Thomas Kuhn. From an educational research perspective, the word paradigm describes a researcher's "world view", where "world view" is regarded as the school of thought used to interpret the research data (Kivunja and Kuyini, 2017:26). According to Lather (1986), a research paradigm is a reflection of the researcher's beliefs about the world in which he/she wants to live. It can be viewed as the conceptual lens through which the researcher evaluates the methodological aspects of the project that are used to eventually establish the method of research and data analysis that will be applied. Paradigms are therefore important, as they give guidance on how a study should be done and also how the results of the study should be interpreted (Kivunja and Kuyini, 2017:26).

A research paradigm has four essential elements, namely epistemology, ontology, methodology and axiology which comprise the basic assumptions, beliefs, norms and values that each paradigm holds (Kivunja and Kuyini, 2017:26-27). These four elements will be discussed next.

Epistemology of a Paradigm: This element describes how a person comes to know reality or the truth. It also addresses what counts as knowledge and ways of knowing, how we should study the world, what evidence or insights are meaningful, and how knowledge arises (McGregor and Murnane, 2010; Kivunja and Kuyini, 2017:27). Epistemology focuses on the

nature and forms of knowledge, including how it can be obtained and communicated to other people. An example of epistemology in the context of this study is how knowledge can be gained on the application of PMMs and ASDMs.

Ontology of a Paradigm: Ontology is related to assumptions that are made in order to believe that something is real or has a clear meaning. It is related to what the objective of the study should be, what human nature is, what it means to be human, how choices are made, what we believe about the nature of reality and how reality can be meaningfully depicted. It is about what counts as nature, reality, feeling, existence or being (Kivunja and Kuyini, 2017:27). Typical examples would be to understand what the terms PMMs and ASMDs mean.

Methodology of a Paradigm: A methodology refers to the research design, methods, approaches and procedures that are applied in a well-planned investigation to determine something (Kivunja and Kuyini 2017:27). It includes the process of gathering data, all the participants involved in the research, the instruments used and analysing of data that all forms part of the broad field of methodology. In essence, it contains the logic and systematic processes used during the research project to eventually obtain knowledge about the research problem (Kivunja and Kuyini, 2017:28). An example of methodology in the context of this study is action research and its cyclical process consisting of five phases.

Axiology: This element involves the consideration of ethical issues during the process of planning a research proposal. It further includes the definition, evaluation and understanding of the concepts of right and wrong behaviour with regard to the research (Kivunja and Kuyini, 2017:28). It takes into account the value that will be gained by the participants, the data and the audience to whom the research results will be reported. An example would be to guard against the manipulation of involved parties during a study and to act in the best interest of their well-being.

Paradigms can be categorised into three main taxonomies, namely positivist, interpretivist, or critical paradigms (Kivunja and Kuyini, 2017:30). Each of these will be described in the following sections.

4.1.1 Positivist research paradigm

The positivistic research paradigm dictates that knowledge can only be true if it was generated by means of a scientific method, which means that the data is extracted by means of experimentation and observation (McGregor and Murnane, 2010:423). It assumes that reality is

objectively given and can be described by means of measurable properties which are independent of the researcher (Myers, 1997).

Positivistic research claims that empirical methods should be applied for process verification purposes, as these methods have no influence on the investigation and are perceived as objective (Kim, 2003:11).

Research within this paradigm depends on deductive logic, hypotheses testing, generating operational definitions and mathematical equations, calculations, and expressions to draw conclusions (Kivunja and Kuyini, 2017:30). It attempts to deliver explanations and make predictions, based on outcomes that are measurable. To help researchers to better understand the meaning and expectations of research conducted within this paradigm, there are four assumptions that can be evaluated:

- *Determinism*: Other factors determine the events that are observed. If the casual relationships among factors are understood, it should be possible to predict and control the impacts of explanatory factors (Kivunja and Kuyini, 2017:30).
- *Empiricism:* Verifiable empirical data that supports the chosen theoretical framework and makes testing the hypotheses possible needs to be collected in order to be able to investigate the research problem (Kivunja and Kuyini, 2017:30).
- *Parsimony:* This involves the researcher's attempts to explain the study in an economical manner (Kivunja and Kuyini, 2017:30).
- *Generalisability:* This means the identification of occurrences in the particular phenomenon that was studied and being able to generalise about what can be expected elsewhere in the world (Kivunja and Kuyini, 2017:30).

In addition to the four assumptions, the positivistic paradigm has the following elements that must be understood:

Objectivist epistemology: This means that knowledge can be acquired through research which relates to what is investigated. Therefore, we become more objective in understanding the aspects of the world by gaining knowledge through research (Kivunja and Kuyini, 2017:31).

Naïve realist ontology: This element involves accepting the following beliefs (Searle, 2015; Putnam, 2012):

- A world of material objects exists.
- The truth about some objects is known through sense-experience.
- Objects of perception are assumed to be perception-independent.

- Objects may have properties that might be perceived as they exist, even when they are not being perceived.
- The world is directly perceived by means of our senses.

Experimental methodology: This element involves the manipulation of one variable to establish if changes in that variable cause changes in another variable during the research (Kivunja and Kuyini, 2017:31).

Beneficence axiology: This element involves the requirement that the research should have the purpose of maximising good outcomes for the research project and humanity, including the research participants. Avoiding or minimising risk, harm, or wrongdoing during the research should also be a focus point (Kivunja and Kuyini, 2017:31).

The following characteristics are normally located within the positivist paradigm (Oates 2006:286:

The existing world is independent of humans: The existing world is a physical and social world that is more than what exists in a person's mindset and needs to be captured, studied and measured.

Measurement and modelling: This world will be discovered by the researcher by making observations, measurements and producing models, for example various hypotheses and theories of how the world works. There will be one explanation for any aspect of the world that can be interpreted as "the truth".

Objectivity: The researcher will be an impartial observer who will behave neutrally and objectively with his personal views and beliefs not affecting any discoveries made.

Hypothesis testing: The researcher is lead to the confirmation and refutation of theories and hypotheses by basing the research on empirical testing.

Quantitative data analysis: Mathematical modelling and evidence is preferred, as mathematics provides a logical and objective means for analysing observations and results.

Universal laws: Universal laws, patterns and facts are searched for to prove that they can be true, irrespective of the researcher and the occasion.

A positivistic paradigm allows the subject being studied and the researcher not to impact each other and measures the subject against universal laws. This paradigm follows a scientific methodology and ensures that scientific data is collected which is precise, can be interpreted as quantitative data and can be analysed by means of statistical and mathematical techniques. Positivist research aims to explain or measure research statements against reality.

The interpretive research paradigm will be discussed in the next section.

4.1.2 Interpretive research paradigm

Interpretive research is concerned with understanding the social context of a research problem: the social processes by which it is developed and construed by people and through which it influences, and is influenced by, its social setting (Oates, 2006:292).

According to Myers (1997), interpretive research assumes that reality can only be accessed through social constructions, such as language, consciousness and shared meanings. Interpretive research tries to make meaning of phenomena by the explanation that people assign to them. Imperative methods of research are “aimed at producing an understanding of the context of the information system, and the process whereby the information system influences and is influenced by the context" (Myers 1997). Interpretive research focuses on the full complexity of human sense making and does not predefine dependent and independent variables.

Yanow (2006:5-26) suggests that the understanding of an interpretivist paradigm lies at the philosophical level of a researcher’s world view. Ontologically, selecting a paradigmatic camp is replete with underlying assumptions about reality which poses the questions of whether the researcher believes there is one objective real world, or whether the world is viewed as social construct (Hathaway, 1995:543-547). While positivists uphold that there is one reality, interpretivists believe reality is subjective, multiple and socially constructed (Krauss, 2005:758-761). Interpretivists further believe that data cannot be collected or removed from context and as such, promotes the generation, discovery or construction of knowledge (Yanow, 2014:97-119). Methodologically, many positivists adopt quantitative methods to make generalisability claims, whereas interpretivists draw on a range of methods, tools and techniques to secure an in-depth understanding of the phenomenon under investigation (Denizin and Lincoln, 2011:9,117-118).

Interpretivism includes all factors that are similar within the social environment and has the following characteristics (Oates, 2006:292–293):

- Various subjective realities: There is more than one version of the truth. What is perceived as being real is only constructed in the researcher’s mind.
- Dynamic, socially constructed meaning: Whatever a person or group perceives as reality can only be accessed and conveyed to others through communication, understanding and meaning.

- Researcher reflexivity: The researcher cannot be seen as neutral, as his/her own assumptions, beliefs, meaning, actions and values will affect the research process, thereby unavoidably influencing and directing the research process.
- People are studied in their natural social setting: People are studied without the researcher manipulating or influencing the environment under investigation.
- Qualitative data analysis: The researcher prefers the gathering and analysing of qualitative data.
- Multiple interpretations: The researcher expects that there will be more than one answer or explanation to the research question.

Although both qualitative and quantitative data can be used in interpretivistic research, the correct data collection and data analysis techniques should be applied to ensure that the interpretations are supported by the data. This research paradigm aims to understand the interpretations of the research, including the reality in which the interpretations where made.

Critical social theory will be explained in the section to follow.

4.1.3 Critical social theory

Oates (2006:296) defines critical social research as research that "is concerned with identifying power relationships, conflicts and contradictions, and empowering people to eliminate them as sources of alienation and domination". In addition, Kincheloe and McLaren (2003:435) describe critical social theory as research concerned with power, race, class, gender, beliefs and other social institutions and cultural dynamics that are working together to form a social system. The aim of critical social research is social critique, where circumstances that limit and alienate are revealed (Myers, 2011:19).

The following characteristics are normally associated with critical social research (Oates, 2006:297-298):

Emancipation: Critical research aims to free people from the power relations that mostly determine the shape of organisations and society. In addition, understanding and explaining critical researchers also look to empowering people. Compared to researchers in the other paradigms, critical researchers are more activist.

Critique of tradition: Critical researchers question and challenge rather than accept the *status quo*. Current taken-for-granted assumptions and patterns of power are highlighted and confronted.

Non-performative intent: Research projects with the objective of improving managerial efficiency and control where maximum outputs are achieved through minimum inputs are rejected by critical researchers.

Critique of technological determinism: The concept of technological development following its own rules and people and societies adapting to the technology is challenged by critical researchers. They highlight such ideas and enable those with vested interests in the technology to increase their power over others. By contrast, critical researchers argue that the technology they developed can be shaped by people and society.

Reflexivity: The possibility of objective, value-free knowledge is questioned by critical researchers. They highlight that those with power and vested interests mostly shape research projects.

The following principles are proposed for critical research (Myers and Klein, 2011:25):

1. *The principle of using core concepts from critical social theorists*: Critical theorists' core concepts and ideas should be taken into consideration by critical researchers in their data collection and analysis activities.
2. *The principle of taking a value position*: Values, such as open democracy, equal opportunity, or discursive ethics are recommended by critical theorists. These values dictate the basis for principles four through six.
3. *The principle of revealing and challenging prevailing beliefs and social practices*: Important beliefs and social practices should be identified and challenged with conflicting arguments and evidence.
4. *The principle of individual emancipation*: Critical social theory aims to facilitate the realisation of human needs and potential, critical self-reflection, and associated self-transformation.
5. *The principle of improvements in society*: This principle indicates that it is possible to improve society. The objective is not only to reveal the existing forms of domination, but also to recommend how unwarranted uses of power might be conquered. It is assumed by most critical theorists that social improvements can realise, but to various differing degrees.
6. *The principle of improvements in social theories*: Critical theorists believe that our theories are frail and that it is possible to improve social theories. Critical researchers also believe that competing truth claims can arise from alternative theoretical categories that can lead critical researchers in their interventions and analysis.

Judging the quality of critical research is still evolving, but criteria that could be used are based on the ideas of fairness and authenticity (Oates, 2006:298):

- **Fairness:** did all research stakeholders have equal access to the inquiry process, question choices, responses and their interpretation?
- **Ontological authenticity:** how much were the informants enabled by the research to expand their personal views of their worlds?
- **Educational authenticity:** how much were informants enabled by the research to improve their appreciation and understanding of the construction of others?
- **Catalytic authenticity:** how far were the informants stimulated by the research into action or decision making?
- **Tactical authenticity:** to what extent were the informants empowered by the research to take action?

To conclude, critical research goes beyond understanding a situation and questions structures which may dominate the situation. Researchers in this paradigm assume that social realty is historically constructed which is produced and reproduced by people (Myers, 1997). Critical researchers realise that although people can consciously change their social and economic circumstances, their ability to do so is limited by various forms of social, cultural and political domination. Critical research focuses on the conflicts and disapprovals in society, while trying to eliminate the causes that lead to alienation and domination (Myers, 1997).

4.1.4 Research paradigm used for this study

In this section, the research paradigm applied in this study that attempts to assist organisations to apply a hybrid ASDM according to their own environment and culture to execute software development projects effectively is explained. A critical social theory paradigm is adopted and best suited for this study, as the study is trying to improve the existing situation of projects being late, over budget and not meeting user requirements. For this reason, the critical social research paradigm was chosen for this study. Critical research includes social components, such as freedom, power, social control, and values in terms of the usage, development, and impact of information technology (Myers and Klein, 2011:17).

Critical scientists argue that researchers should share responsibility for social changes and are of the opinion that positivistic methods cannot capture the critical role in knowledge of values that are needed to improve human conditions (Bredo and Feinberg, 1982:115-128). They also claim that the positivistic tradition often neglects the realities of power, ideological beliefs, and social inequities manifest in society (Rettig, Tam, and Yellowthunder, 1995:109-143). The critical science approach also advocates a process of research that yields social change rather than pure knowledge generation.

The principles of critical research in the context of this study can be applied as follows:

- Core concepts from critical social theorists: This study critically investigates the use of ASDMs and PMMs as a basis for integrating these methodologies to improve project success. Theory around these methodologies are studied and applied during the research process.
- Revealing and challenging prevailing beliefs and social practices: Existing ASDMs and PMMs are thoroughly investigated and challenged throughout this study in an attempt to integrate these methodologies to improve project success.
- Individual emancipation: The research involves an attempt to improve project success which will have a social impact on project team morale and motivation in the organisational environment should the result be positive.
- Improvements in society: Building on the previous principle, this principle suggests that improvements may be possible, not just at an individual level, but in society as a whole. Social theory suggests improvements to organisations, institutions, and society (Myers and Klein, 2011:27). This research aims to improve project success which closely relates to organisational improvements.
- Improvements in social theories: This principle focuses on the growth and improvement of theoretical knowledge which is one of the focus points of this study. By integrating ASDMs and PMMs that may improve project success may also enhance theoretical knowledge in the field of ASDMs and PMMs.

4.2 Research methods associated with critical research

Critical research can be performed by following various methods, such as sampling, for example, with the key factor being the outcome of the research to improve or change the situation or even to emancipate someone. Action research is mostly associated with critical research which is the research method that is going to be used for this study and is explained in the next section.

4.3 Research method used for this study

In this study, the action research method is followed. In this section, the term action research is defined, its characteristics are explained and the action research cycle of diagnosing, action planning, action taking, evaluating and learnings is discussed. The advantages and disadvantages of action research are also explained.

4.3.1 Action research

Action research can be traced back to Lewin in the United States of America during the 1940s and 1950s and also to independent work done by the Tavistock Institute in the United Kingdom during the 1950s and 1960s (Oates, 2006:154). Kurt Zadek Lewin, regarded as the "father" of action research, was concerned about social problems, and concentrated on processes to address organisational conflict, crises and change by means of participative groups (Ahmed, 2009:21). Lewin focussed on making social scientist activities useful by applying psychological techniques to practical social problems instead of writing books or papers that would only be read by academics (Oates, 2006:154). He introduced a research process comprising spiral steps which consisted of a circle of planning, action, and fact-finding about the result of the action (Ahmed, 2009:22). Lewin is generally regarded as one of the pioneers of action research and also the first person who referred to the term "action research", meaning a specific research approach where new social knowledge is generated about a social term, while simultaneously trying to change it (Peters and Robinson, 1984:114-115). Another reason for the emergence of action research and its continuous use in the Information System Industry is the conclusion that social systems are better understood where the researcher forms part of the study under research (Baskerville and Wood-Harper, 1996).

Hult and Lennung (1980:247) define action research as research that "simultaneously assists in practical problem solving and expands scientific knowledge, as well as enhances the competencies of the respective actors, being performed collaboratively in an immediate situation using data feedback in a cyclical process aiming at an increased understanding of a given social situation, primarily applicable for the understanding of change processes in social systems and undertaken within a mutually acceptable ethical framework." Greenwoord and Levin (2007:3) refer to action research as research that is done by a team that wants to improve the participants' situation by defining, examining and resolving a social problem. This is typically one of the objectives that the researcher is trying to achieve with this study, by aiming to improve software development project success in the process of integrating ASDMs and PMMs. Corey (1953: 6) argues that action research is where practitioners study problems scientifically in order to evaluate, improve and direct decision making and practice. Action research has also been described as a method of generating knowledge about a social system as a whole, and simultaneously trying to change it (Elden and Chisholm 1993: 121). Carr and Kemmis (1986:162) provided the following definition: "Action research is simply a form of self-reflective enquiry undertaken by participants in social situations in order to improve the rationality and justice of their own practices, their understanding of these practices, and the situations in which the practices are carried out". Kemmis and McTaggart's (1988: 5) definition of action research states that "it is a form of collective, self-reflective inquiry that participants in

social situations undertake to improve: (1) the rationality and justice of their own social or educational practices; (2) the participants' understanding of these practices and the situations in which they carry out these practices".

Action research can be applied in either a positivist or interpretivist or critical research paradigm. The chosen research paradigm must be clearly stated, as each is different in terms of judging the quality of action research (Oates, 2006:156). Action research in the interpretivistic paradigm works with people in a specific social setting, and attempts to determine how they perceive their own world. It results in actions and changes that are regarded as improving the existing situation under research. Action research also fits into the critical social paradigm where changes enable people to stand up against power structures (Oates, 2006:301). Oates (2006:301) writes that action research can also be in the positivistic paradigm by testing existing theories and validating if the theories' predictions occur during the research. This study applies action research in the critical social paradigm where social inputs and feedback from various participants are incorporated in the evaluation of the research objectives.

4.3.2 Action research characteristics

Action research has the following characteristics (Oates, 2006:155):

- *Concentration on practical issues*: Instead of applying abstract hypotheses, laboratory experiments, software programs or mathematical justifications, it deals with concerns and complicated problems raised by people in their everyday lives. Researchers are required to work amongst the people. In this study, the focus is on the complicated problem faced by software developers to produce quality software while facing time and budget constraints. The researcher forms part of the development team.
- *Plan-act-reflect as an iterative cycle*: Action research is about doing research in action. Researchers are doing research in a real world and then respond to the result and learnings whereafter another cycle of plan-act-reflect starts.
- *Emphasis on change*: Researchers focus on making a difference with their research and learning how the change was impacted.
- *Practitioners' collaboration*: The research participants are the people who actively work in the situation being researched.
- *Various data generation methods:* Action research has no data restrictions, as quantitative and qualitative data can be used.
- *Research and action outcomes:* The result of action research can be both a practical achievement and a learning experience about problem-solving processes. It is preferable that both outcomes should be achieved, although it is not always the result.

4.3.3 Action research principles

Action research has the following principles (Kemmis and McTaggart, 1992: 22–5):

- Action research is an approach to improving a problem situation by changing it and learning from the consequences of the changes.
- Action research is participatory in the sense that people work to improve their own practices.
- Action research follows a self-reflective spiral, involving cycles of planning, acting, observing and reflecting. This process is then followed by re-planning, further implementation, observing and reflecting.
- Action research follows a collaborative approach by involving the people responsible for action in improving that action.
- Self-critical communities are established where people are participating in all phases of the research process.
- Action research follows a systematic learning process where people respond deliberately, but remain open and responsive to surprises and opportunities.
- Action research involves people who are curious about circumstances, action and consequences in theorising about their practices, thereby helping them to understand the relationship between circumstances, actions and consequences in their own lives.
- Action research causes people to apply their practices, ideas and assumptions about institutions by enabling them to gather evidence which could convince them that the previous practices, ideas and assumptions were not correct.
- Action research is transparent with evidence, as it involves records which describe the existing situation accurately, including the collection and analysis of judgements, reactions and impressions of the existing situation.
- Action research records two sets of learnings, namely learnings about the practices being studied and learnings about the process of studying them.
- Action research has a political process, as it requires us to make changes that will affect others.
- Action research requires people to critically analyse situations in which they work which are structured institutionally.
- Action research begins small by involving events that a single person can try and then works towards bigger changes.
- Action research starts off by following small cycles of planning, acting, observing and reflecting that normally help to identify issues, ideas and assumptions which helps those

involved to define more complex and powerful questions themselves as the research progresses.

- Action research begins with smaller groups and then expands the groups of action researchers to the extent that more of those who are affected by the research are involved.
- Action research helps to create records of the improvements.
- Action research produces a justified reason for the educational work to others by showing that the research helped to create a developed, tested and critically examined rationale for what was being done.

4.3.4 Action research advantages

The following advantages are associated with action research (Oates, 2006:168):

- Action research focusses on the real world and on what is relevant to people, resulting in actual practical improvements.
- It connects the practical, everyday world with the academic world.
- Systems development and problem-solving methods can be easily associated with action research.
- It makes the research process more acceptable and generates more appreciation for all knowledge types.
- Higher goals can be set compared to other research strategies.

4.3.5 Action research disadvantages

The following disadvantages are associated with action research (Oates, 2006:168):

- Many researchers in the computing industry and some IS researchers are not familiar with action research and do not accept it.
- It is not thorough enough, has limitations in determining cause and effect and has outcomes that may not be applicable to certain situations.
- It may be perceived as consultancy instead of research.
- It is not for people who do not want to work democratically with other people in complicated, problematic and unpredictable real-world situations.
- The needs and expectations of all parties involved are not easily met.

In an attempt to overcome the disadvantages, three cycles across the different types of software development project within a particular organisation were followed in this study, and the success of these projects, based on a combination of a predefined project success criteria and social feedback from the parties involved were evaluated.

The action research cycle will be explained in the following section.

4.4 Action research cycle

Action research attempts to produce knowledge about a social system while at the same time trying to change it. Action research therefore has two basic elements, namely generating knowledge, and changing social systems (Ahmed, 2009:23). Lewin proposed a cyclical process consisting of a "non-linear pattern of planning, acting, observing, and reflecting on the changes in the social situations" (Noffke and Stevenson 1995: 2). A simple model of the cyclical nature of action research is illustrated in Figure 4-1.

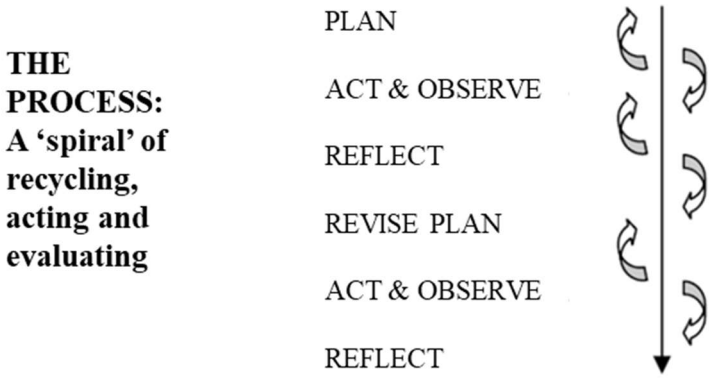

Figure 4-1: Simple Action Research Model (Hopkins, 1985)

Baskerville (1999:14) describes five phases that are conducted within the action research cycle. This follows the compilation of a single plan of action which is implemented. The research gathers data on the outcome of the intervention which is analysed. The results are then interpreted in terms of how successful the action has been. The problem then goes through re-assessment, thereby creating another cycle which continues until such time that the problem is resolved (Drummond and Themessl-Huber, 2007:433).

Figure 4-2 displays five phases of action research that are followed in an iterated manner (Baskerville, 1999:14-15).

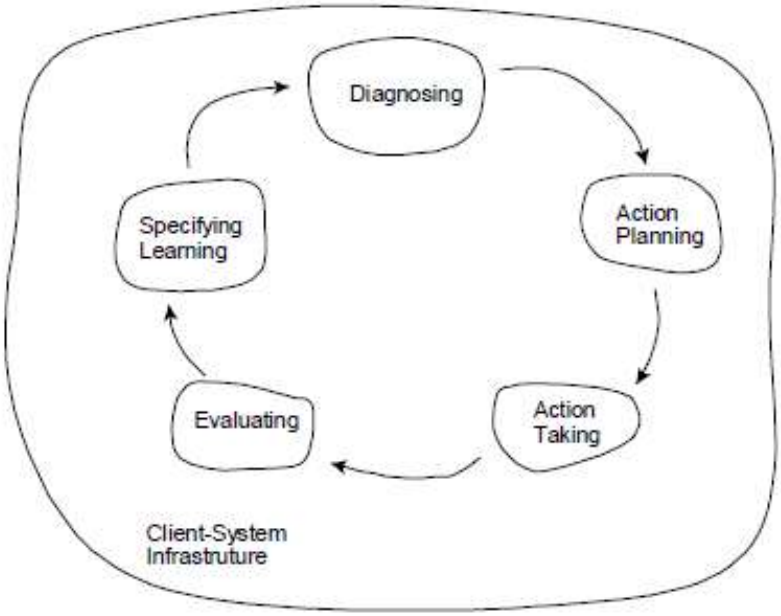

Figure 4-2: The Action Research Cycle (Baskerville, 1999:14)

These five phases and their application in this study will be discussed next.

Diagnosing

During this phase, the primary problems are identified. These are the organisation's reasons why there is a need for change. The organisational problem is self-interpreted holistically instead of by reduction and simplification. Various theoretical assumptions are made that include the nature of the organisation and its related problem domain (Baskerville, 1999:14-15).

The problem of companies that are mostly not able to execute software development projects successfully, and not having a single methodology that can be applied across organisations and industries described in detail in Chapter 1, will be researched by means of action research. The research approach will be to identify and evaluate previously executed software development projects based on predefined criteria that will be used as a measure of successful project execution and deployment.

Action Planning

This phase involves close collaboration between researchers and participants in which the actions that should address and improve the identified problems are specified. A theoretical framework guides the planned actions that point to the desired future state for which the organisation is aiming, including the actions required to deliver such a state. The plan forms the basis of the target for the change and the approach that will be followed to change. It stipulates the organisational actions that should improve or resolve the main problem (Baskerville, 1999:14-15; Baskerville and Wood-Harper, 1996:238).

In this study, the phase involved the planning of how various projects was going to be executed, based on predefined hybrid ASDMs. This was performed for three projects within an organisation, based on different project types.

Action Taking

During this phase, the planned action is implemented by intervening in the client organisation and making certain changes. This can involve the adoption of various forms of intervention strategy, being either directive, where the change is directed, or non-directive, where the change is perceived as indirect (Baskerville, 1999:14-15; Baskerville and Wood-Harper, 1996:238).

This action research will apply a hybrid methodology of ASDMs and PMMs to an organisation with the objective to improve execution and delivery of software development projects. The hybrid methodology will be applied to various projects to test its effectiveness and to establish if the applied methodology improved the execution and delivery of the software development project.

In the context of this study, three projects, based on various ASDM and PMM integrations were executed during this phase. This included the processes followed and the project team's feedback on the ASDM and PMM integration levels. Interviews were conducted with the project team members; this included feedback from the project managers, developers, analysts and customers involved in the projects.

Evaluating

During this phase, the outcomes are evaluated by the collaborative researchers and practitioners. This involves establishing if the theoretical effects of the action were achieved, and whether the problems were addressed by these effects. If the change was successful, it must be critically evaluated whether the success was due to the action undertaken. For unsuccessful changes, a framework for the following iteration in the cycle needs to be determined and must also include hypotheses adaption (Baskerville, 1999:14-15; Baskerville and Wood-Harper, 1996:238).

During this phase of the study, the project execution process was compared with and evaluated against predefined evaluation criteria to determine the effectiveness of the various ASDM and PMM integrations.

Specifying Learning

This is the last activity of the cycle and is mostly an ongoing process. Action research directs the knowledge gained to three audiences (Baskerville, 1999:14-15; Baskerville and Wood-Harper, 1996:238):

- Organisational norms are restructured to display the new knowledge the organisation gained.
- If the change was not successful, the additional knowledge gained can be used as a foundation for further action research interventions.
- The outcome of the research offers useful knowledge to the science community that can be used for future research.

This study included the execution of three software development projects following three cycles of action research. At the end of the first project, lessons learned were incorporated in the second project and further lessons learned from the second project were incorporated in the execution of the third project in an attempt to improve the effectiveness of the selected ASDM and PMM integration.

In this chapter, the research paradigms were explained, culminating in the critical social theory being selected as the paradigm used for this study. Social research methods were described together with action research as the selected research method that was followed in this study. The action research method was explained in more detail, including the action research cycle that was applied to three projects during the research process.

CHAPTER 5: CONTEXTUAL BACKGROUND FOR THE STUDY

In this chapter, an organisation where regular software development projects are conducted is investigated. Its organisational characteristics related to the Information Technology department are explained to understand the environment and processes followed in executing software development projects. This includes system development and authorisation processes, as well as software development standards. Evaluation criteria that evaluate project success and which are applied to a selection of historical software development projects are described.

The selected software development projects are categorised into three categories of small, medium, and big projects and two projects of each category are evaluated against the evaluation criteria. This chapter forms part of the diagnosing phase of the action research cycle that will be applied to software development projects which are discussed in Chapters 6 to 8. In the last section of this chapter, the selection process that was applied to selecting the projects used in this research is explained.

5.1 Organisational environment characteristics

The organisation in which this study is conducted is in the agricultural industry and is over one hundred years old with a staff complement of thirteen thousand people. With a turnover of over eight billion rand per annum the organisation's supply chain ranges from procurement, manufacturing, and distribution to commercial, marketing and sales functions. Information Technology is one of the supporting departments within the organisation and performs software development based on the needs and requirements of the various business units. Application development is a central function available to the organisation and the development teams are co-located at the organisation's head office. The Information Technology department consists of the following application development teams:

- Workflow application development
- Application development
- System integration
- Database administration
- Report writing

A software development life cycle is followed during the process of systems development with formal approval processes and according to software development standards. These processes involve the following:

Systems Development

A structured approach to software development and maintenance is followed to ensure systems are developed and maintained to meet business management goals. Applications follow a formal software development life cycle (SDLC) process to ensure that new and changed software are implemented with a high degree of risk management. The same process is followed when new software is acquired from a third-party software supplier. The organisation follows the following SDLC phases:

System initiation

- A need or opportunity is defined and logged.
- A preliminary analysis or feasibility study is conducted.
- A project charter (if necessary) is formulated.

System requirements analysis

- Analyse user needs and develop user requirements.
- Create a detailed functional requirements document.
- Break down the system, process, or problem into discrete units or modules and utilise diagrams and other visual tools to analyse the situation or need.
- Define and scope any hardware requirements.
- Define any security requirements.
- Identify risks and define mitigation.

System design

- This phase transforms the requirements into a design document.
- The functions and operations of the system or software being designed are described in detail.
- A risk analysis should be done between the system requirements and system design phases.
- A final design review should be done to ensure the design addresses practicality, efficiency, cost, flexibility, and security.

System construction/development

- This phase entails the transformation of the detailed design documents into a finished product or solution.
- Testing at a unit or module level is done throughout this phase by the system or software developers. Security considerations are considered during testing.
- A third-party product may be utilised as a system or software solution if it best fits the user requirements and is more practical from a budgetary and/or resource perspective. However, all the next phases should be followed regardless of whether the solution was developed in-house or purchased.

System testing and acceptance

- This phase should validate or confirm that the developed system or software meets all functional requirements as captured during the system requirements analysis phase.
- All development and testing of software solutions must be done in the development/UAT environment before it is deployed to the production/live environment.
- Representatives separate from the development group should conduct internal quality assurance (QA) testing.
- Representative(s) from the user group should conduct user acceptance testing.
- Documentation during testing should detail and match the testing criteria to the specific requirements.
- While unit and module testing should be done throughout the entire SDLC, this phase entails end-to-end testing of the finished product and the final acceptance testing by the user(s).
- Final security assessment testing is then conducted if required.
- Any problems identified during the previous phases must be resolved or remediated before implementation.

System implementation

- The finished, tested, and user-accepted system or software is moved from the testing environment to production.
- Any user training required should be done prior to or during this phase.
- Objects will be made live by the respective operational environment staff instructed to do so by an approved change control.
- Development and quality assurance staff must not have access to the live environment.

System maintenance

- This phase is the ongoing maintenance of the system or software. Unlike the other phases, this phase only ends when the system or software is decommissioned.
- A customer/user support structure and any other necessary operational support processes should be in place.
- Any planned changes to the system or software should be scheduled, communicated, and documented.
- Mandatory security testing is conducted when any major configuration or architecture change is made.
- System documentation must be kept continuously up to date through all the SDLC phases.

System support

User manuals are made available to users with regular updates and changes as and when they happen.

All issues, enhancements and bugs are logged at a central helpdesk for assignment to the relevant technical staff. A reference number is generated and issued for further inquiries and progress tracking.

Technical staff records the helpdesk reference number on all tasks undertaken.

Systems development authorisation

All systems development and acquisitions require prior approval from the relevant business manager and the group manager of information systems. In some instances, steering committee approval may be required. There is a formal electronic approval and prioritisation system that determines the priorities of the development requests.

Software development standards

Development follows predefined standards as far as possible to ensure the following:

- Product quality
- Deterministic defect control
- Code quality
- Optimising team member interactions
- Traceability
- Velocity
- Maintainability
- Usability
- Application security

Summary

Although the organisation follows a software development life cycle process, it is not evident that a formal software development methodology is followed. Consequently, no formal project management methodology is followed to develop and implement software development projects.

5.2 Evaluation criteria

According to Kerzner (2009:7), project success can be defined as the completion of a project within an agreed time period, within budget, at an acceptable performance or specification level, with acceptance by the customer, with minimum or mutually agreed upon scope changes, without affecting the main workflow of the organisation and without changing the corporate culture. Prabhakar (2008:4) states that project success is more than meeting cost, schedule, and performance specifications and claims that client satisfaction plays a key role in the

perceived success or failure of projects. Furthermore, Sudhakar (2016:163-164) explains that project success relates to the completion of a project in time, within the given budget, and which meets the customer requirements with the specified quality. Similarly, different stakeholders interpret a project to be successful from different perspectives. For example, team members of a project may not deem a project as successful when compared to senior management due to the additional hours they worked to deliver the project. In the same way, a customer may not regard a project as successful compared to the project team due to time delays and cost overruns.

Based on the definitions of project success, the following criteria were identified as important factors to determine if a software development project was successful for a specific organisation. Due to different interpretation of project success by different stakeholders, questionnaires were also incorporated as part of the evaluation criteria in addition to the measurable success factors.

Evaluation criteria for project success:

- *Project scope:* The project scope is used as the baseline requirement to determine if the project was delivered according to customer expectation.
- *Stakeholder size:* The stakeholder size has an impact on the priority of the project and the impact it has on the organisation.
- *Project delivery on time:* The project end date is measured against a baseline schedule that is drawn up during the project planning phase to establish if the project was delivered as promised.
- *Rework post implementation:* This involves the number of defects identified after the project is taken into production which need to be fixed.
- *Modifications due to scope changes:* These are the number of change requests that are required during the project development and implementation phases. These changes are not part of the original scope of the project.
- *Project methodology followed:* This involves the evaluation if the project was executed by means of a project software development methodology.
- *Project priority (High, Medium and Low):* Each project is prioritised, based on calculated criteria which are taken into consideration during the evaluation process.
- *Implemented and in use:* This measures if the project is applicable and being used within the organisation.

5.3 Previously executed projects with no methodology

Part of the diagnosing phase is to investigate previously executed projects to determine how successful historical projects have been in the organisation. The projects are categorised into three categories namely small, medium and big projects. The organisation uses a mathematical model to calculate the priority of a project which relates directly to the size of the project. The mathematical model calculates a weighted average score based on factors that are important to the organisation in terms software development and is completed by the internal Information Technology department prior to starting the project. It is based on what the project team knows before commencing with the project. This includes the following:

- Impact

  These factors are measurements of the overall impact the software development project will have on the organisation and is broken down into the following components:

  - *Pain of the user* – this involves the impact the existing situation has on the customer or business user. It is an indication of the extent to which the user's existing situation can be changed or improved if the project is implemented.
  - *Percentage of customers impacted* – this is an indication of the impact on the number of users and departments within the organisation.
  - *How often the "new feature" will be used* – this component indicates the frequency with which the project once implemented will be used within the organisation.
  - *Customer value* – this component is an indication of the perceived value the customer or business user allocates to the project
  - *Volume of programs to be changed* – this is an indication of the impact the project will have on existing systems or modules of it involves a change within an existing system.
  - *Effect on system* – this component indicates the effect the change will have on the business operations within the organisation.

- Strategic

  This factor indicates the strategic alignment of the project related to the organisation's business strategy and consists of four selections, namely alignment to the business strategy, alignment to the information technology strategy, regulatory requirement or no alignment to any strategy of the organisation.

- Cross-functional

  The factor is an indication of whether the project stretches across organisational functions and departments.

- Complexity

Complexity indicates what the complexity of the project is in terms of the software development.

- Existing system change

  This is an indication of whether the project involves a change in or improvement on an existing system, or if a new system needs to be developed or procured.

- Resource requirements

  This component estimates the number of technical and project resources that are required to execute the project.

- Development time

  This is an estimate of the amount of software development time that would be required to complete the project successfully.

- Risk

  This is an indication of the risk the project poses to the organisation in terms of implementation. It also includes the evaluation of whether an existing risk within the organisation will be addressed by implementing the project.

- IT/Business requirement

  This is an indication of whether the project is related to the business within the organisation or if the project is related to the information technology department.

- Executive Committee influence

  Requirements directly from the executive committee are prioritised higher compared to other software development projects.

- Financial

  This factor indicates the financial impact of the project in terms of the funds required to implement the software development project.

Six historical projects were selected based on the criteria explained in the previous section. Two projects of each category were selected which categorised them into two big projects, two medium projects and two small projects. Table 5-1 is a summary of all the completed projects from 2010 to 2019. Historical projects selected for this study were more recent projects and fall within the average priority score of each category.

Table 5-1: Historical project selection data

Project classification	Project priority	Number of projects	Minimum priority	Maximum priority	Average priority
Big	High	59	187	352	270

Table 5-1: Historical project selection data (continued)

Medium	Medium	139	119	186	152
Small	Low	34	70	118	94
Total		**232**			

The selected historical projects were all recently completed during the time of the study which makes them more representative and represents the majority of the type of software development projects that are mostly executed within the organisation. The priority scores are aligned with the average priority scores of all historical projects. Table 5.2 displays the priority scores and classification of the selected projects.

Table 5-2: Project priority scores and classification

Project Description	**Project classification**	**Project priority**	**Project priority score**
Capital expenditure application	Big	High	237
Fleet management system	Big	High	230
Dealsheet management system	Medium	Medium	175
Frontend – Integrated business	Medium	Medium	168
Mobile device ordering system	Small	Low	101
Product configuration	Small	Low	95

The selected historical projects that were evaluated based on the criteria in sections 5.2.and 5.3 will be discussed in the following sections.

5.3.1 Capital expenditure application system (big project)

This project involved the development of a system to manage capital expenditure within the organisation, thereby ensuring that capital applications are within the agreed budgets and are approved prior to the spending of any capital.

Project scope

The scope of the project was to develop a capital expenditure (capex) application and approval workflow system used as the platform to apply for capital expenditure within the organisation. The aim of the project was to ensure that all capital expenditure gets approved against a predefined budget before capital can be spent. The system also needed to provide the necessary management information and reports related to the organisation's capital expenditure.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-3.

Table 5-3: Customer satisfaction feedback for capex application system

Customer	Score (1 = Poor, 10 = Excellent)
Group Manager: Procurement	6
Group Financial Manager	7
Procurement Manager: Indirect Spend	5
Procurement Manager: Direct Spend	6
Financial Accountant	6
Financial Controller: Fixed Assets	7

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-4 presents the stakeholders who were involved in the project and Table 5-5 summarises the agreed project milestones by comparing the baseline timelines with the timelines that were actually achieved.

Table 5-4: Project stakeholders for capex application system

Stakeholder Position	Interest
Business Representatives	
Group Manager: Procurement	Customer
Group Financial Manager	Customer
Procurement Manager: Indirect Spend	Customer
Procurement Manager: Direct Spend	Customer
Financial Accountant	Customer
Financial Controller: Fixed Assets	Customer
Technical Team	
Group Manager: Information Services	System Delivery
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
System Analyst ERP	Developer

Table 5-4: Project stakeholders for capex application system (continued)

Applications Architect	Applications Architect
Business Intelligence Manager	Information Delivery
Business Information Analyst	Report Developer

Table 5-5: Project delivery against agreed milestones for capex application system

Project Phases	Baseline Milestones	Actual Delivery
Project Proposal	1 August 2017	1 August 2017
Design	9 August 2017	9 August 2017
Development	10 January 2018	15 January 2018
Technical testing 1	24 January 2018	26 January 2018
User Acceptance Testing 1	7 February 2018	20 February 2018
Modifications		7 March 2018
Technical testing 2		13 March 2018
User Acceptance Testing 2		30 March 2018
Modifications		6 April 2018
Technical testing 3		11 April 2018
User Acceptance Testing 3		25 April 2018
Implementation	14 February 2018	30 April 2018
Go-Live	14 February 2018	30 April 2018

Rework post implementation

Table 5-6 demonstrates all the changes and fixes that were required after the project was deployed and implemented.

Table 5-6: Rework post-project implementation for capex application system

Call Nr	Description	Reported	Completed	Duration (Working Days)
00010828	Errors on Capex on live server	1 May 2018	1 May 2018	1
00010855	Capex Application - Access to edit Draft Capexes	3 May 2018	18 May 2018	12
00010861	Fix Advanced Search view on Capex	9 May 2018	24 May 2018	12

Table 5-6: Rework post-project implementation for capex application system (continued)

00010872	Fixed Copy To - built a field containing all required recipients and used the field in the mail event.	10 May 2018	10 May 2018	1
00010904	User is unable to approve Capex from the supplied link	10 May 2018	10 May 2018	1
00010946	Declined CAPEX that could not be retrieved	24 May 2018	11 June 2018	13
00010967	The Capex was submitted but never landed on the worklist of the approver	25 May 2018	4 June 2018	7
00010970	Motivation on Capex not displayed correctly	29 May 2018	12 June 2018	11
00010991	Capex display not calculating correctly	31 May 2018	5 June 2018	4
00011117	Users are unable to add group members on Capex	6 June 2018	11 June 2018	4
00011332	Capex - ensure messages direct the user to the new menu form rather than the old site.	7 June 2018	21 June 2018	11
00011816	Capex Folio issue	8 June 2018	8 June 2018	1
00011983	Update Capex application so that the K2 Originator control is not used in the workflow	8 June 2018	3 July 2018	18
00012573	Capex - Investigate speed on Capex Approval Screen	10 June 2018	14 June 2018	4
00012586	The approver screen on Capex is very slow	11 June 2018	21 June 2018	9
00013518	Address speed issues on the Capex application	13 June 2018	16 June 2018	3
00013529	Capex has the incorrect status at certain points in the workflow	15 June 2018	15 June 2018	1
Total Count = 17				**113**

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-7.

Table 5-7: Modifications during the project due to scope changes for the capex application system

Call Nr	Description
00010842	Capex - users need to see the progress of Approvals in the Capex application
00010879	The Final approver is not able to update the authorised amount for a Capex
00010891	Capex - Add SSRS report
00010899	Add Print Button to Capex
00011117	Users are unable to add group members on Capex
00011289	Add notifiers to the Capex to reduce the number of approvals

Table 5-7: Modifications during the project due to scope changes for the capex application system (continued)

00011539	Capex Batch Approval
00012386	Allow attachments to be added to a Capex until final approval step
00012437	Please add reviewer option to the CAPEX system
00012698	Capex spend monitoring and budget integration
00013032	Capex - Improve Capex Notifications
00013028	Capex - Change Branch Selection on Capex project Screen
00013110	Standardise Capex buttons
00013285	Capex projects Enhancement
00013285	Restrict access to specific projects to the Requester
00013285	Block existing application number on Main Application
00013285	Check error on duplicate
00013285	Add a warning: Total budget value of the sub-votes is greater than the Budget on the main application
00013285	Remove the "Step 1/2/3" part of the heading on the application screens. Change Existing Asset to "Existing Asset (if applicable)"
00013285	Add check box on Existing Asset screen to show or hide the information.
00013285	Refresh the list of projects on close of display.
00013285	No changes to the main project detail after the main application has been approved. The user may make changes to the sub-votes.
00013285	Allow the user to capture sub-votes with a zero-budget value.
00013285	Ensure that the Responsible Person and the Creator of the project have access to the Capex Applications for the project.
00013285	Add Application Amount and Date Approved to Annexure D and the main application display on the project form.
00013285	Open the workflow item for sub-votes from the project screen.
00013285	Add validations to the capex application: At least one motivation, at least one quotation.
00013285	Add Capex projects Menu options.
00013285	Add a link to the Annexure D report to the project form.
00013578	Separate affiliates' data on the Capex application - views and reports
00013665	Add approval step for CFO to Capex process
00013727	Change the Capex approval cycle to allow email approvals
00013730	Capex project management - add sub-sub-votes
Total Count = 33	

Project priority

The project classification is based on a priority calculation that is presented in Table 5-8. The calculation takes various components regarded as important to the organisation into

consideration. They are used to calculate a priority score that determines the priority of the project. This calculation is performed during the project evaluation phase during which the project team completes a questionnaire which is used to calculate a priority score. The project was classified as a high priority project.

Table 5-8: Project priority classification for the capex application system

IMPACT	
Pain of User	4
% of Customers Impacted	21-50
How often the "new feature" will be used?	Daily
Customer value	4
Volume of programs to be changed	New System
Effect on system	High
STRATEGIC	
Is it strategically aligned?	Aligned to business
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-Functional)	Over Business Units
COMPLEXITY	
What is the complexity of the request?	High
EXISTING SYSTEM CHANGE	
What level of the existing system will require change?	New System
RESOURCE REQUIREMENTS	
What resources are required?	Labour > 160 hours
DEVELOPMENT TIME	
Development Time	> 4 Weeks
RISK	
Risk	High
IT/BUSINESS REQUIREMENT	
IT/Business	Business
EXCO INFLUENCE	
Exco Influence	Yes
FINANCIAL	
Capex Required	No
PRIORITY SCORE	**237**
PRIORITY SCORE RESULT	**High Priority**

Implemented and in use

One of the key measurements to determine if a project was successful is to establish if the solution is used by the customer. Software development solutions that are not used by the customers are indications of project failures. This project was implemented successfully and is currently in use by the customer.

Project methodology followed

This evaluation is to determine if the project followed any kind of project or development methodology to deliver the end solution. The project was executed by means of the following actions and processes:

- No structured methodology was followed for project execution.
- No formal software development methodology was followed.
- Requirements were gathered, and development was purely based on the requirements document.
- Users were only involved in the first user acceptance testing phase.
- No continuous customer involvement was followed.

Problems/Challenges identified

The following problems were experienced through the project life cycle:

- Late project delivery – The project was completed two-and-a-half months behind schedule.
- Thirty-three modifications were required during user acceptance testing that was not in scope.
- Delays due to errors in development were experienced with a total of seventeen rework calls after the implementation of the project.
- No PMM was followed.
- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic, especially during user acceptance testing.
- The customer experience was poor, based on feedback from the customer.

Table 5-9 summarises the outcome of the evaluation criteria as it was applied and discussed in this section.

Table 5-9: Summary of the Capital expenditure application system

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Capex Application System	Big project	High	6.1	33	2.5 months overrun	17

5.3.2 Fleet management system (big project)

This project involved the development of a system to manage the fleet claims with an automated workflow. The previous process was a manual process which caused delays in the completion of the documentation needed to complete claims related to incidents.

The process involved role players, such as the organisation employees, organisation managers, fleet manager, flexi-fleet and the insurance broker. The process included actions, such as the manual logging of a call, the electronic initiation and submission of a claim in the organisation's fleet claim database, and routing of documents, depending on the workflow.

Project Scope

The scope of the project included the following:

- Creating a workflow system to handle document routing between the organisation's employee, organisation manager, fleet manager, flexi-fleet and the insurance broker.
- Establishing an improved method of communication between different role players in order to avoid delays.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-10.

Table 5-10: Customer satisfaction feedback for the fleet management system

Customer	**Score (1 = Poor, 10 = Excellent)**
Group Manager: Procurement	5
Manager Fleet	5
Shipping Manager	6
Administrative Controller: Fleet	6

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-11 presents the stakeholders who were involved in the project and Table 5-12 summarises the agreed project milestones by comparing the baseline timelines with the timelines that were actually achieved.

Table 5-11: Project stakeholders for the fleet management system

Stakeholder Position	**Interest**
Business Representatives	
Group Manager: Procurement	Customer

Table 5-11: Project stakeholders for the fleet management system (continued)

Manager Fleet	Customer
Shipping Manager	Customer
Administrative Controller: Fleet	Customer
Various Branches	Customer
Technical Team	
Group Manager: Information Services	System Delivery
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
Applications Architect	Applications Architect
Integration Architect	Integration Developer

Table 5-12: Project delivery against agreed milestones for the fleet management system

Project Phases	**Baseline Milestones**	**Actual Delivery**
Project Proposal	4 April 2017	4 April 2017
Design	13 April 2017	13 April 2017
Development	5 June 2017	13 June 2017
Technical testing 1	19 June 2017	29 June 2017
User Acceptance Testing 1	3 July 2017	10 August 2017
Modifications		24 August 2017
Technical testing 2		7 September 2017
User Acceptance Testing 2		11 October 2017
Modifications		30 October 2017
Technical testing 3		10 November 2017
User Acceptance Testing 3		7 February 2018
Implementation	17 July 2017	19 February 2018
Go-Live	17 July 2017	19 February 2018

Rework post implementation

Table 5-13 demonstrates all the changes and fixes that were required after the project was deployed and implemented.

Table 5-13: Rework post-project implementation for the fleet management system

Call Nr	Description	Reported	Completed	Duration (Working Days)
00010422	Fleet claims is allowing calls to be logged against inactive vehicles - should only use active vehicles.	11 August 2017	14 August 2017	2
00010557	Cannot delete any request once captured	15 August 2017	5 September 2017	16
00010798	Replace Fleet system	20 August 2017	21 August 2017	1
00010993	Update the Make and Model of Fleet vehicles	23 August 2017	24 August 2017	1
00011291	Change the links on the Fleet Management e-mails to point to the new menu	4 September 2017	10 September 2017	5
00012125	Fleet Management - Order Escalation	6 September 2017	1 November 2017	41
00012478	Adjust the fleet list import with changes from	23 September 2017	16 October 2017	16
00012748	Add new cost codes to Fleet system	6 October 2017	7 October 2017	1
00012864	Fleet list Management - Maintain forklifts on fleet list	7 October 2017	28 October 2017	15
00013537	Invalid data sent from causing problems in Fleet system	15 October 2017	22 October 2017	6
00013960	Import fuel card data into the Fleet database	17 October 2017	18 October 2017	1
Total Count = 11				**105**

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-14.

Table 5-14: Modifications during the project due to scope changes for the fleet management system

Call Nr	Description
00010544	Include the request number on the .pdf to be sent to EQSTRA
00010544	Add an approval schedule list showing where the request is in the approval cycle on each request form
00010544	Create the Branch Vehicle Report and submit to the branch notifiers scheduled on the 15th of the month
00010544	Create the process to upload the fleet data to the insurance application, scheduled on the 23rd and 24th of the month
00010544	Add a Vehicle Type report that can be downloaded to excel

Table 5-14: Modifications during the project due to scope changes for the fleet management system (continued)

00010544	Ensure that the Transfers do not require duplicate approvals when the from and to branch approvers are the same
00010622	Allow the addition of new vehicles which are not managed on the Fleet vehicle list
00010856	Fleet list management - Vehicle Management – Trucks
00011340	Fleet Management changes for Toyota integration
00011649	K2 Fleet Request - Add responsible person's ID number
00011972	Add Masakane split as well
00012051	Update security on the Fleet Management System
00012052	Add the old fleet number on reports for re-fleet requests
00012103	Change the display of Fleet Requests to use a smart object that includes active and discharged employee details.
00012696	Link NIA to vehicle on Fleet list
00012871	Fleet Management - New additional vehicles must have a print function
00012872	Fleet Management - to change the fields so that the reporter can edit them
00013027	Create new fleet category for Masakane
00013611	Create Procurement Fleet SharePoint site.
00013815	Add field for fuel card numbers
00013814	Take over functionality and tasks from Fleet Company
Total Count = 21	

Project priority

The project was classified as a high priority project, based on the priority calculation explained in section 5.1.2.1. The priority classification is presented in Table 5-15.

Table 5-15: Project priority classification for the fleet management system

IMPACT	
Pain of User	4
% of Customers Impacted	0-5
How often the "new feature" will be used?	Daily
Customer value	4
Volume of programs to be changed	New System
Effect on system	High
STRATEGIC	
Is it strategic?	Aligned to business strategy
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Not Cross-functional
COMPLEXITY	
What is the complexity of the request?	High

Table 5-15: Project priority classification for the fleet management system (continued)

EXISTING SYSTEM CHANGE What level of the existing system will require change?	New System
RESOURCE REQUIREMENTS What resources are required?	Labour > 160 hours
DEVELOPMENT TIME Development Time	> 4 Weeks
RISK Risk	Medium
IT/BUSINESS REQUIREMENT IT/Business	Business
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**230** **High Priority**

Implemented and in use

This project was implemented successfully and is currently in use by the customer.

Project methodology followed

This evaluation is to determine if the project followed any kind of project or development methodology to deliver the end solution. This project was executed by means of the following actions and processes:

- No structured methodology was followed for project execution.
- No formal software development methodology was followed.
- Requirements were gathered, and development was purely based on the requirements document without user involvement until user acceptance testing.
- Users were involved from the first user acceptance testing phase.
- No continuous customer involvement was followed.

Problems/Challenges Identified

The following problems were experienced through the project life cycle:

- Late project delivery – Project was completed 6.7 months behind schedule.
- Twenty-one modifications were required during user acceptance testing that were not in the project scope.
- Delays due to errors in development were experienced with a total of eleven rework calls.
- No PMM was followed.

- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic during user acceptance testing.
- The customer experience was poor, based on feedback from the customer.

Table 5-16 summarises the outcome of the evaluation criteria as it was applied and discussed in this section.

Table 5-16: Summary of the Fleet management system

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Fleet management system	Big project	High	5.5	21	6.7 months late	11

5.3.3 Dealsheet management system (medium project)

A system was required to manage deals that are negotiated with a buyer during a customer engagement. The system had to ensure that the following functions were performed with regard to a concluded deal:

- Deal Capture
- Deal Approval
- Key account deal capture

The existing process at that time was a manual process with no approval. All deals below a promotional price had to go for special approval to the marketing department and once approved was communicated manually to the pricing department. The supply chain planners were notified accordingly to plan for all deals to ensure that the required stock was available.

The new process had to include the capturing of deals, the approval of deals, capturing of advertisement periods and the auto-population of prices.

The process involved role players, such as the employees in the key accounts department, marketing, pricing and supply chain planning departments.

Project Scope

The scope of the project included the following:

- Develop a workflow and approval process to ensure deals are not below promotional prices and automating the capturing of prices to prevent price claims.
- Generate timeous notifications of deals and prices to the involved departments thereby ensuring that the pricing is correct so that supply chain efficiencies are met.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-17.

Table 5-17: Customer satisfaction feedback for the dealsheet management system

Customer	**Score (1 = Poor, 10 = Excellent)**
Group Manager: Key Accounts	6
Key Account Managers	5
Manager: Pricing	6
Brand Managers	4

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-18 presents the stakeholders who were involved in the project and Table 5-19 summarises the agreed project milestones by comparing the baseline timelines with the timelines that were actually achieved.

Table 5-18: Project stakeholders for the dealsheet management system

Stakeholder Position	**Interest**
Business Representatives	
Group Manager: Key Accounts	Customer
Key Account Managers	Customer
Manager: Pricing	Customer
Brand Managers	Customer
Technical Team	
Group Manager: Information Services	System Delivery
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
Applications Architect	Applications Architect

Table 5-19: Project delivery against agreed milestones for the dealsheet management system

Project Phases	Baseline Milestones	Actual Delivery
Project Proposal	4 June 2016	4 June 2016
Design	10 August 2016	15 August 2016
Development	21 September 2016	5 October 2016
Technical testing 1	10 October 2016	21 October 2016
User Acceptance Testing 1	24 October 2016	10 February 2017
Modifications		10 March 2017
Technical testing 2		14 March 2017
User Acceptance Testing 2		28 August 2017
Modifications		11 September 2017
Technical testing 3		19 September 2017
User Acceptance Testing 3		8 December 2017
Implementation	7 November 2016	5 February 2018
Go-Live	7 November 2016	5 February 2018

Rework post implementation

Table 5-20 demonstrates all the changes and fixes that were required after the project was deployed and implemented.

Table 5-20: Rework post-project implementation for the dealsheet management system

Ref No	Brief Description	Reported	Completed	Duration (Working Days)
00011423	DMS - Deal loading more time effective	12 March 2017	16 March 2017	4
00011468	DMS - Retracting of Deals	20 March 2017	15 June 2017	64
00011714	DMS - Approval Cycle determined on Product split	10 April 2017	24 April 2017	11
00011914	DMS - Basket Deals - Channel and Promo Prices does not pick up correctly when basket is selected	17 April 2017	18 April 2017	2
00012000	Deal Confirmation - Notification does not pick up the description of the Basket if the deal is loaded with a Basket Code	24 April 2017	4 July 2017	52
00012072	Deal Confirmation - Incorrect when loaded on a Basket	2 May 2017	2 June 2017	24

Table 5-20: Rework post-project implementation for the dealsheet management system (continued)

00012154	DMS Request - Error cannot complete the Child Requests	16 May 2017	17 May 2017	2
00012384	Auto Approval - Status not updated	29 May 2017	2 June 2017	5
00012419	DMS price not the same on KAS, see attachment	13 June 2017	14 June 2017	2
00012449	DMS - Products do not save when adding a new product	18 June 2017	17 July 2017	21
00012954	DMS - Speed Improvement	29 June 2017	5 July 2017	5
00013004	Please see below no price is currently on KAS for the Snack Pack – we cannot load cycle deals on DMS as it shows 0	16 July 2017	17 July 2017	1
00013241	DMS - Fixes	19 July 2017	9 September 2017	38
00013956	DMS - Fix Activity spelling (Marketing Approval)	23 July 2017	13 August 2017	15
00014055	DMS - User not able to work due to win 2000 / post win 2000 user ID issue	15 August 2017	17 August 2017	3
00014113	DMS - Copy Deal - Records don't delete from deal when copied and then deleted before submitting	17 August 2017	11 September 2017	18
Total Count = 16				**267**

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-21.

Table 5-21: Modifications during the project due to scope changes for the dealsheet management system

Call Nr	Description
00010399	Attachments of documents functionality to DMS
00010400	Easy Product Search
00011353	DMS - Export Deal
00011424	DMS Reporting
00011445	DMS – Integration with KAS
00011597	DMS - Consolidated Printout for multiple KAM deals
00011893	Automate DMS into KAS
00011940	Change stored procedure getting channel prices for the DMS system to use the key account regions reporting table for linking to the channel price
00011967	DMS Report

Table 5-21: Modifications during the project due to scope changes for the dealsheet management system (continued)

00012142	DMS Deal Report - Not Including Basket Deals
00012193	Increase the Alternate Deal Number on the Cycle \ Promo \ Customer Promo tables to 60 characters as required by the DMS system
00012223	Deal Management System - Marketing approval
00012334	DMS Deal - Deal Capture to KAM
00012335	DMS - Warning when prices are captured higher than list or channel
00012541	Add Namibia to this system
00012679	DMS Enhancements
00012702	DMS Enhancements
00012984	DMS - Marketing Approval Automated retraction
00013081	Marketing Events - Integration with DMS
00013175	DMS - Deals
00013253	IDEAS report
00013304	DMS - Marketing Email Approval \ Below Super Promo Configuration Settings
00013945	DMS - Kas Integration Changes - New System
Total Count = 23	

Project priority

The project was classified as a medium priority project, based on the priority calculation explained in section 5.1.2.1. The priority classification is presented in Table 5-22.

Table 5-22: Project priority classification for the dealsheet management system

IMPACT	
Pain of User	4
% of Customers Impacted	21-50
How often the "new feature" will be used?	Daily
Customer value	5
Volume of programs to be changed	New System
Effect on system	High
STRATEGIC	
Is it strategic?	No alignment
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Over Business Units
COMPLEXITY	
What is the complexity of the request?	High
EXISTING SYSTEM CHANGE	
What level of the existing system will require change?	New System

Table 5-22: Project priority classification for the dealsheet management system (continued)

RESOURCE REQUIREMENTS What resources are required?	Labour > 160 hours
DEVELOPMENT TIME Development Time	> 4 Weeks
RISK Risk	Low
IT/BUSINESS REQUIREMENT IT/Business	Business
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**175** **Medium Priority**

Implemented and in use

This project was implemented successfully and is currently in use by the customer.

Project methodology followed

- No structured methodology was followed during the project execution.
- No formal software development methodology was followed during the project execution.
- Requirements were gathered, and development was purely based on the requirements document without user involvement until user acceptance testing.
- Users were only involved from the first user acceptance testing phase.
- No continuous customer involvement was followed.

Problems/Challenges Identified

The following problems were experienced through the project life cycle:

- Late project delivery – The project was completed 15.1 months behind schedule.
- Twenty-three modifications were required during user acceptance testing that were not in the project scope.
- Delays due to errors in development were experienced with a total of sixteen rework calls.
- No PMM was followed.
- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic during user acceptance testing.
- Customer experience was poor, based on feedback from the customer.

Table 5-23 summarises the outcome of the evaluation criteria as it was applied and discussed in this section.

Table 5-23: Summary of the dealsheet management system

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Dealsheet management system	Medium project	Medium	5.3	23	15.1 months late	16

5.3.4 Frontend – Integrated business planning (medium project)

Integrated business planning (IBP) is a strategy for connecting the planning functions of each department in an organisation to align operations and strategy with the organisation's financial performance. The organisation required an application where master data could be managed and maintained for the central integrated business planning system.

Project Scope

The scope of the project included the following:

- Create a frontend to maintain the master data for the integrated business planning system.
- No further detail scoping could be done due to business not knowing what would be required.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-24.

Table 5-24: Customer satisfaction feedback for the integrated business planning system frontend

Customer	Score (1 = Poor, 10 = Excellent)
Group Manager: Supply Chain Planning	5
Demand Planners	4
Supply Planners	4
Deployment Planners	5

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-25 presents the stakeholders who were involved in the project and Table 5-26 summarises the agreed project milestones. No baselines were defined for the various project phases. An implementation date and go-live date were dictated as this project was a dependency for the main IBP project.

Table 5-25: Project stakeholders for the integrated business planning system frontend

Stakeholder Position	Interest
Business Representatives	
Group Manager: Supply Chain Planning	Customer
Demand Planners	Customer
Supply Planners	Customer
Deployment Planners	Customer
Technical Team	
Group Manager: Information Services	System Delivery
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
Applications Architect	Applications Architect

Table 5-26: Project delivery against agreed milestones for the integrated business planning system frontend

Project Phases	Baseline Milestones	Actual Delivery
Development	Not defined	Various
Technical testing	Not defined	Various
User Acceptance Testing	Not defined	Various
Modifications	Not defined	Various
Implementation	September 2016	3 February 2017
Go-Live	September 2016	3 February 2017

Rework post implementation

A total of 193 changes and fixes were requested after the go-live date which prevented the project from being closed until November 2017.

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-27.

Table 5-27: Modifications during the project due to scope changes for the integrated business planning system frontend

Ref No	Brief Description
00013835	IBP project - Add routing method between Agilisys and LX
00013855	IBP - FE0144 - Referential integrity on SQL
00013856	IBP - FE0149 - Route - Source and destination may not be modified
00013861	IBP - FE0193 - Customer Item Directs - New Maintenance Screen
00013877	IBP - Route Maintenance - Lead Time
00013898	FE0061 - IBP - Code Master Maintenance
00013901	IBP - FE0187 - Add a new field to the Item Group maintenance screen
00013902	FE0188 - IBP - Item Maintenance - Add a delete button on the recipes tab
00013914	IBP - KG Per Unit - not displayed correctly
00014033	IBP - FE106 - Item Location
00014170	IBP Masters - Item Group - Short Supply Percentage
00011507	IBP Front Ends project
00011526	IBP - Change column field length in tables and on Related IBP Frontends mCode_Type, mCode_Master, mCode_Conversion
00011531	IBP - Where we have Boolean (True/False) fields please make True = 1 rather than -1.
00011532	IBP - New Frontend Validations on ITEM_Sourcing and Item_Location_Storage
00011534	IBP - change validation on tMilk_Forecast
00011563	Enhancements on the IBP Milk Forecast Front-end
00011567	Milk Forecast front-end. Add Day of week on frontend, Add location filter on IBP graph, change forecast end date to Newest
00011818	Rename fields on the IBP Database tables
00012148	IBP Batch Copy function on frontends
00012169	IBP - Location Maintenance
00012170	IBP - Code Conversions

Table 5-27: Modifications during the project due to scope changes for the integrated business planning system frontend (continued)

00012171	IBP - Item Location
00012172	IBP - Route Maintenance - Incoming and Outgoing
00012173	IBP - Item Route Maintenance
00012238	Compare IBP masters to Master Front End
00012254	Build a Batch Job Admin for IBP
00012287	IBP Status field Integrity control
00012337	IBP - Milk Imports to Empty Data before Every Import
00012354	IBP Item Master - Default Date to00/00/0000
00012462	Additions to the Item Maintenance IBP Screen
00012463	Update mRoute Table on IBP Masters Front-end
00012481	IBP - Route Maintenance display
00012482	IBP Location Maintenance Lead Time
00012506	IBP check for Satellite of Item Location on AS400
00012518	IBP project Import and upgrade from Unipaas to XPA
00012519	IBP Masters Test for Location Satellite on all route Forms
00012533	Change the FSCIBP job to please also extract the daily efficiency field to the FSCIBP table
00012582	IBP - mRecipe Status management
00012658	IBP - Do not overwrite data on milk cost Import
00012659	IBP Milk cost Import - Add Export Button for Milk Cost Import
00012668	IBP Intermediate Group selection to look at Intermediate group table
00012669	IBP - Change the Milk Transfer Import to be as per attachment
00012671	IBP Add new Field to mRecipe table and add functionality
00012673	IBP - Add export functionality to Milk Transfer cost
00012750	IBP Milk Cost Frontend changes
00012753	New Front-end for IBP Master Frontend mCompReqRatio
00012755	IBP Master-Frontend - functionality to auto navigate when location not active
00012756	IBP Frontend de-activate Insert mode when users scroll down to bottom of list in modify mode
00012760	IBP Masters Frontend- Disable Scroll when on Item Master
00012761	IBP - Masters Frontend - Disable Insert on all lists when in modify and scrolling down

Table 5-27: Modifications during the project due to scope changes for the integrated business planning system frontend (continued)

00012774	IBP Masters Frontend - Enable DOUBLE Click function to go into Edit mode on all forms
00013119	IBP Masters Frontend - Code Conversion - Cannot Insert
00013123	IBP Masters Frontend - Location Maintenance - Cannot Clear Filter criteria
00013207	IBP - Master Screen Issues
00013224	IBP Masters: Intransit_Split
00013255	IBP - Location Maintenance screen slow response due to incorrect MCode_Conversion Factor
00013439	IBP Masters - Location Status update cascade to related tables error
00013673	IBP project - Add routing method between Agilisys and LX
00013699	IBP frontend - Locations Maintenance - Error
00013725	IBP Masters - Fixes
00013740	IBP - Item Maintenance - Base UOM
00013743	IBP Masters - By Product Dropdown not retrieving information
00013810	IPB Masters - IBP route deactivation
00013857	IBP - FE0150 - Item Routes - Source and destination may not be modified
00013860	IBP - FE0192 - Route Maintenance - Column Validations
Total count = 67	

Project priority

The project was classified as a medium priority project, based on the priority calculation explained in section 5.1.2.1. The priority classification is presented in Table 5-28.

Table 5-28: Project priority classification for the integrated business planning system frontend

IMPACT	
Pain of User	4
% of Customers Impacted	21-50
How often the "new feature" will be used?	Daily
Customer value	4
Volume of programs to be changed	New System
Effect on system	High
STRATEGIC	
Is it strategic?	No alignment

Table 5-28: Project priority classification for the integrated business planning system frontend (continued)

CROSS-FUNCTIONAL Does it stretch over departments and/or systems? (Cross-functional)	Not Cross-functional
COMPLEXITY What is the complexity of the request?	High
EXISTING SYSTEM CHANGE What level of the existing system will require change?	New System
RESOURCE REQUIREMENTS What resources are required?	Labour > 160 hours
DEVELOPMENT TIME Development Time	> 4 Weeks
RISK Risk	Low
IT/BUSINESS REQUIREMENT IT/Business	Business
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**168** **Medium Priority**

Implemented and in use

The project was implemented and is currently in use, despite all the changes and modifications that were required.

Project methodology followed

- No structured methodology followed for project execution.
- No formal software development methodology was followed.
- No formal requirements were specified.
- Users were involved from the first user acceptance testing phase.
- Users were involved periodically from the first development requests.

Problems/Challenges Identified

The following problems were experienced through the project life cycle:

- Late project delivery – Project was completed five months behind the planned schedule.
- Sixty-seven modifications were required during user acceptance testing that were not in scope, causing project delays.
- Delays due to errors in development were experienced with a total of 193 rework calls.

- No PMM was followed.
- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic during user acceptance testing.
- Customer experience was poor, based on feedback from the customer.

Table 5-29 summarises the outcome of the evaluation criteria as it was applied and discussed in this section.

Table 5-29: Summary of the Frontend – Integrated business planning

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Frontend – Integrated Business Planning	Medium project	Medium	N/A	67	Not complete	N/A

5.3.5 Mobile device ordering system (small project)

This project involved the development of an application to automate the mobile device ordering process.

Project Scope

The scope of the project was to develop a system to track orders, upgrades and repairs of all mobile devices.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-30.

Table 5-30: Customer satisfaction feedback for the mobile device ordering system

Customer	Score (1 = Poor, 10 = Excellent)
Group Manager: Information Services	5
Manager iSeries and Voice	5
Administrative Controller: CIS	6

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-31 presents the stakeholders who were involved in the project

and Table 5-32 summarises the agreed project milestones by comparing the baseline timelines with the timelines that were actually achieved.

Table 5-31: Project stakeholders for the mobile device ordering system

Stakeholder Position	Interest
Business Representatives	
Group Manager: Information Services	Customer
Manager iSeries and Voice	Customer
Administrative Controller: CIS	Customer
Technical Team	
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
System Reporting	Developer

Table 5-32: Project delivery against agreed milestones for the mobile device ordering system

Project Phases	Baseline Milestones	Actual Delivery
Project Proposal	1 August 2017	21 August 2017
Design	25 August 2017	25 August 2017
Development	15 September 2017	29 September 2017
Technical testing 1	2 October 2017	11 October 2017
User Acceptance Testing 1	20 October 2017	17 November 2017
Modifications		1 December 2017
Technical testing 2		11 December 2017
User Acceptance Testing 2		23 February 2018
Implementation	6 November 2017	2 March 2018
Go-Live	6 November 2017	5 March 2018

Rework post implementation

Table 5-33 demonstrates all the changes and fixes that were required after the project was deployed and implemented.

Table 5-33: Rework post-project implementation for the mobile device ordering system

Call Nr	Description	Reported	Completed	Duration (Working Days)
00011654	Mobility Management - Device Receipt Notification	12 October 2017	15 October 2017	2
00012746	Mobility Device eForm - all changes are not saved	17 October 2017	7 November 2017	17
00013303	Mobility Management - User not able to create contract	23 October 2017	24 October 2017	2
00013562	Cannot retrieve submitted order	29 October 2017	1 November 2017	3
00014241	Escalation error in workflow	20 November 2017	30 November 2017	9
Total Count = 5				**33**

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-34.

Table 5-34: Modifications during the project due to scope changes for the mobile device ordering system

Call Nr	Description
00009282	Include process for purchase of new phone on Mobility application
00011471	Mobility Management - Add Service Provider to system
00012076	Mobility Management Reports.
00010544	Create the process to upload the fleet data to the insurance application, scheduled on the 23rd and 24th of the month
00011971	Mobile Device Asset Management Monthly Report
00013083	K2 - Mobility Management System - Phase 2 - Stolen Device \ Device Repair
00012122	Mobile Device Asset Management - Management Reports
Total Count = 7	

Project priority

The project was classified as a low priority project based on the priority calculation explained in section 5.1.2.1. The priority classification is presented in Table 5-35.

Table 5-35: Project priority classification for the mobile device ordering system

IMPACT	
Pain of User	2
% of Customers Impacted	51 – 80
How often the "new feature" will be used?	Daily
Customer value	4
Volume of programs to be changed	New System
Effect on system	Low
STRATEGIC	
Is it a Strategic?	No alignment
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Over IT Applications
COMPLEXITY	
What is the complexity of the request?	Low
EXISTING SYSTEM CHANGE	
What Level of the existing system will require change?	New System
RESOURCE REQUIREMENTS	
What resources are required?	Labour 50-100 hours
DEVELOPMENT TIME	
Development Time	> 4 Weeks
RISK	
Risk	Low
IT/BUSINESS REQUIREMENT	
IT/Business	IT
EXCO INFLUENCE	
Exco Influence	No
FINANCIAL	
Capex Required	No
PRIORITY SCORE	**101**
PRIORITY SCORE RESULT	**Low Priority**

Implemented and in use

This project was implemented successfully and is currently in use by the customer.

Project methodology followed

- No structured methodology was followed during the project execution.
- No formal software development methodology was followed during the project execution.
- Requirements were gathered, and development was purely based on the requirements document without user involvement until user acceptance testing.
- Users were only involved from the first user acceptance testing phase.
- No continuous customer involvement was followed.

Problems/Challenges Identified

The following problems were experienced through the project life cycle:

- Late project delivery – Project was completed 3.9 months behind schedule.
- Five modifications were required during user acceptance testing that were not in the project scope.
- Delays due to errors in development were experienced with a total of seven rework calls.
- No PMM was followed.
- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic during user acceptance testing.
- Customer experience was poor, based on feedback from the customer.

Table 5-36 summarises the outcome of the evaluation criteria as it was applied and discussed in this section.

Table 5-36: Summary of the mobile device ordering system

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Mobility Management	Small project	Low	5.5	7	3.9 months late	5

5.3.6 Product configuration management (small project)

This project involved the development of a workflow system to enable managers to initiate a request for a new product that needs to be launched. Once research has been done by the research and development department, the brand manager initiates a request for a new product to be created. Changes to existing products are also managed on this system.

Project Scope

The scope of the project was to develop a bill of materials (BOM) workflow system to manage and dictate the product configuration management (PCM) process.

Customer satisfaction was measured by means of a questionnaire in which the project team was asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 5-37.

Table 5-37: Customer satisfaction feedback for the product configuration management system

Customer	Score (1 = Poor, 10 = Excellent)
Group Manager: Production	5
Group Manager PIT	6
Quality Service Manager	6
Formulation Technologist	5
Packaging Technologist	6
Brand Manager Marketing	6
Supply Chain Planners	4

Stakeholders

The stakeholder size has an impact on the size and priority of the project, including the effect it has on the organisation. Table 5-38 presents the stakeholders who were involved in the project and Table 5-39 summarises the agreed project milestones by comparing the baseline timelines with the timelines that were actually achieved.

Table 5-38: Project stakeholders for the product configuration management system

Stakeholder Position	Interest
Business Representatives	
Group Manager: Production	Customer
Group Manager PIT	Customer
Quality Service Manager	Customer
Formulation Technologist	Customer
Packaging Technologist	Customer
Brand Manager Marketing	Customer
Supply Chain Planners	Customer
Technical Team	
Group Manager: Information Services	System Delivery
System Manager	System Delivery
System Analyst Workflow	Developer
System Analyst Applications	Developer
Applications Architect	Applications Architect
Integration Architect	Integration Developer

Table 5-39: Project delivery against agreed milestones for the product configuration management system

Project Phases	Baseline Milestones	Actual Delivery
Project Proposal	13 March 2017	13 March 2017
Design	27 April 2017	19 May 2017
Development	21 September 2017	27 December 2017
Technical testing 1	17 January 2018	12 January 2018
User Acceptance Testing 1	21 February 2018	6 March 2018
Modifications		29 March 2018
Technical testing 2		13 April 2018
User Acceptance Testing 2		11 May 2018
Modifications		18 May 2018
Technical testing 3		22 May 2018
User Acceptance Testing 3		1 June 2018
Implementation	9 March 2018	8 June 2018
Go-Live	9 March 2018	8 June 2018

Rework post implementation

Table 5-40 demonstrates all the changes and fixes that were required after the project was deployed and implemented.

Table 5-40: Rework post-project implementation for the product configuration management system

Ref No	Brief Description	Reported	Completed	Duration (Working Days)
00009098	Error with BOM Workflow	6 March 2018	9 March 2018	4
00009851	Fix display of completed multi-change requests on BOM Workflow	15 March 2018	6 May 2018	37
00010833	Fix errors on PCM application	4 April 2018	18 May 2018	33
00010866	Change the user's ID on BOM Workflow forms	17 April 2018	18 April 2018	2
00011215	PCM issues on live	24 April 2018	28 April 2018	4
Total Count = 5				**80**

Project modifications

During and after the project, various scope changes were required that resulted in many development modifications which are summarised in Table 5-41.

Table 5-41: Modifications during the project due to scope changes for the product configuration management system

Call Nr	Description
00008397	Add notifier to BOM workflow
00008523	Procurement - User requires reports on BOM, Vendor and Budget information
00008999	Multi-change on BOM Workflow
00011780	BOM process enhancement
00012126	PCM - Single BOM change
00012163	BOM K2 Integration
00012259	Review the scrap factors
00013026	PCM - Add Cancel option on workflow to allow users to cancel BOM workflows themselves
Total Count = 8	

Project priority

The project was classified as a low priority project based on the priority calculation explained in section 5.1.2.1. The priority classification is presented in Table 5-42.

Table 5-42: Project priority classification for the product configuration management system

IMPACT	
Pain of User	2
% of Customers Impacted	6-20
How often the "new feature" will be used?	Daily
Customer value	2
Volume of programs to be changed	New System
Effect on system	New System
STRATEGIC	
Is it a Strategic?	No alignment
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Not Cross-functional
COMPLEXITY	
What is the complexity of the request?	R&D
EXISTING SYSTEM CHANGE	
What level of the existing system will require change?	New System

Table 5-42: Project priority classification for the product configuration management system (continued)

RESOURCE REQUIREMENTS What resources are required?	Labour 101-160 hours
DEVELOPMENT TIME Development Time	> 4 Weeks
RISK Risk	Low
IT/BUSINESS REQUIREMENT IT/Business	IT
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**95** **Low Priority**

Implemented and in use

This project was implemented and is currently in use by the customer.

Project methodology followed

- No structured methodology was followed during the project execution.
- No formal software development methodology was followed during the project execution.
- Requirements were gathered, and development was purely based on the requirements document without user involvement until user acceptance testing.
- Users were only involved from the first user acceptance testing phase.
- No continuous customer involvement was followed.

Problems/Challenges Identified

The following problems were experienced through the project life cycle:

- Late project delivery – Project was completed three months behind schedule.
- Eight modifications were required during user acceptance testing that were not in the project scope.
- Delays due to errors in development were experienced with a total of five rework calls.
- No PMM was followed.
- No ASDM was followed.
- Customers were not continuously involved throughout the project.
- Availability of customers was problematic during user acceptance testing.
- Customer experience was poor, based on feedback from the customer.

Table 5-43 summarises the outcome of the evaluation criteria as applied and discussed in this section.

Table 5-43: Summary of the product configuration management

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Product Configuration Management	Small project	Low	5.4	8	3 months late	5

5.3.7 Summary

Table 5-44 summarises the results from the previously executed projects and indicates per priority what the project results were according to on-time delivery and quality of the project related to the extent of rework that was required.

All projects were delivered behind schedule and required a specified amount of rework to complete the project. The project scope of all projects also changed throughout project execution and the overall customer satisfaction of these projects was rated between 5 and 6 out of a total score of 10. This indicated that the customer satisfaction in terms of project execution and delivery was not good.

Table 5-44: Summary of previously executed projects

Project	Project classification	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Capex Application System	Big project	6.1	33	2.5 months late	17
Fleet Management System	Big project	5.5	21	6.7 months late	11
Dealsheet Management System (DMS)	Medium project	5.3	23	15.1 months late	16
Frontend – Integrated Business Planning	Medium project	N/A	67	Not complete	N/A
Mobility Management	Small project	5.5	7	3.9 months late	5
Product Configuration Management	Small project	5.4	8	3 months late	5

5.4 Project selected for this study

The organisation has a backlog of business approved projects which was used in the selection process of identifying projects for this study. Three projects based on the evaluation criteria discussed in sections 5.2 and 5.3 were selected from projects which were approved by the organisation for implementation. Projects from each project classification were selected consisting of a small, medium and big project. The projects selected are as follows:

- Quality concessions
- Overseas travel control system
- Cashbook

The project classifications and prioritisation calculations are discussed in Chapters 6, 7 and 8 which are explained in the context of the action research cycle.

Table 5-45 demonstrates the priority score for the selected projects compared to the project priority scores and classifications of the historical projects explained in section 5.3.

Table 5-45: Project priority scores and classification

Project Description	**Project classification**	**Project priority**	**Project priority score**
Selected projects for this study			
Quality concessions	Small	Low	102
Overseas travel control system	Medium	Medium	164
Cashbook	Big	High	246
Historical projects			
Mobile device ordering system	Small	Low	101
Product configuration management	Small	Low	95
Dealsheet management system	Medium	Medium	175
Frontend – Integrated business	Medium	Medium	168
Capital expenditure application	Big	High	237
Fleet management system	Big	High	230

In the next chapter, the first project that followed the action research cycle to investigate the impact of the ASDM and PMM integration will be discussed.

CHAPTER 6: ACTION RESEARCH CYCLE 1 – SMALL PROJECT

In Chapter 5, a number of projects were selected and evaluated against predefined success criteria. Software projects previously executed where no formal methodology was followed were discussed, including the evaluation criteria related to project success.

In this chapter, a brief overview of the first project is given and how the action research cycle for each phase is applied to the first selected project categorised as a small project is explained. In sections 6.2, 6.3 and 6.4, how the action research cycle of diagnosing, action planning, action taking, evaluating and specifying learning are applied to the project, based on the criteria combinations the team selected from the ASDM and PMM integration matrix discussed in Chapter 3 is explained.

6.1 Quality concessions (small project)

The organisation supplies consumer products to the market and occasionally requires approval to release products that are close to their expiry dates, or which quality does not conform fully to the required quality standards. These products are then released and sold at lower prices. During this process, no traceability existed on requested concessions which affected decision making and response times to requests, including market supply. There was therefore a need for a workflow system to assist with the paperwork and decisions made in rejecting or accepting concessions. Concessions are a critical part of the quality control function that needs to release products for market supply.

The purpose of the project was to design a workflow system to control the quality concession process within the organisation, from the business request up to approval by the quality manager.

6.2 Diagnose

The selection and analysis of historical projects discussed in section 5.3 form part of the diagnose phase for this project and explains the primary problems experienced during the execution of these projects.

Based on the size and classification of the quality concessions project, it can be compared to the small size historical projects displayed in Table 6-1.

Table 6-1: Summary of the small historical projects

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Mobility Management	Small project	Low	5.5	7	3.9 months late	5
Product Configuration Management	Small project	Low	5.4	8	3 months late	5

These historical projects were delivered behind schedule and required a specified amount of rework to complete the project. The project scope of these projects also changed throughout project execution and the overall customer satisfaction was rated between 5 and 6 out of a total score of 10 which indicated that customer satisfaction in terms of project execution and delivery was not satisfactory.

6.3 Action planning

The project was measured against the evaluation criteria described in section 5.2. In the following section, the evaluation results used in planning the action that was taken to improve software development projects more successfully are described.

Project priority

Table 6-2 illustrates the project priority which was classified as a low priority.

Table 6-2: Project priority classification for quality concessions

IMPACT	
Pain of User	3
% of Customers Impacted	6-20
How often the "new feature" will be used?	*Ad hoc*
Customer value	2
Volume of programs to be changed	New System
Effect on system	New System
STRATEGIC	
Is it strategic?	Aligned to business
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Over Business Units
COMPLEXITY	
What is the complexity of the request?	Low
EXISTING SYSTEM CHANGE	
What level of the existing system will require change?	New System

Table 6-2: Project priority classification for quality concessions (continued)

RESOURCE REQUIREMENTS What resources are required?	Labour 50-100 hours
DEVELOPMENT TIME Development Time	2-3 weeks
RISK Risk	Low
IT/BUSINESS REQUIREMENT IT/Business	Business
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**102** **Low Priority**

Stake holders:

The project was categorised as a small project based on its priority calculation which required a small development team. The project team was therefore lean and structured as per Table 6-3.

Table 6-3: Project stakeholders for the quality concession system

Stakeholder Position	Interest
Business Representatives	
Group Manager: Quality	Customer
Quality Manager: Laboratory Services	Customer
Technical Team	
System Manager	System Delivery
System Analyst	Developer

In the following section, how the selected ASDMs were integrated with PMBOK for the quality concessions system is described.

According to the nature and culture of the organisation, ASDM and PMM integration were based on the integration matrix as per Table 6-4. ASDMs as per PMBOK's knowledge areas were selected based on the applicability within the specific organisation. The applicability is indicated in colour codes where green indicates a good fit for the project, yellow a partial fit and red no fit. Green and yellow selections were applied in the project execution and the red selections were discarded. In the following section, the ASDMs and PMM integration for the quality concession

project and is illustrated in Table 6-4 is described. This integration was compiled based on the organisational characteristics which determined the methodology components that were applied in the project execution process.

Table 6-4: Integration of PMBOK and ASDM for the Quality Concessions

PMBOK	Software development methodologies							
	SCRUM	Organisation Characteristics	XP	Organisation Characteristics	DSDM	Organisation Characteristics	FDD	Organisation Characteristics
Project Integration Management								
Develop project charter	Software requirements are formulated and prioritised by the product owner as stories	Frequently changing requirements	User stories or requirements are created by customers and the development team	Frequently changing requirements	Feasibility and business study as evaluation and planning mechanisms	Not required for small projects as developing is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Develop project management plan	Sprint planning events	Frequently changing requirements	Stories are converted into iterations	Frequently changing requirements	Prioritisation is essential	Frequently changing requirements	Building a feature list	Object-oriented modelling not commonly followed in the organisation
Direct and manage project work	Strong change management procedure with product and sprint backlog	Frequently changing requirements	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Frequently changing requirements	Baseline of requirements and functionality is at a high level	Requirements needs to be well defined and in detail. Planning is also done in detail and not at a high level	Planning and designing by feature	Object-oriented modelling not commonly followed in the organisation
Monitor and control project work	Retrospective and Next Sprint Planning	The organisation prefers to complete iterations first and changes identified during an iteration are only incorporated in the next iteration	Coding takes priority over all tasks	The development environment classifies requirements and testing as important as coding	Relies heavily on techniques to develop an application	No techniques are currently being used. Prioritisation and timeboxing will be applied for this project.	Prioritise based on features	Object-oriented modelling not commonly followed in the organisation
Integrated change control			Continuous design-coding-testing-listening cycles	The development environment classifies requirements and testing as important as coding	Uses prototypes	Not practical as a prototype will require most of the development. Also not required for medium projects		
Close project					Good for projects with tight time constraints	Project needs to be delivered within a specified timeline		
					Testing integrated throughout the lifecycle	Existing development does not follow iterations. Develop solution first followed by testing		
Project Scope Management								
Scope planning	Product Backlog Creation	Frequently changing requirements. Scope planning required	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Business studies not required for small projects as developing is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Collect requirements	Sprint Planning and Sprint Backlog Creation	Frequently changing requirements	Release planning and prioritisation by customer	Frequently changing requirements	Focusing on the customer needs	Frequently changing requirements	Develop feature list	Object-oriented modelling not commonly followed in the organisation
Scope definition	User stories are defined and prioritised	Frequently changing requirements	Continuous feedback from customer	Frequently changing requirements	Active user involvement	Customer is closely involved during the development process		
Create work breakdown structure	Review continuously to improve development process	Frequently changing requirements	Small releases	Frequently changing requirements	Frequent releases	Frequent releases make continuous testing possible		
Scope validation								
Scope control								

Table 6-4: Integration of PMBOK and ASDM for the Quality Concessions (continued)

Project Time Management								
Plan schedule	Plan sprint durations	Time estimation required	Iterations time planning	Planning is done for estimation purposes	Time and resource are not adjusted	Time can be adjusted based on requirements changes. Resources are unlikely to change	Plan by feature	Object-oriented modelling not commonly followed in the organisation
Define activities	Sprint lasts ± 2 weeks	Can be flexible, but not longer than two weeks for a medium project	Release planning	Planning is done for estimation purposes	Deliver the projects as early as possible without affecting the quality	Time is not the primary focus and is flexible. Functionality is a priority.	Plan order of features based on dependencies	Object-oriented modelling not commonly followed in the organisation
Sequence activities	Task board for tracking progress	Customer requires fast and regular feedback			Good control over quality of product, cost and time	Big focus on quality in terms of functionality. No cost is involved and time is not the primary focus	Determine development sequence	Development is prioritised
Estimate activity resources	Daily stand-up meetings to discuss progress	Customer requires fast and regular feedback, depending on customer availability			Frequent delivery of products	Frequently changing requirements		
Estimate activity durations Develop schedule Control schedule								
Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	No cost implication	Programming team prepares the costs of carrying out iterations	No cost implication	Deals effectively with project cost	No cost implication	No primary focus	No cost implication
Cost budgeting			Double development cost	No cost implication	Gives priority to time, quality, and cost	No cost implication		
Cost control			Reduce cost via small iterations	No cost implication	Costly to implement	No cost implication		
Project Quality Management								
Quality planning	Quality goals sustained during sprints	Functionality and quality are important factors in terms of success criteria	Pair programming for higher quality code	Development teams are lean	Good control over quality of product, risk, cost and time	Not a big focus for medium projects. No costs are involved and mostly low risk projects	Continuous testing of the code	Frequently changing requirements
Perform quality assurance	Retrospective and Next Sprint Planning to improve development process	The organisation prefers to complete iterations first and changes identified during an iteration are only incorporated in the next iteration	Extreme testing in development phase	Development teams are lean. Developers perform their own technical testing.	Delivering the projects without affecting the quality of the project	Quality cannot be compromised	Coding standards	Coding standards are followed to ensure quality and support
Perform quality control	Regular scrum meetings	Functionality and quality are important factors in terms of success criteria	Continuous customer involvement and testing	Frequently changing requirements	Focus on frequent releases rather than quality	Quality more important than time	Measuring audits and metrics in the code	Not a big focus for medium projects
	Customer inputs for further improvements during iterations	Frequently changing requirements	Simple design	Not a primary focus as functionality and quality are more important				
			Coding standards	Coding standards are followed to ensure quality and support				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross-functional	Developers are highly skilled and can work independently	Pair programming	Development teams are lean	Teams empowered to make decisions	Developers are highly skilled and can work independently	Client and development team develop overall model	Object-oriented modelling not commonly followed

Table 6-4: Integration of PMBOK and ASDM for the Quality Concessions (continued)

Acquire project team	Daily scrum meetings	Frequently changing requirements	Small teams	Development teams are lean	Facilitated Workshops	Workshops are facilitated to ensure all customer requirements are captured	Highly skilful teams	Developers are highly skilled and can work independently
Develop project team	Small teams	Development teams are lean	Customer part of development team	Frequently changing requirements			Feature teams	Develop according to sprint backlog tasks and not according to features
Manage project team			Team-oriented methodology	Regular develop-test cycles require a team focussed methodology				
			Avoid working overtime	Not a focus point for the organisation				
Project Communications Management								
Communications planning	Permanent communication and close cooperation between the stakeholders at each step	Frequently changing requirements	Continual communication with the customer and amongst the team	Frequently changing requirements	Facilitated workshops	Workshops are facilitated to ensure all customer requirements are captured	Continuous review meetings	Frequently changing requirements
Information distribution	Daily scrum meetings	Frequently changing requirements	Collaborative workspaces	Teams work closely together with customer involvement	Environment ideal for the formation of ideas	Not a primary focus area	Feature teams deliver features	Object-oriented modelling not commonly followed
Performance reporting	Group is self-organising and collaboratively managed	Developers are highly skilled and can work independently	Co-location of development and business space	Teams work closely together with customer involvement	Accurate and quick decision making	Frequently changing requirements		
Manage stakeholders			Paired development	Development teams are lean				
			Frequently changing pair partners	Development teams are lean				
			Short stand-up meetings	Frequently changing requirements				
			Unit tests and verbal communication	Frequently changing requirements				
Project Risk Management								
Risk management planning	Iterative and incremental approach to control risk	Frequently changing requirements	Identify risks early in the project	Small projects are mostly very low risk to no risk projects	Changes are reversible during development	Frequently changing requirements	Guidance of skilled, experienced developers	Only skilled developers
Risk identification	Sprints limit risk to one calendar month of cost	No cost implication	Frequent releases	Frequently changing requirements	Feasibility study identifies risks involved	Not required for medium projects as development is an internal function of the organisation	Feature planning take risks into account	Object-oriented modelling not commonly followed
Qualitative risk analysis	Project risks seen more quickly	Medium projects are mostly low risk projects	Unit testing	Frequently changing requirements	Guidelines for risk management	No specific guidelines for risk management		
Quantitative risk analysis	Decisions to control risk are made based on the perceived state of the artefacts	Risk decisions are made based on project status and test results	Customer part of development team	Frequently changing requirements	Reduce risk through incrementally delivering the solution	Frequently changing requirements		
Risk response planning	Meetings to review risks	Regular stand-up meetings cover risks as well	Collective ownership reduces risks of programmer dependency.	Development teams are lean				
Risk monitoring and control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting Request Seller Responses Select Sellers Contract Administration Contract Closure								

Project integration management - Organisational characteristics

The organisation required the project to be flexible as requirements may change during project execution. To achieve this, fast feedback mechanisms and the ability to adapt quickly to changes were required. The organisation preferred to complete iterations first and changes, identified during an iteration, were only incorporated in the next iteration. The development environment classified requirements and testing as important as coding. A feasibility and business study as an evaluation and planning mechanism were not performed as the project was classified as small and development is an internal function of the organisation. Because the project was classified as a small project, various techniques and prototyping as defined by DSDM were not required. The organisation expected projects to be delivered according to agreed timelines. Developing an overall model as per FDD was not done as the organisation did not follow object-oriented modelling methodologies. Feature lists were not compiled as the project was managed by means of product and sprint backlogs.

Project scope management - Organisational characteristics

Frequent changing requirements within the organisation require the scope of the project to be compiled and managed by means of a product and sprint backlog. The customer preferred to be closely involved in the project and frequent releases ensured that the changing requirements were met on a continuous basis.

Project time management - Organisational characteristics

The organisation required a project plan to indicate when the project deliverable would be expected. Because of continuous changing requirements, sprints did not have to be completed within two weeks and could run for longer periods, although this never happened due to the size of the project. Regular feedback meetings fitted the environment of the organisation, as the customer preferred to be involved throughout the project process. Planning needed to be conducted in conjunction with the customer to estimate when the project had to be completed.

Project cost management - Organisational characteristics

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management - Organisational characteristics

The organisation saw quality and functionality as higher priorities compared to time and prerequisites for the success of the project. The organisation preferred to complete iterations first and changes identified during an iteration were only incorporated in the next iteration. The development team was lean; that made pair programming as per XP not ideal. To improve the

quality of the products delivered to the organisation and to minimise the dependency on key developers, coding standards were followed as far as possible. Audits and metrics were not a major focus point for small projects.

Project human resource management - Organisational characteristics

The organisation had a highly skilled development team who worked independently. Daily meetings were preferred to ensure good communication during project execution. The development team was lean that made pair programming as per XP not ideal. The organisation's developers were reluctant to write tests first and code later as stipulated by XP. Overtime was not encouraged in the organisation and was not required as the delivery of the project within a specified timeframe took a lower priority than quality and functionality. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs were preferred to feature development.

Project communications management - Organisational characteristics

The organisation had a highly skilled development team who worked independently and closely together. Daily meetings were preferred to ensure good communication during project execution and to address the continuous changing requirements of the customer. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs were preferred to feature development.

Project risk management - Organisational characteristics

There were no costs involved, as development of small projects was executed internally by an in-house development team. Iterations therefore did not have to be limited to one calendar month to minimise the risk of cost overruns. Risk management in terms of quality and functionality were important to the organisation where decisions around risk were focussed to improve the overall status of the project. Daily meetings were preferred to manage the risk on continuous changing requirements and to deliver products according to expectation. Collective code ownership was not a priority due to the limited number of developers on the project. Feasibility and business studies as evaluation and a planning mechanism were not required, as this particular project was small, and development was an internal function of the organisation. Guidelines for risk management as per DSDM were not required for small projects within the organisation. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs were preferred above feature development.

Project Procurement Management - Organisational characteristics

No procurement management was required.

6.4 Action taking

In this section, the processes that were followed and the project team's feedback on the integration levels that were selected are described. The integration levels between PMBOK and the ASDMs for the quality concessions project which is summarised in Table 6-5 are also explained. The selected integration levels what were applied during the execution of the project are indicated in green. Interviews were conducted with the project team members that included feedback from the project manager, developers, analysts and customers involved in the project. The feedback obtained from the project team and related parties is incorporated in the following sections.

Table 6-5: Project team feedback

PMBOK	Software development methodologies							
	SCRUM	Project Team Feedback	XP	Project Team Feedback	DSDM	Project Team Feedback	FDD	Project Team Feedback
Project Integration Management								
Develop project Charter	Software requirements are formulated and prioritised by the product owner as stories	Formulated by project team and prioritised by development team	User stories or requirements are created by customers and the development team	Formulated by project team and prioritised by development team	Feasibility and business study as evaluation and planning mechanisms	Not applicable	Developing an overall model	Not applicable
Develop project management Plan	Sprint planning events	Works well	Stories are converted into iterations	Yes, different sprints	Prioritisation is essential	Works well	Building a feature list	Not applicable
Direct and Manage project Work	Strong change management procedure with product and sprint backlog	Works well	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Works well. Development team only	Baseline of requirements and functionality is at a high level	Not applicable	Planning and designing by feature	Not applicable
Monitor and Control project Work	Retrospective and Next Sprint Planning	Changes included in active sprint when discussed in stand-up. Not included in next sprint.	Coding takes priority over all tasks	Not applicable	Relies heavily on techniques to develop an application	Not applicable	Prioritise based on features	Not applicable
Integrated Change Control			Continuous design-coding-testing-listening cycles	Works well	Uses prototypes			
Close project					Good for projects with tight time constraints	Timelines are difficult to manage due to continuous changing requirements.		
					Testing integrated throughout the lifecycle			
Project Scope Management								
Scope Planning	Product Backlog Creation	Works well	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Not applicable	Developing an overall model	Not applicable
Collect Requirements	Sprint Planning and Sprint Backlog Creation	Works well	Release planning and prioritisation by customer	Release planning works well. Prioritisation done by development team	Focusing on the customer needs	Works well	Develop feature list	Not applicable

Table 6-5: Project team feedback (continued)

Scope Definition	User stories are defined and prioritised	Works well	Continuous feedback from customer	Works well	Active user involvement	Works well		
Create Work Breakdown Structure	Review continuously to improve development process	Works well. The result might be less rework	Small releases	Works well	Frequent releases	Works well		
Scope Validation Scope Control								
Project Time Management								
Plan Schedule	Plan sprint durations	Provides good estimate to communicate to business and manager. Move sprint timelines out due to other priorities and support	Iterations time planning	Provides good estimate to communicate to business and manager. Move sprint timelines out due to other priorities and support	Time and resource are not adjusted	Not applicable	Plan by feature	Not applicable
Define Activities	Sprint lasts +- 2 weeks	Works well	Release planning	Works well	Deliver the projects as early as possible without affecting the quality	Not applicable	Plan order of features based on dependencies	Not applicable
Sequence Activities	Task board for tracking progress	Works well			Good control over quality of product, cost and time		Determine development sequence	Works well
Estimate Activity Resources	Daily stand-up meetings to discuss progress	Not daily, but regularly			Frequent delivery of products	Works well. Duration similar than before, but quality much better with minimal rework		
Estimate Activity Durations Develop Schedule Control Schedule								
Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	Not applicable	Programming team prepares the costs of carrying out iterations	Not applicable	Deals effectively with project cost	Not applicable	No primary focus	Not applicable
Cost Budgeting			Double development cost	Not applicable	Gives priority to time, quality, and cost	Not applicable		
Cost Control			Reduce cost via small iterations	Not applicable	Costly to implement	Not applicable		
Project Quality Management								
Quality Planning	Quality goals sustained during sprints	Works well. Show to business and therefore technical testing more thorough	Pair programming for higher quality code	Not applicable	Good control over quality of product, risk, cost and time	Not applicable	Continuous testing of the code	Works well
Perform Quality Assurance	Retrospective and Next Sprint Planning to improve development process	Works well. Lessons learned in sprints were applied in following sprints	Extreme testing in development phase	Works well	Delivering the projects without affecting the quality of the project	Works well	Coding standards	Works well
Perform Quality Control	Regular scrum meetings	Regular stand-up meetings worked well	Continuous customer involvement and testing	Works well	Focus on frequent releases rather than quality	Not applicable	Measuring audits and metrics in the code	Not applicable
	Customer inputs for further improvements during iterations	Works well	Simple design	Not applicable				
			Coding standards	Works well				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross functional	Works well	Pair programming	Not applicable	Teams empowered to make decisions	Sprint planning and coding decisions done by developer	Client and development team develop overall model	Not applicable
Acquire project Team	Daily scrum meetings	Works well	Small teams	Only one developer	Facilitated workshops	Progress/stand-up meetings	Highly skilful teams	Yes

Table 6-5: Project team feedback (continued)

Develop project Team	Small teams	Works well if dedicated/not distracted	Customer part of development team	Works well			Feature teams	Not applicable
Manage project Team			Team-oriented methodology	One developer				
			Avoid working overtime	Not applicable				
Project Communications Management								
Communications Planning	Permanent communication and close cooperation between the stakeholders at each step	Works well	Continual communication with the customer and amongst the team	Works well	Facilitated Workshops	Works well	Continuous review meetings	Works well
Information Distribution	Daily scrum meetings	Works well	Collaborative workspaces	Developers and Business work separately, except for Stand-up meetings	Environment ideal for the formation of ideas		Feature teams deliver features	Not applicable
Performance Reporting	Group is self-organising and collaboratively managed	Works well	Co-location of development and business space	Same location, but different office blocks between development team and customer	Accurate and quick decision making	Works well		
Manage Stakeholders			Paired development	Not applicable				
			Frequently changing pair partners	Not applicable				
			Short stand-up meetings	Works well				
			Unit tests and verbal communication	Works well				
Project Risk Management								
Risk Management Planning	Iterative and incremental approach to control risk	Works well	Identify risks early in the project	Works well	Changes are reversible during development	Works well	Guidance of skilled, experienced developers	Not Applicable
Risk Identification	Sprints limit risk to one calendar month of cost	Not Applicable	Frequent releases	Works well	Feasibility study identify risks involved	Not Applicable	Feature planning take risks into account	Not Applicable
Qualitative Risk Analysis	Project risks seen more quickly	Works well	Unit testing	Works well	Guidelines for risk management	Not Applicable		
Quantitative Risk Analysis	Decisions to control risk are made based on the perceived state of the artefacts	Not Applicable	Customer part of development team	Works well	Reduce risk through incrementally delivering the solution	Works well		
Risk Response Planning	Meetings to review risks	Works well	Collective ownership reduces risks of programmer dependency.	Not Applicable				
Risk Monitoring and Control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting								
Request Seller Responses								
Select Sellers								
Contract Administration								
Contract Closure								

Project integration management

The project team selected various combinations of ASDMs listed in Table 6-5. The ASDMs selected are indicated at the end of each point in brackets.

Processes followed

The following ASDM tasks in terms of project integration were applied:

- Software requirements were formulated and prioritised by the customer and project team as stories (SCRUM, XP). According to Table 6-5, scrum and extreme programming formulate software requirements in the form of user stories. This is done by both the project team and the customer who prioritise the requirements in terms of a development sequence. Table 6-5 displays the selected ASDM components in green.
- Development was done in iterations (SCRUM, XP, DSDM, FDD).
- Development tasks were planned by means of a sprint backlog (SCRUM, XP).
- Retrospective and next sprint planning were incorporated, and no changes were made during sprints (SCRUM).
- Development followed a continuous cycle of design, coding, testing and listening (SCRUM, XP).

Project team feedback and improvements (Evaluating)

The project requirements were formulated by the project team which included the customer, but prioritisation was done by the development team only. The sprint planning process by means of the product and sprint backlog worked well and enabled the project team to plan the project thoroughly. This also helped with the change management processes that kept the project team and customer up to date with sprint releases. Changes required during a sprint were not allowed and were only incorporated in the next sprint. Timelines were difficult to manage due to continuous changing requirements, but the final project was delivered according to expectation and with no changes that had to be incorporated at the end of the project.

Project scope management

Processes followed

The following processes were followed and worked well to ensure the delivery of the project:

- A product backlog was created that was used to define and prioritise the customer requirements in the form of user stories. These were defined by the development team and the customer (SCRUM, XP, DSDM).
- The product backlog defined the scope of the project (SCRUM, DSDM).
- The requirements were broken down into separate development tasks in the form of a sprint backlog, listing the effort and time estimates for each task (SCRUM).
- Sprints were reviewed on a continuous basis that improved the development process (SCRUM, XP).
- The customer received continuous feedback from the development team throughout the development process (SCRUM, XP).

- Development was released frequently and in small iterations as defined by the various sprints (SCRUM, XP, DSDM).
- Developing an overall model dictated that the project scope as per FDD would not be done as the organisation did not follow object-oriented modelling methodologies.

Project team feedback and improvements

The product and sprint backlog creation worked well in defining and managing the scope of the project and continuously improved the development process. The customer was primarily involved which helped to deliver the project according to expectation. The backlogs helped the team to plan releases more accurately, but the development tasks were prioritised by the project team without the customer's involvement. The continuous customer feedback and small releases kept the focus on the project and ensured that the project progressed according to expectation.

Project time management

Processes followed

Project time management was conducted as follows:

- The project was planned and prioritised by means of sprints that indicated the development tasks and sequence (SCRUM, FDD).
- The various tasks as per the sprint backlog were used to track progress (SCRUM).
- Daily stand-up meetings were conducted to address concerns and to ensure the project stayed on track (SCRUM).
- Priority was given to time and quality with close customer involvement (DSDM).
- Adjusting time and resources as per DSDM was not recommended as it was required due to continuous changing requirements.
- The organisation saw quality as a higher priory than time.

Project team feedback and improvements

The sprint planning process improved project delivery estimations for the various iterations and acted as a good communication mechanism with regard to the customer. The time it took to deliver the project was similar to that of historical projects, but the quality improved tremendously, as no rework was required after the project was delivered. The developing sequence as specified by the sprint backlog enabled the development team to stay focussed and to prioritise the development tasks effectively. Although the baseline timelines were missed, the final product was of such good quality that the customer accepted the revised deadlines.

Project cost management

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management

Processes followed

Quality was ensured by means of the following processes:

- The quality goals were defined during the product backlog creation process in conjunction with the customer (SCRUM).
- Regular meetings between the development team and the customer ensured that the development team delivered quality code for every iteration, thereby incorporating all customer inputs for further improvements (SCRUM).
- Extensive and continuous testing was performed during the development phase with continuous customer involvement (SCRUM, XP, FDD).
- Development was done by following predefined coding standards (XP, FDD).

Project team feedback and improvements

The quality goals were met and sustained by regular stand-up and feedback meetings with the customer which forced the technical team to do thorough technical testing more frequently. Retrospective and next sprint planning enabled the project team to apply lessons learned from previous sprints to next and future sprints. The team preferred to incorporate changes during a sprint instead of during the next sprint which was only applied and tested in the next project. Regular, constructive feedback from the customer helped the project team to deliver quality products with every iteration. Coding standards improved the quality of the delivered products as it made the code easier to review and ensured that the most efficient code was used.

Project human resource management

Processes followed

The project in terms of human resource management was structured as follows:

- Teams were self-organised, cross-functional and empowered to make decisions (SCRUM, DSDM).
- Daily stand-up meetings were conducted between the development team and the customer (SCRUM, XP).
- The development team was small and highly skilled with the customer part of the team (SCRUM, XP, FDD).

- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The development team, being self-organised and highly skilled, helped in the delivery of quality work according to priority and plan. The daily stand-up meetings helped to keep the team focussed and to deliver releases according to customer expectation. Because it was a small project, only one developer who was empowered to make decisions was assigned to the project. This worked well with the delivery of the project.

Project communications management

Processes followed

Project communications management was conducted as follows:

- There was close cooperation between all stakeholders with continuous communication throughout the execution of the project (SCRUM, XP).
- Daily stand-up meetings were held to monitor progress and to review the development process (SCRUM, XP, FDD).
- Team members were self-organised and collaboratively managed (SCRUM, XP).
- Teams were co-located performing unit tests with good communication (XP).
- Workshops were facilitated (DSDM).
- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The frequent stand-up meetings improved the communication between all parties on the project. The project team worked in a collaborative area that did not include the customer, as the customer could not be dedicated to the project on a full-time basis. The scrum meetings were efficient enough for communication between customer and project team. Unit testing and verbal communication formed part of the iterations and improved the development quality.

Project risk management - Organisational characteristics

Processes followed

Project risks were managed and mitigated as follows:

- An iterative and incremental approach with frequent code releases was followed to control and to identify risks early in the project (SCRUM, XP, DSDM).

- The customer formed part of the development team that helped to reduce risks (SCRUM, XP, DSDM, FDD).
- Collective code ownership was not a priority, as the development team was lean (XP).

Project team feedback and improvements

The product and sprint backlogs enabled the team to identify and manage risks related to the project schedule effectively. The methodology of developing in iterations enhanced this process further. The stand-up meetings also assisted with identifying and discussing risks in time thereby enabling the team to address them in advance.

Project procurement management - Organisational characteristics

Procurement management was not required.

6.5 Evaluating

The project execution process was compared and evaluated against the evaluation criteria described in section 5.2 and delivered the following results:

Customer satisfaction feedback

Customer satisfaction was measured by means of a questionnaire in which the project team members were asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 6-6.

Table 6-6: Customer Satisfaction Feedback for quality concessions

Customer	Score (1 = Poor, 10 = Excellent)
Group Manager: Quality	9
Quality Manager: Laboratory Services	10
System Manager	9
System Analyst	9

Project delivery against agreed milestones

The project was baselined at the beginning of the project and measured against the baseline after the project was delivered. Table 6-7 displays the product backlog that indicates when the various project phases were completed against the planned completion dates. Although the project was not completed on the baseline date, the project was completed with no rework. Previous projects required considerable rework that caused major project delays.

Table 6-7: Product backlog for quality concessions

Item	Baseline Milestones	Actual Delivery
Business requirements	3 September 2019	6 September 2019
Employee submits a concession request	9 September 2019	17 September 2019
Approval of concession request	16 September 2019	19 September 2019
Create concessions request workflow	17 September 2019	23 September 2019
Employee submits review of concession request	19 September 2019	23 September 2019
Reporting	20 September 2019	3 October 2019
User Acceptance Testing	20 September 2019	11 October 2019
Deployment	27 September 2019	14 October 2019

Table 6-8 indicates the different sprints and sprint tasks with the estimate and actual hours per project sprint task. The tasks in yellow were changes requested during the stand-up meetings which were incorporated during the project and eliminated the need for rework at the end of the project.

Table 6-8: Sprint backlog for quality concessions

Sprint	Product Backlog Item	Sprint Task	Est hours	Act hours	Diff	Planned Date	Date Ended
1	Employee submits a concession request	Create database tables	1.5	1.5	0	2019-09-09	2019-09-09
		Create K2 smart objects	1	0.5	0.5	2019-09-09	2019-09-09
		Create K2 Menu for application	1	1	0	2019-09-09	2019-09-09
		Create K2 master data views with required validations	4	5	-1	2019-09-10	2019-09-12
		Create K2 transaction views with required validations	4	6	-2	2019-09-11	2019-09-16
		Create K2 master data form(s) with required validations	3	2	1	2019-09-12	2019-09-16
		Create K2 transaction form(s) with required task buttons	4	4	0	2019-09-12	2019-09-17
		Development technical testing	2	1.5	0.5	2019-09-13	2019-09-19
		Create Master for approver list	1	1	0	Added	2019-09-20
		Create Unit of measure table	1	1	0	Added	2019-09-20
2	Approval of concession request	Create K2 transaction form(s) with required task buttons	3	3	0	2019-09-16	2019-09-19
		Create approval tracking	1	1	0	2019-09-16	2019-09-19
		Development technical testing	2	2	0	2019-09-16	2019-09-19

Table 6-8: Sprint backlog for quality concessions (continued)

3	Create concessions request workflow	Create workflow e-mail templates	1	1	0	2019-09-17	2019-09-19
		Create workflow events and actions	4	4	0	2019-09-18	2019-09-20
		Setup workflow reminders	1	1	0	2019-09-18	2019-09-23
		Development technical testing	2	1	1	2019-09-18	2019-09-23
4	Employee submits review of concession request	Create K2 transaction form(s) with required task buttons	2	3	-1	2019-09-19	2019-09-23
		Development technical testing	0.5	0.5	0	2019-09-19	2019-09-23
5	User Required Changes - this was requested once the above development was completed	Add a concession outcome type e.g. Batch or Specific Period	0.25	0.25	0	Added	2019-10-02
		Create Master for concession outcome type	0.5	0.5	0	Added	2019-10-02
		Add to 'Get Reason' view	0.5	0.5	0	Added	2019-10-02
		Add additional review step to inform PIT of concession (Product Development Review)	1	1	0	Added	2019-10-02
		Change List Editing to 'on click'	0.25	0.25	0	Added	2019-10-02
6	Reporting	Create user report in SSRS	2.5	3	-0.5	2019-09-20	2019-10-03
7	User Acceptance Testing	User Acceptance testing	8	8	0	2019-09-20	2019-10-11
8	Deployment	Package and Deploy	4	0	4	2019-09-20	2019-10-14

Rework post implementation

No rework was required or reported for a two-week period after the project was delivered and taken into production.

Modifications during the project due to scope changes

None to date

Implemented and in use

- The project was implemented successfully and is currently in use**.**

Project methodology followed

- The PMM and ASDM integration matrix as per Table 6-4 were applied to manage the project and to develop and implement the solution.
- Requirements were gathered, and development was based on the sprints and iterations as defined by the product and sprint backlogs of the project.
- The customer was continuously involved in all iterations and sprints throughout the project.

- The product and sprint backlogs made it easy to estimate timelines which could not be done previously.
- The compiling sprints enabled the team to plan development better and more constructively.

Comparison to historical projects

Based on the project classification, size and priority, the Quality concessions project can be compared to the historical projects which were categorised as "small" projects. Table 6.9 compares the project results of these projects.

Table 6.9: Project results comparison of small projects

Project Results	Quality concessions	Mobility management	Product configuration management
Project classification	Small	Small	Small
Priority	Low	Low	Low
Customer Satisfaction	9.25	5.5	5.4
Scope Creep (Modifications)	0	7	8
Timeline	0.56 months late	3.9 months late	3 months late
Rework (Calls)	0	5	5

The quality concessions project required no scope changes and no rework during the execution of the project. Although the project was not delivered within the projected timeline, the overrun was much shorter compared to that of the historical projects. Consequently, the project produced a better customer experience that resulted in a good customer satisfaction score.

6.6 Specifying learning

During the project process, various lessons learned were identified by the project team. The lessons were used to determine if the subsequent projects could be done differently to further improve the project process. The following limitations were identified during the project life cycle:

- The team found the process of not making changes during a sprint as a limitation.
- The timelines were impacted due to continuous changing requirements.
- Sprint timelines had to be moved due to the development team not being dedicated to the project only as they had to focus on other system support and priorities as well.
- Documentation was limited due to time constraints. It was also not updated during the project due to time constraints, although the requirements that changed continuously were logged and managed.

- Daily stand-up meetings could not always take place due to the unavailability of various parties.

In this chapter, the results of the quality concessions project that was executed according to the ASDM and PMM integration matrix explained in Chapter 3 are explained. Action research was applied by following the action research cycle of diagnosing, action planning, action taking, evaluating and specify learning. The final project results based on the evaluation criteria explained in Chapter 5 were better compared to the historical projects that were classified as small projects.

In the next chapter, the second project that followed the action research cycle to investigate the impact of the ASDM and PMM integration will be discussed.

CHAPTER 7: ACTION RESEARCH CYCLE 2 – MEDIUM PROJECT

In this chapter, a brief overview of the second project is given and how the action research cycle for each phase is applied to the second selected project categorised as a medium project is explained. In sections 7.2, 7.3 and 7.4, how the action research cycle of diagnosing, action planning, action taking, evaluating and specifying learning are applied to the project based on the criteria combinations the team selected from the ASDM and PMM integration matrix discussed in Chapter 3 is explained.

7.1 Overseas travel control system (medium project)

Currently, foreign travel costs are managed and controlled via a manual system running on Excel. It is mainly used to claim back foreign expenses incurred during a trip and contains a summary of all costs related to the foreign trip.

The customer and custodian for this system is the finance department which is a central function of the organisation. The finance department has the following characteristics that should be taken into consideration:

- Based on previous development projects, the finance team had a reputation of frequently changing requirements as the project progressed throughout the development life cycle.
- The customer preferred to be closely involved throughout the process, but was not dedicated permanently to the project. For this reason, daily stand-up meetings were not possible at all times.
- The functionality and quality of the system took a higher priority compared to delivery of the project.

7.2 Diagnose

The selection and analysis of historical projects discussed in section 5.3 form part of the diagnose phase for this project and explain the primary problems experienced during the execution of these projects.

Based on the size and classification of the overseas travel control system project, it can be compared to the medium size historical projects displayed in Table 7-1.

Table 7-1: Summary of the medium historical projects

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Dealsheet management system	Medium project	Medium	5.3	23	15.1 months late	16
Front end – Integrated business planning	Medium project	Medium	N/A	67	Not complete	N/A

The medium size historical projects were delivered behind schedule and required a specified amount of rework to complete the project. The project scope of these projects also changed throughout project execution and the overall customer satisfaction was rated below 6 out of a total score of 10 which indicated that customer satisfaction in terms of project execution and delivery was not satisfactory. The Frontend integrated business planning project was not completed during the time of the study which is why no scoring could be allocated to the customer satisfaction and amount of rework evaluation criteria.

7.3 Action planning

The project was measured against the evaluation criteria described in section 5.2. In the following section, the evaluation results used in planning the action that was taken to improve software development projects more successfully are described. The lessons learned as per section 6.6 were taken into consideration to improve the process and methodology in executing the overseas travel control system project. This involved the incorporation of changes during a sprint compared to only making changes at the beginning of a new sprint. In addition, the daily stand-up meetings were changed to weekly stand-up meetings and sometimes twice per week, depending on when the team had to discuss changes with the customer.

Project scope

There was a requirement for a workflow system to control the full foreign travel expense process, from request, to approval, to ordering, to the submission of the claim after the trip.

Project priority

Table 7-2 illustrates the project priority which was classified as a low priority.

Table 7-2: Project priority classification for the overseas travel control system

IMPACT	
Pain of User	3
% of Customers Impacted	6-20
How often the "new feature" will be used?	*Ad hoc*
Customer value	2
Volume of programs to be changed	New System
Effect on system	New System
STRATEGIC	
Is it strategic?	Aligned to business
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Over Business Units
COMPLEXITY	
What is the complexity of the request?	Low
EXISTING SYSTEM CHANGE	
What level of the existing system will require change?	New System
RESOURCE REQUIREMENTS	
What resources are required?	Labour 101-160 hours
DEVELOPMENT TIME	
Development Time	3-4 weeks
RISK	
Risk	Low
IT/BUSINESS REQUIREMENT	
IT/Business	Business
EXCO INFLUENCE	
Exco Influence	No
FINANCIAL	
Capex Required	No
PRIORITY SCORE	**164**
PRIORITY SCORE RESULT	**Medium Priority**

Stakeholders

The project was categorised as a medium project, based on its priority calculation which determined that the development consisted of a small number of developers. The project team was therefore lean and structured as per Table 7-3.

Table 7-3: Project stakeholders for the overseas travel control system

Stakeholder Position	Interest
Business Representatives	

Table 7-3: Project stakeholders for the overseas travel control system (continued)

Group Financial Controller	Customer
Financial Manager	Customer
Financial Controller: Accounts Payable	Customer
Technical Team	
System Manager	System Delivery
System Analyst Workflow	Developer
Integration Developer	Developer
Business Intelligence Analyst	Report Developer

In the following section, how the selected ASDMs were integrated with PMBOK for the overseas travel control system is described.

Due to the nature and culture of the organisation, ASDM and PMM integration was based on the integration matrix as per Table 7-4. ASDMs as per PMBOK's knowledge areas were selected based on the applicability within the specific organisation. The applicability is indicated in colour codes where green indicates a good fit for the project, yellow a partial fit and red no fit. Green and yellow selections were applied in the project execution and the red selections were discarded. In the following section, the ASDMs and PMM integration for the overseas travel control system project is described and illustrated in Table 7-4. This integration was compiled, based on the organisational characteristics, and determined the methodology components that were applied in the project execution process.

Table 7-4: Integration of PMBOK and ASDM for the overseas travel control system

PMBOK	Software development methodologies							
	SCRUM	Organisation Characteristics	XP	Organisation Characteristics	DSDM	Organisation Characteristics	FDD	Organisation Characteristics
Project Integration Management								
Develop project Charter	Software requirements are formulated and prioritised by the product owner as stories	Frequently changing requirements	User stories or requirements are created by customers and the development team	Frequently changing requirements	Feasibility and business study as evaluation and planning mechanisms	Not required as development is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Develop project management Plan	Sprint planning events	Frequently changing requirements	Stories are converted into iterations	Frequently changing requirements	Prioritisation is essential	Frequently changing requirements	Building a feature list	Object-oriented modelling not commonly followed in the organisation
Direct and Manage project Work	Strong change management procedure with product and sprint backlog	Frequently changing requirements	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Frequently changing requirements	Baseline of requirements and functionality is at a high level	Requirements need to be well defined and in detail	Planning and designing by feature	Object-oriented modelling not commonly followed in the organisation

Table 7-4: Integration of PMBOK and ASDM for the overseas travel control system (continued)

Monitor and Control project Work	Retrospective and Next Sprint Planning	Changes identified in specific sprints are incorporated instead of carrying them over, to the next sprint	Coding takes priority over all tasks	The development environment classifies requirements and testing as important as coding	Relies heavily on techniques to develop an application	Not required	Prioritise based on features	Object-oriented modelling not commonly followed in the organisation
Integrated Change Control			Continuous design-coding-testing-listening cycles	The development environment classifies requirements and testing as important as coding	Uses prototypes	Not required		
Close project					Good for projects with tight time constraints	Project needs to be delivered within a specified timeline		
					Testing integrated throughout the life cycle	Existing development process was changed to iterations development and continuous testing.		
Project Scope Management								
Scope Planning	Product Backlog Creation	Frequently changing requirements	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Not required as development is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Collect Requirements	Sprint Planning and Sprint Backlog Creation	Frequently changing requirements	Release planning and prioritisation by customer	Frequently changing requirements	Focusing on the customer needs	Frequently changing requirements	Develop feature list	Object-oriented modelling not commonly followed in the organisation
Scope Definition	User stories are defined and prioritised	Frequently changing requirements	Continuous feedback from customer	Frequently changing requirements	Active user involvement	Customer is closely involved during the development process		
Create Work Breakdown Structure	Review continuously to improve development process	Frequently changing requirements	Small releases	Frequently changing requirements	Frequent releases	Frequent releases make continuous testing possible		
Scope Validation Scope Control								
Project Time Management								
Plan Schedule	Plan sprint durations	Time estimation required	Iterations time planning	Planning is done for estimation purposes	Time and resource are not adjusted	Time can be adjusted based on requirements changes	Plan by feature	Object oriented modelling not commonly followed in the organisation
Define Activities	Sprint lasts ± 2 weeks	Can be flexible, but no longer than one week for small project	Release planning	Planning is done for estimation purposes	Deliver the projects as early as possible without affecting the quality	Time is not the primary focus and is flexible	Plan order of features based on dependencies	Object oriented modelling not commonly followed in the organisation
Sequence Activities	Task board for tracking progress	Customer requires fast and regular feedback			Good control over quality of product, cost and time	Big focus on quality in terms of functionality	Determine development sequence	Development is prioritised
Estimate Activity Resources	Daily stand-up meetings to discuss progress	Customer requires fast and regular feedback			Frequent delivery of products	Frequently changing requirements		
Estimate Activity Durations Develop Schedule Control Schedule								
Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	No cost implication	Programming team prepares the costs of carrying out iterations	No cost implication	Deals effectively with project cost	No cost implication	No primary focus	No cost implication
Cost Budgeting			Double development cost	No cost implication	Gives priority to time, quality, and cost	No cost implication		
Cost Control			Reduce cost via small iterations	No cost implication	Costly to implement	No cost implication		

Table 7-4: Integration of PMBOK and ASDM for the overseas travel control system (continued)

Project Quality Management								
Quality Planning	Quality goals sustained during sprints	Functionality and quality are important factors in terms of success criteria	Pair programming for higher quality code	Development teams are lean	Good control over quality of product, risk, cost and time	Not a big focus for medium projects	Continuous testing of the code	Frequently changing requirements
Perform Quality Assurance	Retrospective and next sprint planning to improve development process	Changes identified in specific sprints are incorporated instead of carrying them over, to the next sprint	Extreme testing in development phase	Functionality and quality are important factors in terms of success criteria	Delivering the projects without affecting the quality of the project	Quality is not comprisable	Coding standards	Coding standards are followed to ensure quality and support
Perform Quality Control	Regular scrum meetings	Functionality and quality are important factors in terms of success criteria	Continuous customer involvement and testing	Frequently changing requirements	Focus on frequent releases rather than quality	There is no compromise on quality	Measuring audits and metrics in the code	Not a big focus for medium projects
	Customer inputs for further improvements during iterations	Frequently changing requirements	Simple design	Not a primary focus as functionality and quality are more important				
			Coding standards	Coding standards are followed to ensure quality and support				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross functional	Developers are highly skilled and can work independently	Pair programming	Development teams are lean	Teams empowered to make decisions	Developers are highly skilled and can work independently	Client and development team develop overall model	Object-oriented modelling not commonly followed
Acquire project Team	Daily scrum meetings	Frequently changing requirements	Small teams	Development teams are lean	Facilitated Workshops	Workshops are facilitated to ensure all customer requirements are captured	Highly skilled teams	Developers are highly skilled and can work independently
Develop project Team	Small teams	Development teams are lean	Customer part of development team	Frequently changing requirements			Feature teams	Develop according to sprint backlog tasks and not according to features
Manage project Team			Team-oriented methodology	Regular develop-test cycles require a team-focussed methodology				
			Avoid working overtime	Not a focus point for the organisation				
Project Communications Management								
Communications Planning	Permanent communication and close cooperation between the stakeholders at each step	Frequently changing requirements	Continual communication with the customer and amongst the team	Frequently changing requirements	Facilitated Workshops	Workshops are facilitated to ensure all customer requirements are captured	Continuous review meetings	Frequently changing requirements
Information Distribution	Daily scrum Meetings	Frequently changing requirements	Collaborative workspaces	Teams work closely together with customer involvement	Environment ideal for the formation of ideas	Not a primary focus area	Feature teams deliver features	Object-oriented modelling not commonly followed
Performance Reporting	Group is self-organising and collaboratively managed	Developers are highly skilled and can work independently	Co-location of development and business space	Teams work closely together with customer involvement	Accurate and quick decision making	Frequently changing requirements		
Manage Stakeholders			Paired development	Development teams are lean				
			Frequently changing pair partners	Development teams are lean				
			Short stand-up meetings	Frequently changing requirements. Meetings can take longer in the early stages of the project				
			Unit tests and verbal communication	Frequently changing requirements				

Table 7-4: Integration of PMBOK and ASDM for the overseas travel control system (continued)

Project Risk Management								
Risk Management Planning	Iterative and incremental approach to control risk	Frequently changing requirements	Identify risks early in the project	Medium projects are mostly low risk projects	Changes are reversible during development	Frequently changing requirements	Guidance of skilled, experienced developers	Only skilled developers
Risk Identification	Sprints limit risk to one calendar month of cost	No cost implication	Frequent releases	Frequently changing requirements	Feasibility study identifies risks involved	Not required for as development is an internal function of the organisation	Feature planning take risks into account	Object-oriented modelling not commonly followed
Qualitative Risk Analysis	Project risks seen more quickly	Risks management important	Unit testing	Frequently changing requirements	Guidelines for risk management	No specific guidelines for risk management		
Quantitative Risk Analysis	Decisions to control risk are made based on the perceived state of the artefacts	Risk decisions are made based on project status	Customer part of development team	Frequently changing requirements	Reduce risk through incrementally delivering the solution	Frequently changing requirements		
Risk Response Planning	Meetings to review risks	Regular meetings cover risks as well	Collective ownership reduces risks of programmer dependency.	Development teams are lean				
Risk Monitoring and Control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting								
Request Seller Responses								
Select Sellers								
Contract Administration								
Contract Closure								

Project integration management - Organisational characteristics

The organisation required the project to be flexible as requirements normally change during project execution. To achieve this fast feedback, mechanisms and the ability to adapt quickly to changes were required. Based on the lessons learned in the quality concessions project, the project team accommodated changes within specific sprints instead of carrying changes, identified in a sprint, over to the next sprint. Project management was mainly performed by the technical team with the customer as a project team member. The development environment classified requirements and testing as important as coding. The organisation expected projects to be delivered according to the agreed timelines. The previously followed development process was changed to iterations development to incorporate continuous testing. A feasibility and business study as an evaluation and planning mechanism was not performed, as the project was classified as medium and development was an internal function of the organisation. Requirements needed to be well defined, as high-level requirements were not preferred. Various techniques and prototyping, as defined by DSDM, were not required. Quality and functionality were classified as more important than time which allowed for the timelines being flexible. Developing an overall model as per FDD was not done, as the organisation did not follow object-oriented modelling methodologies. Feature lists were not compiled as the project was managed by means of product and sprint backlogs.

Project scope management - Organisational characteristics

Frequent changing requirements within the organisation required the scope of the project to be compiled and managed by means of a product and sprint backlog. The customer preferred to be closely involved in the project and frequent releases ensured that the changing requirements were met on a continuous basis. Business studies were not required, as development was an internal function of the organisation. Feature lists were not required, as object-oriented modelling was not commonly followed in the organisation.

Project time management - Organisational characteristics

The organisation required an estimate of when the project deliverable could be expected. Because of continuous changing requirements, sprints did not have to be completed within two weeks and could run for longer periods. Regular feedback meetings fitted the environment of the organisation, as the customer preferred to be involved throughout the project process. Planning needed to be conducted in conjunction with the customer to estimate when the project would be completed. Time could be adjusted based on requirements changes as they were identified during sprints. Timelines were not the primary focus of the project and could be flexible. There was a big focus on the quality and functionality of the project which took priority over timelines. Feature planning was not required as object-oriented modelling was not commonly followed in the organisation.

Project cost management - Organisational characteristics

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management - Organisational characteristics

The organisation saw quality and functionality as higher priorities compared to time and prerequisites for the success of the project. Changes identified in specific sprints were incorporated in the same sprint instead of carrying them over to the next sprint. Continuous customer inputs during the iterations improved the overall quality of the product that was delivered. The development team was lean; this made pair programming as per XP not ideal. There was no primary focus on simple designs as functionality and quality were more important. To improve the quality of the products delivered to the organisation and to minimise the dependency on key developers, coding standards were followed as far as possible. Cost and time were not primary focus areas for medium size projects. Audits and metrics were not a major focus point for small projects.

Project human resource management - Organisational characteristics

The organisation had a highly skilled development team that worked independently. Regular meetings were preferred to ensure good communication during project execution. The development team was lean; this made pair programming as per XP not ideal. The organisation's developers were reluctant to write tests first and code later as stipulated by XP. Developers were highly skilled and could work independently. Workshops were facilitated to ensure all customer requirements were captured and incorporated in the product. Overtime was not encouraged in the organisation and was not required as the delivery of the project within the specified time took a lower priority than quality and functionality. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs was preferred to feature development.

Project communications management - Organisational characteristics

The organisation had a highly skilled development team who worked independently and closely together. Regular meetings were preferred to ensure good communication during project execution and to address the continuous changing requirements of the customer. Pair development was not done as the development team was lean. Workshops were facilitated to ensure all customer requirements were captured. The environment was the primary focus area for requirements gathering as the technical team and the customer resided on the same premises. Regular interactions between the project team and the customer allowed for accurate and quick decision making. Object oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs were preferred above feature development.

Project risk management - Organisational characteristics

There were no costs involved as the development of projects was executed internally by a lean in-house development team. Iterations therefore did not have to be limited to one calendar month to minimise the risk of cost overruns. Risk management in terms of quality and functionality was important to the organisation and decisions around risk were focussed on improving the overall status of the project. Regular meetings were preferred to manage the risk of continuous changing requirements and to deliver products according to expectation. Collective code ownership was not a priority due to the limited number of developers on the project. Feasibility and business studies as evaluation and planning mechanisms were not required as development was an internal function of the organisation. Guidelines for risk management as per DSDM were not required for medium projects within the organisation. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs was preferred above feature development.

Project procurement management - Organisational characteristics

No procurement management was required.

7.4 Action taking

In this section, the processes that were followed and the project team's feedback on the integration levels that were selected are described. The integration levels between PMBOK and the ASDMs for the overseas travel control system project which is summarised in Table 7-5 are also explained. The selected integration levels that were applied during the execution of the project are indicated in green. Interviews were conducted with the project team members that included feedback from the project manager, developers, analysts and customers involved in the project. The feedback obtained from the project team and related parties is incorporated in the following sections.

Table 7-5: Project Team Feedback

PMBOK	Software development methodologies							
	SCRUM	Project Team Feedback	XP	Project Team Feedback	DSDM	Project Team Feedback	FDD	Project Team Feedback
Project Integration Management								
Develop project Charter	Software requirements are formulated and prioritised by the product owner as stories	Developer plays role as BA and compile the requirements/stories in conjunction with the customer	User stories or requirements are created by customers and the development team	Workshops with Business to enable developer to compile requirements	Feasibility and business study as evaluation and planning mechanisms	Not applicable	Developing an overall model	Not applicable
Develop project management Plan	Sprint planning events	Developer doing sprint planning.	Stories are converted into iterations	Works well	Prioritisation is essential	Sprints are prioritised according to development life cycle and logical sequence of business process.	Building a feature list	Not applicable
Direct and Manage project Work	Strong change management procedure with product and sprint backlog	Changes logged as issues within active sprint if impacts on later development	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Business not involved in plan and time of iterations	Baseline of requirements and functionality is at a high level	Not applicable	Planning and designing by feature	Not applicable
Monitor and Control project Work	Retrospective and Next Sprint Planning	Changes included in active sprint when discussed in stand-up. Not included in next sprint.	Coding takes priority over all tasks	Not applicable	Uses heavily on techniques to develop an application	Not applicable	Prioritise based on features	Not applicable
Integrated Change Control			Continuous design-coding-testing-listening cycles	Works well	Uses prototypes			
Close project					Good for projects with tight time constraints	Timelines were difficult to manage due to continuous changing requirements		
					Testing integrated throughout the lifecycle	Worked well		
Project Scope Management								
Scope Planning	Product Backlog Creation	Works well	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Not applicable	Developing an overall model	Not applicable

Table 7-5: Project Team Feedback (continued)

Collect Requirements	Sprint Planning and Sprint Backlog Creation	Works well	Release planning and prioritisation by customer	Release planning works well. Prioritisation done by development team	Focusing on the customer needs	Works well as the users are continuously involved and saw progress of project, they were able to identify additional requirements/gaps to improve both the solution and their business process	Develop feature list	Not applicable
Scope Definition	User stories are defined and prioritised	Works well	Continuous feedback from customer	Works well	Active user involvement	Works well as the users are continuously involved and saw progress of project, they were able to identify additional requirements/gaps to improve both the solution and their business process		
Create Work Breakdown Structure	Review continuously to improve development process	Works well. The result might be less rework	Small releases	Works well	Frequent releases	Works well		
Scope Validation		Initial discussions were very high level and they only realised the detail due to being part of the process						
Scope Control		The ever-changing requirements prevented scope control, but some discussions lead to a proposal for a second phase of the project						
Project Time Management								
Plan Schedule	Plan sprint durations	Initial estimates done, but changed after each meeting with Business	Iterations time planning	Initial estimates done, but keeps on changing after each meeting with Business	Time and resource are not adjusted	Not applicable	Plan by feature	Not applicable
Define Activities	Sprint lasts ± 2 weeks	Some sprints took less than two weeks, but two weeks sprints worked well.	Release planning	Not used in medium project	Deliver the projects as early as possible without affecting the quality	Not applicable	Plan order of features based on dependencies	Not applicable
Sequence Activities	Task board for tracking progress	No physical boards were used, but Electronic Boards worked well for IT to track progress.			Good control over quality of product, cost and time	Cost was not applicable, but we were able to ensure a quality product	Determine development sequence	Determined by the developer based on experience, but changed due to availability of other resources and other priorities.
Estimate Activity Resources	Daily stand-up meetings to discuss progress	Stand-up meetings were held to monitor progress, but resulted in constant changing requirements. This in the end delivered a better solution, but also caused the target date to move out. Daily stand-ups were too frequent and meetings were held 2-3 times per week to allow the developer to make progress between meetings			Frequent delivery of products	Works well		
Estimate Activity Durations								
Develop Schedule								
Control Schedule								

Table 7-5: Project Team Feedback (continued)

Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	Not applicable	Programming team prepares the costs of carrying out iterations	Not applicable	Deals effectively with project cost	Not applicable	No primary focus	Not applicable
Cost Budgeting			Double development cost	Not applicable	Gives priority to time, quality, and cost	Not applicable		
Cost Control			Reduce cost via small iterations	Not applicable	Costly to implement	Not applicable		
Project Quality Management								
Quality Planning	Quality goals sustained during sprints	Works well. Show to business and therefore technical testing more thorough	Pair programming for higher quality code	Not applicable	Good control over quality of product, risk, cost and time	Not applicable	Continuous testing of the code	Works well
Perform Quality Assurance	Retrospective and Next Sprint Planning to improve development process	Works well. Lessons learned in sprints were applied in following sprints	Extreme testing in development phase	Works well	Delivering the projects without affecting the quality of the project	Works well	Coding standards	Works well
Perform Quality Control	Regular scrum meetings	Regular stand-up meetings worked well	Continuous customer involvement and testing	Works well	Focus on frequent releases rather than quality	Not applicable	Measuring audits and metrics in the code	Not applicable
	Customer inputs for further improvements during iterations	Works well	Simple design	Not applicable				
			Coding standards	Works well				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross functional	Works well	Pair programming	Not applicable	Teams empowered to make decisions	Works well in case where policy, or business rules were not changed. Senior business management was consulted when policy changes were deemed necessary	Client and development team develop overall model	Not applicable
Acquire project Team	Daily scrum meetings	Regular stand-up meetings worked well. *Ad hoc* meetings held with other team members/developers as necessary	Small teams	Works well if developers are dedicated to the project, without interruptions.	Facilitated Workshops	Works well.	Highly skilled teams	Works well.
Develop project Team	Small teams	Works well if developers are dedicated to the project, without interruptions.	Customer part of development team	Works well, because customer has more buy-in and took ownership.			Feature teams	Not applicable
Manage project Team			Team-oriented methodology	Development done by one developer only with the assistance of an integration developer.				
			Avoid working overtime	Not applicable				
Project Communications Management								
Communications Planning	Permanent communication and close cooperation between the stakeholders at each step	Works well	Continual communication with the customer and amongst the team	Works well	Facilitated Workshops	Works well	Continuous review meetings	Works well
Information Distribution	Daily scrum Meetings	Regular stand-up meetings worked well. *Ad hoc* meetings held with other team members/developers as necessary	Collaborative workspaces	Developers and Business work separately, except for Stand-up meetings	Environment ideal for the formation of ideas	Meetings allowed for development of ideas to solve user problems/gaps	Feature teams deliver features	Not applicable
Performance Reporting	Group is self-organising and collaboratively managed	Works well. Junior developers would not have achieved the same result.	Co-location of development and business space	Same location, but different office blocks between development team and customer	Accurate and quick decision making	Project team was not always able to make quick decisions and had to consult with snr management.		

Table 7-5: Project Team Feedback (continued)

Manage Stakeholders			Paired development	Not applicable				
			Frequently changing pair partners	Not applicable				
			Short stand-up meetings	It was tested, but failed. Meetings were at least an hour.				
			Unit tests and verbal communication	Works well				
Project Risk Management								
Risk Management Planning	Iterative and incremental approach to control risk	Works well	Identify risks early in the project	Works well	Changes are reversible during development	Works well. Changes were requested to change payment option after dev was completed.	Guidance of skilled, experienced developers	Not Applicable
Risk Identification	Sprints limit risk to one calendar month of cost	Not Applicable	Frequent releases	Works well	Feasibility study identify risks involved	Not Applicable	Feature planning takes risks into account	Not Applicable
Qualitative Risk Analysis	Project risks seen more quickly	Risks were identified in the frequent meetings and addressed. Example approvals on executive level.	Unit testing	Testing was only performed at the end of the development cycle.	Guidelines for risk management	Not Applicable		
Quantitative Risk Analysis	Decisions to control risk are made based on the perceived state of the artefacts	Not Applicable	Customer part of development team	Works well	Reduce risk through incrementally delivering the solution	Due to the agile process, the team was able to reduce risk by discussing it in the meetings and make relevant changes to the application.		
Risk Response Planning	Meetings to review risks	No separate risk meetings held	Collective ownership reduces risks of programmer dependency.	Not Applicable				
Risk Monitoring and Control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting Request Seller Responses Select Sellers Contract Administration Contract Closure								

Project integration management

The project team selected various combinations of ASDMs listed in Table 7-5. The ASDMs selected are indicated at the end of each point in brackets.

Processes followed

The following ASDM tasks in terms of project integration were applied:

- Software requirements were formulated and prioritised by the customer and project team as stories (SCRUM, XP). According to Table 7-5, scrum and extreme programming formulate software requirements in the form of user stories. This is done by both the project team and the customer who prioritise the requirements in terms of a development sequence. Table 5-47 displays the selected ASDM components in green.
- Development was done in iterations (SCRUM, XP, DSDM, FDD).
- Development tasks were planned by means of a sprint backlog (SCRUM, XP).

- Retrospective and next sprint planning were incorporated during sprints and not afterwards. (XP).
- Development followed a continuous cycle of design, coding, testing and listening (SCRUM, XP).

Project team feedback and improvements (Evaluating)

The project requirements were formulated by the project team which included the customer, but prioritisation was done by the development team only. The sprint planning process by means of the product and sprint backlog worked well and enabled the project team to plan the project thoroughly. The sprint backlog which was used for iteration planning and task prioritisation was compiled by the technical team only. These artefacts helped with the change management processes that kept the project team and customer up to date with sprint releases. Changes were accommodated during sprints and were not carried over to next sprints. Although the timelines were difficult to manage, the continuous design, coding and testing cycles as per the iterations were followed and enabled the project team to deliver releases according to customer expectation.

Project scope management

Processes followed

The following processes were followed and worked well to ensure delivery of the project:

- A product backlog was created that was used to define and prioritise the customer requirements in the form of user stories. These were defined by the development team and the customer (SCRUM, XP, DSDM).
- The product backlog defined the scope of the project (SCRUM, DSDM).
- The requirements were broken down into separate development tasks in the form of a sprint backlog listing the effort and time estimates for each task (SCRUM).
- Sprints were reviewed on a continuous basis; this improved the final product delivered to the customer rather than the development process (SCRUM, XP).
- The customer received continuous feedback from the development team throughout the development process (SCRUM, XP).
- Development was released frequently and in small iterations as defined by the various sprints (SCRUM, XP, DSDM).
- Developing an overall model as per FDD was not done, as the organisation did not follow object-oriented modelling methodologies.

Project team feedback and improvements

Initial discussions were at a very high level. The customer being closely involved throughout the process enabled the team to understand and validate the scope more thoroughly than was the case with previous projects. The product and sprint backlog creation worked well in defining and managing the scope of the project and continuously improved the development process. The customer was primarily involved which helped to deliver the project according to expectation. The backlogs helped the team to plan releases more accurately, but the development tasks were prioritised by the project team without the customer's involvement. Continuous customer feedback and small releases kept the focus on the project and ensured that the project progressed according to expectation. This enabled the customer to continuously see progress on the projects and to identify further improvements, both to the solution and the business process. Scope control was not possible due to the continuous changing requirements.

Project time management

Processes followed

Project time management was conducted as follows:

- The project was planned and prioritised by means of sprints that indicated the development tasks and sequence (SCRUM, FDD).
- The various tasks as per the sprint backlog were used to track progress (SCRUM).
- Daily stand-up meetings were conducted to address concerns and to ensure the project stayed on track (SCRUM).
- Priority was given to time and quality with close customer involvement (DSDM).
- Adjusting time and resources as per DSDM was not recommended as it was required due to continuous changing requirements.
- The organisation viewed quality as a higher priory than time.

Project team feedback and improvements

The sprint planning process improved project delivery estimations for the various iterations, and served as a good communication mechanism for the customer. The initial estimates changed due to customer requirement changes. The time it took to deliver the project was similar to that of historical projects, but the quality improved tremendously as no rework was required after the project was delivered. The developing sequence as specified by the sprint backlog, thus enabling the development team to stay focussed and to prioritise the development tasks effectively. Some sprints took less than two weeks, but two-week sprints worked well. No physical story boards were used. Electronic boards worked well to track progress. The stand-

up meetings were held to monitor progress, which resulted in constant changing requirements thereby delivering a better solution. This caused the target date to be moved out. Daily stand-up meetings were too frequent and the meeting frequency was changed to two to three times per week to allow the developer to make progress between meetings. The methodology ensured that good quality control could be maintained throughout the project process. The development sequence had to be changed due to resource limitations, as developers were not solely dedicated to the project and had to focus on other priorities throughout the project. Although the baseline timelines were missed, the final product was of such good quality that the customer accepted the revised deadlines.

Project cost management

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management

Processes followed

Quality was ensured by means of the following processes:

- The quality goals were defined during the product backlog creation process in conjunction with the customer (SCRUM).
- Regular meetings between the development team and the customer ensured that the development team delivered quality code for every iteration, thereby incorporating all customer inputs for further improvements (SCRUM).
- Extensive and continuous testing was performed during the development phase with continuous customer involvement (SCRUM, XP, FDD).
- Development was done by following predefined coding standards (XP, FDD).

Project team feedback and improvements

The quality goals were met and sustained by regular stand-up and feedback meetings with the customer which forced the technical team to do thorough technical testing more frequently. Retrospective and next sprint planning enabled the project team to apply lessons learned from previous sprints to next and future sprints. Regular, constructive feedback from the customer helped the project team to deliver quality products with every iteration. Coding standards improved the quality of the delivered products, as it made the code easier to review and ensured that the most efficient code was used.

Project human resource management

Processes followed

The project in terms of human resource management was structured as follows:

- Teams were self-organised, cross-functional and empowered to make decisions (SCRUM, DSDM).
- Daily stand-up meetings were conducted between the development team and the customer (SCRUM, XP).
- The development team was small and highly skilled with the customer part of the team (SCRUM, XP, FDD).
- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The development team being self-organised and highly skilled helped in delivering quality work according to priority and plan. The daily stand-up meetings helped to keep the team focussed and to deliver releases according to customer expectation. *Ad hoc* meetings were held with other team members/developers as necessary. Because the development team was lean and could not be dedicated to the project full-time, some interruptions occurred that impacted the task priorities and timelines. The customer took more ownership of the project as he was part of the project team. A team-oriented methodology was followed, although the technical team was very lean. Teams were empowered to make decisions as they are highly specialised, but where company policy changes were required, senior management had to be consulted for direction and decision making. Facilitated workshops worked well.

Project communications management

Processes followed

Project communications management was conducted as follows:

- There was close cooperation between all stakeholders with continuous communication throughout the execution of the project (SCRUM, XP).
- Daily stand-up meetings were held to monitor progress and to review the development process (SCRUM, XP, FDD).
- Team members were self-organised and collaboratively managed (SCRUM, XP).
- Teams were co-located, performing unit tests with good communication (XP).
- Workshops were facilitated (DSDM).
- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The frequent stand-up meetings improved the communication between all parties on the project. *Ad hoc* meetings with other team members/developers were conducted as necessary. Developers were highly skilled and worked independently. The project team worked in a collaborative area that did not include the customer as the customer could not be dedicated to the project on a full-time basis. The project team was not always able to make quick decisions and had to consult with senior management as policy-related decisions were required that could only be taken by senior management. The scrum meetings were efficient enough for communication between customer and the project team. Unit testing and verbal communication formed part of the iterations and improved the development quality.

Project risk management - Organisational characteristics

Processes followed

Project risks were managed and mitigated as follows:

- An iterative and incremental approach with frequent code releases was followed to control and to identify risks early in the project (SCRUM, XP, DSDM).
- The customer formed part of the development team that helped to reduce risks (SCRUM, XP, DSDM, FDD).
- Collective code ownership was not a priority as the technical team was lean (XP).
- Changes were reversible during development (DSDM).

Project team feedback and improvements

The product and sprint backlogs enabled the team to identify and manage risks related to quality and the project schedule effectively. The methodology of developing in iterations enhanced this process further. The stand-up meetings also assisted with identifying and discussing risks in time which enabled the team to address them in advance.

Project procurement management - Organisational characteristics

Procurement management was not required.

7.5 Evaluating

The project execution process was compared and evaluated against the evaluation criteria described in section 5.2 and delivered the following results:

Customer satisfaction feedback

Customer satisfaction was measured by means of a questionnaire in which the project team members were asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 7-6.

Table 7-6: Customer satisfaction feedback for the overseas travel control system

Customer	Score (1 = Poor, 10 = Excellent)
Group Financial Controller	8
Financial Manager	9
Financial Controller: Accounts Payable	9
System Manager	9
System Analyst Workflow	9
Integration Developer	9
Business Intelligence Analyst	9

Project delivery against agreed milestones

The project was baselined at the beginning of the project and measured against the baseline after the project was delivered. Table 7-7 displays the product backlog that indicates when the various project phases were completed against the planned completion dates. Although the project was not completed on the baseline date the project was completed with no rework. Previous projects required a vast amount of rework that caused major project delays.

Table 7-7: Product backlog for the overseas travel system

Item	Baseline Milestones	Actual Delivery
Submit a request for foreign travel	26 August 2019	26 August 2019
Approval process for travel request according to the delegation of authority	30 August 2019	30 August 2019
Update request with details of expenses for the approved trip	13 September 2019	13 September 2019
Automatically create purchase order(s) for the approved travel request.	27 September 2019	17 September 2019
Re-approval of updated request if the total value of the trip now exceeds the approved amount.	Change request	17 September 2019
Modify the details of the approved travel request prior to departure.	Change request	18 September 2019
Traveller completes and submits a claim form on return	4 October 2019	2 October 2019

Table 7-7: Product backlog for the overseas travel system (continued)

Financial Controller to update and correct details on the request from received invoices regarding the trip.	4 October 2019	2 October 2019
Approval of the completed Travel Claim	9 October 2019	8 October 2019
Integration with payroll via e-mail	9 October 2019	8 October 2019
Changes from User Testing	Change requests	18 October 2019
User acceptance testing	15 October 2019	25 October 2019
Deployment	16 October 2019	28 October 2019

Table 7-8 indicates the different sprints and sprint tasks with the estimate and actual hours per project sprint task. The tasks in yellow were changes requested during the stand-up meetings which were incorporated during the project and eliminated the need for rework at the end of the project.

Table 7-8: Sprint backlog for the overseas travel system

Sprint	Product Backlog Item	Sprint Task	Est hours	Act hours	Diff	Planned Date	Date Ended
1	Submit a request for foreign travel	Create database tables	4.0	6.0	-2.0	2019-08-22	2019-08-20
		Create K2 master data views with required validations	8.0	8.0	0.0	2019-08-22	2019-08-22
		Create K2 transaction views with required validations	8.0	6.0	2.0	2019-08-26	2019-08-26
		Create K2 master data form(s) with required validations	4.0	4.0	0.0	2019-08-22	2019-08-22
		Create K2 transaction form(s) with required task buttons	4.0	4.0	0.0	2019-08-26	2019-08-26
		Create K2 Menu for application	6.0	6.0	0.0	2019-08-20	2019-08-22
2	Approval process for Travel request according to the delegation of authority.	Create and Update Master tables	2.0	2.0	0.0	2019-08-26	2019-08-26
		Create and update Transaction tables	4.0	4.0	0.0	2019-08-26	2019-08-26
		Create K2 views with required validations	8.0	2.0	6.0	2019-08-28	2019-08-29
		Update K2 form for current processing	4.0	4.0	0.0	2019-08-28	2019-08-28
		Create approval workflow	8.0	7.0	1.0	2019-08-28	2019-08-29
		Create Reminders for Travel Request Approval Cycle	6.0	6.0	0.0	2019-08-30	2019-08-30
3	Update request with details of expenses for the approved trip.	Create and update database tables and SmartObjects	10.0	9.0	1.0	2019-09-13	2019-09-13
		Create K2 views with required validations	12.0	10.0	2.0	2019-09-13	2019-09-13
		Update K2 form for current processing	8.0	3.0	5.0	2019-09-13	2019-09-13
		Update approval workflow to include the request update step.	4.0	4.0	0.0	2019-09-13	2019-09-13
		Check and update request tables, views and form to ensure all required data is available to create purchase orders.	6.0	7.0	1.0	2019-09-13	2019-09-12

Table 7-8: Sprint backlog for the overseas travel system (continued)

		Update Travel Request Header - (Table,View,Form)	6.0	5.0	1.0	2019-09-13	2019-09-12
4	Automatically create purchase order(s) for the approved travel request.	MQ Administrators to assist in checking & testing the MQ integration	6.0	6.0	0.0	2019-09-16	2019-10-15
		Create tables for purchase order data	4.0	4.0	0.0	2019-09-20	2019-09-17
		Ensure that the required MQ queues are created - send message and receive response	1.0	1.0	0.0	2019-09-25	2019-09-25
		Create Purchase Order Message (MQ)	4.0	2.0	2.0	2019-09-25	2019-09-17
		Process to send PO messages	6.0	4.0	2.0	2019-09-25	2019-09-17
		Process to Receive the Purchase Order lines	5.0	3.0	2.0	2019-09-27	2019-09-17
		Ensure that the XML caters for the requirements for the creation of the Foreign Travel Purchase Orders.	4.0	4.0	0.0	2019-09-27	2019-09-27
5	Re-approval of updated request if the total value of the trip now exceeds the approved amount.	Create and update database tables	2.0	2.0	0.0	Change request	2019-09-17
		Create K2 views with required validations	4.0	4.0	0.0	Change request	2019-09-17
		Update K2 form for current processing	4.0	4.0	0.0	Change request	2019-09-17
		Add line manager approval step to workflow	4.0	2.0	2.0	Change request	2019-09-17
6	Modify the details of the approved travel request prior to departure.	Set up controls to determine whether a request can be modified.	3.0	3.0	0.0	Change request	2019-09-18
		Update the K2 views to allow the required modifications.	3.0	3.0	0.0	Change request	2019-09-18
		Update the K2 form(s) to allow the required modifications.	4.0	4.0	0.0	Change request	2019-09-18
		Update the approval workflow to allow re-submission of modified form.	4.0	3.0	1.0	Change request	2019-09-18
		Add approval steps to workflow to send for re-approval of changed Travel Request. Add approval steps to workflow to send for re-approval of changed Travel Request.	4.0	2.0	2.0	Change request	2019-09-18
		Update workflow to cater for creation and confirmation of new purchase orders.	4.0	2.0	2.0	Change request	2019-09-18
7	Traveller completes and submits a claim form on return	Create database objects for Travel Claim Form	10.0	10.0	0.0	2019-10-04	2019-10-02
		Create K2 master data views with required validations	1.0	1.0	0.0	2019-10-04	2019-09-27
		Create K2 transaction views with required validations	12.0	12.0	0.0	2019-10-04	2019-10-02
		Create K2 master data form(s) with required validations	1.0	0.5	0.5	2019-10-04	2019-09-27
		Create K2 transaction form(s) with required task buttons	15.0	15.0	0.0	2019-10-04	2019-10-02
8	Financial Controller to update and correct details on the request from received invoices regarding the trip.	Create Table, View and Form fields to indicate Foreign Travel Accountant's evaluation of the completed claim.	12.0	7.0	5.0	2019-10-04	2019-10-02
		Add evaluation step for Foreign Travel Accountant - the accountant can return the claim to the traveller AND make changes to the claim.	6.0	2.5	3.5	2019-10-04	2019-10-02

Table 7-8: Sprint backlog for the overseas travel system (continued)

9	Approval of the completed Travel Claim	Update workflow with Reminders and Escalations to ensure timeous completion of the claim form.	10.0	10.0	0.0	2019-10-09	2019-10-08
		Add "Finalise" button and workflow step for Foreign Travel Accountant.	4.0	3.0	1.0	2019-10-04	2019-10-04
		Claim Authorisation and approval	8.0	8.0	0.0	2019-10-04	2019-10-04
		Notify Traveller and Foreign Travel Accountant.	4.0	2.0	2.0	2019-10-09	2019-10-08
10	Integration with payroll via e-mail	Send claim to traveller to decide on payment method if traveller needs to re-pay	2.0	1.0	1.0	2019-10-09	2019-10-08
		Send request to Payroll Administrator	4.0	4.0	0.0	2019-10-09	2019-10-08
		Send request to Financial controller for checking if traveller selects EFT	4.0	2.0	2.0	2019-10-09	2019-10-08
11	Changes from User Testing	Create "Print to PDF" button for Travel Claim.	8.0	8.0	0.0	Change request	2019-10-14
		Initiate a claim if purchase orders have been allocated to a Travel Request and the request is cancelled	8.0	6.5	1.5	Change request	2019-10-14
		Change Return Date and related calculations	8.0	2.0	6.0	Change request	2019-10-10
		Change table and column headings	1.0	1.0	0.0	Change request	2019-10-10
		Change approval of request for EXCO travellers	8.0	3.0	5.0	Change request	2019-10-14
		Add "Expenses prior to travel" for traveller	4.0	3.4	0.6	Change request	2019-10-10
		Remove option to select EFT re-payment	1.0	1.0	0.0	Change request	2019-10-14
		Allow user to change forex rate for daily allowance claim	2.0	0.8	1.3	Change request	2019-10-11
		Add Motivation for selected expense types	4.0	3.3	0.8	Change request	2019-10-11
		Validate Expenses prior to Departure against PO value	3.0	1.0	2.0	Change request	2019-10-11
		Fix display on printouts	3.0	2.0	1.0	Change request	2019-10-15
		Update documentation	16.0	4.0	12.0	Change request	2019-10-18
12	User Acceptance Testing	User acceptance testing				2019-10-15	2019-10-25
13	Deployment	Package and Deploy				2019-10-16	2019-10-28

Rework post implementation

No rework was required or reported for a two-week period after the project was delivered and taken into production.

Modifications during the project due to scope changes

None to date

Implemented and in use

- The project was implemented successfully and is currently in use.

Project methodology followed

- The PMM and ASDM integration matrix as per Table 7-4 was applied to manage the project and to develop and implement the solution.
- Requirements were gathered, and development was based on the sprints and iterations as defined by the product and sprint backlogs of the project.
- The customer was continuously involved in all iterations and sprints throughout the project.
- The product and sprint backlogs made it easy to estimate timelines which could not be done previously.
- Compiling sprints enabled the team to plan development better and more constructive.

Comparison to historical projects

Based on the project classification, size and priority, the overseas travel control system project can be compared to the historical projects which were categorised as “medium” projects. Table 7.9 compares the project results of these projects.

Table 7.9: Project results comparison of small projects

Project Results	Overseas travel control system	Dealsheet management system	Frontend – Integrated business planning
Project classification	Medium	Medium	Medium
Priority	Medium	Medium	Medium
Customer Satisfaction	8.85	5.3	N/A
Scope Creep (Modifications)	0	23	67
Timeline	0.38 months late	15.1 months late	Not complete
Rework (Calls)	0	16	N/A

The quality concessions project required no scope changes and no rework during the execution of the project. Although the project was not delivered within the projected timeline, the overrun was much shorter compared to that of the historical projects. Consequently, the project produced a better customer experience that resulted in a good customer satisfaction score.

7.6 Specifying learning

During the project process, various lessons learned were identified by the project team. The lessons were used to determine if the next projects could be done differently to further improve the project process. The following limitations were identified during the project life cycle:

- Making changes within specified sprints work better than posting changes to next iterations. This was adopted and will be followed in the next project.

- The timelines were impacted due to continuous changing requirements.
- Sprint timelines had to be moved due to the development team not being solely dedicated to the project, as they had to focus on other system support and priorities as well.
- Documentation was limited due to time constraints. It was also not updated during the project due to time constraints, although the requirements that changed continuously were logged and managed.
- Daily stand-up meetings could not always take place due to the unavailability of various parties. These meetings took place two to three times per week that suited all project team members and were enough to identify changes and to keep the customer updated on progress.

In this chapter, the results of the overseas travel control system project that was executed according to the ASDM and PMM integration matrix explained in Chapter 3 were explained. Action research was applied by following the action research cycle of diagnosing, action planning, action taking, evaluating and specify learning. The final project results based on the evaluation criteria explained in Chapter 5 were better compared to those of the historical projects that were classified as medium projects.

In the next chapter, the third project that followed the action research cycle to investigate the impact of the ASDM and PMM integration will be discussed.

CHAPTER 8: ACTION RESEARCH CYCLE 3 – BIG PROJECT

In this chapter, a brief overview of the third project is given and how the action research cycle for each phase is applied to the third selected project categorised as a big project is explained. In sections 8.2, 8.3 and 8.4, how the action research cycle of diagnosing, action planning, action taking, evaluating and specifying learning is applied to the project, based on the criteria combinations the team selected from the ASDM and PMM integration matrix discussed in Chapter 3 is explained.

8.1 Cashbook (big project)

The purpose of this project was to develop a new cashbook system for the organisation. This involved development, integration and reporting related to the cash payments of the organisation. It was categorised as a big project due to the complexity, hours required and importance of the project. As for the overseas travel control system, the customer and custodian for this system was the finance department. Therefore, the same departmental characteristics mentioned in Chapter 7 apply to this project.

8.2 Diagnose

The selection and analysis of historical projects discussed in section 5.3 form part of the diagnose phase for this project and explain the primary problems experienced during the execution of these projects.

Based on the size and classification of the cashbook project, it can be compared to the big size historical projects displayed in Table 8-1.

Table 8-1: Summary of the big historical projects

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Capex Application System	Big project	High	6.1	33	2.5 months overrun	17
Fleet management system	Big project	High	5.5	21	6.7 months late	11

The big size historical projects were delivered behind schedule and required a specified amount of rework to complete the project. The project scope of these projects also changed throughout project execution and the overall customer satisfaction was rated between 5 and 6 out of a total

score of 10 which indicated that customer satisfaction in terms of project execution and delivery was not satisfactory.

8.3 Action planning

The project was measured against the evaluation criteria described in section 5.2. In the following section, the evaluation results used in planning the action that was taken to improve software development projects are described. The lessons learned as pointed out in section 6.6 in Chapter 6 and section 7.6 in Chapter 7 were taken into consideration to improve the process and methodology in executing the cashbook project. This involved the incorporation of changes during a sprint compared to only making changes at the beginning of a new sprint. In addition, the daily stand-up meetings were changed to weekly stand-up meetings and sometimes twice per week, depending on when the team had to discuss changes with the customer.

Project scope

The project scope involved the development of a new cashbook system for the finance department that included the following:

- Bank statement template maintenance
- Bank statement import
- Bank statement rollback
- Bank master maintenance

Project priority

Table 8-2 illustrates the project priority which was classified as a high priority.

Table 8-2: Project priority classification for the cashbook

IMPACT	
Pain of User	3
% of Customers Impacted	51-80
How often the "new feature" will be used?	Daily
Customer value	5
Volume of programs to be changed	New System
Effect on system	New System
STRATEGIC	
Is it strategic?	Aligned to business
CROSS-FUNCTIONAL	
Does it stretch over departments and/or systems? (Cross-functional)	Over Business Units

Table 8-2: Project priority classification for the cashbook (continued)

COMPLEXITY What is the complexity of the request?	High
EXISTING SYSTEM CHANGE What level of the existing system will require change?	New System
RESOURCE REQUIREMENTS What resources are required?	Labour > 160 hours
DEVELOPMENT TIME Development Time	>4 weeks
RISK Risk	High
IT/BUSINESS REQUIREMENT IT/Business	Business
EXCO INFLUENCE Exco Influence	No
FINANCIAL Capex Required	No
PRIORITY SCORE **PRIORITY SCORE RESULT**	**246** **High Priority**

Stakeholders

The project was categorised as a high priority project based on its priority calculation which determined that the development consisted out of a medium size development team. The project team was therefore structured as per Table 8-3.

Table 8-3: Project stakeholders for the cashbook

Stakeholder Position	Interest
Business Representatives	
Group Financial Manager	Customer
Financial Manager	Customer
Management Accountant	Customer
Assistant Accountant	Customer
Technical Team	
System Manager	System Delivery
System Analyst Applications	Developer
Integration Developer	Developer
Business Intelligence Analyst	Report Developer

In the following section, how the selected ASDMs were integrated with PMBOK for the cashbook project is described.

Due to the nature and culture of the organisation, ASDM and PMM integration was based on the integration matrix as per Table 8-4. ASDMs integrated with PMBOK's knowledge areas were selected, based on the applicability within the specific organisation. The applicability is indicated in colour codes where green indicates a good fit for the project, yellow a partial fit and red no fit. Green and yellow selections were applied in the project execution and the red selections were discarded. In the following section, the ASDMs and PMM integration for the cashbook project is described and illustrated in Table 8-4. This integration was compiled, based on the organisational characteristics, and determined the methodology components that were applied in the project execution process.

Table 8-4: Integration of PMBOK and ASDM for the cashbook

PMBOK	Software development methodologies							
	SCRUM	Organisation Characteristics	XP	Organisation Characteristics	DSDM	Organisation Characteristics	FDD	Organisation Characteristics
Project Integration Management								
Develop project Charter	Software requirements are formulated and prioritised by the product owner as stories	Frequently changing requirements	User stories or requirements are created by customers and the development team	Frequently changing requirements	Feasibility and business study as evaluation and planning mechanisms	Not required for small projects as developing is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Develop project management Plan	Sprint planning events	Frequently changing requirements	Stories are converted into iterations	Frequently changing requirements	Prioritisation is essential	Frequently changing requirements	Building a feature list	Develop according to sprint backlog tasks and not according to features
Direct and Manage project Work	Strong change management procedure with product and sprint backlog	Frequently changing requirements	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Frequently changing requirements	Baseline of requirements and functionality is at a high level	Requirements need to be well defined and in detail. Planning is also done in detail and not at a high level	Planning and designing by feature	Develop according to sprint backlog tasks and not according to features
Monitor and Control project Work	Retrospective and Next Sprint Planning	Changes are incorporated in existing sprints and not carried over	Coding takes priority over all tasks	The development environment classifies requirements and testing as important as coding	Uses techniques to develop an application	No techniques are currently being used. Prioritisation, and timeboxing will be applied for this project.	Prioritise based on features	Develop according to sprint backlog tasks and not according to features
Integrated Change Control			Continuous design-coding-testing-listening cycles	The development environment classifies requirements and testing as important as coding	Uses prototypes	Not practical as a prototype will require most of the development. Also not required for medium projects		
Close project					Good for projects with tight time constraints	Project needs to be delivered within a specified timeline		
					Testing integrated throughout the lifecycle	Based on previous learnings testing will be integrated throughout the lifecycle		

Table 8-4: Integration of PMBOK and ASDM for the cashbook (continued)

Project Scope Management								
Scope Planning	Product Backlog Creation	Frequently changing requirements. Scope planning required	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Business studies not required for small projects as developing is an internal function of the organisation	Developing an overall model	Object-oriented modelling not commonly followed in the organisation
Collect Requirements	Sprint Planning and Sprint Backlog Creation	Frequently changing requirements	Release planning and prioritisation by customer	Frequently changing requirements	Focusing on the customer needs	Frequently changing requirements	Develop feature list	Object-oriented modelling not commonly followed in the organisation
Scope Definition	User stories are defined and prioritised	Frequently changing requirements	Continuous feedback from customer	Frequently changing requirements	Active user involvement	Customer is closely involved during the development process		
Create Work Breakdown Structure	Review continuously to improve development process	Frequently changing requirements	Small releases	Frequently changing requirements	Frequent releases	Frequent releases make continuous testing possible		
Scope Validation								
Scope Control								
Project Time Management								
Plan Schedule	Plan sprint durations	Time estimation required	Iterations time planning	Planning is done for estimation purposes	Time and resource are not adjusted	Time can be adjusted based on requirements changes. Resources are unlikely to change	Plan by feature	Object-oriented modelling not commonly followed in the organisation
Define Activities	Sprint lasts ± 2 weeks	Can be flexible and longer than two weeks for a big project	Release planning	Planning is done for estimation purposes	Deliver the projects as early as possible without affecting the quality	Time is not the primary focus and is flexible. Functionality is a priority.	Plan order of features based on dependencies	Object oriented modelling not commonly followed in the organisation
Sequence Activities	Task board for tracking progress	Customer requires fast and regular feedback			Good control over quality of product, cost and time	Big focus on quality in terms of functionality. No cost is involved and time is not the primary focus	Determine development sequence	Development is prioritised
Estimate Activity Resources	Daily stand-up meetings to discuss progress	Customer requires fast and regular feedback, depending on customer availability			Frequent delivery of products	Frequently changing requirements		
Estimate Activity Durations								
Develop Schedule								
Control Schedule								
Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	No cost implication	Programming team prepares the costs of carrying out iterations	No cost implication	Deals effectively with project cost	No cost implication	No primary focus	No cost implication
Cost Budgeting			Double development cost	No cost implication	Gives priority to time, quality, and cost	No cost implication		
Cost Control			Reduce cost via small iterations	No cost implication	Costly to implement	No cost implication		

Table 8-4: Integration of PMBOK and ASDM for the cashbook (continued)

Project Quality Management								
Quality Planning	Quality goals sustained during sprints	Functionality and quality are important factors in terms of success criteria	Pair programming for higher quality code	Development teams are lean	Good control over quality of product, risk, cost and time	Good control over quality and functionality are important.	Continuous testing of the code	Frequently changing requirements
Perform Quality Assurance	Retrospective and Next Sprint Planning to improve development process	Changes identified in specific sprints are incorporated instead of carrying them over, to the next sprint	Extreme testing in development phase	Functionality and quality are important factors in terms of success criteria	Delivering the projects without affecting the quality of the project	Quality cannot be compromised	Coding standards	Coding standards are followed to ensure quality and support
Perform Quality Control	Regular scrum meetings	Functionality and quality are important factors in terms of success criteria	Continuous customer involvement and testing	Frequently changing requirements	Focus on frequent releases rather than quality	Quality more important than time	Measuring audits and metrics in the code	Not a big focus point for the organisation
	Customer inputs for further improvements during iterations	Frequently changing requirements	Simple design	Not a primary focus as functionality and quality are more important				
			Coding standards	Coding standards are followed to ensure quality and support				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross functional	Developers are highly skilled and can work independently	Pair programming	Development teams are lean	Teams empowered to make decisions	Developers are highly skilled and can work independently	Client and development team develop overall model	Object-oriented modelling not commonly followed
Acquire project Team	Daily scrum meetings	Frequently changing requirements	Small teams	Development teams are lean	Facilitated Workshops	Workshops are facilitated to ensure all customer requirements are captured	Highly skilful teams	Developers are highly skilled and can work independently
Develop project Team	Small teams	Development teams are lean	Customer part of development team	Frequently changing requirements			Feature teams	Develop according to sprint backlog tasks and not according to features
Manage project Team			Team-oriented methodology	Regular develop-test cycles require a team-focussed methodology				
			Avoid working overtime	Not a focus point for the organisation				
Project Communications Management								
Communications Planning	Permanent communication and close cooperation between the stakeholders at each step	Frequently changing requirements	Continual communication with the customer and amongst the team	Frequently changing requirements	Facilitated Workshops	Workshops are facilitated to ensure all customer requirements are captured	Continuous review meetings	Frequently changing requirements
Information Distribution	Daily scrum Meetings	Frequently changing requirements	Collaborative workspaces	Teams work closely together with customer involvement	Environment ideal for the formation of ideas	Not a primary focus area	Feature teams deliver features	Object-oriented modelling not commonly followed
Performance Reporting	Group is self-organising and collaboratively managed	Developers are highly skilled and can work independently	Co-location of development and business space	Teams work closely together with customer involvement	Accurate and quick decision making	Frequently changing requirements		
Manage Stakeholders			Paired development	Development teams are lean				
			Frequently changing pair partners	Development teams are lean				

Table 8-4: Integration of PMBOK and ASDM for the cashbook (continued)

			Short stand-up meetings	Frequently changing requirements				
			Unit tests and verbal communication	Frequently changing requirements				
Project Risk Management								
Risk Management Planning	Iterative and incremental approach to control risk	Frequently changing requirements	Identify risks early in the project	Risks management important	Changes are reversible during development	Frequently changing requirements	Guidance of skilled, experienced developers	Only skilled developers
Risk Identification	Sprints limit risk to one calendar month of cost	No cost implication	Frequent releases	Frequently changing requirements	Feasibility study identifies risks involved	Not required for medium projects as development is an internal function of the organisation	Feature planning takes risks into account	Object-oriented modelling not commonly followed
Qualitative Risk Analysis	Project risks seen more quickly	Risks management is important	Unit testing	Frequently changing requirements	Guidelines for risk management	No specific guidelines for risk management		
Quantitative Risk Analysis	Decisions to control risk are made based on the perceived state of the artefacts	Risk decisions are made based on project status and test results	Customer part of development team	Frequently changing requirements	Reduce risk through incrementally delivering the solution	Frequently changing requirements		
Risk Response Planning	Meetings to review risks	Regular stand-up meetings cover risks as well	Collective ownership reduces risks of programmer dependency	Development teams are lean				
Risk Monitoring and Control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting Request Seller Responses Select Sellers Contract Administration Contract Closure								

Project integration management - Organisational characteristics

The organisation required the project to be flexible as the requirements might have changed during project execution. Fast feedback mechanisms and the ability to adapt quickly to changes were required. Based on the lessons learned in the previous projects, the project team accommodated changes within specific sprints instead of carrying changes, identified in a sprint, over to the next sprint. Project management was mainly performed by the technical team with the customer as a project team member. The development environment classified requirements and testing as important as coding. The organisation expected projects to be delivered according to agreed timelines. The previous development process was changed to an iteration development process that incorporated continuous testing. A feasibility and business study as an evaluation and planning mechanism was not performed, as the project was managed

internally. Requirements needed to be well defined, as high-level requirements were not accepted. Various techniques and prototyping, as defined by DSDM, were not required. Quality and functionality were classified as more important than time which allowed for the timelines being flexible. Developing an overall model as per FDD was not done as the organisation did not follow object-oriented modelling methodologies. Feature lists were not compiled, as the project was managed by means of product and sprint backlogs.

Project scope management - Organisational characteristics

Frequent changing requirements within the organisation required the scope of the project to be compiled and managed by means of a product and sprint backlog. The business requirements were defined as stories and prioritised accordingly. The customer preferred to be closely involved in the project and frequent releases ensured that the changing requirements were met on a continuous basis. Business studies were not required, as development was an internal function of the organisation. Feature lists were not required, as object-oriented modelling was not commonly followed in the organisation.

Project time management - Organisational characteristics

The organisation required an estimate of when the project deliverable could be expected. Because of continuously changing requirements, sprints did not have to be completed within two weeks and could run for longer periods. Regular feedback meetings fitted the environment of the organisation, as the customer preferred to be involved throughout the project process. Planning needed to be conducted in conjunction with the customer to estimate when the project could be completed. Timelines could be adjusted, based on requirements changes as they were identified during sprints. Timelines were not the primary focus of the project and could be flexible. There was a major focus on the quality and functionality of the project which took priority over timelines. Feature planning was not required, as object-oriented modelling was not commonly followed in the organisation.

Project cost management - Organisational characteristics

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management - Organisational characteristics

The organisation viewed quality and functionality as higher priorities when compared to time and as prerequisites for the success of the project. Changes identified in specific sprints were incorporated in the same sprint instead of carrying them over to the next sprint. Continuous customer inputs during the iterations improved the overall quality of the product that was

delivered. The development team consisted of different developers from a technology point of view which was not ideal for pair programming as per XP. There was no primary focus on simple designs as functionality and quality were more important. Quality and functionality were regarded as important factors that needed to be managed and controlled. To improve the quality of the products delivered to the organisation and to minimise the dependency on key developers, coding standards were followed as far as possible. Cost and time were not the primary focus areas for medium size projects. Audits and metrics were not a major focus point for this project.

Project human resource management - Organisational characteristics

The organisation had a highly skilled development team that worked independently. Regular meetings were preferred to ensure good communication during project execution. The development team comprised different developers from a technology point of view which was not ideal for pair programming as per XP. The organisation's developers were also reluctant to write tests first and code later as stipulated by XP. Developers were highly skilled and worked independently. Workshops were facilitated to ensure all customer requirements were captured and incorporated in the product. Overtime was not encouraged in the organisation and was not required as the delivery of the project within a specified timeframe took a lower priority than quality and functionality. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs was preferred above feature development.

Project communications management - Organisational characteristics

The organisation had a highly skilled development team that worked independently and closely together. Regular meetings were preferred to ensure good communication during project execution and to address the continuous changing requirements of the customer. Pair development was not done due to the different types of developer on the project. Workshops were facilitated to ensure all customer requirements were captured. The environment was the primary focus area for requirements gathering, as the technical team and the customer resided on the same premises. Regular interactions between the project team and the customer allowed for accurate and quick decision making. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs was preferred above feature development.

Project risk management - Organisational characteristics

Project risks were relatively small, as there were no costs involved due to development being executed internally. Iterations therefore did not have to be limited to one calendar month to

minimise the risk of cost overruns. Risk management in terms of quality and functionality were important to the organisation and decisions around risk were focussed on improving the overall status of the project. Regular meetings were preferred to manage the risk of continuously changing requirements and to deliver products according to expectation. Collective code ownership was not a priority due to the different types of developer on the project. Feasibility and business studies as evaluation and planning mechanisms were not required as development was an internal function of the organisation. Guidelines for risk management as per DSDM were not required for projects within the organisation. Object-oriented modelling was not commonly followed within the organisation and development according to product and sprint backlogs was preferred above feature development.

Project procurement management - Organisational characteristics

No procurement management was required.

8.4 Action taking

In this section, the processes that were followed and the project team's feedback on the integration levels that were selected are described. The integration levels between PMBOK and the ASDMs for the cashbook project which is summarised in Table 8-5 are also explained. The selected integration levels what were applied during the execution of the project are indicated in green. Interviews were conducted with the project team members that included feedback from the project team, developers, analysts and customers involved in the project. The feedback obtained from the project team and related parties is incorporated in the following sections.

Table 8-5: Project Team Feedback for the cashbook

PMBOK	Software development methodologies							
	SCRUM	Project Team Feedback	XP	Project Team Feedback	DSDM	Project Team Feedback	FDD	Project Team Feedback
Project Integration Management								
Develop project Charter	Software requirements are formulated and prioritised by the product owner as stories	Requirements were formulated by the project team and the customer, but the tasks were prioritised by the technical team	User stories or requirements are created by customers and the development team	Works well	Feasibility and business study as evaluation and planning mechanisms	Not applicable	Developing an overall model	Not applicable
Develop project management Plan	Sprint planning events	Sprints' timelines were impacted due to frequently changing requirements. Planning done by technical team	Stories are converted into iterations	Works well	Prioritisation is essential	Sprint tasks are prioritised by the development team	Building a feature list	Not applicable
Direct and Manage project Work	Strong change management procedure with product and sprint backlog	Works well	Programming team and business prepares the plan, time, and costs of carrying out the iterations	Determined by development team	Baseline of requirements and functionality is at a high level	Not applicable	Planning and designing by feature	Not applicable

Table 8-5: Project Team Feedback for the cashbook (continued)

Monitor and Control project Work	Retrospective and Next Sprint Planning	Changes included in active sprint when discussed in stand-up. Not included in next sprint.	Coding takes priority over all tasks	Not applicable	Uses heavily on techniques to develop an application	Not applicable	Prioritise based on features	Not applicable
Integrated Change Control			Continuous design-coding-testing-listening cycles	Works well	Uses prototypes			
Close project					Good for projects with tight time constraints	Timelines were difficult to manage due to continuous changing requirements		
					Testing integrated throughout the lifecycle	Works well		
Project Scope Management								
Scope Planning	Product Backlog Creation	Works well	Customer defines user stories	Customer primarily involved	Requirements list as per the business study defines the scope	Not applicable	Developing an overall model	Not applicable
Collect Requirements	Sprint Planning and Sprint Backlog Creation	Works well	Release planning and prioritisation by customer	Release planning works well. Prioritisation done by development team	Focusing on the customer needs	Works well as the users are continuously involved and saw progress on the project. They were able to identify additional requirements/gaps to improve both the solution and their business process	Develop feature list	Not applicable
Scope Definition	User stories are defined and prioritised	Works well	Continuous feedback from customer	Works well	Active user involvement	Customer is closely involved during the development process		
Create Work Breakdown Structure	Review continuously to improve development process	Works well. The result will be less rework	Small releases	Works well	Frequent releases	Works well		
Scope Validation		Initial discussions were very high level and they only realised the detail due to being part of the process						
Scope Control		The ever-changing requirements prevented scope control, but some discussions lead to a proposal for a second phase of the project						
Project Time Management								
Plan Schedule	Plan sprint durations	Works well	Iterations time planning	Works well	Time and resource are not adjusted	Not applicable	Plan by feature	Not applicable
Define Activities	Sprint lasts ± 2 weeks	Can be flexible and longer than two weeks for big projects	Release planning	Works well	Deliver the projects as early as possible without affecting the quality	Not applicable	Plan order of features based on dependencies	Not applicable
Sequence Activities	Task board for tracking progress	No physical boards were used, but Electronic Boards worked well for IT to track progress. Progress was mostly tracked in Stand-up meetings			Good control over quality of product, cost and time	Cost was not applicable, but the team was able to ensure a quality product	Determine development sequence	Determined by the developer based on experience, but changed due to availability of other resources and other priorities.

Table 8-5: Project Team Feedback for the cashbook (continued)

Estimate Activity Resources	Daily stand-up meetings to discuss progress	Stand-up meetings were held to monitor progress, but resulted in constant changing requirements. This in the end delivered a better solution, but also caused the target date to move out. Daily stand-ups were too frequent and meetings were held 2-3 times per week to allow the developer to make progress between meetings			Frequent delivery of products	Works well		
Estimate Activity Durations Develop Schedule Control Schedule								
Project Cost Management								
Cost Estimating	Scrum team defines cost estimates related to stories during sprint planning	Not applicable	Programming team prepares the costs of carrying out iterations	Not applicable	Deals effectively with project cost	Not applicable	No primary focus	Not applicable
Cost Budgeting			Double development cost	Not applicable	Gives priority to time, quality, and cost	Not applicable		
Cost Control			Reduce cost via small iterations	Not applicable	Costly to implement	Not applicable		
Project Quality Management								
Quality Planning	Quality goals sustained during sprints	Works well. Show to business and therefore technical testing more thorough	Pair programming for higher quality code	Not applicable	Good control over quality of product, risk, cost and time	Iterations helped to control quality and functionality	Continuous testing of the code	Works well
Perform Quality Assurance	Retrospective and Next Sprint Planning to improve development process	Works well. First sprint review and testing are prerequisites for next sprint. When sprints were completed, tasks were added to next sprints should another requirement applicable to sprint one development be discovered	Extreme testing in development phase	Works well	Delivering the projects without affecting the quality of the project	Works well as quality remained a key priority throughout the project	Coding standards	Works well
Perform Quality Control	Regular scrum meetings	Regular stand-up meetings worked well	Continuous customer involvement and testing	Works very well (ensures that requirements gathered was correctly implemented)	Focus on frequent releases rather than quality	Not applicable	Measuring audits and metrics in the code	Not applicable
	Customer inputs for further improvements during iterations	Works well	Simple design	Simple design not possible due to complexity				
			Coding standards	Works well				
Project Human Resource Management								
Human Resource Planning	Scrum teams self-organised and cross functional	Works well	Pair programming	Not applicable	Teams empowered to make decisions	Works well	Client and development team develop overall model	Not applicable

Table 8-5: Project Team Feedback for the cashbook (continued)

Acquire project Team	Daily scrum meetings	Regular stand-up meetings worked well. They could not be held daily, but two to three times per week	Small teams	Works well if developers are dedicated to the project and not distracted. Developers could not be dedicated and also had other responsibilities	Facilitated Workshops	Works well.	Highly skilled teams	Works well.
Develop project Team	Small teams	Works well if developers are dedicated to the project and not distracted. Developers could not be dedicated and also had other responsibilities	Customer part of development team	Works well, because customer had more buy-in and took ownership.			Feature teams	Not applicable
Manage project Team			Team-oriented methodology	Due to lean team and operational requirements of the customer, it may be required to add another developer				
			Avoid working overtime	May be required for tight deadlines				
Project Communications Management								
Communications Planning	Permanent communication and close cooperation between the stakeholders at each step	Works well	Continual communication with the customer and amongst the team	Works well	Facilitated Workshops	Works well	Continuous review meetings	Works well
Information Distribution	Daily scrum Meetings	Regular stand-up meetings worked well. *Ad hoc* meetings held with other team members/developers as necessary	Collaborative workspaces	Developers and Business work separately, except for Stand-up meetings	Environment ideal for the formation of ideas	Meetings allowed for development of ideas to solve user problems/gaps	Feature teams deliver features	Not applicable
Performance Reporting	Group is self-organising and collaboratively managed	Works well. Junior developers would not have achieved the same result.	Co-location of development and business space	Same location, but different office blocks between development team and customer	Accurate and quick decision making	Project team was not always able to make quick decisions and had to consult with snr management.		
Manage Stakeholders			Paired development	Not applicable				
			Frequently changing pair partners	Not applicable				
			Short stand-up meetings	It was tested, but failed. Meetings were at least an hour.				
			Unit tests and verbal communication	Works well				
Project Risk Management								
Risk Management Planning	Iterative and incremental approach to control risk	Works well	Identify risks early in the project	Works well, although no risks were identified	Changes are reversible during development	Works well	Guidance of skilled, experienced developers	Not Applicable
Risk Identification	Sprints limit risk to one calendar month of cost	Not Applicable	Frequent releases	Works well	Feasibility study identifies risks involved	Not Applicable	Feature planning take risks into account	Not Applicable
Qualitative Risk Analysis	Project risks seen more quickly	Risks were identified and addressed in the frequent meetings.	Unit testing	Testing was only performed at the end of the development cycle per iteration.	Guidelines for risk management	Not Applicable		

Table 8-5: Project Team Feedback for the cashbook (continued)

Quantitative Risk Analysis	Decisions to control risk are made based on the perceived state of the artefacts	Not Applicable	Customer part of development team	Works well - Allows users to customize the solution	Reduce risk through incrementally delivering the solution	Due to the agile process, the team was able to reduce risk by discussing it in the meetings and make relevant changes to the application.		
Risk Response Planning	Meetings to review risks	Works well. Risks reviewed in stand-up meetings	Collective ownership reduces risks of programmer dependency.	Not Applicable				
Risk Monitoring and Control								
Project Procurement Management								
Plan Purchases and Acquisitions	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable	No information	Not Applicable
Plan Contracting								
Request Seller Responses								
Select Sellers								
Contract Administration								
Contract Closure								

Project integration management

The project team selected various combinations of ASDMs listed in Table 8-5. The ASDMs selected are indicated at the end of each point in brackets.

Processes followed

The following ASDM tasks in terms of project integration were applied:

- Software requirements were formulated and prioritised by the customer and project team as stories (SCRUM, XP). According to Table 8-5, scrum and extreme programming formulate software requirements in the form of user stories. This is done by both the project team and the customer who prioritise the requirements in terms of a development sequence. Table 8-5 displays the selected ASDM components in green.
- Development was done in iterations (SCRUM, XP, DSDM, FDD).
- Development tasks were planned by means of a sprint backlog (SCRUM, XP).
- Retrospective and next sprint planning were incorporated during sprints and not afterwards. (XP).
- Development followed a continuous cycle of design, coding, testing and listening (SCRUM, XP).

Project team feedback and improvements (Evaluating)

The project requirements were formulated by the project team that included the customer, but prioritisation was done by development team only. The sprint planning process by means of the product and sprint backlog worked well and enabled the project team to plan the project

thoroughly. The sprint backlog was compiled by the technical team only and was used from iteration planning and task prioritisation. These artefacts helped with the change management processes that kept the project team and customer up to date with sprint releases. Changes were accommodated during sprints and were not carried over to next sprints. Although the timelines were difficult to manage, the continuous design, coding and testing cycles as per the iterations were followed and enabled the project team to deliver releases according to customer expectation.

Project scope management

Processes followed

The following processes were followed and worked well to ensure delivery of the project:

- A product backlog was created that was used to define and prioritise the customer requirements in the form of user stories. These were defined by the development team and the customer (SCRUM, XP, DSDM).
- The product backlog defined the scope of the project (SCRUM, DSDM).
- The requirements were broken down into separate development tasks in the form of a sprint backlog, listing the effort and time estimates for each task (SCRUM).
- Sprints were reviewed on a continuous basis; this improved the final product delivered to the customer rather than the development process (SCRUM, XP).
- The customer received continuous feedback from the development team throughout the development process (SCRUM, XP).
- Development was released frequently and in small iterations as defined by the various sprints (SCRUM, XP, DSDM).
- Developing an overall model as per FDD was not done as the organisation does not follow object-oriented modelling methodologies.

Project team feedback and improvements

Initial discussions were very high level, but with the customer being closely involved throughout the process the team was able to understand and validate the scope more thoroughly than for previous projects. The product and sprint backlog creation worked well in defining and managing the scope of the project and continuously improved the development process. The customer was primarily involved which helped to deliver the project according to expectation. The backlogs helped the team to plan releases more accurately; however, the development tasks were prioritised by the project team without the customer's involvement. The continuous customer feedback and small releases kept the focus on the project and ensured that the project progressed according to expectation. This enabled the customer to continuously see

progress on the projects and to identify further improvements, both to the solution and the business process. Scope control was not possible due to the continuously changing requirements.

Project time management

Processes followed

Project time management was conducted as follows:

- The project was planned and prioritised by means of sprints that indicated the development tasks and sequence (SCRUM, FDD).
- The various tasks as per the sprint backlog were used to track progress (SCRUM).
- Daily stand-up meetings were conducted to address concerns and to ensure the project stayed on track (SCRUM).
- Priority was given to time and quality with close customer involvement (DSDM).
- Adjusting time and resources as per DSDM was not recommended as it was required due to continuously changing requirements.
- The organisation perceived quality as a higher priory than time.

Project Team Feedback and improvements

The sprint planning process improved project delivery estimations for the various iterations and served as a good communication mechanism for the customer. The initial estimates changed due to customer requirement changes. The time it took to deliver the project was similar to that of historical projects, but the quality improved tremendously as no rework was required after the project was delivered. The developing sequence as specified by the sprint backlog enabled the development team to stay focussed and to prioritise the development tasks effectively. Some sprints took more than two weeks, but overall, two-week sprints worked well. No physical story boards were used. Electronic boards worked effectively to track progress. The stand-up meetings were held to monitor progress, but resulted in constant changing requirements which delivered a better solution, although it caused the target date to move out. Daily stand-ups were too frequent and meetings were rather held twice a week to allow the developer to make progress between meetings. The methodology ensured that good quality control could be maintained throughout the project process. The development sequence had to be changed due to resources limitations. Developers were not solely dedicated to the project. They had to focus on other priorities throughout the project.

Project cost management

Project cost management was not applicable for this project as no costs were involved. The project was executed by an internal development team.

Project quality management

Processes followed

Quality was ensured by means of the following processes:

- The quality goals were defined during the product backlog creation process in conjunction with the customer (SCRUM).
- Regular meetings between the development team and the customer ensured that the development team delivered quality code for every iteration, thereby incorporating all customer inputs for further improvements (SCRUM).
- Extensive and continuous testing was performed during the development phase with continuous customer involvement (SCRUM, XP, FDD).
- Development was done by following predefined coding standards (XP, FDD).

Project Team Feedback and improvements

The quality goals were met and sustained by regular stand-up and feedback meetings with the customer which forced the technical team to do thorough technical testing more frequently. Retrospective and next sprint planning enabled the project team to apply lessons learned from previous sprints to next and future sprints. Regular, constructive feedback from the customer helped the project team to deliver quality products with every iteration. Coding standards improved the quality of the delivered products as it made the code easier to review and ensured that the most efficient code was used.

Project human resource management

Processes followed

The project human resource management was structured as follows:

- Teams were self-organised, cross-functional and empowered to make decisions (SCRUM, DSDM).
- Daily stand-up meetings were conducted between the development team and the customer (SCRUM, XP).
- The development team was small and highly skilled, with the customer as part of the team (SCRUM, XP, FDD).
- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The development team being self-organised and highly skilled helped in delivering quality work according to priority and plan. The daily stand-up meetings helped to keep the team focussed and to deliver releases according to customer expectation. *Ad hoc* meetings were held with other team members/developers as necessary. Because the development team was lean and could not be dedicated full-time to the project, some interruptions occurred that impacted the task priorities and timelines. The customer took greater ownership of the project as he was part of the project team. A team-oriented methodology was followed in spite of the technical team being very lean. Teams were empowered to make decisions as they are highly specialised, but where company policy changes were required, senior management had to be consulted for direction and decision making. Facilitated workshops worked well.

Project communications management

Processes followed

Project communications management was conducted as follows:

- There was close cooperation between all stakeholders with continuous communication throughout the execution of the project (SCRUM, XP).
- Daily stand-up meetings were held to monitor progress and to review the development process (SCRUM, XP, FDD).
- Team members were self-organised and collaboratively managed (SCRUM, XP).
- Teams were co-located performing unit tests with good communication (XP).
- Workshops were facilitated (DSDM).
- Workshops were conducted to ensure that the customer requirements were captured accurately (DSDM).

Project team feedback and improvements

The frequent stand-up meetings improved the communication between all parties on the project. *Ad hoc* meetings with other team members/developers were conducted as necessary. Developers were highly skilled and worked independently. The project team worked in a collaborative area that did not include the customer, as the customer could not be dedicated to the project on a full-time basis. The project team was not always able to make quick decisions and had to consult with senior management, as policy-related decisions were required that could only be taken by senior management. The scrum meetings were efficient enough for communication between customer and project team. Unit testing and verbal communication formed part of the iterations and improved the development quality.

Project risk management - Organisational characteristics

Processes followed

Project risks were managed and mitigated as follows:

- An iterative and incremental approach with frequent code releases was followed to control and to identify risks early in the project (SCRUM, XP, DSDM).
- The customer formed part of the development team which helped to reduce risks (SCRUM, XP, DSDM, FDD).
- Collective code ownership is not a priority, as the technical team is lean (XP).
- Changes are reversible during development (DSDM).

Project Team Feedback and improvements

The product and sprint backlogs enabled the team to identify and manage risks related to quality and the project schedule effectively. The methodology of developing in iterations enhanced this process further. The stand-up meetings also assisted with identifying and discussing risks in time so that the team could address them in advance.

Project procurement management - Organisational characteristics

Procurement management was not required.

8.5 Evaluating

Customer satisfaction was measured by means of a questionnaire in which the project team members were asked to give an overall score in terms of how the project was executed and if its requirements and expectations were met. The results are summarised in Table 8-6.

Table 8-6: Customer satisfaction feedback for cashbook

Customer	**Score (1 = Poor, 10 = Excellent)**
Group Financial Controller	9
Financial Manager	10
Management Accountant	9
Assistant Accountant	10
System Manager	9
System Analyst Applications	9
Integration Developer	9
Business Intelligence Analyst	9

Project delivery against agreed milestones

The project was baselined at the beginning of the project and measured against the baseline after the project was delivered. Table 8-7 displays the product backlog that indicates when the various project phases were completed against the planned completion dates. Although the project was not completed on the baseline date, the project was completed with no rework and much better quality. Previous projects required plenty of rework that caused major project delays.

Table 8-7: Product backlog for the cashbook system

Item	Baseline Milestones	Actual Delivery
User Imports Banks Statements	23 August 2019	1 October 2019
View / Process Cash Book	9 September 2019	11 October 2019
View / Process Bank Statement Transactions	23 September 2019	17 October 2019
Bank Reconciliation	4 October 2019	30 October 2019
Interfaces	9 October 2019	5 November 2019

Table 8-8 indicates the different sprints and sprint tasks with the estimated and actual hours per project sprint task. The tasks in yellow were changes requested during the stand-up meetings which were incorporated during the project; this eliminated the need for rework at the end of the project.

Table 8-8: Sprint backlog for the cashbook system

Sprint	Product Backlog Item	Sprint Task	Est hours	Act hours	Diff	Planned Date	Date Ended
1	User Imports Bank Statements	Bank Master - Documentation	5	4	1	2019-08-02	2019-08-02
		Bank Master - DB Changes	1	0.9	0.1	2019-08-12	2019-08-12
		Bank Master - Development	4	4	0	2019-08-12	2019-08-12
		Bank Master – Testing	2	2	0	2019-08-12	2019-08-12
		Bank Master - Multi Currency Preparation	3	1	2	2019-08-12	2019-08-12
		Bank Master - Bank Account Identifier	1	1	0	2019-08-12	2019-08-12
		Import Template - Documentation	4	3.5	0.5	2019-08-12	2019-08-12
		Import Template - Database Changes	1	0	1	2019-08-12	2019-08-12
		Import Template - Development	6	0	6	2019-08-12	2019-08-12
		Import Template - Testing	2	0	2	2019-08-12	2019-08-12

Table 8-8: Sprint backlog for the cashbook system (continued)

		Open / Close Period – Documentation	12	12	0	2019-08-15	2019-08-29
		Open / Close Period - DB Changes	4	4	0	2019-08-15	2019-09-03
		Open / Close Period - Development	10	5.12	4.88	Change request	2019-09-06
		Open/Close Period - Testing	4	0.64	3.36	Change request	2019-09-06
		Bank Statement Import – Documentation	4	4	0	2019-08-19	2019-09-06
		Bank Statement Import - Database Changes	1	1	0	2019-08-21	2019-09-09
		Bank Statement Import - Development	4	3.8	0.2	2019-08-21	2019-09-10
		Bank Statement Import - Testing	1	0	1	2019-08-22	2019-09-11
		Bank Statement Rollback - Documentation	20	8	12	2019-08-22	2019-09-19
		Bank Statement Rollback - Database Changes	1	4	-3	2019-08-23	2019-09-20
		Bank Statement Rollback - Development	4	3.8	0.2	Change request	2019-09-23
		Bank Statement Rollback – Testing	1	0.95	0.05	Change request	2019-09-24
		Allowed Transaction Type per bank Account	2	1.5	0.5	Change request	2019-09-25
		Allowed Payment Type per Bank Account	1	1	0	Change request	2019-09-25
		Allowed Tax Codes per Bank Account	1	1	0	Change request	2019-09-26
		Multi-Currency	16	16	0	Change request	2019-09-27
		Auto Approval - configured transactions	4	4	0	Change request	2019-09-30
		Journal approval level change (Capturer / Approver)	8	8	0	Change request	2019-10-01
2	View / Process Cash Book	Documentation	12	0	12	2019-08-27	2019-10-02
		DB Changes	4	4	0	2019-08-29	2019-10-03
		Develop Cashbook view	12	16	-4	2019-09-04	2019-10-09
		Develop Cashbook Transaction capture	12	16	-4	2019-09-06	2019-10-10
		Develop Transaction Reversal	4	4	0	2019-09-09	2019-10-11
		Develop Transaction Adjustment	4	2	2	2019-09-09	2019-10-11
		Testing	4	6	-2	2019-09-10	2019-10-11
3	View / Process Bank Statement Transactions	Documentation	20	0	20	2019-09-13	2019-10-14
		DB Changes	4	3	1	2019-09-16	2019-10-14
		Develop Transaction view	12	16	-4	2019-09-19	2019-10-15
		Develop Transaction Adjustment	4	2	2	2019-09-20	2019-10-15
		Develop Transaction Allocation	4	4	0	2019-09-23	2019-10-15
		Lookup / Validation - GL Accounts	8	8	0	Change request	2019-10-17
4	Bank Reconciliation	Documentation	12	0	12	2019-09-25	2019-10-21

Table 8-8: Sprint backlog for the cashbook system (continued)

		DB Changes	4	1.5	2.5	2019-09-26	2019-10-21
		Develop Bank Reconciliation Processing (Auto Match)	16	8	8	2019-09-30	2019-10-23
		Develop Quick Match (Manual Reconciliation)	12	10	2	2019-10-02	2019-10-25
		Develop Bank Recon Result View	12	4	8	2019-10-04	2019-10-28
		Develop Transaction Adjustment	4	0	4	2019-10-04	2019-10-28
		Testing	4	0	4	2019-10-05	2019-10-29
		Receipt Export / Import / Reconciliation	6	6	0	Change request	2019-10-29
		Auto Populate / Recon from bank statement	6	6	0	Change request	2019-10-30
		Remittance Advice	4	1	3	Change request	2019-10-30
5	Interfaces	UIC - Payment Interface	8	8	0	2019-10-06	2019-10-31
		Journal Interface	8	6	2	2019-10-07	2019-11-04
		EFT Payments	8	6	2	2019-10-08	2019-11-05
		Testing	4	4	0	2019-10-09	2019-11-05

Rework post implementation

No rework was required or reported for a two-week period after the project was delivered and taken into production.

Modifications during the project due to scope changes

None to date.

Implemented and in use

- The project was implemented successfully and is currently in use.

Project methodology followed

- The PMM and ASDM integration matrix as per Table 8-4 were applied to manage the project and to develop and implement the solution.
- Requirements were gathered, and development was based on the sprints and iterations as defined by the product and sprint backlogs of the project.
- The customer was continuously involved in all iterations and sprints throughout the project.
- The product and sprint backlogs made it easy to estimate timelines which could not be done previously.
- Compiling sprints enabled the team to plan development better and more constructively.

Comparison to historical projects

Based on the project classification, size and priority, the cashbook project can be compared to the historical projects which were categorised as “big” projects. Table 8.9 compares the project results of these projects.

Table 8.9: Project results comparison of small projects

Project Results	Overseas travel control system	Capex Application System	Fleet management system
Project classification	Big	Big	Big
Priority	High	High	High
Customer Satisfaction	9.25	6.1	5.5
Scope Creep (Modifications)	0	21	33
Timeline	0.87 months late	2.5 months late	6.7 months late
Rework (Calls)	0	17	11

The cashbook project required no scope changes and no rework during the execution of the project. Although the project was not delivered within the projected timeline, the overrun was much shorter compared to that of the historical projects. Consequently, the project produced a better customer experience that resulted in a good customer satisfaction score.

8.6 Specifying learning

During the project process, various lessons were learned and identified by the project team. The following limitations were identified during the project life cycle:

- The team found the process of not making changes during a sprint as a limitation.
- The timelines were impacted due to constantly changing requirements.
- Sprint timelines had to be moved due to the development team not being solely dedicated to the project, as they had to focus on other system support and priorities as well.
- Documentation was limited due to time constraints. It was also not updated during the project due to time constraints, although the requirements that changed continuously were logged and managed.
- Daily stand-up meetings could not always take place due to the unavailability of various parties.

In this chapter, the results of the cashbook project that was executed according to the ASDM and PMM integration matrix explained in Chapter 3 was explained. Action research was applied by following the action research cycle of diagnosing, action planning, action taking, evaluating

and specify learning. The final project results based on the evaluation criteria explained in Chapter 5 were compared to the historical projects that were classified as big projects.

In the next chapter, the overall project results will be discussed and the study will be concluded.

CHAPTER 9: RESULTS DISCUSSION AND CONCLUSION

9.1 Introduction

The aim in this chapter is to discuss and conclude the results from the three projects that were executed according to the action research cycles discussed in Chapters 6-8. The results obtained for each project will be analysed and discussed. An attempt will also be made to answer the research problem by evaluating if the study will assist organisations to execute software development projects effectively by following a hybrid methodology between ASDMs and PMMs according to its organisational environment. Lastly, it will be evaluated if the following secondary objectives have been achieved:

- Study literature on the execution of software development projects within companies;
- Study different ASDMs that are most commonly used by organisations;
- Study PMMs that are most commonly used by organisations;
- Study literature on previous studies where ASDMs and PMMs were applied in conjunction to deliver software projects;
- Gather information on previous software development projects for a particular organisation and evaluate the execution against predefined success criteria in terms of ASDMs and PMMs that were used;
- Develop an ASDM and PMM framework that can be applied to various organisations and industries;
- Identify an applicable ASDM and PMM framework for a particular organisation and apply the integration on software development projects; and
- Evaluate the effectiveness of the applied ASDM and PMM integration framework.

9.2 Research objectives

The findings of the study will be evaluated and it will be determined if the aims and objectives explained in section 1.5 have been achieved. The secondary objectives will first be discussed followed by the main objective of the study.

9.2.1 Study literature on the execution of software development projects within companies

The outcome of this objective was discussed in section 1.3 where previous studies on the integration of ASDMs with PMMs were investigated and discussed, including studies on agile and traditional project management. The conclusion was that research on the integration of agile software development methodologies with project management was limited and that previous studies were more focused on comparisons and the adoption of agile methodologies

rather than the integration between these methodologies and project management methodologies. Previous studies further showed that although a considerable amount of research has been done on the topics of ASDMs and PMMs, these studies were carried out in isolation. It is therefore evident that research on how these methodologies can be used in conjunction with one another is still in the early stages.

9.2.2 Study different ASDMs that are most commonly used by organisations

The outcome of this objective was discussed in Chapter 3 where the applications of the identified ASDMs, namely Scrum, Extreme Programming (XP), Dynamic Systems Development Method (DSDM) and Feature-driven Development (FDD) were explained by means of examples and case studies. To get a better understanding of the selected ASDMs, literature on these methodologies was discussed in detail, including the related advantages and disadvantages.

9.2.3 Study PMMs that are most commonly used by organisations

The outcome of this objective was discussed in Chapter 2 where the applications of the identified PMMs, namely Project Management Body of Knowledge (PMBOK), PRoject IN Controlled Environment (PRINCE2) and Agile Project Management (APM) were explained by means of examples and case studies. To get a better understanding of the execution of software development projects, literature on the identified PMMs was discussed in detail, including the advantages and disadvantages of these methodologies.

9.2.4 Study literature on previous studies where ASDMs and PMMs were applied in conjunction to deliver software projects

In section 1.3, the outcome and results of this objective where previous studies on studies on agile and traditional project management were listed and explained were discussed. The conclusion was that previous studies on ASDM and PMM integrations are limited and that neither one of these studies has been applied or tested in practice. Most studies where ASDMs and PMMs were compared mainly concluded that they are complementary and can improve project success.

9.2.5 Gather information on previous software development projects for a particular organisation and evaluate the execution against a predefined success criteria in terms of ASDMs and PMMs that were used

In section 5.3, six historical projects executed in a particular organisation were described. The success of these projects was evaluated against evaluation criteria based on best practice and

on what was viewed as important factors for the organisation. In Table 9.1, the results of the evaluation are summarised and it is concluded that these projects were not successful.

Table 9-1: Results of previously executed projects

Project	Project classification	Priority	Customer Satisfaction	Scope Creep (Modifications)	Timeline	Rework (Calls)
Capex Application System	Big project	High	6.1	33	2.5 months late	17
Fleet Management System	Big project	High	5.5	21	6.7 months late	11
Dealsheet Management System (DMS)	Medium project	Medium	5.3	23	15.1 months late	16
Frontend – Integrated Business Planning	Medium project	Medium	N/A	67	Not complete	N/A
Mobility Management	Small project	Low	5.5	7	3.9 months late	5
Product Configuration Management	Small project	Low	5.4	8	3 months late	5

9.2.6 Develop an ASDM and PMM matrix that can be applied to various organisations and industries

The outcome of this objective was described in section 3.3 where the identified ASDMs and PMM, namely PMBOK, were integrated and compiled in the form of a matrix. The result was illustrated in Table 3.4 which was defined as the base for the hybrid ASDM that was be applied to three projects for the specific organisation.

9.2.7 Identify an applicable ASDM and PMM framework for a particular organisation and apply the integration on software development projects

The hybrid ASDM as per Table 3.4 was used as a framework from which project teams selected specific components, based on the project type and characteristics of the organisation. Three different ASDM combinations, based on the framework, were selected and applied to each project type respectively.

9.2.8 Evaluate the effectiveness of the applied ASDM and PMM integration framework

Three software development projects of different project sizes were selected from a specific organisation and executed in sequence following action research cycles. The projects were executed following on one another with the principle of addressing lessons learned in the preceding project. This process assisted with the identification of hybrid ASDMs related to specific project classifications. During this process, the hybrid ASDMs were also refined according to the characteristics of the organisation and the tasks that worked well to effectively execute software development projects.

The evaluation and specifying learning phases of the action research cycles described and evaluated the results of the applied ASDMs in detail. The outcome of each project was again evaluated against the success criteria as described in sections 7.3 and 8.3 to determine the effectiveness of the applied ASDM. This involved the incorporation of changes during a sprint compared to only making changes at the beginning of a new sprint. In addition, the daily stand-up meetings were changed to weekly stand-up meetings and sometimes to twice per week, depending on when the team had to discuss changes with the customer. The level of documentation and quality thereof also improved during these processes.

Chapters 6, 7 and 8 described the results and outcome of each of these projects and highlighted improvements in scope management, rework, and customer satisfaction. All three projects resulted in no scope changes and no rework was required or reported for the period after the project was delivered and taken into production. Although these projects were not delivered within the projected timelines, the overruns were much shorter compared to the historical projects and resulted in greater customer satisfaction. Due to closer customer involvement throughout these projects, the customers' expectations could be managed more closely which contributed further to the improvement in customer satisfaction. Overall, all three projects were classified as successful.

9.3 Discussion of results

In the following sections, the results obtained from the three projects that were executed according to the proposed ASDM and PMM integration framework are summarised.

9.3.1 Project 1: Small project

This project was the first project that was done by following a hybrid ASDM integrated with PMBOK selected by the organisation's project team. The project was classified as a small project, based on the prioritisation evaluation and the ASDM category selections were based on

the organisational characteristics. The project was executed following the hybrid ASDM and according to the action research phases of diagnosing, action planning, action taking, evaluating and specifying leaning.

The hybrid ASDM worked well and the tasks that were less successful were identified during the project development process. These items were evaluated in the specifying learning phase and approached differently in the next project. This improved the ASDM in the next action research cycle.

Interviews were conducted with the project team members, comprising the project team and customers, who were asked to rate the overall project on a scale from one to ten, where the score of one was poor and ten excellent. This included the overall experience they had throughout the project, including the methodology followed, quality of the solution, conformance with requirements, team communication, team collaboration, product defects, product rework and project delivery. The overall scores were very good with an average rating of 9.25 out of ten. This is an indication that the ASDM was successful, although some improvement areas were identified. Compared to historical projects, the quality of the solution delivered to the customer was very good, as no rework was required after the project was implemented. Due to the close collaboration with the customers, they were continuously updated regarding the project progress, including their continuous requirement changes. This ensured that the final product was delivered according to the customer requirements in terms of functionality, although the estimated timelines were not met. As per the organisational characteristics, the quality of the solution was more important to the customer than the timelines which is why they were satisfied with the end result.

One concern raised by the project team was that timelines could not be managed accurately due to the constantly changing requirements. Although this was a concern for the technical team, the customer was comfortable postponing the timelines, as they knew that it was mainly due to their requirement changes. This can be problematic from a resource management perspective, especially when resources are not dedicated to projects which was the case on this project. Another area that needed more attention was the compilation of documentation. Documentation is mostly required for training purposes from a technical and business perspective. Because the customers were closely involved throughout the project, no detail training was required which limited the need for documentation.

9.3.2 Project 2: Medium project

This project was classified as a medium priority, based on the prioritisation evaluation. The lessons learned in the previous project were applied in this project and generated better results according to the feedback received from the project team.

The results experienced during this project were similar to those of the first project with the exception of the changes that were accommodated based on the first lessons learned. This project scored an overall rating of 8.85 out of ten. As was the case with the first project, the results showed that the product quality was very good compared to the timelines. This project delivered a solution that was seven days behind the planned schedule.

9.3.3 Project 3: Big project

Based on the prioritisation evaluation, this project was classified as a high priority. The lessons learned in the previous two projects were applied in this project thereby generating very good results according to the feedback received from the project team.

The results experienced during this project were similar to those of the first two projects with the exception of the changes that were accommodated based on the previous lessons learned. This project scored an overall rating of 9.25 out of ten which is the same result as that of the first project. As was the case with the first project, the results showed that the product quality when compared to the timelines was very good. This project delivered a solution that was seven days behind the planned schedule. Based on the complexity, priority and size of the project, the results compared to those of historical projects are much better, as the project evaluation changes from unsuccessful to successful.

The contribution of this study is discussed in the next section.

9.4 Contribution of this study

This study and research contributed to the following areas:

- This study involved a practical analysis and application of a hybrid ASDM where organisational factors and characteristics were taken in consideration.
- A practical combination of ASDMs and PMMs was applied and tested to determine the effectiveness of a hybrid ASDM and PMM combination.
- The outcome of the projects that were executed according to the proposed ASDM were more successful compared to previous projects that were executed without following any methodology related to software development and project management.

- The proposed ASDM generated software development solutions of much better quality and functionality, based on development rework and customer feedback.
- Interviews conducted with the project teams and customers were very positive and the proposed ASDM was adopted by the organisation for future software development projects.

An ASDM framework was created where selected ASDMs and a PMM was integrated which can be used by companies to manage and execute software development projects. This framework can also be adapted by selecting only certain components based on the specific characteristics of an organisation. The framework can be used as a template or guideline for executing software development projects. The ASDM consists of the PMBOK knowledge areas which are integrated with four popular ASDMs. This framework discussed in Chapter 3, section 3.3 was used as the basis for the execution of three software development projects and was adapted according to lessons learned from previous projects and knowledge gained from this study. In Table 9.2, the final framework that can be used by organisations to manage and implement software development projects is presented.

Table 9-2: Integration of PMBOK and agile software development methodologies

PMBOK	Software development methodologies			
	SCRUM	**XP**	**DSDM**	**FDD**
Project Integration Management				
• Develop project Charter • Develop project management Plan • Direct and Manage project Work • Monitor and Control project Work • Integrated Change Control • Close project	• Software requirements are formulated and prioritised by the product owner as stories • Sprint planning events • Strong change management procedure with product and sprint backlog • Retrospective and Next Sprint Planning	• User stories or requirements are created by customers and the development team • Stories are converted into iterations • Programming team and business prepares the plan, time, and costs of carrying out the iterations • Coding takes priority over all tasks • Continuous design-coding-testing-listening cycles	• Feasibility and business study as evaluation and planning mechanisms • Prioritisation is essential • Baseline of requirements and functionality is at a high level • Relies heavily on techniques to develop an application • Uses prototypes • Good for projects with tight time constraints • Testing integrated throughout the lifecycle	• Developing an overall model • Building a feature list • Planning and designing by feature • Prioritise based on features
Project Scope Management				
• Scope Planning • Collect Requirements • Scope Definition • Create Work Breakdown Structure • Scope Validation • Scope Control	• Product Backlog Creation • Sprint Planning and Sprint Backlog Creation • User stories are defined and prioritised • Review continuously to improve development process	• Customer defines user stories • Release planning and prioritisation by customer • Continuous feedback from customer • Small releases	• Requirements list as per the business study defines the scope • Focusing on the customer needs • Active user involvement • Frequent releases	• Developing an overall model • Develop feature list

Table 9-2: Integration of PMBOK and agile software development methodologies (continued)

Project Time Management				
• Plan Schedule • Define Activities • Sequence Activities • Estimate Activity Resources • Estimate Activity Durations • Develop Schedule • Control Schedule	• Plan sprint durations • Sprint lasts ± 2 weeks • Task board for tracking progress • Daily stand-up meetings to discuss progress	• Iterations time planning • Release planning	• Time and resource are not adjusted • Deliver the projects as early as possible without affecting the quality • Good control over quality of product, cost and time • Frequent delivery of products	• Plan by feature • Plan order of features based on dependencies • Determine development sequence
Project Cost Management				
• Cost Estimating, • Cost Budgeting, and • Cost Control project	• Scrum team defines cost estimates related to stories during sprint planning	• Programming team prepares the costs of carrying out iterations • Double development cost • Reduce cost via small iterations	• Deals effectively with project cost e • Gives priority to time, quality, and cost • Costly to implement	• No primary focus
Project Quality Management				
• Quality Planning • Perform Quality Assurance • Perform Quality Control	• Quality goals sustained during sprints • Retrospective and Next Sprint Planning to improve development process • Regular scrum meetings • Customer inputs for further improvements during iterations	• Pair programming for higher quality code • Extreme testing in development phase • Continuous customer involvement and testing • Simple design • Coding standards	• Good control over quality of product, risk, cost and time • Delivering the projects without affecting the quality of the project • Focus on frequent releases rather than quality	• Continuous testing of the code • Coding standards • Measuring audits and metrics in the code
Project Human Resource Management				
• Human Resource Planning • Acquire project Team • Develop project Team • Manage project Team	• Scrum teams self-organised and cross functional • Daily scrum meetings • Small teams	• Pair programming • Small teams • Customer part of development team • Team-oriented methodology • Avoid working overtime	• Teams empowered to make decisions • Facilitated Workshops	• Client and development team develop overall model • Highly skilled teams • Feature teams
• **Project Communications Management**				
• Communications Planning • Information Distribution • Performance Reporting • Manage Stakeholders	• Permanent communication and close cooperation between the stakeholders at each step • Daily scrum Meetings • Group is self-organising and collaboratively managed	• Continual communication with the customer and amongst the team • Collaborative workspaces • Co-location of development and business space • Paired development • Frequently changing pair partners • Short stand-up meetings • Unit tests and verbal communication	• Facilitated Workshops • Environment ideal for the formation of ideas • Accurate and quick decision making	• Continuous review meetings • Feature teams deliver features
Project Risk Management				

Table 9-2: Integration of PMBOK and agile software development methodologies (continued)

• Risk Management Planning • Risk Identification • Qualitative Risk Analysis • Quantitative Risk Analysis • Risk Response Planning • Risk Monitoring and Control	• Iterative and incremental approach to control risk • Sprints limit risk to one calendar month of cost • project risks seen more quickly • Decisions to control risk are made based on the perceived state of the artefacts • Meetings to review risks	• Identify risks early in the project • Frequent releases • Unit testing • Customer part of development team • Collective ownership reduces risks of programmer dependency.	• Changes are reversible during development • Feasibility study identifies risks involved • Guidelines for risk management • Reduce risk through incrementally delivering the solution	• Guidance of skilled, experienced developers • Feature planning take risks into account
Project Procurement Management				
• Plan Purchases and Acquisitions • Plan Contracting • Request Seller Responses • Select Sellers • Contract Administration • Contract Closure	• No information	• No information	• No information	• No information

The limitations documented for this study are subsequently discussed.

9.5 Limitations of this study

The limitations of this study are as follows:

- Even though three projects of different types were executed based on the proposed hybrid ASDM following action research cycles, it is recommended that more projects should be executed according to the ASDM.
- The projects executed according to the ASDM were classified into small, medium and large projects for a particular organisation. Other types of project based on different classifications may be selected to test the effectiveness of the proposed ASDM framework.
- The ASDM framework was only tested on one organisation in a particular industry. Testing it on other types of organisation and industry will deliver a more accurate indication of the effectiveness of the framework.

From this study, a few ideas regarding future work emerged and are discussed next

9.6 Future work

Future work and research which can be considered include:

The hybrid ASDM proposed in this study is a framework that can be used by various organisations and industries for software development projects. This framework was only applied to one organisation and it is therefore recommended that the framework be applied, amended and improved to make it more acceptable, implementable and applicable in practice.

There is no one ASDM, PMM or combination of ASDM and PMM that can work for all organisations and all industries, although this framework can be adapted to different sizes and different types of project. Future work is required to determine if additional components or ASDMs should be added to the framework. This would need to be evaluated in different organisations, industries and business environments to investigate and improve the effectiveness even further.

9.7 Conclusion

In this chapter, the aim was to present the findings gained from the action research and the analysed results. The secondary objectives set for this study have been achieved. The application and effectiveness of a hybrid ASDM was also discussed and summarised. The integration of ASDMs with a PMM was discussed and a basic framework was developed which can be used as a guideline by companies and organisations.

Results obtained by applying the hybrid ASDM framework were discussed and the effectiveness was evaluated. The components of this framework can be adapted in future research as a limitation of this study was that the hybrid ASDM framework could only be tested on one organisation.

This study contributed to improving the knowledge regarding the application of hybrid ASDMs integrated with a PMM and improved the success rate of completing software development projects successfully for a specific organisation. This study can serve as a basis for future work and research by adapting and applying the ASDM framework to other organisations and industries.

BIBLIOGRAPHY

Abdelghany, A.S., Darwish, N.R. and Hefny, H.A. 2017. Towards a hybrid approach for software project management using ontology alignment. *International Journal of Computer Applications,* 168(6):12-19.

Aguanno, K. 2004. Managing agile projects. 1st ed. Ontario: Multi-Media Publications Inc.

Alami, A. 2016. Why do information technology projects fail? *Procedia Computer Science,* 100(1):62-71.

Alexander, M. 2017. How to pick the best project management methodology for success. https://www.cio.com/article/2950579/methodology-frameworks/how-to-pick-a-project-management-methodology.html Date of access: 18 Mar. 2018.

Alqudah, M. and Razali, R. 2017. Key factors for selecting an agile method: A systematic literature review. *International Journal on Advanced Science Engineering Information Technology*, 7(2):526-536.

Altameem, A.E. 2015. Impact of agile methodology on software development. *Computer and Information Science,* 8(2):9-14.

Al-zewairi, M., Biltawi, M., Etaiwi, W. and Shaout, A. 2017. Agile software development methodologies: Survey of Surveys. *Journal of Computer and Communications*, 5:74-97.

Ahmed, J.U. 2009. Action research: A new Look. *KASBIT Business Journal*, 2(1&2):19-32.

Anand, R. and Dinakaran, M. 2016. Popular agile methods in software development: Review and analysis. *International Journal of Applied Engineering Research*, 11(5):3433-3437.

Anwer, F., Aftab, S., Waheed, U. and Muhammad, S.S. 2017. Agile software development Models TDD, FDD, DSDM, and Crystal Methods: A Survey. *International Journal of Multidisciplinary Sciences and Engineering*, 8(2):1-10.

APMG-Australasia. 2005. Project Management and cultural change at the University of Western Australia Library. https://www.prince2.com/zar/secure-downloads/university-of-

western-australia-project-management-cultural-change-150b9f6a8c227953c1f6caa5b7145573.pdf Date of access: 24 Sep. 2019.

APMG-Australasia. 2008. PRINCE2® help to deliver excellence at Vocalink. https://www.prince2.com/zar/secure-downloads/prince2-helps-to-deliver-excellence-at-vocalink-ffda78cc28d3630f2cb04e6592ec2838.pdf Date of access: 24 Sep. 2019.

Arcidiacono, G. 2017. Comparative research about high failure rate of IT projects and opportunities to improve. *PM World Journal*, 6(2):1-10.

Aubry, M., Müller, R., Hobbs, B., Blomquist, T. 2010. Project management offices in transition. *International Journal of project. Management*, 28(8):766-778.

Avison, D.E. and Fitzgerald, G. 2003. Information systems development: Methodologies, techniques and tools. 3rd ed. London: McGraw-Hill Publishing Company.

Axelos. 2017. Managing successful projects with PRINCE2. Norwich: The Stationery Office. Pages - https://www.scribd.com/document/366985496/managing-successful-projects-with-prince2-2017-6th-edition Date of access: 16 Mar. 2019.

Azanha, A., A.R.T.T. Argoud and de Camargo Junior, J.B. 2017. Agile project management with scum: A case study of a Brazilian pharmaceutical company IT project. https://www.researchgate.net/publication/312536118_Agile_project_management_with_Scrum_A_case_study_of_a_Brazilian_pharmaceutical_company_IT_project Date of access: 30 Sep. 2019.

Badiru, A.B. and Osisanya, S.O. 2013. Project management for the oil and gas industry: A World System Approach. Broken Sound Parkway: CRC Press.

Bahrudin, I.A., Abdullah, M.E., Hanifa, R.M., Kasim, N. and Roslan, R. 2014. Challenges of gathering user requirement in eXtreme programming project: A Case Study of Highway Construction Monitoring System. *Key Engineering Materials,* 594-595:511-515.

Baskerville, R.L. 1999. Investigating information systems with action research. *Communications of the AIS*, 2(19).

Baskerville, R. and Wood-Harper, T. 1996. A Critical perspective on action research as a method for information system research. *Journal of Information Technology*, 11(3):235-246.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J. and Thomas, D. 2001. Manifesto for agile software development. http://agilemanifesto.org/ Date of access: 18 Mar. 2018.

Beck, K. 2005. Extreme programming explained – Embrace Change. 2nd ed. Boston: Addison-Wesley.

Bredo, E., and Feinberg, W. 1982. The interpretive approach to social and educational research. Philadelphia: Temple University Press.

Brinkkemper, S. 1996. Method engineering: engineering of information system development methods and tools. *Information and software technology*, 38:275-280.

Brolsma, D. and Kouwenhoven, M. 2017. Prince2 2017 edition foundation courseware – English. 1st ed. Zaltbommel: Van Haren Publishing.

Bruegge, B. and Du Toit, A.H. 2010. Object-oriented software engineering: using UML, patterns, and Java. 3rd ed. Upper Saddle River, NJ: Prentice-Hall.

Bunsiri, T. and Kumprom, T. 2016. Benefits of agile project management. *Apheit Journal*, 5(1):23-29.

Carton, F.L., Adam, F. and Sammon, D. 2008. Project management: a case study of a successful ERP implementation. https://www.researchgate.net/publication/241700177_Project_management_a_case_study_of_a_successful_ERP_implementation Date of access: 23 Sep. 2019.

Chapram, S.B. 2018. An appraisal of agile DSDM approach. *International Journal of Advance Research, Ideas and Innovations in Technology,* 4(3):512-515.

Charvat, J. 2003. Project management methodologies: Selecting, Implementing, and Supporting methodologies and Processes for projects. Hoboken, NJ: John Wiley and Sons, Inc.

Chaves, M.S., De Araujo, C.C.S., Teixeita, L.R., Rosa, D.V., Gloria, I. and Nogueira, C.D. 2015. A new approach to managing Lessons Learned in PMBoK process groups: the Ballistic 2.0 Model. *International Journal of Information Systems and project management,* 4(1):27-45.

Check, J., and Schutt, R. K. 2012. Research methods in education. Thousand Oaks, CA: Sage Publications.

Chin, C. 2012. Development of a project management methodology for use in a university-industry collaborative research environment. University of Nottingham. (Thesis – PhD).

Chin, M.M. and Spowage, A.C. 2012. Project management methodologies: A comparative analysis. *Journal for the Advancement of Performance Information and Value,* 4(1):106-118.

Cho, J. 2008. Issues and challenges of agile software development with Scrum. *Issues in information systems,* 9(2):188-195.

Ciric, D. and Gracanin, D. 2017. Agile project management beyond software industry. https://www.iim.ftn.uns.ac.rs/is17/papers/60.pdf Date of access: 22 Mar. 2019.

Cleland, D. I. 2004. Field guide to project management. 2nd ed. Hoboken: John Wiley.

Coery, S. M. 1953. Action research to improve school practice. New York: Teachers College, Columbia University.

Couglan, M. 2009. Interviewing in qualitative research. *International Journal of Therapy and Rehabilitation,* 16(6):309-314.

Daniel, A. and Rwakatiwana, R. 2009. Application of traditional and agile project management in consulting firms. Sweden: Umea University. (Masters - Thesis).

Denzin, N. K. and Lincoln, Y. S. 2011. The SAGE handbook of qualitative research. 4th ed. Thousand Oaks, CA: Sage.

Despa, M. L. 2014. Comparative study on software development methodologies. *Database Systems Journal*, 5(3):37-56.

De Jaeger, J. 2019. PMBOK (PMI). Retrieved from https://www.12manage.com/methods_pmi_pmbok.html Date of access: 26 Mar. 2019.

Dingsoyr, T., Nerur, S., Balijepally, V. and Brede Moe, N. 2012. A decade of agile methodologies: Towards explaining agile software development. *The Journal of Systems and Software*, 85(6):1213-1221.

Drummond, J.S. and Themessl-Huber, M. 2007. The cyclical process of action research: The contribution of gilles deleuze. *Action Research*, 5(4):430-448.

Ebad, S.A. 2016. Influencing Factors for IT Software project Failures in Developing Countries — A Critical Literature Survey. *Journal of Software*, 11(11):1145-1153.

Edeki C. 2015. Agile software development methodology. *European Journal of Research in Medical Sciences,* 3(1):22-27.

Edureka. 2019. PMBOK® Guide 6th Edition – What's New in Edition 6? Retrieved from https://www.edureka.co/blog/pmbok-6th-edition-guide/ Date of access: 17 Jan. 2019.

Elden, M. and Chisholm, R. 1993. Emerging varieties of action research. *Human Relations*, 46(2):121-141.

Errihani, S., Elfezazi, S. and Benhinda, K. 2015. Adaptation and application of project management according to the PMBOK to a set of it projects in a public body. *Journal of Theoretical and Applied Information Technology,* 79(2):191-202.

Fahad, M., Qadri, S., Ullah, S., Husnain, M., Qaiser, R., Qureshi, S.A., Ahmed, W., Muhammad, S.S. 2017. A Comparative analysis of DXPRUM and DSDM. *IJCSNS International Journal of Computer Science and Network Security,* 17(5):259-264.

Friedrich, T., Schlauderer, S., Weidinger, J. and Raab, M. 2017. On the research paradigms and research methods employed in the BISE Journal – A ten-year update. (*In* Leimeister, J.M.; Brenner, W. (Hrsg.): Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI 2017), St. Gallen, S. 1111-1125).

Firdaus, A., Ghani, I. and Yasin, I.M. 2013. Developing secure websites using feature driven development (FDD): A Case Study. *Journal of Clean Energy Technologies,* 1(4):322-326.

Flora, H.K. and Chande, S.V. 2014. A systematic study on agile software development methodologies and practices. *International Journal of Computer Science and Information Technologies,* 5(3):3626-3627.

Gasik, S. 2015. An analysis of knowledge management in PMBOK Guide. *PM World Journal,* 4(1):1-13.

Ghosh, S., Forrest, D., Dinetta, T., Wolfe, B. and Lambert, D.C. 2015. Enhance PMBOK by comparing it with P2M, ICB, PRINCE2, APM and Scrum project management standards. *PM World Journal,* 4(9):1-75.

Gist, P. and Langley, D. 2007. Application of standard project management tools to research – A case study from a multi-national clinical trial. *The Journal of Research Administration,* 38(2):51-58.

Goncalves, E.F. Drumond, G.M. and Mexas, M.P. 2016. Evaluation of PMBOK and Scrum practices for software development in the vision of specialists. *Independent Journal of Management & Production,* 8(5):569-582.

Gonzalez-Perez, C. and Henderson-Sellers, B. 2005. Templates and resources in software development methodologies. *Journal of Object Technology,* 4(4):173-190.

Greenwood, D.J. and Levin, M. 2007. Introduction to action research. 2nd ed. London: Sage Publications, Inc.

Gulbrandsen, M and Kyvik, S. 2010. Are the concepts basic research, applied research and experimental development still useful? An empirical investigation among Norwegian academics. *Science and public policy*, 37(5):343-353.

Gustavsson. T. 2016. Benefits of agile project management in a non-software development context – A literature review. *PM World Journal*, 5(8):1-12.

Hathaway, R. S. 1995. Assumptions underlying quantitative and qualitative research: Implications for institutional research. *Research in Higher Education*, 36(5):535-562.

Hedeman, B. and Seegers, R. 2009. PRINCE2 2009 Edition. A pocket guide. Zaltbommel: Van Haren Publishing.

Highsmith, J. 2010. Agile project management: creating innovative products. 2nd ed. Boston: MA: Addison-Wesley.

Hirschheim, R. and Klein, H.K. 1989. Four paradigms of information systems development. *Communications of the ACM,* 32(10):1199-1216.

Hiwarkar, K. Doshi, A., Chinta, R. and Manjula R. 2016. Comparative analysis of agile software development methodologies - A review. *Journal of Engineering Research and Applications*, 3(6):80-85.

Hopkins, D. 1985. A Teacher's Guide to Classroom Research. Philadelphia: Open University Press.

Hult, M. and Lennung, S. 1980. Towards a definition of action research: A note and a bibliography. *Journal of Management Studies*: 17(2):241-250.

Hussain, A., Mkpojiogu E.O.C and Kamal, F.M. 2016. The Role of requirements in the success or failure of software projects. *International Review of Management and Marketing*, 6(S7):306-311.

Ilies, L., Crisan, E. and Muresan. 2010. Best practices in project management. *Review of International Comparative Management*, 1(11):43-51.

Iivari, J., Hirscheim R. and Klein, H.K. 1998. A paradigmatic analysis contrasting information systems development approaches and methodologies. *Information Systems Research*, 9(2):164-193.

Iivari, J., Hirscheim R. and Klein, H.K. 2000-2001. A dynamic framework for classifying information systems development methodologies and approaches. *Journal of Management Information Systems,* 17(3):179-218.

Iivari, J. and Maansaari, J. 1998. The usage of systems development methods: are we stuck to old practices? *Information and software technology*, 40:501-510.

ILX. 2014. PRINCE2® comes to credit risk insurance experts. https://www.prince2.com/zar/secure-downloads/prince2-comes-to-credit-risk-insurance-experts-29d4a5256100f77325842c3dab941d7c.pdf Date of access: 24 Sep. 2019.

ILX Marketing Team. 2017. The History of PRINCE2. https://www.prince2.com/zar/blog/the-history-of-prince2 Date of access: 15 Jan. 2019.

ITS Project Management Group. 2014. ITS Project management methodology. https://its.ucsc.edu/project-management/docs/pm-docs/pm-methodology-v2.1.pdf Date of access: 23 Sep. 2019.

Ingason, H.T., Gestsson, E. and Jonasson, H.I. 2013. The project Kanban wall: Combining Kanban and Scrum for coordinating software projects. *PM World Journal*, 2(8):1-23.

International project management Association. 2006. ICB - IPMA Competence baseline version 3.0

Introna, L.D. and Whitley, E.A. 1997. Against method-ism exploring the limits of method. *Information Technology & People*, 10(1):31-45.

ISO (International Organization for Standardization). 2017. Quality management – Guidelines for quality management in projects. Geneva: ISO copyright office. ISO 10006:2017(E).

Jainendrakumar, T.D. 2015. Project Scope Management in PMBOK made easy. *PM World Journal*, 4(4):1-10.

Jaferi, F., Sajadi, S.M and Alinaghian, M. 2014. Using the PMBOK guideline for investigation of project communication management in project-based organizations case study: National gas company of Lorestan province. https://www.researchgate.net/publication/289621118_Using_the_PMBOK_guideline_for_investigation_of_project_communication_management_in_project-based_organizations_case_study_National_gas_company_of_lorestan_province Date of access: 23 Sep. 2019.

Jamali, G. and Oveisi, M. 2016. A Study on project management based on PMBOK and PRINCE2. *Modern Applied Science*, 10(6):142-146.

Jaziri, R., El-Mahjoub, O. Boussaffa, A. 2018. Proposition of a hybrid methodology of project management. *American Journal of Engineering Research,* 7(4):113-127.

Joslin, R. and Müller, R. 2015. Relationships between a project management methodology and project success in different project governance contexts. *International Journal of Project Management*, 33:1377-1392.

Karaman, E. and Kurt, M. 2015. Comparison of project management methodologies: prince 2 versus PMBOK for IT projects. *Int. Journal of Applied Sciences and Engineering Research*, 4(4):572-579.

Kerlinger, F.N. 1979. Behavioural research: a conceptual approach. New York: Holt, Rinehart, and Winston.

Kerzner, H. 2009. Project management: A systems approach to planning, scheduling, and controlling. 10th ed. Hoboken: John Wiley.

Kerzner, H. 2013. Project management: A systems approach to planning, scheduling, and controlling. 11th ed. Hoboken: John Wiley.

Kemmis, S. and McTaggart, R. 1992. The action research planner. 3rd ed. Geelong, Vic.: Deakin University Press.

Kim, S. 2003. Research paradigms in organizational learning and performance: competing modes of inquiry. *Information Technology, Learning, and Performance Journal*, 21(1):9-18.

Kivunja, C. and Kuyini, A.B. 2017. Understanding and applying research paradigms in educational contexts. *International Journal of Higher Education,* 6(5):26-41.

Khosravi, A., Gandomani, T.J., Fahimian, H. 2017. Introduction of Scrum in an elite team: A case study. *Journal of Software,* 12(3):173-179.

Krauss, S. E. 2005. Research paradigms and meaning making: A primer. *The Qualitative Report*, 10(4):758-770.

Landry, J. P. and McDaniel, R. 2016. Agile preparation within a traditional project management course. *Information Systems Education Journal (ISEDJ)*, 14(6):27-33.

Larmen, C. 2003. Agile & iterative development, a manager's guide. 1st ed. Boston: Addison-Wesley.

Lather, P. 1986. Research as praxis. *Harvard Educational Review*, 56(3):257-277.

Layman, L., Williams, L. and Cunningham, L. 2004. Exploring extreme programming in context: an industrial case study. Agile Development Conference at the Agile Development Conference, Salt Lake City, UT, USA.

Leach, L.P. 2014. Critical chain project management. 3rd ed. Boston: Artech House.

Lehtinen, T.O.A, Mäntylä, M.V., Vanhanen, J., Itkonen, J. and Lassenius, C. 2014. Perceived causes of software project failures – An analysis of their relationships. *Information and Software Technology*, 56(1):623-643.

Livari, J. and Huisman, M. 2007. The relationship between organisational culture and the deployment of systems development methodologies. *MIS Quarterly*, 31(1):35-58.

Madampe, K. 2017. Successful adoption of agile project management in software Development Industry. *International Journal of Computer Science and Information Technology Research,* 5(4):27-33.

Mahajan, R. and Kaur, P. 2010. Extreme programming: Newly acclaimed agile system development process. *International journal of information technology and knowledge management*, 3(2):699-705.

Mahalakshmi, M. and Sundararajan, DR. M. 2013. Traditional SDLC vs Scrum methodology – A Comparative Study. *International Journal of Emerging Technology and Advanced Engineering,* 3(6):192-196.

Mahnic, V. and Drnovscek, S. 2005. Agile software project management with Scrum. https://www.researchgate.net/profile/Viljan_Mahnic/publication/228967959_Agile_Software_Proj

ect_Management_with_Scrum/links/09e4150644170a03ea000000/Agile-Software-Project-Management-with-Scrum.pdf Date of access: 30 Sep. 2019.

Mandal, A. and Pal, S.C. 2015. Identifying the reasons for software project failure and some of their proposed remedial through BRIDGE process models. *International Journal of Computer Sciences and Engineering*, 3(1):118-126.

Matos, S. and Lopes, E. 2013. Prince2 or PMBOK – a question of choice. *Procedia Technology,* 9(1):787-974.

May, E.L. and Zimmer, B.A. 1996. The evolutionary development model for software. http://www.hpl.hp.com/hpjournal/96aug/aug96a4.pdf Date of access: 10 Mar. 2018.

McGregor, S.L.T. and Murane, J.A. 2010. Paradigm, methodology and method: Intellectual integrity in consumer scholarship. *International Journal of Consumer Studies*, 34(4):419-427.

Monteiro de Carvalho, M. 2013. Six sigma project: The portfolio management perspective. *The Journal of Modern Project Management*, 1(2):40-49.

MSG (Management Study Guide). 2008. Project management - Definition and Important Concepts. https://www.managementstudyguide.com/project-management.htm Date of access: 7 Jan. 2019.

Myers, M. 1997. Qualitative research in information systems. https://www.qual.auckland.ac.nz/ Date of access: 4 Mar. 2019.

Myers, M.D. and Klein, K.K. 2011. A Set of Principles for Conducting Critical Research in Information Systems. *MIS Quarterly,* 35(1):17-36.

Nasir, M.H.N. and Sahibuddin, S. 2015. How the PMBOK addresses critical success factors for software projects: A multi-round Delphi study. *Journal of Software,* 10(11):1283-1300.

Nawaz, Z., Aftab, S. and Anwer, F. 2017. Simplified FDD process model. *I.J. modern education and computer science,* 9(1):53-59.

Nazir, A.K., Zafar, I. and Abbas, M. 2017. The impact of agile methodology (DSDM) on software project management. International Conference on Engineering, Computing & Information Technology (ICECIT 2017). https://pdfs.semanticscholar.org/8437/33664dc56367e0c61a6a854a84b844798c45.pdf Date of access: 8 May 2019.

Noble, J., Marshall, S., Marshall, S. and Biddle, R. 2004. Less extreme programming. https://www.academia.edu/12601926/Less_extreme_programming Date of access: 2 June 2019.

Noffke, S.E. and Stevenson, R.B. 1995. Educational action research: Becoming practically critical. New York: Teachers College Press.

Nuottila, J., Aaltonen, K. and Kujala, J. 2016. Challenges of adopting agile methods in a public organization. *International Journal of Information Systems and Project Management*, 4(3):65-85.

Oates, B. J. 2006. Researching information systems and computing. United Kingdom: Sage Publications.

Obrutsky, S. 2016. Comparison and contrast of project management methodologies PMBOK and SCRUM. Eastern Institute of Technology https://www.researchgate.net/publication/305969672_Comparison_and_contrast_of_project_management_methodologies_PMBOK_and_SCRUM Date of access: 12 Jan. 2019.

Office of Government Commerce. 2009. Managing successful projects with PRINCE2. 5th ed. Norwich: The Stationery Office.

Owen, R.L. and Koskela, L. 2006. Agile construction project management. https://pdfs.semanticscholar.org/11db/7d28d71c03fd5e4522fc250c04ba52cea611.pdf?_ga=2.115941190.1915916334.1569326897-880180839.1569132830 Date of access: 24 Sep. 2019.

Oxford University Press. 2019. Methodology. https://en.oxforddictionaries.com/definition/methodology Date of access: 7 January 2019.

Oxford University Press. 2019. Project. https://en.oxforddictionaries.com/definition/project Date of access: 5 January 2019.

Pawar, R.P. and Mahajan, K. N. 2017. Benefits and issues in managing project by PRINCE2 methodology. *International Journal of Advanced Research in Computer Science and Software Engineering,* 7(3):190-195.

Permana, P. A. G. 2015. Scrum Method Implementation in a software development project management. *International Journal of Advanced Computer Science and Applications*, 6(9):198-204.

Peters, M. and Robinson, V. 1984. The Origins and Status of Action Research. *The Journal of Applied Behavioral Science*, 20(2):113-124.

PMBOK GUIDE. 2013. A Guide to the project management body of knowledge. 5th ed. Newtown Square, PA: PMI Publishers.

Ponto, J. 2015. Understanding and evaluating survey research. https://www.researchgate.net/publication/286445115_Understanding_and_Evaluating_Survey_Research/fulltext/5ae19346a6fdcc91399ef8e6/Understanding-and-Evaluating-Survey-Research.pdf Date of access: 3 May 2020.

Popli, R. and Chauhan, N. 2011. Scrum: An agile Framework. *International Journal of Information Technology and Knowledge Management*, 4(1):147-149.

Prabhakar, G. P. 2008. What is project success: A literature review. *International Journal of Business and Management,* 3(9):3-10.

Priyanka, A. 2016. Critical Evaluation of Prince2 and agile project management methodologies for a complex project. *International Journal of Engineering and Computer Science.* 5(10):18702-18706.

Project Management Association of Japan. 2005. A Guidebook of project & Program Management for Enterprise Innovations. 7th ed. Japan: PMAJ.

Project Management Methodology Guide. 2016. PM² Project Management Methodology Guide. Open ed. Brussels: EU Publication.

Pressman, R. S. and Maxim, B.R. 2014. Software Engineering: A Practitioner's Approach. 8th ed. New York: McGraw Hill.

Putnam, H. 2012. How to Be a Sophisticated "Naive Realist"'. (*In* 'Philosophy in an Age of Science'. Cambridge, Mass: Harvard University Press.)

Rahman, S.S.M.M., Mollah, S.A., Anirban, S., Rahman, M.H., Rahman, M. Hassan, M.M., and Sharif, M.H. 2018. OSCRUM: A modified Scrum for open source software development. *International Journal of Simulation: Systems, Science & Technology*, 19(3):20.1-20.7.

Ramy, A., Krishna, T.S., Phani Kanth, CH., Phani Krishna, CH.V., Vamsi Krishna, T.V. 2011. Survey on extreme programming in software engineering. *International Journal of Computer Trends and Technology*, 2(2):21-24.

Rao, K.N., Naidu, G.K., Chakka, P. 2011. A Study of the agile software development methods, Applicability and Implications in Industry. *International Journal of Software Engineering and Its Applications*, 5(2):35-46.

Rebaiaia, M. and Vieira, D.R. 2014. Integrating PMBOK Standards, Lean and agile Methods in project management Activities. *International Journal of Computer Applications*, 88(4):40-46.

Rettig, K. D., Tam, V. C., and Yellowthunder, L. 1995. Family policy and critical science research: Facilitating change. *Journal of Family and Economic Issues*, 16(1):109-143.

Ruhe, G and Wohlin, C. 2014. Software project management in a Changing World. Berlin, Heidelberg: Springer.

Salameh, H. 2014. What, When, Why, and How? A Comparison between agile project management and Traditional project management Methods. *International Journal of Business and Management Review*, 2(5):52-74.

Sani, A. Firdaus, A., Jeong, S.R. and Ghani, I. 2013. A Review on software development Security Engineering using Dynamic System Method (DSDM). *International Journal of Computer Applications*, 69(5):37-44.

Saravanan, K. 2017. Systems development methodologies: Conceptual study. *Indian J.Sci.Res.,* 14(1):27-37.

Schach, S.R. 1997. Software Engineering with Java. New York: Irwin.

Schwaber, K. 2004. Agile project management with Scrum. Washington: Microsoft Press.

Schwaber, K. and Sutherland, J. 2017. The Definitive Guide to Scrum:
The Rules of the Game. The Scrum Guide.
https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf#zoom=100
Date of access: 8 June 2019.

Schwalbe, K. 2014. Information technology project management. 7th ed. Boston: Cengage Learning

Schwalbe, K. 2017. An Introduction to project management. 6th ed. Minneapolis: Schwalbe.

Seaman, C.B. 1999. Qualitative methods in empirical studies of software engineering. *IEEE transactions on software engineering*, 25(4):557-572.

Searle, J. R. 2015. Seeing Things as They Are; A Theory of Perception. Oxford University Press.

Serrador, P. and Pinto, J.K. 2015. Does agile work? — A quantitative analysis of agile project success. *International Journal of Project Management*, 33:1040-1051.

Seymour, T. and Hussein, S. 2014. The history of project management. *International Journal of Management & Information Systems,* 18(4):233-240.

Shankarmani, R., Pawar, R., Mantha, S.S. and Babu, V. 2012. Agile Methodology Adoption: Benefits and Constraints. *International Journal of Computer Applications,* 58(15):31-37.

Siegelaub, J.M. 2004. How PRINCE2® Can Complement the PMBOK® Guide and Your PMP. https://docplayer.net/2392732-How-prince2-can-complement-pmbok-and-your-pmp-jay-m-siegelaub-impact-strategies-llc-abstract-about-prince2.html Date of access: 26 Mar. 2019.

Software Engineering Institute. 2001. Capability maturity model integration. Pittsburgh: Carnegie Mellon University.

Spundak, M. 2014. Mixed agile/traditional project management methodology – reality or illusion? *Procedia - Social and Behavioral Sciences*, 119:939-948.

Stare, A. 2014. Agile project management in product development projects. *Procedia - Social and Behavioral Sciences,* 119(1):295-304.

Stoica, M., Mircea, M. and Ghilic-Micu, B. 2013. Software development: Agile vs. Traditional. *Informatica Economica*, 17(4):64-76.

Sudhakar, G.P. 2016. Understanding the meaning of "project success". *Binus Business Review,* 7(2):163-169.

Sutherland, J. 2001. Agile can scale: Inventing and reinventing SCRUM in five companies. *Cutter IT Journal*, 14(12):5-11.

Sutherland, J. 2012. The Scrum Papers: Nut, Bolts, and Origins of an agile Framework. Cambridge: Scrum, Inc.

Tavan, F. and Hosseinib, M. 2016. Comparison and analysis of PMBOK 2013 and ISO 21500. *Journal of Project Management,* 1(1):27-34.

Tavares, B.G., Da Silva, C.E.S. and De Souza, A.D. 2017. Risk Management in Scrum projects: A Bibliometric Study. *Journal of Communications Software and Systems,* 13(1):1-8.

Thakur, S. and Kaur, A. 2013. Role of agile methodology in software development. *International Journal of Computer Science and Mobile Computing*, 2(10):89-90.

Vaskimo, J. 2011. Project management methodologies: an invitation for Research. https://www.scribd.com/document/282407104/Vaskimo-Projektinhallinnan-metodologiat-1-pdf Date of access: 27 Mar. 2019.

Vijay, D. and Ganapathy, G. 2014. Guidelines to minimize the cost of software quality in agile Scrum process. *International Journal of Software Engineering & Applications*, 5(3):61-69.

Weinreich, R., Neumann, N., Riedel, R. and Muller, E. 2016. Scrum as method for agile project management outside of the product development area.

https://www.researchgate.net/publication/292979323_Scrum_as_Method_for_Agile_Project_Management_Outside_of_the_Product_Development_Area Date of access: 1 June 2019.

Wells, H. 2012. How effective are project management methodologies? An explorative Evaluation of their Benefits in Practice. *Project management Journal*, 43(6):43-58.

Wideman, M. 2005. PMBOK® Guide, Third Edition – Is more really better? http://www.maxwideman.com/papers/pmbok3/pmbok3.pdf Date of access: 25 Mar. 2019.

Williams, C. 2007. Research Methods. *Journal of business & economic research,* 5(3):65-72.

Wynekoop, J.L. and Russo, N.L. 1993. System Development Methodologies: Unanswered questions and the research-practice gap. (*In* DeGross, J.I., Bostrom, R.P. & Robey, D., *eds*. Proceedings of the Fourteenth International Conference on Information Systems. Orlando, FL: ACM. p.181-190).

Yadav, A. 2015. Agile: Software development model. *International Journal of Engineering and Technical Research (IJETR),* 3(3):11-17.

Yanow, D. and Schwrtz-Shea, P. 2006. Interpretation and method: Empirical research methods and the interpretive turn. 2nd ed. London: M. E. Sharpe.

Yanow, D. and Schwrtz-Shea, P. 2014. Interpretation and method: Empirical research methods and the interpretive turn. 2nd ed. London & New York: M. E. Sharpe & Routledge.

Yeong, A. 2007. The marriage proposal of PRINCE2 and PMBOK. https://www.scribd.com/document/112343434/Marriage-of-PRINCE2-and-PMBOK Date of access: 27 Mar. 2019.

Young, D.C. 2013. Software Development Methodologies. https://www.researchgate.net/profile/David_Young30/publication/255710396_Software_Development_Methodologies/links/0c960520515475fc8e000000/Software-Development-Methodologies.pdf Date of access: 28 Sep. 2019.

Ziolkowski, A. 2014. Hybrid approach in project management – Mixing capability maturity model integration with agile practices. *Social Sciences*, 3(85):64-71.