Inmon versus Kimball: The agile development of a data warehouse

I Scholtz

22357467

Dissertation submitted in partial fulfilment of the requirements for the degree *Magister Scientiae* in Computer Science at the Potchefstroom Campus of the North-West University

Supervisor: Prof HM Huisman

Co-supervisor: Ms MJ Grobler

May 2016

It all starts here ™

NORTH-WEST UNIVERSITY
YUNIBESITI YA BOKONE-BOPHIRIMA
NOORDWES-UNIVERSITEIT
®

Abstract

The aim of this study was to develop a new data warehouse development approach known as the agile data warehouse methodology using either Inmon's or Kimball's approach as the foundation along with the best practices associated with three agile system development methodologies (extreme programming, SCRUM, and feature driven development). The agile data warehouse methodology aims to provide guidelines for a development team that tackles the unique technical aspects of data warehouse development along with the issues associated with it, using agile best practice in order to develop a data warehouse in an agile environment.

In order to achieve this aim, an in-depth literature study on agile system development methodologies was undertaken. The literature study started off by identifying the reasons behind the birth of agile system development methodologies, followed by a definition and discussion of an agile system development methodology. This was followed by a brief explanation and discussion of extreme programming, SCRUM, and feature driven development.

In addition to the literature study done on agile system development methodologies, a literature study was carried out on data warehousing to reveal the history of data warehouses. The discussion on the history of data warehouses was followed by the discussion of business intelligence in order to resolve the uncertainty between the terms, data warehousing and business intelligence, and to clarify the relationship between the two terms. This was followed by the justification of a data warehouse system and the literature study on data warehouses was completed by discussing the approaches of the two giants (Inmon and Kimball) in data warehouse development.

Following the literature study, two questionnaires were developed and distributed to determine the best suited candidates for the development of the agile data warehouse methodology. The data warehouse questionnaire was targeted at revealing whether Inmon's or Kimball's approach is best suited for the agile development of a data warehouse system and revealed that Kimball's approach is best suited in this regard. The agile system development methodology questionnaire was focused on determining which of the three agile system development methodologies (extreme programming, SCRUM, and feature driven development) is best suited for each major phase of development for Kimball's approach and revealed that SCRUM is best suited for the planning and the implementation phases, feature driven development is best suited for the design phase, and extreme programming is best suited for the development phase.

Based on the knowledge gained from the data warehouse and agile system development methodology questionnaires, version 0 of the agile data warehouse methodology was developed and tested in collaboration with an industry partner by implementing the agile data warehouse (version 0) methodology in the actual development of a data warehouse system in a real-world environment. In order to evaluate the effectiveness of the implementation of the agile data warehouse (version 0) methodology, the version 0 questionnaire was developed and distributed and revealed that the agile data warehouse (version 0) methodology succeeded in the successful and agile development of a data warehouse system. Based on the implementation and evaluation of the agile data warehouse (version 0) methodology, version 1 was developed to result in an improved version of the agile data warehouse methodology. Version 1 of the agile data warehouse methodology was evaluated by developing and distributing the version 1 questionnaire, which aimed to verify the usefulness of the agile data warehouse (version 1) methodology which consisted of Kimball's approach as the foundation along with the best practices of extreme programming, SCRUM, and feature driven development.

The final verdict of the study is - **The agile data warehouse (version 1) will provide assistance and support to a project team in the successful and agile development of a data warehouse system in a development environment characterised by uncertainty and changing requirements.**

In conclusion, the agile data warehouse (version 1) methodology introduces a new innovative approach to data warehouse development to provide assistance and support to project teams for the successful and agile development of a data warehouse system. Therefore, the agile data warehouse methodology is recommended to project teams who desire to shift their data warehouse development process towards a more agile environment or for data warehouse development projects faced with uncertainty and changing requirements.

Keywords: Data warehousing (DW), system development methodology (SDM), agile system development methodology (ASDM), extreme programming (XP), SCRUM, feature driven development (FDD), agile data warehouse (ADW) methodology.

Opsomming

Die doel van hierdie studie was om 'n nuwe data pakhuis ontwikkelingsbenadering te ontwikkel bekend as die ratse data pakhuis metodologie wat gebruik maak van Inmon of Kimball se benadering as die grondslag saam met die beste tegnieke van die drie ratse ontwikkelingstelselmetodologieë (ekstreme programmering, skrum, en funksiegedrewe ontwikkeling). Die doel van die ratse data pakhuis metodologie is om riglyne vir 'n ontwikkelingspan voor te stel wat die unieke tegniese aspekte van data pakhuis ontwikkeling en ook die probleme geassosieer met die ontwikkeling van data pakhuise aan te pak deur gebruik te maak van die beste tegnieke van die drie ratse ontwikkelingstelselmetodologieë vir die ratse ontwikkeling van ’n data pakhuis.

Om die bogenoemde doel te bereik, is 'n grondige literatuurstudie oor ratse ontwikkelingstelselmetodologieë uitgevoer. Die literatuurstudie het begin met die identifikasie van die redes agter die ontstaan van ratse ontwikkelingstelselmetodologieë, gevolg deur 'n definisie en bespreking daarvan. Die is gevolg deur 'n kort verduideliking en bespreking van ekstreme programmering, skrum, en funksiegedrewe ontwikkeling.

Addisioneel tot die literatuurstudie uitgevoer op ratse ontwikkelingstelselmetodologieë, is 'n literatuurstudie oor data pakhuise gedoen wat begin met die bespreking van die geskiedenis van data pakhuise. Die bespreking oor die geskiedenis van data pakhuise is gevolg deur ’n bespreking van besigheidsintelligensie om die onsekerheid tussen die twee terme, data pakhuise en besigheidsintelligensie, en die verhouding tussen die twee terme te verduidelik. Dit is gevolg deur die regverdiging van 'n data pakhuis stelsel en die literatuurstudie oor data pakhuise sluit af met die bespreking van die benaderings van die twee reuse in data pakhuis ontwikkeling, naamlik Inmon en Kimball.

Na aanleiding van die literatuurstudie, is twee vraelyste ontwikkel en versprei om te bepaal wat die beste geskikte kandidate vir die ontwikkeling van die ratse data pakhuis metodologie is. Die data pakhuis vraelys was gemik op die identifikasie van die beste geskikte benadering tussen Inmon of Kimball vir die ontwikkeling van 'n ratse data pakhuis stelsel en het uitgewys dat Kimball se bendaring die geskikte keuse is. Die ratse ontwikkelingstelselmetodologie vraelys was daarop gemik om te bepaal watter ratse ontwikkelingstelselmetodologie van die drie (ekstreme programmering, skrum, en funksiegedrewe ontwikkeling) die beste geskik is vir elke groot fase van ontwikkeling vir Kimball se benadering en het uitgewys dat skrum die mees geskikte is vir die beplanning en implementering fases, funksiegedrewe ontwikkeling die mees geskikte is vir die ontwerp fase, en ekstreme programmering die mees geskikte is vir die ontwikkeling fase.

Gegrond op die kennis opgedoen uit die data pakhuis en ratse ontwikkelingstelselmetodologie vraelyste, is weergawe 0 van die ratse data pakhuis metodologie ontwikkel en getoets in samewerking met 'n bedryfsvennoot deur die implementering van die ratse data pakhuis (weergawe 0) metodologie in die ontwikkeling van 'n data pakhuis stelsel in 'n werklike omgewing. Deur om te bepaal hoe effektief die implementering van die ratse data pakhuis (weergawe 0) metodologie was, is die weergave 0 vraelys ontwikkel en versprei en het uitgewys dat die ratse data pakhuis (weergawe 0) methodologie suksesvol was in die ratse ontwikkeling van ’n data pakhuis . Gebaseer op die die evaluering en implementering van die ratse data pakhuis (weergawe 0) metodologie, is weergawe 1 ontwikkel en lei na n verbeterde weergawe van die ratse data pakhuis metodologie. Weergawe 1 van die ratse data pakhuis metodologie is geëvalueer by wyse van die ontwikkeling en verspreiding van die weergawe 1 vraelys wat daarop gemik was om die nut van die ratse data pakhuis (weergawe 1) metodologie wat bestaan uit Kimball se benadering as die grondslag saam met die beste tegnieke van ekstreme programmering, skrum, en funksiegedrewe ontwikkeling te verifieer.

Die finale uitspraak van die studie, naamlik – **die ratse data pakhuis (weergawe 1) metodologie sal hulp en ondersteuning aan 'n projekspan verskaf vir die suksesvolle en ratse ontwikkeling van 'n data pakhuis stelsel in 'n ontwikkelingsomgewing onderhewig aan onsekerheid en vereistes wat gereeld verander.**

Ten slotte, die ratse data pakhuis (weergawe 1) metodologie stel 'n nuwe innoverende benadering tot data pakhuis ontwikkeling om hulp en ondersteuning te verleen aan 'n projekspan vir die suksesvolle en ratse ontwikkeling van 'n data pakhuis stelsel. Daarom is die ratse data pakhuis metodologie aanbeveel vir projekspanne wat begeer om hul data pakhuis ontwikkelingsproses te skuif na 'n meer ratse omgewing of vir data pakhuis ontwikkelingsprojekte onderhewig aan onsekerheid en vereistes wat gereeld verander.

Sleutelwoorde: Data pakhuise, ontwikkelingstelselmetodologie, ratse ontwikkelingstelselmetodologie, ekstreme programmering, skrum, funksiegedrewe ontwikkeling, ratse data pakhuis metodologie.

Acknowledgements

"Yea, though I walk through the valley of the shadow of death, I will fear no evil: for thou art with me; thy rod and thy staff they comfort me." Psalm 23:4

First, and foremost, I give all the glory, praise and honour to JESUS the CHRIST for giving me the strength, perseverance and intelligence to walk through my valley of death (my dissertation) and rise victorious.

A special thanks goes out to my supervisor, Prof Magda Huisman, for her insight, patience, and encouragements that helped me to produce meaningful work that I am proud of.

I would also like to thank my parents for their support, prayers, and encouragements as well as my roommate, Stefan Smuts, for listening to my complaints and providing insight and encouragements.

Lastly I would like to thank THRIP for their financial support without which this study would not be possible.

Table of contents

Abstract ii
Abstrak iv
Acknowledgements vi
Table of contents vii
List of figures xi
List of tables xiii

Chapter 1 : Introduction

1.1 Proposed title 1
1.2 Background 1
1.2.1 Data warehouse 1
1.2.2 System development methodologies 3
1.3 Problem statement and substantiation 8
1.4 Research aim and objectives 10
1.5 Research methodology 11
1.6 Layout of the study 13

Chapter 2 : Agile system development methodologies

2.1 Introduction 15
2.2 Definition of system development methodology 16
2.3 Agile system development methodology (ASDM) 19
2.3.1 Definition of agile system development methodology 19
2.4 The three agile system development methodologies 22
2.4.1 Extreme programming (XP) 23
2.4.2 SCRUM 30
2.4.3 Feature driven development (FDD) 34
2.5 Conclusion 38

Chapter 3 : Data warehouse (DW)

3.1 Introduction 40
3.2 Data warehousing 41
3.3 Business intelligence (BI) 42
3.3.1 Components of BI 43
3.4 The connection between DW and BI 45
3.5 Justification of a DW system 48
3.6 Inmon's approach to the development of a DW 49
3.6.1 Inmon's lifecycle 50
3.6.2 Collecting requirements 59
3.6.3 Data modelling 60
3.6.4 Extract, transform and load (ETL) 63
3.6.5 Data access and deployment 65
3.7 Kimball's approach to the development of a DW 67
3.7.1 Kimball's lifecycle 68
3.7.2 Collecting requirements 74
3.7.3 Data modelling 76
3.7.4 Extract, transform and load (ETL) 81
3.7.5 Data access and deployment 86
3.8 Conclusion 87

Chapter 4 : Research methodology

4.1 Introduction 89
4.2 Research plan 90
4.3 Positivism 92
4.4 Design and create 94
4.5 Questionnaires 96
4.6 Quantitative analysis 98
4.7 Conclusion 99

Chapter 5 : Proposed framework

5.1 Introduction 101
5.2 Evaluation of Inmon versus Kimball for agile development 102
5.2.1 Inmon's approach 106

5.2.2 Kimball's approach 110
5.3 Evaluation of the three ASDMs 115
5.3.1 Planning 119
5.3.2 Design 125
5.3.3 Development 130
5.3.4 Implementation 135
5.4 Version 0 of the agile data warehouse (ADW) methodology 139
5.4.1 ADW (version 0) methodology lifecycle 141

Chapter 6 : Evaluation

6.1 Introduction 150
6.2 Implementation of version 0 of the ADW methodology 151
6.2.1 Background of the DW development project 151
6.2.2 Development process 152
6.3 Evaluation of version 0 of the ADW methodology 161
6.3.3 The version 0 questionnaire 167
6.4 Version 1 of the ADW methodology 171
6.4.1 ADW (version 1) methodology lifecycle 171
6.5 Evaluation of version 1 of the ADW methodology 176
6.6 Conclusion 182

Chapter 7 : Conclusion

7.1 Introduction 184
7.2 Evaluation of Inmon's and Kimball's approaches 185
7.3 Evaluation of XP, SCRUM, and FDD for DW development 186
7.4 Version 0 of the ADW methodology 187
7.5 Evaluation of the ADW (version 0) methodology 189
7.6 Version 1 of the ADW methodology 190
7.7 Evaluation of the ADW (version 1) methodology 193
7.8 Knowledge gained, limitations and future work 194
7.8.1 Knowledge gained 194
7.8.2 Limitations 195
7.8.3 Future work 196

References **197**
Appendix A **204**
The DW questionnaire 204
The ASDM questionnaire 205
The version 0 questionnaire 206
The version 1 questionnaire 213

List of figures

Chapter 1 : Introduction

Figure 1.1 - Overview of Chapter 1 1
Figure 1.2 - Research methodology process steps 12

Chapter 2 : Agile system development methodologies

Figure 2.1 - Overview of Chapter 2 15
Figure 2.2 - Graphical illustration of the metaphor of an SDM 17
Figure 2.3 - XP's strucutre 24
Figure 2.4 - XP's lifecycle 27
Figure 2.5 - The SCRUM lifecycle 32
Figure 2.6 - FDD processes 36
Figure 2.7 - FDD component assembly 38

Chapter 3 : Data warehouse (DW)

Figure 3.1 - Overview of Chapter 3 40
Figure 3.2 - DW/BI system 47
Figure 3.3 - Difference between the operational and DW development cycles 50
Figure 3.4 - Coporate information factory 51
Figure 3.5 - Data warehouse development - Meth 2 52
Figure 3.6 - Heuristic DSS development - Meth 4 57
Figure 3.7 - Feedback loop between the DSS analyst and the data architect 60
Figure 3.8 - Corporate data model 61
Figure 3.9 - The four contructs that make up the midlevel data model 62
Figure 3.10 - Kimball's lifecycle diagram 69
Figure 3.11 - Enterprise wide DW bus diagram 71
Figure 3.12 - Aspects of the project the business requirements affect 74
Figure 3.13 - Dimensional model for DVD(s) rented and returned by customer example 80
Figure 3.14 - Main driving force for the dimensional model 81

Chapter 4 : Research methodology

Figure 4.1 - Overview of Chapter 4..... 89
Figure 4.2 - The research plan 90

Chapter 5 : Proposed framework

Figure 5.1 - Overview of Chapter 5 part 1..... 101
Figure 5.2 - Overview of Chapter 5 part 2..... 101
Figure 5.3 - ADW (version 0) methodology lifecycle diagram 142

Chapter 6 : Evaluation

Figure 6.1 - Overview of Chapter 6..... 150
Figure 6.2 - Dimensional model created during the implementation of version 0 156
Figure 6.3 - BI application cube.....159
Figure 6.4 - Simple drag and drop query 160
Figure 6.5 - Pie chart of the query presented in Figure 6.4.....160
Figure 6.6 - ADW (version 1) methodology lifecycle 172

Chapter 7 : Conclusion

Figure 7.1 - Overview of Chapter 7..... 185
Figure 7.2 - Recap of the ADW (version 0) methodology lifecycle diagram 189
Figure 7.3 - Recap of the ADW (Version 1) methodology lifecycle diagram..... 192

List of tables

Chapter 1 : Introduction

Table 1.1 - Table of SDM definitions that have been introduced over the years 6

Chapter 2 : Agile system development methodologies

Table 2.2 - ASDM definitions that have emerged over the years 21

Chapter 5 : Proposed framework

Table 5.1 - Scoring system 103
Table 5.2 - Inmon's scores from the DW questionnaire 103
Table 5.3 - Kimball's scores from the DW questionnaire 104
Table 5.4 - Statistical evaluation of the DW questionnaire 106
Table 5.5 - Data generated by the ASDM questionnaire 117
Table 5.6 - Statistical summary of the data generated by the ASDM questionnaire 118

Chapter 6 : Evaluation

Table 6.1 - Incomplete feature list used in the implementation of version 0 155
Table 6.2 - Scoring system 162
Table 6.3 - Data generated from the version 0 questionnaire 162
Table 6.4 - Statistical analysis of the data generated by the version 0 questionnaire 165
Table 6.5 - Data generated from the version 1 questionnaire 178
Table 6.6 - Statistical analysis of the data generated by the version 1 questionnaire 179

Chapter 1: Introduction

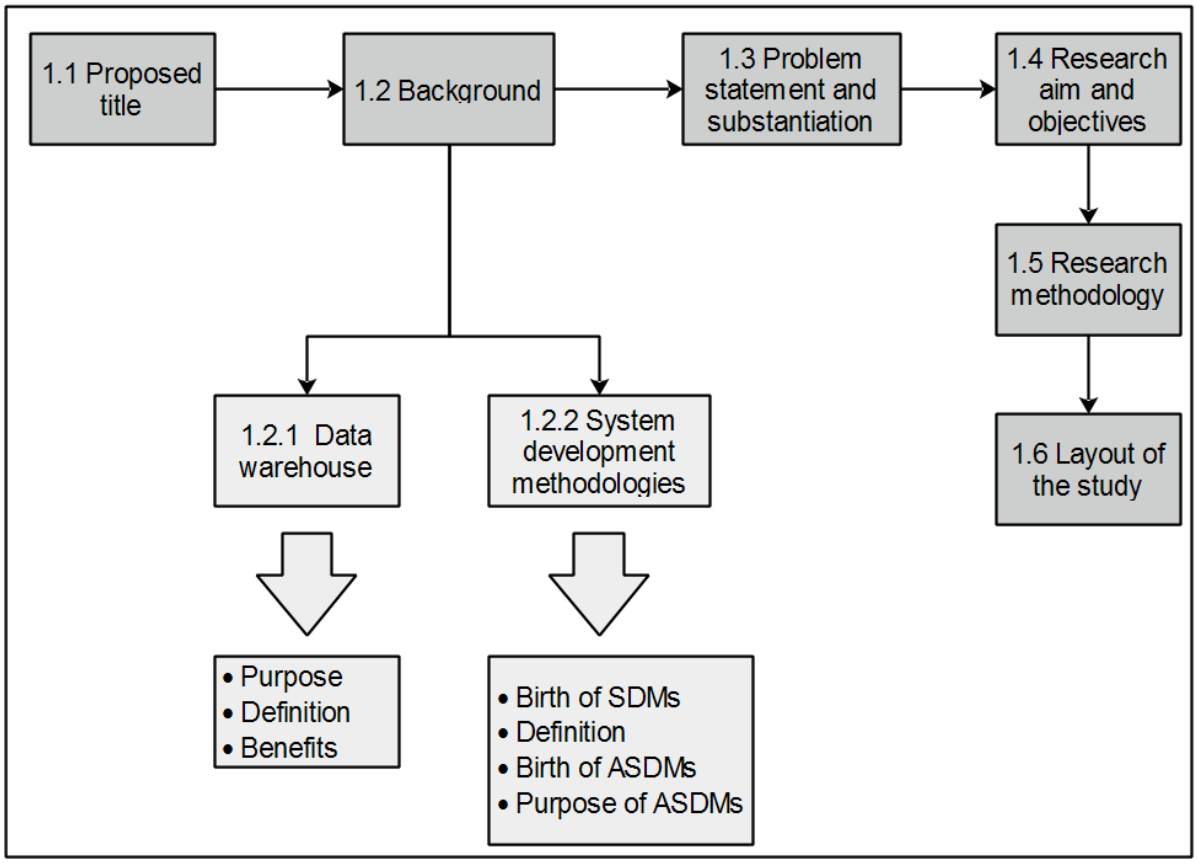


Figure 1.1 – Overview of Chapter 1

1.1 Proposed title

Inmon versus Kimball: The agile development of a data warehouse

1.2 Background

In this section a background of the relevant components of this study will be provided in order to better understand the components that will lead to the problem statement and points of interest in the study.

1.2.1 Data warehouse

In the highly competitive business environment in which the world finds itself today, good decision making is a critical success factor for business operations (Guerra & Andrews, 2013:1; Nguyen, 2011:1). The best decisions are made when all the available relevant data is taken into account and a good source for this data is a well-developed data warehouse (DW) (Inmon, 2003:xiii: Nguyen, 2011:1; Guerra & Andrews, 2013:1). Organizations typically perpetuate and utilize a number of operational data sources. These operational data sources

usually include a database and any other data repository necessary for the organization to perform day-to-day operations (Jukic, 2006:83). According to Inmon (2003:xiii), an organization will develop a DW as a separate data store from the operational data, the primary goal of which is data analysis for the support of management's decision-making process.

According to Jukic (2006:84) there are two main reasons that support the development of a DW as a separate analytical data store (Jukic, 2006:84):

- The performance of operational queries used for day-to-day activities can be drastically diminished when they compete for computing resources with analytical queries.
- Although performance might not always be an issue, it is usually impossible to design a database in such a manner that the database can be used for operational and analytical queries.

There is no standardized definition for a data warehouse (Quarles, 2002:6) and everyone has a unique opinion on what it actually is. The definitions of various authors are listed below:

Inmon (2003:31) defines it as "a subject-oriented, integrated, non-volatile, and time-variant collection of data in support of management's decisions".

The Kimball group defines a DW as "the overall process of providing information to support business decisions" (Kimball *et al*., 2008:1.4).

The data warehouse institute (TDWI) (2004:24) defines it as "a data structure that is optimized for distribution. It collects and stores integrated sets of historical data from multiple operational systems and feeds them to one or more data marts. It may also provide end-user access to support enterprise views of data".

In this study a DW will be defined as a collection of integrated, subject-oriented, non-volatile and time-variant collection of data from multiple sources that can be loaded, extracted and transformed to provide decision support and preserve historical and current data that can be accessed and analysed.

A DW will provide the following benefits to an organization (Kimball *et al*., 2008:28; De Silva, 2005:1.2; Rosencrance, 2011; BI-Insider, 2011):

- **Delivers enhanced business intelligence** – By providing relevant data from a number of sources, management and executives can make informed business decisions.

- **Business intelligence from multiple sources** – A DW performs integration of existing data from multiple sources and makes them accessible in a single location. Integration of data from multiple sources into a single repository alleviates the burden of duplicating data gathering efforts and enables the extraction of data that would otherwise be impossible.
- **Saves time** – Employees and business users can quickly access critical data from a number of sources, which saves time compared to retrieving the same data from multiple sources.
- **Enhances data quality and consistency** – Implementing a data warehouse requires the conversion of data from numerous sources into a single common format. This will lead to data from a number of departments to be in line with all the other departments. This ultimately leads to more accurate data**.**
- **Provides historical intelligence** – A DW stores a large amount of historical data so that one can analyse different time periods and trends in order to make more accurate future predictions.
- **Generates a high ROI** – Companies that have implemented a DW have generated more revenue and saved more money compared to companies that have not invested in a DW.

In the next section a brief discussion of system development methodologies will be discussed in order to reveal issues encountered in the field of information system development and how they have been addressed. This is done to later reveal that DW and IS development are pursued by the same issues and that the solutions provided by agile system development methodologies may also prove to address these issues in regard to DW development.

1.2.2 System development methodologies

Although information systems (IS) development is liable to have a low success rate, this issue is not new to the field; in fact in 1967 the NATO science committee called for an international conference in order to analyse the current state of IS development. They came to the conclusion that the field finds itself in what they termed a ‘software crisis’, which in turn led to the popular use of system development methodologies (SDMs). (Wu, 2011:2)

During the 1970’s, the use of computer applications in the commercial arena began to emerge, which caused the number of computer applications to exponentially increase and system development projects were undertaken on a larger and wider scale. With the size and complexity of IS rapidly increasing, an effort to formalize, structure, and use universally accepted good practices amongst IS developers began to spread. This effort, which is still in effect to this day, gave birth to the SDM movement, which began in the 1960’s and Avison

and Fitzgerald have divided it into the following four categories (Avison & Fitzgerald, 2006a:576-589):

Pre-methodology era

During the 1960's and early 1970's IS projects were developed without any explicit or formalized development methodologies. During this era the emphasis was on programming and solving technical problems, mainly the problems that resulted from the hardware limitations of the time (Avison & Fitzgerald, 2006a:576). The gathering of user requirements was poorly done and the user's needs were poorly defined, which led to IS designs that were inadequate to meet the genuine business requirements of the users (Avison & Fitzgerald, 2003:79). Following this approach to IS development, design was implicitly performed in one's mind and documentation was often non-existent (Mishra & Mohant, 2011:24).

Early methodology era

During the late 1970's and early 1980's developers attempted to address the issues facing IS development, which led to the introduction of a new development approach known as the systems development lifecycle (SDLC), more commonly known as the waterfall model. Although it was not yet known at the time, the SDLC was an early SDM, if not the first, and a pioneer in the SDM movement. The SDLC approach introduced a structured approach to the development of IS, which alleviated some of the chaos experienced in the pre-methodology era. The SDLC approach proved to be a step in the right direction for the development of ISs. (Avison & Fitzgerald, 2006b:28)

Methodology era

This era began in the mid 1980's up to the late 1990's and during this era the term methodology was used for the first time. During this era a large number of methodologies emerged in an attempt to deal with the limitations encountered with the SDLC approach. The methodologies or approaches that emerged came from two sources (Avison & Fitzgerald, 2006a:578):

- Methodologies developed from practice
- Methodologies developed from theory and research

The development of these new approaches can also be classified into two movements. The first movement was methodologies developed to improve upon the SDLC approach. The second movement was the proposal of new methodologies that moved away from the traditional SDLC approach. (Avison & Fitzgerald, 2003:81)

Post-methodology era

The success or failure of IS development thus far cannot be directly related to the use, misuse or non-use of methodologies, but this era has been characterized by a serious re-appraisal by practitioners and researchers alike on the concept and usefulness of the early methodologies (Avison & Fitzgerald, 2003:81). Up to this era, methodologies have often been viewed as a panacea to the problems of traditional development approaches and were often chosen and adopted for the wrong reasons. For a number of organizations, the use of a methodology has not always provided the benefits and success its advocates expected (Avison & Fitzgerald, 2006b: 35).

In response to the perceived problems and limitations of SDMs, the development of IS is being approached in a number of ways, namely (Avison & Fitzgerald, 2006a:586-588)

- **Continuing Refinement and Improvement -** In this approach to IS development, some are continuing the search for the methodology holy grail. It is most likely that new SDMs will continue to be developed and existing ones will evolve.
- **External Development -** Some organizations have given up on the idea of in-house development of IS and have decided to meet their IS requirements by purchasing packages. This approach is viewed as a quick and relatively cheap way of implementing IS for organizations that have generalized requirements. To some degree a purchased package modification and integration may still be necessary for the package to successfully meet all the needs of the organization, which may still be performed in-house. This approach may be useful but in the case where the IS is unique to a specific organization or a suitable package is not available, in-house development must be considered for this IS.
- ***Ad hoc* Development -** The *ad hoc* approach may be defined as a return to the approach of the pre-methodology era in which development is undertaken without the use of any formalized methodology. This approach to IS development is heavily dependent on the skills and experience of the developers and the process followed depends on whatever the developers understand and feel will work. It is understandable why some organizations have turned to this approach, but it generates the risk of repeating the same mistakes and encountering the issues prior to the use of SDMs.
- **Contingency -** This approach recommends a structure to assist the developers, but tools and techniques may or may not be used (or even modified) depending on the situation. Situations may differ depending on the organization, type of project and its objectives, users and the development team.

This is currently the era in which the SDM movement finds itself and during this era agile system development methodologies (ASDMs) made their debut and are the primary inspiration for using ASDMs in this study.

Even though SDMs have been around for quite some time, a universally accepted, rigorous and concise definition has not been agreed upon (Huisman & Iivari, 2002:136). At the general level, a methodology is regarded as a recommended series of steps and procedures to be followed in the course of developing an information system (Avison & Fitzgerald, 2006a:567). This definition is far from adequate and raises more questions than it answers (Avison & Fitzgerald, 2006a:567). In order to fully grasp what is meant by an SDM a few definitions are provided below in Table 1.1.

Source	Definition
Avison & Fitzgerald, 2006a:568	A system development methodology is a recommended means to achieve the development, or part of the development, of information systems based on a set of rationales and an underlying philosophy that supports, justifies and makes coherent such a recommendation for a particular context. The recommended means usually include the identification of phases, procedures, tasks, rules, techniques, guidelines, documentation and tools. They might also include recommendations concerning the management and organization of the approach and the identification and training of the participants.
Huisman & Iivari, 2006:32	Defines a systems development methodology as a combination of the following: **A systems development approach** This involves the philosophical view on which the methodology is built. It is the set of goals, guiding principles and beliefs, fundamental concepts, and principles of the systems development process that drives interpretations and actions. Examples are the structured, object-oriented and information modelling approaches. **A systems development process model** A process model is a representation of the sequences of stages through which a system evolves. Some examples are the linear lifecycle model and the spiral model. **A systems development method** A method is a systematic way of conducting at least one complete phase of systems development, consisting of a set of guidelines, activities, techniques, and tools, based on a particular philosophy and the target system. Examples include OMT and IE.

	A systems development technique Development techniques can be defined as procedures, possibly with a prescribed notation, to perform a development activity, for example construction of entity relationship diagrams.
Zaied *et al.*, 2013:20	The framework that is used to structure, plan, and control the process of developing an information system.

Table 1.1 – Table of SDM definitions that have been introduced over the years

This study will define an SDM as a recommended flexible framework to assist in the development of an information system, or part of an information system, that is supported by an underlying belief or philosophical view. This framework consists of an approach, process model, method and tools and techniques to assist in the development process. The framework may also indicate how to implement the IS and training guidelines (a more detailed explanation behind this definition will be provided in section 2.2).

According to Chin (2003), IS projects easily fall victim to constantly changing requirements, project budget and time instability, and the project team's ability to respond to change. These uncertainties and changing requirements have forced SDMs to adapt to the ever evolving business environment, which led to the development of ASDMs. The main objective of implementing ASDMs in an organizational setting is to deliver IS quickly, to quickly adapt to change and to allow for change as frequently as possible (Highsmith, 2002a:4). The term agile refers to "having a quick resourceful and adaptable character" and agile SDMs have appeared as a solution to the long-standing problems related to conventional methods (Aydin *et al.*, 2005:26).

The agile movement is not an anti-methodology movement; in fact, it intends to restore credibility to the word "methodology" as well as balance (Fowler & Highsmith, 2001). Although ASDMs differ from each other they all have the same foundation, which is the agile manifesto.

The Agile Manifesto consists of four purposes and twelve principles, which are (Fowler & Highsmit, 2001)

Purpose

1. Place emphasis on individuals and interactions over processes and tools.
2. Focus on producing working software over comprehensive documentation.
3. Focus on customer collaboration rather than contract negotiation.
4. Place in higher regard the act of responding to change instead of following a plan.

Principles

1. The highest priority of agile is to satisfy the customer through early and continuous delivery of valuable software.
2. Gladly welcome changing requirements, even late in development. Agile processes aim to harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.
4. Business people and developers work together daily throughout the project.
5. Build projects around motivated individuals, give them the environment and support they need and trust them to get the job done.
6. The most efficient and effective method of conveying information with and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhance agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.
11. The best architectures, requirements and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

The main goal of ASDMs is to make an organization agile, giving the organization the ability to adapt to change. According to Highsmith and Cockburn (2001), the practices ASDMs use are not what is new about them, but their recognition of people as the primary drivers of project success, coupled with a primary focus on the manoeuvrability, change and effectiveness.

Since the birth of ASDMs there has been quite a number of agile SDMs that have been developed, each unique in its own way but still adhering to the core principle of the agile manifesto to form the agile family. As agile SDMs began to gain popularity and recognition, a wide number of articles (Hislop *et al.*, 2002; Boehm & Turner, 2003; Conboy & Fitzgerald, 2004; Toflo & Wazlawick, 2008; Schwaber & Sutherland, 2013) on what Martin Fowler calls the "New Methodology" (Fowler & Highsmit, 2001) reflected a growing interest in these new approaches for the development of IS. Among these new approaches are the three relevant ASDMs (XP, SCRUM, and FDD) used in this study.

1.3 Problem statement and substantiation

According to Walker (2009), change management is probably the largest single issue facing data warehouse implementation and maintenance. Ambler (2013) states that in the course of

the development of a DW, requirements are going to change throughout the project. Therefore, DW projects are characterised by change and uncertainty and ASDMs are well-suited for projects faced with these issues (Informatica, 2011:2). Below are a number of reasons why DW projects are suitable candidates for ASDMs (Informatica, 2011:8):

- In regard to data warehousing or data integration, the more a business user drills down into understanding the data in an organizational context, the more new rules will be identified.
- Results from analysing data lead to new requirements.
- Business users expect quick delivery while still refining their requirements.
- Process steps are often skipped or expectations are not set properly.
- Data models are subject to change frequently due to changes in reporting and other business requirement changes.
- Business users are often not clear on the data until they see the data in the target system.
- Changes are made in the source system by the business after starting the project.
- Business logic has frequent changes.
- Most DW projects run for long periods of time and are subject to continual change.
- Required resources can be added as the project grows.

However, ASDMs are targeted at IS development in general and does not provide tools, techniques and best practices to deal with the technical aspects unique to DW development. According to (Jukic, 2006:83), one must choose a suitable data modelling approach in order to successfully develop a DW. Although there are a number of different choices when it comes to DW development, the industry's tools and methodologies are mostly based on only two approaches: those of Inmon and Kimball (Breslin, 2004:7). Bill Inmon, the father of data warehousing (Inmon, 2003), and Ralph Kimball, the dimensional data warehouse expert (Kimball *et al*., 2008) each introduced a different yet effective approach to the development of a DW.

Graziano (2005) stated that ASDMs can indeed be used in the development of a DW, but does not perform any experimentation to back his statement. Previous research conducted by Grey (2006) showed that ASDMs along with Kimball's approach served to be suitable for the development of a DW. Grey (2006:248) mentioned two aspects that proved to limit the generalization of his results. The first limitation was that only a single data mart was developed and the second limitation was that the research was conducted in an academic setting. In addition, the research conducted by Grey (2006) does not introduce a new approach that integrates the best practices into a well-established DW development approach to create a

new innovative methodology, but simply states that an ASDM alongside Kimball's approach proved to be suitable for DW development.

This study aims to determine whether Inmon's or Kimball's approach is best suited for the agile development of a DW system which in turn will provide the foundation for a new methodology to assist a project team in the agile development of a DW. In order to address the issues associated with DW development, a new methodology will be introduced that adopts best practices from three suitable ASDMs (XP, SCRUM, and FDD) that is built upon a foundation provided by either Inmon's or Kimball's approach. Using either Inmon's or Kimball's approach as the foundation will ensure that the technical issues unique to DW development are addressed accompanied by the benefits associated with ASDMs in order to provide guidelines for the agile development of a DW. Therefore, the aim of this research is to propose an agile data warehouse (ADW) methodology to support a project team in the agile development of a DW. The ADW (version 0) methodology will be evaluated by developing a DW in collaboration with an industry partner using real-world data and business users. Based upon the evaluation and the knowledge gained from the implementation of version 0, a final version of the ADW will be developed, evaluated, and presented as the contribution of knowledge intended to assist and support a project team in the agile development of a DW.

1.4 Research aim and objectives

The aim of this study is to develop the ADW methodology using either Inmon's or Kimball's approach as the foundation along with the best practices associated with XP, SCRUM, and FDD. The ADW methodology aims to provide guidelines for a development team that tackles the unique technical aspects of DW development along with the issues associated with DW development using agile best practice in order to develop a DW in an agile environment. In order to achieve this aim the following objectives must be achieved:

- Evaluate Inmon's and Kimball's approaches and determine which is best suited for agile development. This will be performed by developing and distributing the DW questionnaire.
- Evaluate XP, SCRUM, and FDD in order to determine which of the three is best suited for each major phase of the best-suited DW development approach (either Inmon or Kimball). This will be performed by developing and distributing the ASDM questionnaire.
- Develop version 0 of the ADW methodology.
- Implement and test the ADW (version 0) methodology for the actual development of a DW system in collaboration with the industry partner. Evaluate the implementation of

the ADW (version 0) methodology by developing and distributing the version 0 questionnaire.

- Develop version 1 (final) of the ADW methodology based on the evaluation and knowledge gained from the implementation of version 0.
- Evaluate the final version of the ADW methodology by developing and distributing the version 1 questionnaire.
- Present the knowledge gained, limitations, and future work.

1.5 Research methodology

The research for this study will be conducted using the positivistic research paradigm as the underlying philosophical viewpoint in order to provide empirical research that can be generalized for the agile development of a DW. Building upon the underlying paradigm, the research uses the design and create research strategy in order to produce a methodology artefact. The design and create strategy will form the foundation of the research that comprises the process steps followed to produce the desired artefact. Using the process steps introduced by the design and create strategy, the study will use questionnaires to gather the necessary data to perform the required evaluations. Finally, the data will be analysed using quantitative analysis (statistical analysis) to determine the reliability of the generated data and to draw sound conclusions. The research process steps are presented in Figure 1.2 and how they are carried out in the study is briefly discussed.

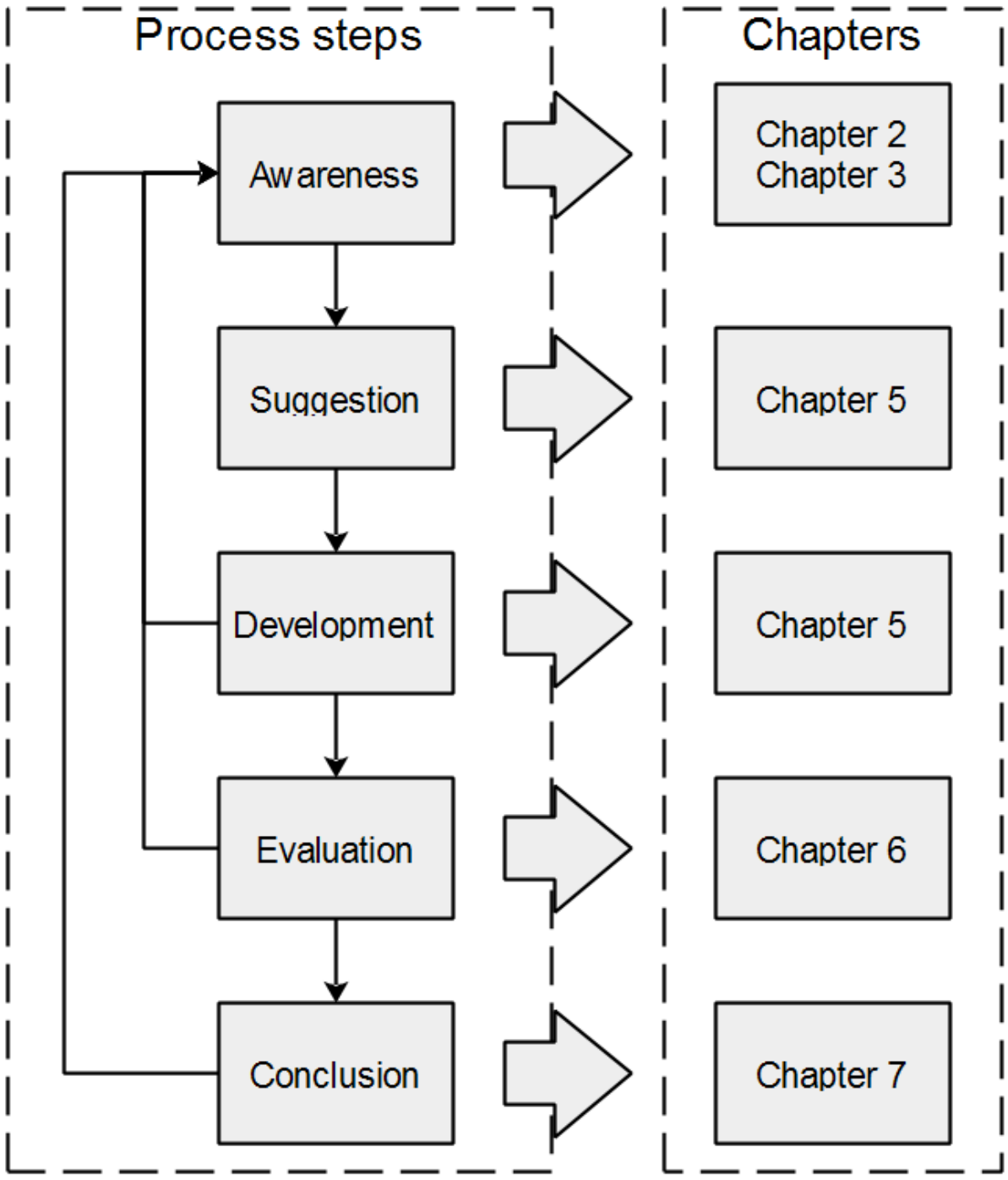


Figure 1.2 – Research methodology process steps

- **Awareness** – The awareness stage will be carried out in Chapters 2 and 3 and consists of a literature study on the following:
    - Chapter 2– Study the available literature in order to understand ASDMs in general and to provide a definition of the term and then introduce and discuss three popular agile SDMs (XP, SCRUM, and FDD) that will serve as possible candidates for the development of the ADW methodology.

    - Chapter 3 – Study the relevant literature in order to determine the origin of data warehousing and business intelligence and how these two terms come together to form a data warehouse system. In addition, the approaches of Inmon and Kimball will be introduced and discussed to serve as potential foundations for the development of the ADW methodology.
- **Suggestion** – The suggestion stage is undertaken in Chapter 5 and in this chapter a number of evaluations is performed in order to determine and suggest the best-suited approach between Inmon and Kimball in regard to agile development. The chapter also suggests the best-suited ASDM of the three (XP, SCRUM, and FDD) for each major phase of DW development for the best suited DW approach. The evaluations discussed in this chapter to arrive at the necessary conclusions to complete the suggestion stage of the research methodology will be implemented by developing and distributing the DW and ASDM questionnaires.
- **Development** – The development stage of the research methodology will be described in Chapter 5. The development phase focuses on developing the ADW (version 0) methodology based on the recommendations provided in the suggestion stage of the research.
- **Evaluation** – The evaluation stage of the research will be done in Chapter 6 and starts off by first evaluating version 0 of the ADW methodology. The evaluation of version 0 will be performed by developing and distributing the version 0 questionnaire to the relevant stakeholders and business users in order to determine whether the implementation of version 0 in the development of a DW system was successful. Based on the evaluation and knowledge gained from the implementation of version 0, version 1 (final) will be developed and evaluated by developing and distributing the version 1 questionnaire to verify the usefulness of the ADW (version 1) methodology for the agile development of a DW system.
- **Conclusion** – The conclusion stage will be addressed in Chapter 7 and will consist of discussing the knowledge gained in order to highlight key factors that led to the successful completion of the aim of the study. The final verdict of the study will be presented and the contribution of the study will be discussed.

1.6 Layout of the study

1. **Introduction** – The research problem and substantiation are stated in order to introduce the reader to the problem the study aims to address. The aims and objectives of the study are discussed followed by a brief overview of the research methodology and chapter outline of the study.

2. **Agile system development methodologies** – An in-depth explanation of the current literature on topics related to the study in regard to agile system development methodologies.
3. **Data warehousing** - An in-depth explanation of the current literature on topics related to the study in regard to data warehousing.
4. **Research methodology** – A formal explanation of the research plan and how it was used to arrive at sound results as well as a brief description of the components relevant to the research methodology used in this study.
5. **Proposed methodology** – The chapter starts off with the evaluation of Inmon versus Kimball in order to determine which of the two approaches is best suited for agile development with the help of the DW questionnaire. Using the best-suited DW development approach for agile development, another evaluation is conducted to determine which ASDM (XP, SCRUM, or FDD) is best suited for each major phase of development in a DW project with the help of the ASDM questionnaire. The chapter ends with the introduction and explanation of version 0 of the ADW methodology that is based on the evaluations performed earlier in the chapter.
6. **Evaluation** – This chapter begins by first discussing the implementation of version 0 of the ADW methodology and is followed by the evaluation of this implementation with the help of the version 0 questionnaire. Based on the evaluation and knowledge gained during the implementation of version 0, the chapter introduces/develops and explains version 1 (final) of the ADW methodology. The chapter concludes with the evaluation of the final version of the ADW methodology with the help of the version 1 questionnaire.
7. **Conclusion** – This chapter will conclude the findings of the study by discussing the objectives mentioned in section 1.4 in order to reveal how the objectives were implemented in order to arrive at the aim and goal of the study. The chapter draws to a conclusion by highlighting key factors that led to the successful completion of the aim to arrive at the final verdict of the study, mentioning the limitations encountered, and recommending future work to further strengthen the verification and/or improve upon the usefulness of the ADW (version 1) methodology.

In the next chapter the awareness stage of the research will be addressed by conducting a literature study on the relevant aspects of ASDMs.

Chapter 2: Agile system development methodologies

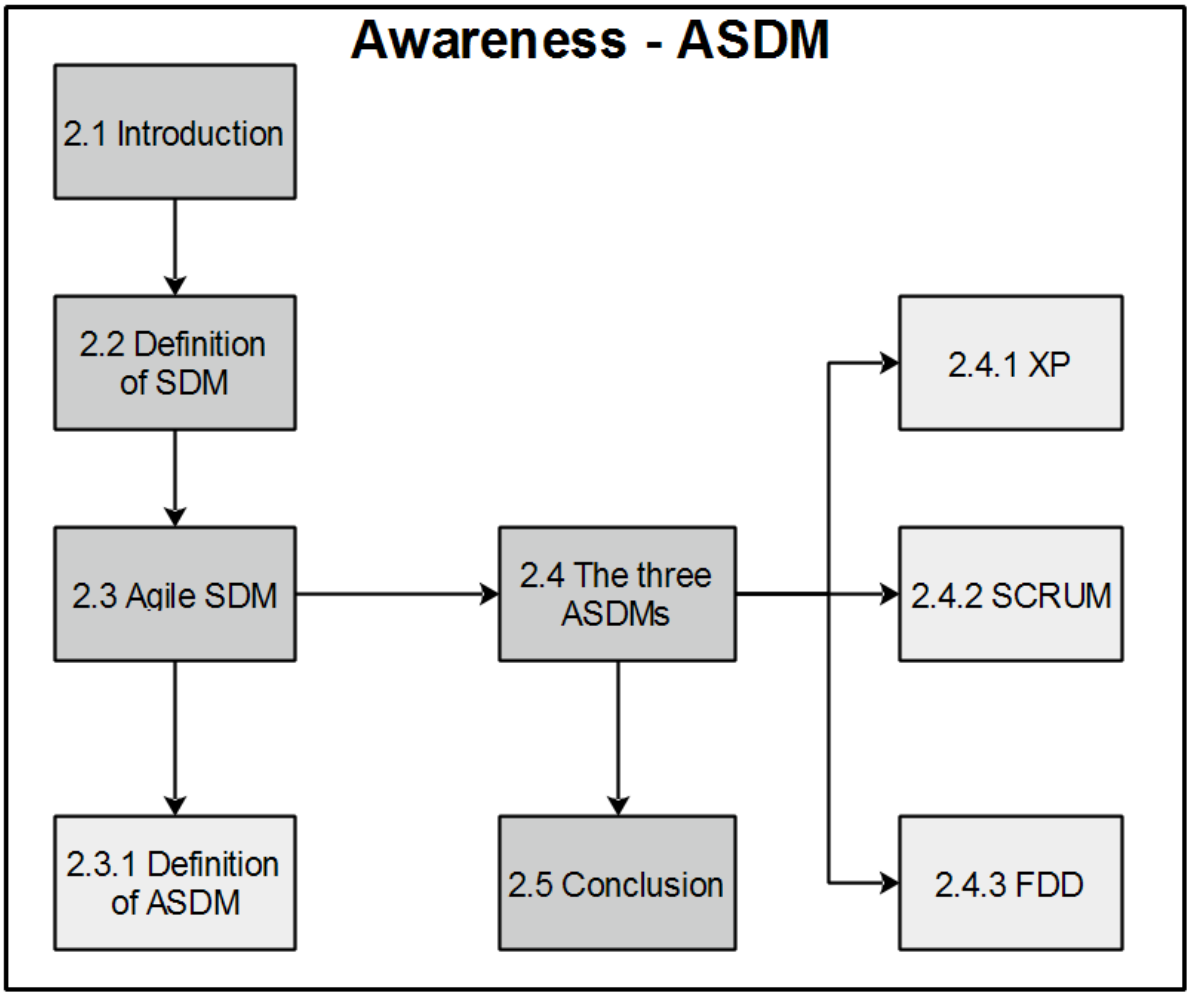


Figure 2.1 – Overview of Chapter 2

2.1 Introduction

In this chapter the definition of an SDM is expanded and explained by using a metaphor and diagram to clearly show how SDMs fit into the development of an IS. The explanation of the definition of an SDM is followed by the discussion of ASDMs in which the origin and reasons for the development of ASDMs are discussed. During the discussion on ASDMs, ASDMs will be defined by looking at a few definitions that have been documented over the years followed by the definition that will be used in this study. Once ASDMs have been defined to express what is meant by the term ASDM, the three ASDMs (XP, SCRUM, and FDD) will be introduced and explained. This section starts off by first explaining why these particular ASDMs were selected for the study followed by an explanation of the XP ASDM. The discussion of XP is

followed by the SCRUM ASDM and lastly the FDD ASDM is discussed. The chapter ends by highlighting important aspects that have been identified in the chapter.

2.2 Definition of system development methodology

In section 1.2.2 this study defined an SDM as a recommended flexible framework to assist in the development of an information system, or part of an information system, that is supported by an underlying belief or philosophical view. This framework consists of an approach, process model, method, and tools and techniques to assist in the development process. The framework may also indicate how to implement the IS and training guidelines.

In order to clearly show what is meant by the concept, definition, and environment of an SDM, the term will be described using a metaphor and diagram that express the view of this study.

Consider that the development environment of a new IS is viewed as a maze with all the possible routes and directions in which this project may progress (see Figure 2.2). The goal is to get from point A (start of a new IS project) to point B (successful completion of the IS project) by the shortest/safest route. This shortest/safest route can be regarded as the development process of the IS where the scope, cost, and time are within an acceptable range and all requirements are met.

In this maze there exist several pitfalls that lead to dead ends, which causes a project to take longer and cost more than expected or which may even lead to its termination. These pitfalls may be common mistakes and errors that have been identified over the years in IS development or new pitfalls unique to the maze in which a project team finds itself.

The complexity and size of each maze depends on each unique project, which in turn affect the use of an SDM. In the case where the maze is small and relatively basic, an SDM can be viewed as being unnecessary and an experienced project team or even just an experienced project manager will be able to determine the shortest/safest route from point A to B without the aid of an SDM. In cases where the maze ranges from moderate to high complexity where the optimal or even the safest route from point A to B cannot be determined without some uncertainty, then the use of an SDM can be justified to aid the project team to successfully reach point B.

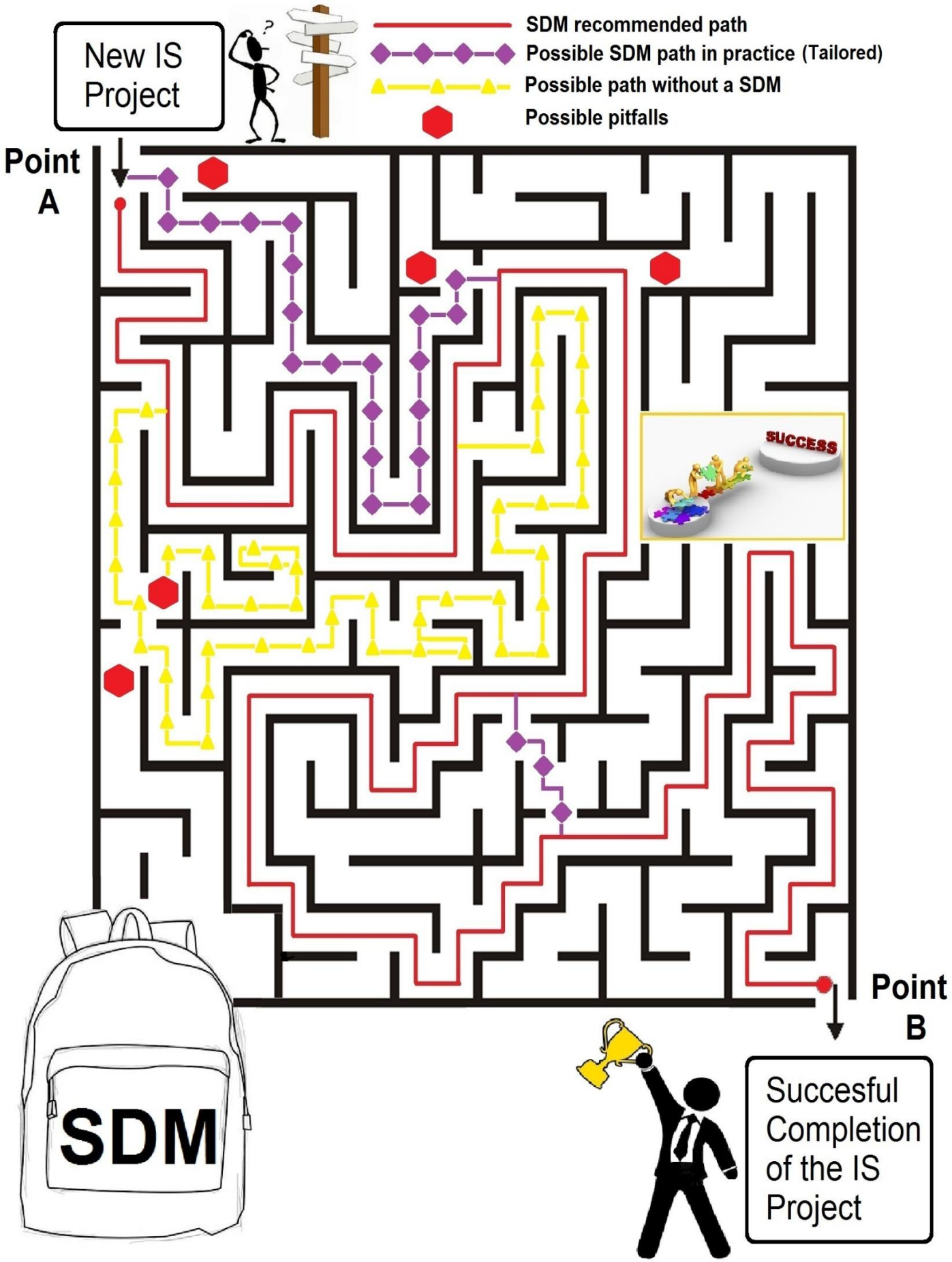


Figure 2.2 – Graphical illustration of the metaphor of an SDM

In the case where the maze ranges from moderate to high complexity there is a number of possible paths to reach point B; these three possible paths are shown in Figure 2.1. In Figure 2.1, one can see that by using an SDM, the project team is provided with a safe and acceptable path to reaching point B. Another path, which is the shortest, is a scenario where an SDM is used but instead of following the SDM to the letter, the project team has tailored the SDM to meet its needs based on its experience. This describes how most SDMs are used in practice and possibly the optimal route for unique IS projects. The last path, also the longest (in most cases) is a path where no SDM is used. As indicated in the diagram, this path leads to uncertainty and pitfalls and requires additional time and resources to reach the goal.

Now consider an SDM as being a pre-packed backpack that contains everything necessary to achieve the goal (reach point B by the optimal/safest route). Now this backpack typically contains:

Manual (IS development *approach*) that states the goals and objectives of using this backpack. It provides all the information of the backpack including the name of this particular backpack; in regard to an SDM this will be the philosophical view. Examples are the structured, object-oriented and information modelling approaches (Huisman & Iivari, 2006:32).

GPS (IS development *process model*) that provides a recommended route from point A to B to avoid possible pitfalls. The route recommended may not always be the optimal or shortest route, but it is an acceptable route, which is relatively safe. This relates to a representation of the sequences of stages through which a system evolves. Examples are the linear lifecycle model and the spiral model (Huisman & Iivari, 2006:32).

Instructions (IS development *method*) that provides guidelines on how to use the GPS and equipment found within the backpack and what to do in a particular situation. This is a systematic way of conducting at least one complete phase of IS development, consisting of a set of guidelines, activities, techniques, and tools, based on a particular philosophy and the target system. Examples are OMT, IE, etc. (Huisman & Iivari 2006:32).

Equipment (IS development *tools and techniques*) refers to the tools and necessary items to assist the project team in its journey through the maze. These tools are to assist with any obstacles or unexpected difficulty encountered on the journey. In regard to an SDM this can be defined as procedures and tools used to perform a development activity (Huisman & Iivari 2006:32).

Now that a clear explanation has been provided for an SDM, the term ASDM will be discussed and defined in the next section.

2.3 Agile system development methodology (ASDM)

According to (Abrahamsson *et al*., 2002:5), the field of IS development is far from shy when it comes to introducing new SDMs. In 1999 a study argued that traditional SDMs are viewed mainly as a necessary fiction that presents the illusion of control or provides a symbolic status. The study also claimed that traditional SDMs are too mechanistic to be used in detail (Abrahamsson *et al*., 2002:5). (Truex *et al*., 2000:53) took an even more extreme position and claim that it is possible that traditional SDMs are "merely unattainable ideals and hypothetical 'straw men' that provide normative guidance to utopian development situations". This caused industrial IS developers to become sceptical about new solutions (SDMs) that are difficult to use (Abrahamsson *et al.,* 2002:5).

Barry Boehm (2002:66) stated, "Plan-driven methods work best when developers can determine the requirements in advance … and when the requirements remain relatively stable, with change rates in the order of one percent per month". According to this statement, plan-driven SDMs with a relatively low change rate will still pose difficulties in the development process.

In the mid 1990's many IS developers found the initial requirements-gathering step frustrating and in some cases, impossible (Highsmith, 2002c). With the industry and technology rapidly improving it is becoming increasingly difficult for customers to precisely define their needs up front (Hislop *et al.*, 2002:170). In order to deal with this issue, several consultants have developed SDMs that embrace, rather than reject, these high rates of uncertainty in the requirements of IS projects (Hislop *et al.*, 2002:170). Most of the SDMs that emerged are based on iterative enhancement and although they were developed independently they share fundamental similarities in their techniques and philosophies (Hislop *et al.*, 2002:170). In February 2001, the authors of these emerging SDMs joined forces and decided to call their similar SDMs "agile" (Hislop *et al.*, 2002:170). These authors formed the agile alliance and wrote a set of agile values and principles, which is known as the Agile Manifesto (discussed in section 1.2.2).

2.3.1 Definition of agile system development methodology

The primary objective of an ASDM is to make an organization agile and according to Highsmith (2002b) the term agile (in regard to an organization) means to be able to deliver quickly, change quickly and to change as often as necessary. Highsmith and Cockburn (2001:120) claim that what is unique about ASDMs is not the practices they utilize but that they give recognition to people as the primary source of project success, combined with an emphasis on flexibility, change and effectiveness. ASDMs share three overlapping philosophies: process

control theory, emergence, and self-organization that need to be discussed in order to define an ASDM (Schwaber, 2001).

Process control theory

Defined processes are processes where a set of pre-defined steps are repeated to lead to a predictable desired outcome, such as the development of an automobile on an assembly line (Hislop *et al.*, 2002:170). Software development from an agile viewpoint is not a defined process where a set of steps is repeated to produce a predictable outcome (Schwaber & Beedle, 2002:45). Instead, IS development calls for rapid "inspect and adapt" cycles and feedback loops. ASDMs make use of short (two-six weeks) iterations that focus on developing a fragment of the software that the customer can inspect. This allows for improved customer collaboration and inspection, which in turn allows the development team to adapt (as well as improve product quality) the development plan and priorities for the next iteration. (Hislop *et al.*, 2002:170-171)

This development process is known as iterative and incremental development and in order to clearly understand this process, these two terms must also be defined.

Incremental – The IS is developed and delivered in fragments. This means that the requirements are broken down into small manageable subsystems. These subsystems are added together after each iteration to form the complete IS. (Lindvall *et al*., 2002:201)

Iterative – Iterative development goes hand-in-hand with incremental development, where incremental development breaks the entire IS into smaller fragments and iterative development focuses on developing one of these fragments in the current iteration. Therefore, iterative development refers to the process of repeating the lifecycle (development process) to continue to develop incremental fragments to be added to the IS to result in the final product. (Boehm & Turner, 2003:16)

Emergence

The authors of ASDMs believe that when faced with changing requirements and technologies, IS requirements cannot be accurately specified in advance. Rather it is believed that the requirements a customer actually desires (as opposed to the initial requirements) should be allowed to emerge along with the development process. Therefore, ASDMs welcome changing requirements and uncertainty at any time in the development process. The reason for this flexibility is to deliver IS that the customer actually wants regardless of constant change and turbulence. (Hislop *et al.*, 2002:171)

Self-organization

In order to determine the best possible way to get the job done, ASDMs give the entire development team the autonomy to self-organize. Team members are not limited by pre-defined roles or expected to perform obsolete task plans. Project managers of agile teams place considerable trust in and responsibility on the entire project team and emphasis is placed on face-to-face communication, rather than on formal documentation. ASDMs also use post-mortem meetings where the project team discusses how it can become more effective. (Hislop *et al.*, 2002:171)

As with SDMs no universally accepted definition for ASDMs exists (Larman, 2004:25), but the following definitions for ASDMs have emerged over the years (see Table 2.1).

Source	Definition
Boehm & Turner, 2003:17	Very lightweight processes that employ short iterative cycles: actively involve users to establish, prioritise, and verify requirements, and rely on tactic knowledge within a team as opposed to documentation.
Rouse, 2007	The creative process that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product. Agile software development focuses on keeping the code simple, testing often, and delivering functional bits of the application as soon as they're ready.
Qumer & Henderson-Sellers, 2008:281	A methodology that is people-focused, communications-oriented, flexible (ready to adapt to expected or unexpected change at any time), speedy (encourages rapid and iterative development of the product in small releases), lean (focuses on shortening timeframe and cost and on improved quality), responsive (reacts appropriately to expected and unexpected changes), and learning (focuses on improvement during and after product development).
Grey, 2011:11	An ASDM is an adaptable/flexible SDM that focuses on integrated communication, user involvement, minimized documentation, and incremental and iterative development to deliver quickly and as often as possible in a constantly changing environment.

Table 2.1 – ASDM definitions that have emerged over the years

Apart from the above definitions, Conboy and Fitzgerald (2004:108) state that agility consists of flexibility and speed. They claim that terms, such as "fast", "rapid", "speed" and "quick" are commonly found in definitions of agility; therefore, for an organization to be agile it must be able to respond "speedily and flexibly" (Conboy & Fitzgerald, 2004:108). Fowler (2001) stated

that an ASDM is more “adaptive than predictive” and more “people-oriented than process-oriented”.

Based on the definitions and explanations given above, this study will define an ASDM as a light-weight and flexible methodology that is people-focused, communications-oriented, relies on tactic knowledge within the team opposed to excessive documentation and implements iterative and incremental development to deliver results quickly and as often as possible in a constantly changing environment.

2.4 The three agile system development methodologies

In this section the three ASDMs that are relevant to this study will be explained. These ASDMS are

- Extreme programming (XP)
- SCRUM
- Feature driven development (FDD)

In section 1.3 the reasons for implementing ASDM best practices for DW development were discussed. In the following paragraphs the reason for selecting the above-mentioned ASDMs will be discussed.

Graziano (2005) mentions that ASDMs, especially XP and FDD, have promising characteristics that can be implemented to contribute to the successful development of a DW. Graziano (2005) then goes further and states that FDD appears to be the most applicable ASDM for the development of a DW project.

Krawatzeck *et al*. (2015:4767) mention six articles that have tested the suitability of SCRUM and XP as the process model for the development of a DW project and mention that each project revealed promising results.

The business intelligence blog (2008) mentions a number of ASDMs that may prove to be well-suited for the agile development of a DW system and includes XP, SCRUM, and FDD.

The claim made by Graziano (2005) is only based on theoretical deductions and has not been tested or proven to be based on sound scientific evidence. Krawatzeck *et al*. (2015:4767) discuss the suitability of SCRUM and XP as revealing promising results but none of the studies implemented SCRUM or XP in the actual development of a DW in a real-world environment. In other words, the studies were performed in a controlled environment. In addition it seems that each methodology was used as is and can therefore not provide a project team with best practices, tools, and techniques to address the unique technical aspects of DW development.

The business intelligence blog (2008) mentions a number of well-suited ASDMs for DW development but, does not support this claim with scientific evidence or proof.

Therefore this study will use the knowledge provided by past studies (Graziano, 2005; the business intelligence blog, 2008; Krawatzeck *et al*., 2015) and further analyse XP, SCRUM, and FDD for the development of a DW. This study differs from the above-mentioned studies by developing a new innovative methodology (ADW methodology) that uses a well-established DW development approach as the foundation for the unique technical aspects of DW development. Building on this foundation, best practices from XP, SCRUM, and FDD will be used to empower the ADW methodology with the benefits of ASDMs to deal with the uncertainty and changing requirements characterising DW development. The initial version of the ADW methodology will be implemented in the actual development of a DW system in collaboration with an industry partner to allow for an uncontrolled development environment. Therefore, this study differs from previous studies by developing a new unique methodology by combining a well-established DW development approach with XP, SCRUM, and FDD to result in the ADW methodology that will be tested and implemented in an uncontrolled development environment. In this study an evaluation of the suitability of XP, SCRUM, and FDD will also be performed to determine which ASDM is best suited for each unique phase of development and this will be carried out by using the ASDM questionnaire that will result in scientific proof to support the suitability of XP, SCRUM and FDD for the development of a DW.

In concluding the section on the selection of XP, SCRUM, and FDD for further analysis, previous work has highlighted the potential of the above-mentioned ASDMs for DW development but contain the following gaps that will be addressed by this study:

- Claims and statements are based on theoretical deductions and not supported by scientific evidence.
- Does not test and implement XP, SCRUM, and FDD in the actual development of a DW system in an uncontrolled environment.

In addition to the gaps that the study will address, the study is unique by contributing a new innovative methodology that combines a well-established DW development approach with the best practices of XP, SCRUM, and FDD for the successful and agile development of a DW in an environment pursued by uncertainty and changing requirements.

2.4.1 Extreme programming (XP)

In 1996 Kent Beck introduced XP and the term itself might strike fear into traditional system development circles. However, taking a closer look at XP reveals the approach to be a sound structure that is based on a number of principles and practices to which each developer can

relate (Hislop *et al.*, 2002:172). XP moves away from traditional SDMs by directing the focus on building simple, clean and tested ISs in small deliverables through the harmonized efforts of a team of developers (Hislop *et al.*, 2002:172). XP is a light-weight SDM that is best suited for small to medium-sized development teams dealing with vague or rapidly changing requirements. XP is part of the agile family and is usually the first ASDM that companies intend to adopt (Toflo & Wazlawick, 2008:1955).

XP can best be described in terms of its main structural components as seen in Figure 2.2. The basis of XP's work ethic is the four values (communication, simplicity, feedback and courage) and the technical backbone is the four basic activities (coding, testing, listening and designing) (Hislop *et al.*, 2002:172).

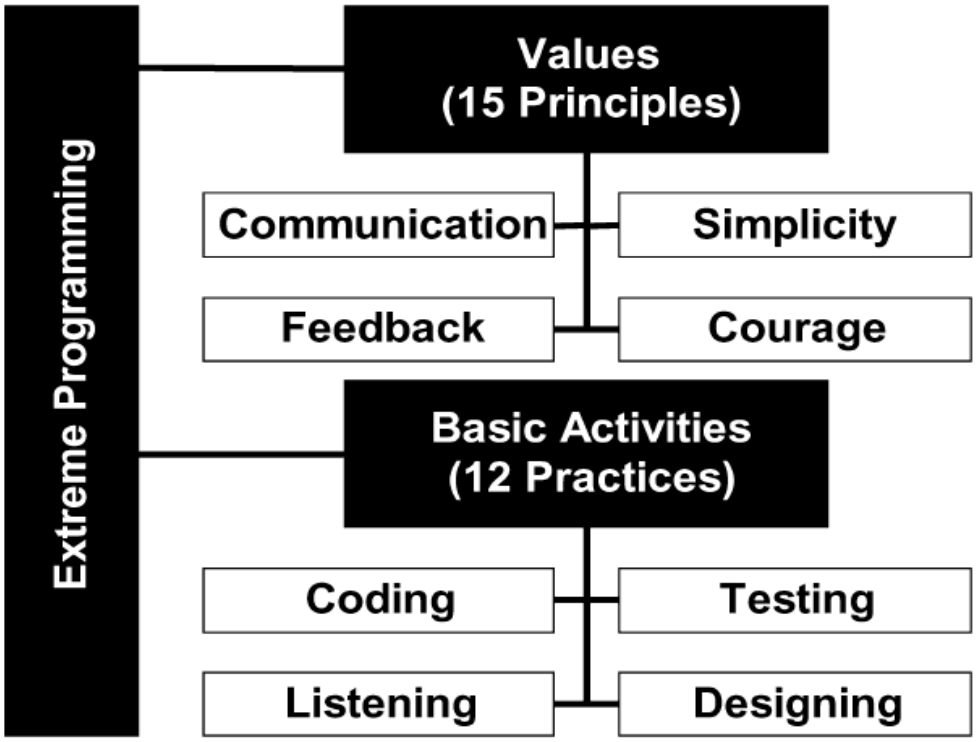


Figure 2.3 – XP's structure (Hislop *et al.*, 2002:173)

The Four values of XP

When using XP, the team's actions are guided by these values and according to Lindstrom and Jeffries (2004:50) they can be used to determine whether XP will suit the project, team, and organization. Lindstrom and Jeffries (2004:45) also mention that XP is the only ASDM that is explicit in its values that produce principles and practices. This combination of values and activities give guidance to the project team on how to react when the practices do not work (using the values) and what to do (using the practices) (Lindstrom & Jeffries, 2004:45; Grey, 2011:24).

Communication - Communication among the project team and between the customers and the team is expected to be relatively high and frequent. In the requirement-gathering process, customers communicate the necessary requirements as stories written on small file cards (known as story cards) that are used by the team to guide the development of the IS. (Hislop *et al.*, 2002:172; Grey, 2011:24-25)

Simplicity – As the name implies, this means that the project team must focus on developing the simplest IS that will get the job done. As features are implemented into the growing IS, the project team must stay true to this value and keep the IS as simple as possible while still meeting all the requirements of the customer. (Hislop *et al.*, 2002:173; Grey, 2011:25)

Feedback – XP promotes the constant and continuous involvement of customers and the project team, which results in immediate feedback on the status of the IS to keep the project team on track (Pearman & Goodwill, 2006:5; Chromatic, 2003:8). Pair programming is a continuous feedback loop between two programmers working together at the same desk (Hislop *et al.*, 2002:173). Continuous testing provides the project team with rapid feedback on bugs in the code (Hislop *et al.*, 2002:174; Grey, 2011:25).

Courage – The project team is responsible for developing an IS that is simple, easy to use and that meets the primary requirements of the customers. In order to achieve this, courageous decisions may need to be taken by the project team. These may include removing large chunks of code or re-engineering the IS to get rid of duplicate code. In short, a project team using XP must not be afraid to make drastic decisions in order to achieve what is desired. (Hislop *et al.*, 2002:174; Grey, 2011:25)

In addition to the four values just discussed, Holcombe and Holcombe (2008:25) discuss a fifth value, respect. Most of the problems faced during the development of an IS are “people issues”, which are normally associated with relationships. The project team members and all the other stakeholders involved must treat each other with the respect they deserve in order to manage relational and people issues. This means that individuals must be allowed to explain their point of view, resolve conflict in an orderly manner and maintain a healthy working relationship. (Holcombe & Holcombe, 2008:25)

XP has 15 principles that are based on these four values that bridge the gap between these values and the practices that implement them are (Hislop *et al*., 2002:173; Pearman & Goodwill, 2006:6):

- Incremental changes
- Embracing change

- Assuming simplicity
- Rapid feedback
- High-quality work delivery
- Learn and teach XP
- Small initial investments
- Concrete experimentation
- Playing to win
- Accepting responsibility
- Promoting honest and open communication
- Working with people's instincts
- Adapting as required
- Travel light
- Honest measurement

The four basic activities of XP

Coding – For any IS project there ultimately has to be a functioning program and therefore, coding is the one activity that cannot be ignored. In XP coding is seen as a learning activity and XP is designed to enhance programming with pair programming and refactoring practices (Baird, 2002:29). Coding helps the developers to test their thinking process and if the thinking process is accurate, then the code will do what it is indented to do (Hislop *et al*., 2002:174).

Testing – Testing is seen as a continuous process through the entire development process as new requirements are implemented into the growing IS (Baird, 2002:29). Developers are required to test every completed feature before integration to ensure that the system is correct and up to date before moving on to the next iteration (Hislop *et al*., 2002:174; Grey, 2011:26).

Listening – As previously mentioned, communication is one of the four values of XP and communication cannot take place without listening. If the project team does not know how to listen to the customers it will be impossible to accurately gather the requirements and the IS project will surely fail. In XP listening is simplified with the use of story cards and the project team must listen to the customers stories so that they can help define the requirements and the team can know what should be developed and tested in order to develop a successful IS (Hislop *et al*., 2002:174).

Designing – Since the agile approach to IS development is to constantly adapt to change, design in XP is a continuous activity of implementing new features and requirements into the evolving IS. Rather than an architectural design, XP uses a metaphor to guide the design

process. A metaphor is a brief description that conveys the IS's main functional attributes (Hislop *et al*., 2002:174).

The twelve core practices of XP

In this section the twelve core practices of XP will be discussed using the XP lifecycle. Figure 2.3 shows the XP lifecycle and how the twelve core practices are integrated into the development process. As seen in Figure 2.4, a spike can be described as an "exploring effort by developing prototype code". This has a specific goal in mind and should not be seen as general technical research (Baird, 2002:59; Grey, 2011:27).

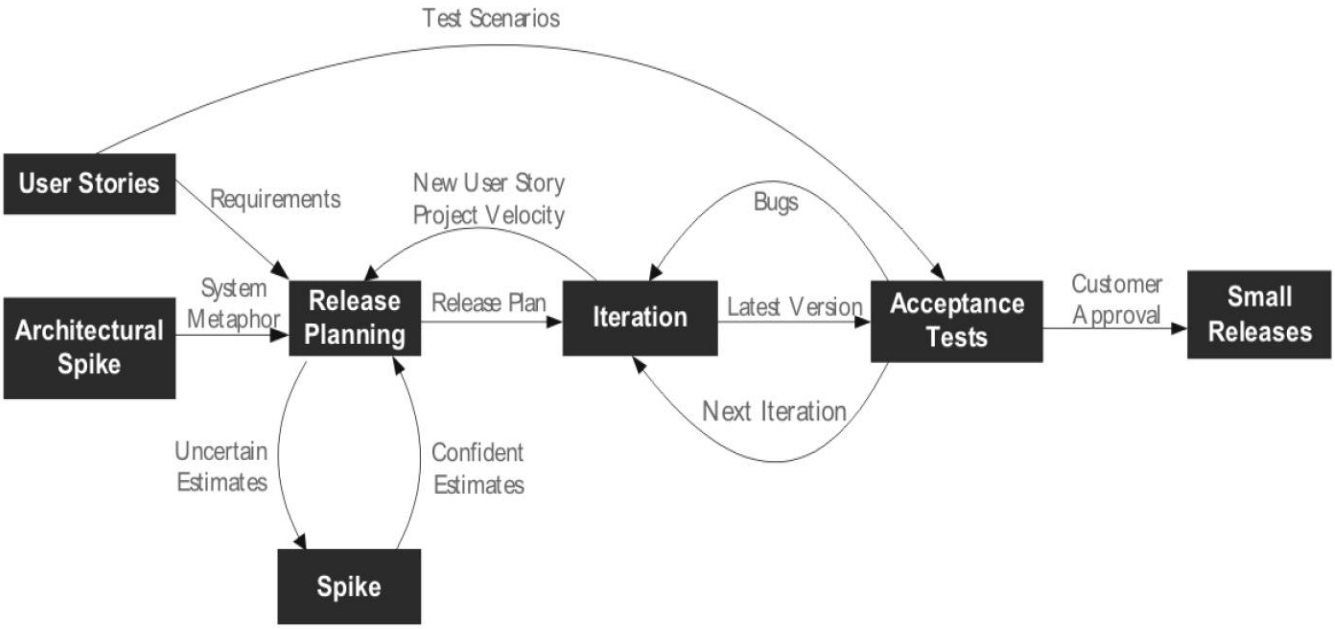


Figure 2.4 – XP's lifecycle (Huisman, 2014:46)

The planning game – At the start of each iteration, the managers, customers and development team meet to discover, prioritize, and predict what should be achieved before the next release as well as to gather the requirements for the next release (Lindstrom & Jeffries, 2004:47; Grey, 2011:27). This activity consists of two key planning steps which are release planning and iteration planning. In release planning the customer describes the desired requirements for the IS, which are called "user stories". The "user stories" are captured on index cards, which are then evaluated by the project team to determine the time and cost of each story. Once an estimate of each story has been provided, the customers choose which stories will be implemented in the current iteration and agree not to make any changes or additions until the next iteration. Iteration planning involves internal planning amongst the project team for the current iteration. Once the current iteration comes to an end the planning

game is played again in order to gather the requirements for the next iteration. (Lindstrom & Jeffries, 2004:47, Pearman & Goodwill, 2006:12; Huisman, 2014:48; Grey, 2011:27-28)

Small releases – IS projects that were developed in shorter timeframes have been found to have a higher success rate than projects that have lengthy delivery dates (Pearman & Goodwill, 2006:13). In XP the IS is planned with short release cycles consisting of the highest priority business requirements (Huisman, 2014:47). A release is a group of iterations that provide valuable features to the users of an IS and iterations are typically one to three weeks in duration (Grenning, 2002:2; Grey, 2011:28). A release is usually delivered in less than three months and this allows the users to see and touch the working IS on a regular basis (Huisman, 2014:47).

Metaphor – Metaphors serve as guidelines or models for the development process and explain the system's behaviour (Grenning, 2002:2; Huisman, 2014:48). The purpose of metaphors is to have a complete and simple way to describe the IS project's vision that defines its scope and purpose (Pearman & Goodwill, 2006:13; Huisman, 2014:48). Metaphors are typically a group of classes (grouped program code) that are arranged to satisfy the requirements from the story cards that are being developed (Holcombe & Holcombe, 2008:29; Grey, 2011:28).

Testing (client tests and test-driven development) – In XP there are two types of test that are performed continuously throughout the development process. Firstly, unit tests are performed to verify that the program code does what developers want it to do and these tests are typically written by the developers (Huisman, 2014:47). Unit testing allows developers to receive immediate feedback on their progress (Lindstrom & Jeffries, 2004:48). Secondly, acceptance tests are performed by the development team but "driven by the client" (Burke & Coyner, 2003:21). These tests are written based on the user stories (currently available to the project team) to test whether the IS does what the user expects from the system (Huisman, 2014:47; Holcombe & Holcombe, 2008:26; Grenning, 2002:2). According to Baird (2002:34), these tests are not just performed at the end of the IS project, but are performed continuously as the project progresses.

Simple design – In XP the design of the IS is kept as simple as possible for the agreed-upon user stories. The development team does not waste its time and energy on features that might be coming later on in the development process; instead it focuses on what has been agreed upon and will shift its focus to new requirements as they are added (Grenning, 2002:3; Lindstrom & Jeffries, 2004:48). According to Baird (2002:34), "simple systems are easy to extend, work faster, are easier to communicate about, and simpler to maintain".

Refactoring – Once the project team realizes that a design is no longer appropriate or of any value to the user, it must be changed (Grey, 2011:29; Huisman, 2014:48). This includes improving the design and code of the IS while still keeping the system functional and simple through removing duplication, improving communication and adding flexibility (Lindstrom & Jeffries, 2004:48; Burke & Coyner, 2003:22; Grey, 2011:29). This will result in a cleaner more understandable and maintainable solution (Holcombe & Holcombe, 2008:33).

Pair programming – Pair programming is where two programmers work at the same computer and both programmers are intimately involved in writing and reviewing the code (Crispin & House, 2003:5; Lindstrom & Jeffries, 2004:48; Grey, 2011:29). The two programmers switch seats periodically and frequently discuss each other's code (Huisman, 2014:47). While the one programmer codes, the other thinks strategically about where the approach will be most effective and about ways to simplify the design (Avison & Fitzgerald, 2006a:480). Of all the practices introduced by XP, pair programming appears to raise the most suspicion (Hislop *et al*., 2002:174). This practice may seem like an ineffective way to program, but pair programming increases the number of programmers that are familiar with the code, ensures effective coding with fewer errors in a shorter time and allows the programmers to learn from each other (Pearman & Goodwill, 2006:10: Holcombe & Holcombe, 2008:26; Grey, 2011:29).

Collective ownership – As the name implies, the entire team has ownership of all the development documents and code and is free to make changes anywhere at all times in the development process (Pearman & Goodwill, 2006:10; Crispin & House, 2003:6; Holcombe & Holcombe, 2008:32). The supporters of XP argue that the more people who work on a piece, the higher the quality of the code (Huisman, 2014:47). Collective ownership increases code quality, reduces bugs and eliminates code duplication (Lindstrom & Jeffries, 2004:49).

Continuous integration – New fragments of code must be integrated as soon and as frequently as possible to ensure that the development team is up to date at all times (Crispin & House, 2003:5; Burke & Coyner, 2003:27: Grey, 2011:30). Once a task is complete it must be integrated into the IS; in the integration process the task is tested and after it has passed the test, it is accepted as part of the current IS (Lindstrom & Jeffries, 2004:48; Grey, 2011:29). Integration occurs after only a few hours of coding or at the end of the day at the latest and this simplifies the integration process and makes it easy to identify the responsible party when integration testing fails (Huisman, 2014:47; Pearman & Goodwill, 2006:12).

Forty-hour week – Development using XP is very demanding and in order for the project team to stay fresh, XP has a rule that the team members must not work more than 40 hours per week. Although this all sounds well and good there are times when 40 hours is just not

enough, therefore XP allows for overtime but insists that overtime must not continue for longer than two subsequent weeks (Grey, 2011:30). This practice prevents the development team from burning out and doing sloppy work that may affect the IS in a negative manner. When the team is well rested and has a positive morale, productivity is boosted and this will prove favourable in the long run (Huisman, 2014:49).

On-site client – As previously mentioned, XP places a great deal of importance on communication, especially between the client and the team, and promotes the idea that a client be on site at all times as part of the project team (Huisman, 2014:50; Grey, 2011:30). This allows the client to contribute to development, to answer questions, to perform acceptance tests, to bring clarity where there is uncertainty and to keep the development team on track (Holcombe & Holcombe, 2008:27).

Coding standards (open work space) – XP requires that programmers agree upon and follow a common coding standard so that code that emphasizes communication appears to have been written by one person (Holcombe & Holcombe, 2008:32).

In the next section the SCRUM ASDM will be discussed.

2.4.2 SCRUM

Ken Schwaber and Jeff Sutherland developed SCRUM in 1995 and described it as a process that is open to the uncertainty associated with the development of IS with a “do what it takes mentality” and the ASDM has been successfully implemented by a number of independent software vendors (Grey, 2011:19). Schwaber and Sutherland (2013:3) define SCRUM as “a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value”. SCRUM is: (Schwaber & Sutherland, 2013:3)

- Lightweight
- Simple to understand
- Difficult to master

The methodology is named after the SCRUM in rugby where a tight formation of forwards bind together and push and shuffle against the opponent’s forwards for possession of the ball (Grey, 2011:19). SCRUM is an enhancement of the iterative and incremental approach for delivering object-oriented software that releases functioning artefacts regularly (Schwaber & Sutherland, 2013:3; Grey, 2011:19). Schwaber and Sutherland (2013:3) state that “SCRUM is not a process or a technique for building products; rather, it is a framework within which you

can employ various processes and techniques. SCRUM makes clear the relative efficacy of your product management and development practices so that you can improve."

The main idea of SCRUM is that requirements are most likely to change during the development process, thus making the development process complex and unpredictable. Therefore, the development process calls for flexibility and adaptability in order to successfully develop the IS in a changing environment. According to (Abrahamsson *et al*., 2002:27), SCRUM is an "empirical approach applying the ideas of industrial process control theory to system development resulting in an approach that reintroduces the ideas of flexibility, adaptability and productivity". SCRUM is concerned with how the project team must act in order to produce a successful IS in a constantly changing environment (Abrahamsson *et al*., 2002:27).

The primary concern of SCRUM rests on a few key management tasks instead of placing emphasis on the way the IS is actually developed (Good, 2003:18). The methodology is based upon the following (Stober & Hansmann, 2010:41-42):

- Three roles – product owner, scrum master and the team
- Three documents and processes – product backlog, sprint backlog and sprint results
- Three meetings – pre-sprint planning, daily scrum and post-sprint (also known as sprint review meeting)

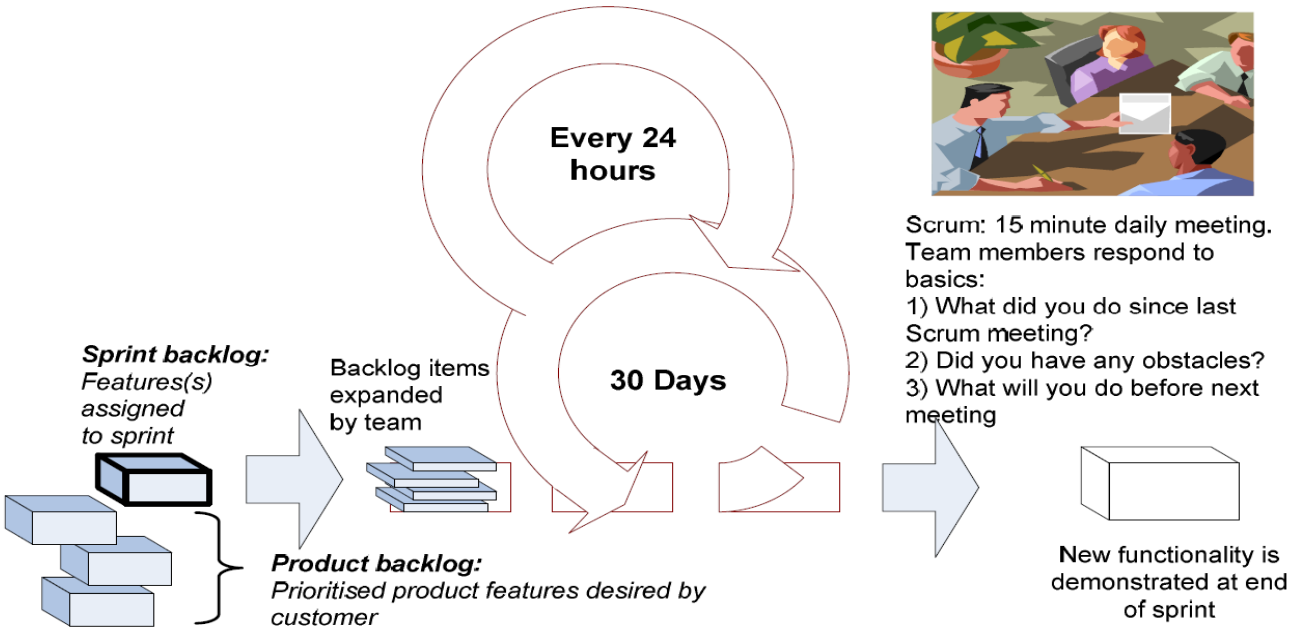


Figure 2.5 – The SCRUM lifecycle (Grey, 2011:20)

The SCRUM ASDM will be explained using the SCRUM lifecycle (Figure 2.5). IT projects, when using SCRUM, are divided into iterations, which are called “sprints” that are periods of up to thirty days (Grey, 2011:21). During the sprint, sectioned tasks will be performed to develop deliverables that meet the requirements of the clients, managers and experts (Huijbers *et al*., 2004:17). The independent project team is the primary player responsible for developing quality deliverables (Kammermeier, 2009:5). As is the case with iterative development, each sprint consists of all the traditional phases associated with IS development (requirements, analysis, design, evaluation, documentation, and delivery) and once a sprint is completed a new one starts and the entire development cycle is repeated (Abrahamsson *et al*., 2002:30: Stober & Hansmann, 2010:42). In the entire development process there can be as many as eight sprints when using the SCRUM methodology.

The product backlog (see Figure 2.5) is part of the planning phase and contains all the work required during the entire project, which includes requirements obtained from the developers, clients, and experts (Grey, 2011:20). According to (Abrahamsson *et al*., 2002:29), the product backlog “includes the definition of the system being developed, project team, tools and other resources, risk assessment and controlling issues, training needs and verification management approval”. The requirements listed in the product backlog are prioritized according to descending order of importance and the product backlog must be updated as frequently as possible and prioritized when new requirements are added (Grey, 2011:20). The product owner is burdened with the responsibility for the product backlog and he/she represents the interests, expectations, and constraints of the stakeholders involved in the project (Kammermeier, 2009:5; Haugan, 2011:289). The product owner is also responsible for ensuring that the necessary funding is available to the project and signs off on all deliverables that are completed (Stober & Hansmann, 2010:41). The architecture phase is completed along with planning, which includes any changes made and the problems these changes may cause when implementing the product backlog if the implemented IS needs improvement (Abrahamsson *et al*., 2002:29). In order to implement these changes a design review meeting is held (Grey, 2011:20).

Before the project team can begin with a sprint, it first needs to have a pre-sprint planning session where it identifies the tasks necessary to achieve the defined sprint goal. These identified requirements and tasks are moved from the product backlog to the sprint backlog after they have been prioritized to be completed in the next sprint cycle (Schwaber & Sutherland, 2013:9). Once the first sprint cycle has been completed, a post-sprint meeting is held where the project team evaluates the project and decides whether it should continue with the project. If the team decides to continue with the project, a pre-sprint meeting is held to identify tasks that need to be completed during the next sprint.

The sprint backlog is the starting point for every sprint and it contains all the requirements and tasks that need to be carried out during the current sprint (Schwaber & Sutherland, 2013:9; Grey, 2011: 21). Using the prioritized lists of the product backlog, the tasks that need to be completed for the current sprint are identified by the scrum team, the scrum master and the product owner during the pre-sprint planning (Abrahamsson *et al*., 2002:33). The scrum master (can be seen as the project manager) is responsible for the success of the entire project, including the delivery of quality products, facilitation of communication, removal of impediments, and the process as a whole (Stober & Hansmann, 2010:41: Haugan, 2011:289). The scrum master is also responsible for the way in which the SCRUM ASDM is used and should adhere to the rules, values, process and practices of SCRUM if the team wishes to reap the full benefits of the SCRUM ASDM.

Every morning, a short meeting of about fifteen minutes is held in order to keep track of the development process. In these meetings, the scrum team specifies what has been done since the previous meeting and discusses what tasks need to be done before the next meeting (Abrahamsson *et al*., 2002:34). These meeting are used to identify problems encountered and solutions to these problems are discussed to keep the team focused on the goal. Additionally, these meetings increase communication and keep the stakeholders and team members involved and up to date (Zelkowitz, 2004:15). According to (Abrahamsson *et al*., 2002:35), these meetings help the team to believe in its own abilities as well as the client to believe in the team.

Once a sprint has been completed, a post-sprint meeting is held between the scrum master, the team, and product owner to analyse the progress and to present the IS to management and clients (Zelkowitz, 2004:15). During this meeting, new requirements may be identified to further improve the IS. The new requirements are then added to the prioritized product backlog and the next sprint is planned based on the updated product backlog (Grey, 2011:22).

SCRUM is mainly concerned with key management tasks instead of the way in which the IS is actually developed (Good, 2003:18). This can be seen throughout the lifecycle where SCRUM placed more emphasis on the necessary meetings required to develop the IS rather than the tools and techniques that will assist the team in the actual development. The project team is responsible for the development of the actual system and it can choose tools and techniques it deems fit for the occasion.

In the next section the FDD ASDM will be discussed.

2.4.3 Feature driven development (FDD)

FDD was created by Jeff De Luca and Peter Coad in 1997 and later published in 1999 (De Luca, 2005). FDD was created when De Luca and Coad became consultants for a project (the large lending system project at United Overseas Bank in Singapore) that was in dire need of professional help (Grey, 2011:32). As consultants on the project, Coad applied feature-oriented development techniques on the project and De Luca used a streamlined, lightweight process framework (Hislop *et al.*, 2002:175). They then combined their ideas into a methodology that became known as FDD in order to save the extremely complex Singapore project from failing (Hislop *et al.*, 2002:175; Grey, 2011:33). Before Jeff De Luca and Peter Coad joined the project as consultants, the previous development team spent over two years on this project, writing over 3 500 pages of documentation without any code and declared the project impossible (Highsmith, 2002a:5; Hislop *et al.*, 2002:175). FDD was then used for the development of the failing project and within two months began releasing a functioning product in increments to a very surprised client (Highsmith, 2002a:5; Hislop *et al.*, 2002:175; Grey, 2011:33).

FDD is an agile and adaptive methodology for developing ISs and focuses on the design and building phases rather than on the entire IS development process (Abrahamsson *et al.*, 2002:47). It places emphasis on quality aspects throughout the entire development process and includes frequent and tangible deliveries with precise monitoring of the project's progress (Abrahamsson *et al.*, 2002:47). The FDD methodology is constructed according to a core set of best practices that complements and strengthens each other (Palmer & Felsing, 2002:35).

FDD best practices include (Palmer & Felsing, 2002:35: Westfall, 2009:139-140; Grey, 2011:33):

- **Domain object modelling** – The problem domain is defined and investigated to provide a framework where features can be added.
- **Developing by feature** – Progress is monitored using a list of small, functionally decomposed and client-valued functions.
- **Individual class ownership** – An individual is burdened with the responsibility of performance, consistency and conceptual integrity of a particular class.
- **Feature teams** – Small, dynamic teams.
- **Inspection** – Inspection is done using the best-known defect-detection mechanisms.
- **Regular builds** –This practice ensures that there will always be a running basic system available where new features can be added.
- **Configuration management** – The latest versions of each source code file are identified and tracked using configuration management.

- **Progress reporting** – This practice requires that reporting be done at an organizational level on all completed features.

The lead designer dismantles these business best practices into features to plan, design and code these features (Abrahamsson *et al.*, 2002:47). Similar to story cards in XP, features are small pieces that are useful to the users and should be written in a language that all parties can understand and should not take longer than two days (Grey, 2011:33). FDD encompasses iterative development along with best practices to manage and monitor the development process and to deliver completed features to clients in two-week cycles (Abrahamsson *et al*., 2002:47; Hislop *et al*., 2002:175).

Highsmith (2002c:273) mentioned some core values that are suited for developing an IS using the FDD methodology. These values are

- A way for building IS is necessary in order to scale to larger projects.
- A simple, well-defined process works best.
- Process steps should be logical and their value immediately obvious to each team member.
- "Process pride" (when developers believe their process will work, although a much better process for the current situation exists) can keep the real work from happening.
- Good processes move to the background so that team members can focus on results.
- Short iterative, feature-driven lifecycles are best.

FDD consists of five sequential processes, involving techniques, guidelines, roles, goals, artefacts, timelines and methods for developing an IS. FDD is described using the ETVX (Entry criteria, Task, Verification, eXit) scheme (Highsmith, 2002c; Schuh, 2004:28). Figure 2.6 illustrate the five processes of FDD.

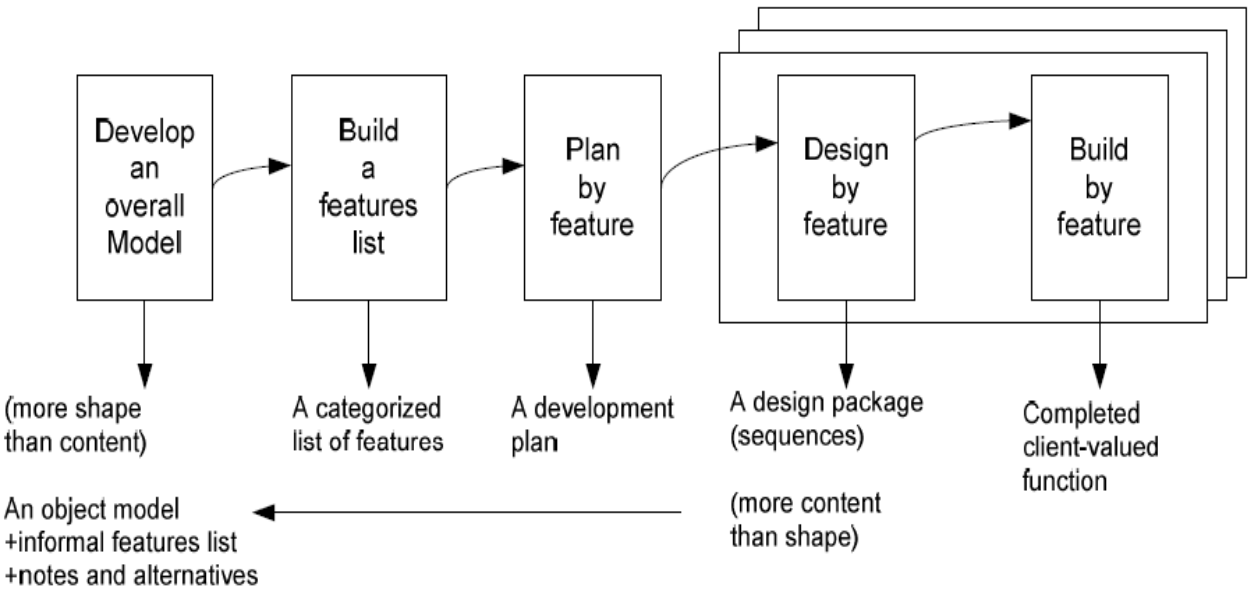


Figure 2.6 – FDD processes (De Luca, 2005:10)

Develop an overall model – In Figure 2.6, the FDD process starts with an in-depth walk-through with the client and in this step the domain experts become accustomed with the scope, context and the requirements of the IS (Haugan, 2011:289: Westfall, 2009:138). Once the domain experts are familiar with the idea that the clients have for the IS, they present a walk-through version of the IS where the chief architect, the one responsible for the overall IS design and the team members are informed of the main requirements and system definition (Abrahamsson *et al*., 2002:52: Westfall, 2009:138). The overall domain of the IS is divided into a number of smaller domain areas and every domain is concerned with building object models for every domain area (Abrahamsson *et al*., 2002:48). The overall model is built by choosing the relevant object models for each domain and this process performs the following tasks (Highsmith, 2002c:275):

- form the modelling team;
- conduct a domain walk-through;
- study the relevant documents;
- develop the model;
- refine the overall project model;
- write model notes; and
- conduct external and internal assessments.

Build a feature list – In this process the information gathered from the walk-through and the defined requirements is used by the project team to create a list of client-valued features included in the IS (Westfall, 2009:138; Haugan, 2011:289; Grey, 2011:35). This list is made up of the main feature sets that represent each of the domain areas and these major feature sets are further divided into sub-sets that "represent different activities within specific domain areas" (Abrahamsson et al., 2002:49). Features that will take longer than ten days to complete are broken down into sub-features (Grey, 2011:35). Once the feature list has been broken down into bite-size chunks, the sponsors and users will review the list to ensure that the list is complete and valid.

Plan by feature – This process includes the creation of a high-level plan, where the features are listed according to their priority and dependency and assigned to the chief programmer (Westfall, 2009:138). Chief programmers are experienced developers who lead small teams in the analysis, design and development of new features. Additionally, chief programmers

assign class ownership and responsibility to other individual developers. During this process, the approximate time for features and feature sets is determined (Rychly & Ticha, 2007:200).

Design by feature and build by feature – These processes are managed by the feature owner and focus on the design, development and implementation of individual features (Rychly & Ticha, 2007:200). The chief programmer selects a small group of features from the feature list by combining features that use the same classes, thus starting the iterative design and build processes (Westfall, 2009:139). The chosen features are planned in more detail, developed, tested, and integrated iteratively within two weeks (Abrahamsson *et al*., 2002:49; Schuh, 2004:29).

Once an iteration, which includes coding, code inspection, testing, and integration, has successfully been completed, the iteration is released and a new one is started. At the start of each iteration the chief programmer chooses a number of features from the feature set/s to be completed during the iteration. Figure 2.7 illustrates the layout of the feature list, feature sets and features. As can be seen in Figure 2.7, the complete feature list consist of a number of subject areas that contains a number of feature sets and finally the feature sets consist of a number of individual features.

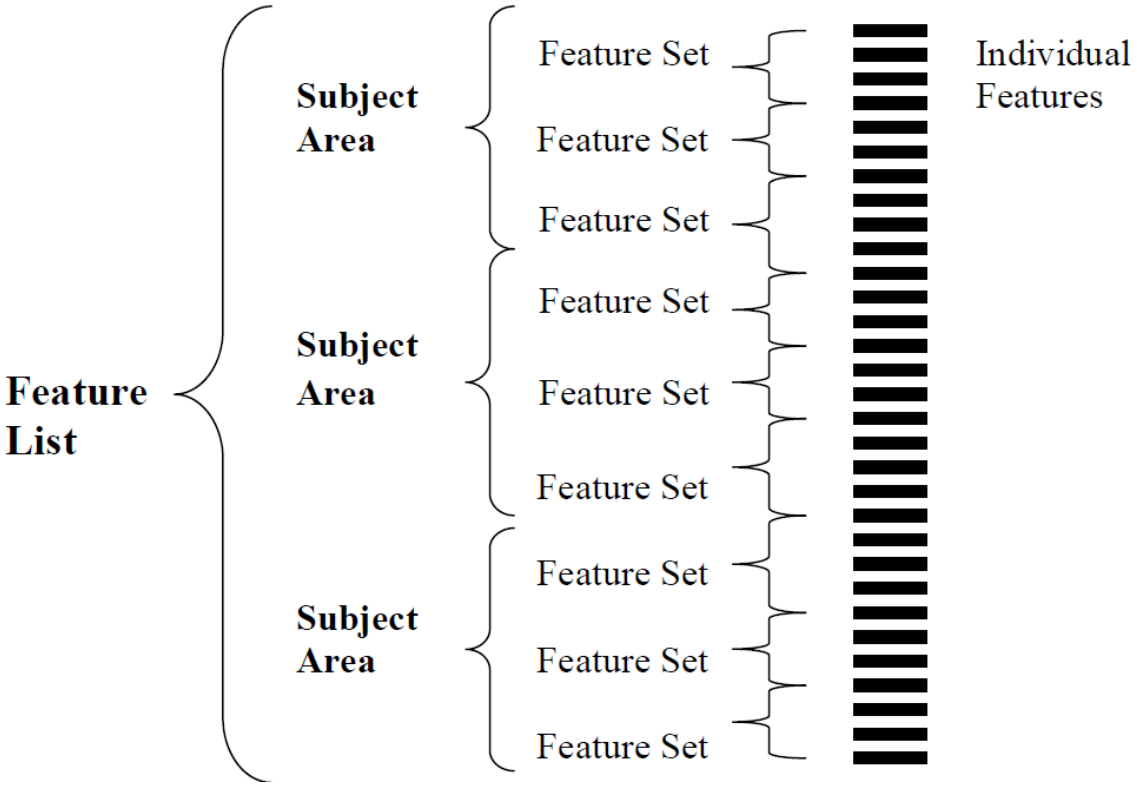


Figure 2.7 – FDD component assembly (Grey, 2011:36)

In the next section the chapter will be summarised and key components identified in the chapter will be highlighted.

2.5 Conclusion

This chapter started off with the expansion of the definition of an SDM by using a metaphor and it was shown that an SDM can be viewed as a backpack that contains a manual, GPS, instructions, and equipment to assist a project team through a maze that is the development of an IS.

After the detailed explanation of the definition of an SDM was discussed, the chapter moved on to the origin and definition of an ASDM and this section revealed that IS developers grew tired of SDMs' inability to deal with changing requirements and uncertainty, which led to the birth of the ASDM that provides a flexible, adaptable, and light-weight approach to the development of ISs. The definition of different authors of ASDM were provided and resulted in the formal definition and viewpoint of an ASDM for this study. The ASDM section can be concluded by mentioning that the direction of moving from SDMs to ASDMs proved to be a move in the right direction with increased project success and user satisfaction by focusing on flexibility and welcoming change (Schwaber, 2001; Hislop *et al.*, 2002; Toflo & Wazlawick, 2008; Grey, 2011; Schwaber & Sutherland, 2013).

The ASDM section is followed by the discussion of the three relevant ASDMs (XP, SCRUM, and FDD) of the study. In this section the following was identified for each of the three ASDMs:

XP is an ASDM that focuses on best practices for the **development** of an IS. It recommends best practices that aim to assist the project team to develop an IS by primarily focusing on techniques to assist with the development aspects but with the secondary focus on the improved management of the team and the requirements and expectations of the user.

SCRUM is an ASDM that places its primary focus on the **management** of IS development. SCRUM introduces best practices to improve the management of the project team, changing requirements, user expectations, and progress. SCRUM focuses on best practices to improve the analysis and planning and management of IS projects and leaves the responsibility of design and development with the project team to decide what works best for it.

FDD is an ASDM that primarily focuses on the **design** and development of an IS project by introducing best practices to assist the project team in **breaking down** the IS into features. Features allow for simplicity in design and development with rapid delivery and quick development.

It can be seen that each of these ASDMs is unique in its own way (XP is development driven, SCRUM is management driven, and FDD is design driven by focusing on breaking down the IS into features) and focuses on different areas of IS development, but all of them adhere to the principles of the Agile Manifesto by focusing on incremental iterative development, short delivery time, flexibility, and welcoming change. Therefore, a combination of these three ASDMs for the development of a DW holds promising potential that may prove to introduce a methodology that is more effective with the combined power of the three ASDMs compared to the individual effectiveness of each of these three ASDMs. The results of combining these three ASDMS will produce a methodology that is management (SCRUM), design (FDD), and development (XP) driven as opposed to only focusing on one of these aspects. Therefore, the ADW methodology will aim to select best practices from each of the three ASDMs to develop an approach that benefits from the power of combining these three ASDMs to result in a an evolved methodology that tackles IS development from multiple angles (management, design, and development).

In the next chapter the awareness stage will be completed by undertaking a literature study on the relevant aspects of data warehousing.

Chapter 3: Data warehouse (DW)

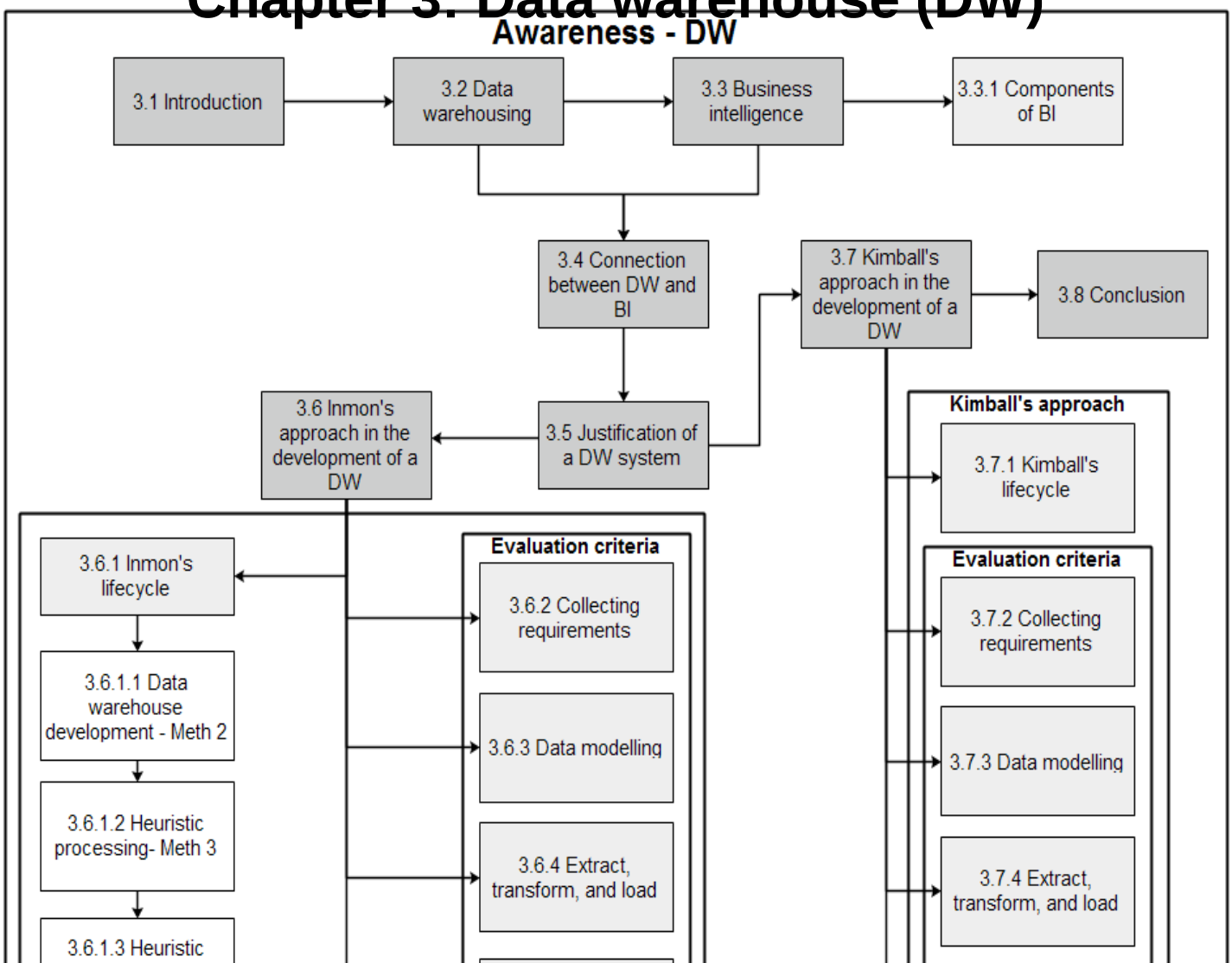


Figure 3.1 – Overview of Chapter 3

3.1 Introduction

In this chapter the origin of the DW system will be discussed, followed by what is meant by business intelligence and what it consists of. Once the origin of DW has been discussed and BI has been defined, the chapter discusses the relationship between the two terms and how they relate to each other. The chapter then describes the justification of a DW system that uses both DW and BI technologies to provide value to an organization.

The rest of the chapter then introduces and explains Inmon's and Kimball's approaches in regard to the lifecycle of their approaches. Each approach is then further discussed according to the four major phases (collecting requirements, data modelling, ETL, and data access and deployment) associated with DW system development.

3.2 Data warehousing

Databases and database theory have been around for quite some time and the main focus of the early database systems was operational (usually transactional) processing (Inmon, 2003:Xiii; Slazinski, 2004:353). According to Inmon (2003:Xiii), a more sophisticated version of the database has surfaced that serves informational or analytical needs and needs to be separated from the operational database. The reason for separating operational and informational databases is (Inmon, 2003:xiii)

- The data serving operational needs is physically different data from that serving informational or analytical need.
- The supporting technology for operational processing is fundamentally different from the technology used to support informational or analytical needs.
- The user community for operational data is different from the one served by informational or analytical data.
- The processing characteristics for the operational environment and the informational environment are fundamentally different.

Therefore if one wishes to develop an informational or analytical system one needs to separate it from the operational database. Inmon (2003:xiii) termed the informational or analytical database/system as a “data warehouse”, which is at the heart of informational, decision support system processing (Inmon, 2003:xiii; Nguyen, 2011:1).

When William H. Inmon (the father of data warehousing (Nguyen, 2011:4)) published his first edition of *Building the Data Warehouse*, database theorists mocked the idea of a data warehouse (Inmon, 2003:xiv). One theoretician said that data warehousing set back the information technology (IT) industry by 20 years and another said that the founder of data warehousing must not be allowed to speak in public. On the release of the second edition the world was all for anything related to the internet and one venture capitalist stated, “Why do we need a data warehouse when we have the Internet?” (Inmon, 2003:xiv).

Regardless of all the negative reviews, data warehousing has surpassed the database theoreticians who wanted to put all the data in a single database and it survived the dot.com tragedy fuelled by the short-sighted venture capitalists (Inmon, 2003:xiv). According to Inmon (2003:xiv), “in an era where technology in general is rejected by Wall Street and Main Street, data warehousing has never been stronger and today data warehousing is not just a theory but a fact of life”.

With the release of the third edition of *Building the Data Warehouse*, Inmon stated that over the years it has been discovered what works and what does not and knows the following with certainty (Inmon, 2003:xv):

- Data warehouses are built under a different development methodology than applications. Not keeping this in mind is a recipe for disaster.
- Data warehouses are fundamentally different from data marts. The two do not mix – they are like oil and water.
- Data warehouses deliver on their promise, unlike many overhyped technologies that simply faded away.
- Data warehouses attract huge amounts of data, to the point that entirely new approaches to the management of large amounts of data are required.

Additionally, one of the most fascinating aspects of a data warehouse that has been learned over the years, is that data warehouses build a foundation for other forms of processing (Inmon, 2003:xv).

In 2008 (Kimball *et al*., 2008:xxxi) stated that there has been remarkable transformation in the last nine years and the DW industry has reached full maturity and achieved acceptance across the business world. Business users are realizing the benefits of high quality data and more importantly, a new name, *business intelligence (BI),* that reflects the true purpose of a DW, has been introduced (Kimball *et al*., 2008:xxxi). In order to emphasize the importance of BI in a DW system, Kimball *et al*. (2008:xxxi) refer to a DW system as a DW/BI system. This shift towards BI puts the mission of a DW, which is the necessary platform for BI into perfect focus (Kimball *et al*., 2008:xxxi). Kimball *et al*. (2008:xxxi-xxxii) said that "of course we strive for world class business intelligence, but world class business intelligence is only possible if you have a world class data warehouse. And conversely, a data warehouse without business intelligence will fail spectacularly". Ranjan (2009:62) agrees with Kimball *et al*. and said, "the main key to a successful BI system is consolidating data from the many different enterprise operational systems into an enterprise data warehouse".

In the next section the term BI will be discussed in more detail.

3.3 Business intelligence (BI)

In an age of deregulation, cost control and high market demands gaining competitive advantage by making good decisions based on complete information have become a necessary requirement (De Silva, 2005:1.1; Nguyen, 2011:1). Management has turned towards IT to give it the strategic advantage within the business arena. Information has become the greatest tool in the marketplace to evaluate both the internal and external

environments and it is through DW and BI that organizations can access this information (De Silva, 2005:1.1; Nguyen, 2011:6).

According to Ranjan (2009:60), the term BI has two different meanings associated with the use of the term intelligence. The first refers to the human intelligence capacity used in business activities. This refers to the investigation of the application of human cognitive faculties and artificial intelligence technologies in regard to the management and decision support for different business issues. The second meaning pertains to intelligence as information valued for its currency and relevance. In other words this relates to expert information, knowledge and technologies efficient in the management of organizational and individual business. Ranjan (2009:60) therefore defines BI as "a broad category of applications and technologies for gathering, providing access to, and analysing data for the purpose of helping enterprise users make better business decisions".

Stackowiak *et al*. (2007:30) define BI as "the process of taking large amounts of data, analysing that data, and presenting a high-level set of reports that condenses the essence of that data into the basis of business actions".

"Business intelligence means using your data assets to make better business decisions. It is about access, analysis, and uncovering new opportunities." (Almeida *et al*., 1999:1)

From these definitions it can clearly be seen that BI uses some form of data in order to perform decision support, on-line analytical processing, statistical analysis, forecasting and data mining (Ranjan, 2009:61).

Therefore, this study will define BI as a component of a DW system that uses the data in the DW database to perform decision support, on-line analytical processing, statistical analysis, forecasting and data mining, which serve as the connection between the user and the data in the DW database. In the next section the components of BI will be discussed.

3.3.1 Components of BI

The components of BI will be mentioned and briefly discussed and these components are (Ranjan, 2009:61-62; Bălăceanu, 2007:70-72):

On-line analytical processing (OLAP)

OLAP refers to how users can navigate through data using advanced tools that assist in the navigation of dimensions, such as time or hierarchies. It provides multidimensional, summarized views of business data that is used for reporting, analysis, modelling and planning for optimizing the business. These tools and techniques can be used with DW or data marts that have been designed for advanced intelligence systems.

Advanced analytics

Advanced analytics relates to data mining, forecasting or predictive analysis and it makes use of statistical analysis techniques to predict or provide certainty measures on facts (Ranjan, 2009:62).

Corporate performance management (portals, scorecards, dashboards)

This component is a general category that typically provides a container where several pieces are plugged into that tells some kind of story. For example, let us say it provides a balanced scorecard that presents portals for financial metrics combined with organizational learning and growth metrics. It serves as the navigation interface for users to provide them with some kind of report, statistical analysis or intelligence.

Real-time BI

This is a simple but vital component that allows for real-time distribution of metrics through e-mail, messaging system and/or user-friendly interactive displays.

Data warehouse and data marts

The reports/information that the BI system produces are/is only as trustworthy as the data from which the conclusions have been drawn and therefore the data warehouse is one the most significant components of a successful BI system. It is responsible for the physical propagation of data by handling the large amount of enterprise records for integration, cleaning, aggregation and query tasks.

A data mart is a collection of subject areas that have been organized for the purpose of decision support based on the unique needs of a particular department. Accounting has its data mart, sales have theirs and marketing has theirs and so forth and the data mart for one department only slightly resembles the data mart of another department. Each department provides its own interpretation of what a data mart should look like and each department's data mart is unique based on its needs. Data marts are similar to a data warehouses due to the fact that they contain operational data that business experts use to make informed decisions based on analyses of historical trends and experience. The primary difference

between data marts and data warehouses is that the design of a data mart is focused on the specific, predefined need for a certain grouping and configuration of selected data.

Data sources

Data sources can be operational databases, historical data, external data or data from an existing data warehouse environment. Data sources can be any form of data that supports the line of business applications, it can be structured information, such as tables or spreadsheets or it can be unstructured information, such as plain-text files or pictures and other multimedia information.

As one might have noticed, DW was mentioned as a component of BI and in the next section the relationship between DW and BI will be discussed.

3.4 The connection between DW and BI

The term BI initially emerged in the 1990's and it referred to the reporting and analysis of data stored in a warehouse (Kimball *et al*., 2008:10). When the term first appeared and caught the attention of the industry's radar, the Kimball group was dumbfounded by the excitement the term generated because they claim that they have been advocating the practices introduced by BI for years (Kimball *et al*., 2008:10). In order to get to the bottom of the excitement that the term BI generated, the Kimball group discovered that many organizations have built DWs as if they were archival librarians that have no intention of delivering the data to the business users in a useful manner (Kimball *et al*., 2008:10). It is due to this confusion regarding what was meant by a DW that the term BI emerged to perform what DW (if done correctly) initially set out to perform. This led to two separate terms that are quite the same and in the next few pages the confusion will be resolved.

Rainardi (2006) stated that many people, even those working in the industry, do not know the difference between DW and BI. Most people use the two terms interchangeably (Kimball *et al*., 2008:10). There are those who say that BI is not just a DW but it is something more, but when asked, 'What BI systems are not DW systems?' or 'What part of BI systems are not data warehouses?' most of them have difficulty answering the question (Rainardi, 2006).

In Chapter 1 (section 1.2.1) a DW was defined as an integrated, subject-oriented, non-volatile and time-variant collection of data from multiple sources that can be loaded, extracted and transformed to provide decision support and preserve historical and current data that can be accessed and analysed.

Previously in this chapter BI was defined as an IS (DW system) that uses some form of data (DW database) in order to perform decision support, on-line analytical processing, statistical analysis, forecasting and data mining

Therefore, a DW is the collection of data that has gone through some kind of transformation to provide the foundation upon which BI can be performed to provide relevant information to business users (Kimball *et al*., 2008:10). Ranjan (2009:64) sums up the definition of a BI system as a "combination of data warehousing and decision support system". Most BI systems are based on DW systems, but there are BI systems that are not or that do not make use of DW systems (Rainardi, 2006; Kimball *et al*., 2008:10). BI systems that do not use a DW system take data directly from the source and analyse the data without any transformation and are very limited due to the data not being in the proper form for unique analysis and the correct term for these systems is BI applications instead of a BI system (Rainardi, 2006). DW systems or rather DW databases can exist without implementing BI and are just a database that uses dimensional storage (Rainardi, 2006). Therefore, DW and BI can be seen as peanut butter and jam, each is enjoyed on a sandwich but when put together they complement each other and produce magic. The terms used for a system that uses both DW and BI technologies can be referred to as a DW system or a BI system as opposed to a DW database (enterprise wide database in the form of dimensional storage) or BI applications (applications that takes data directly from the source to perform BI) (Rainardi, 2006).

Figure 3.2 demonstrates the relationship between a DW database and BI applications/tools to form a DW system.

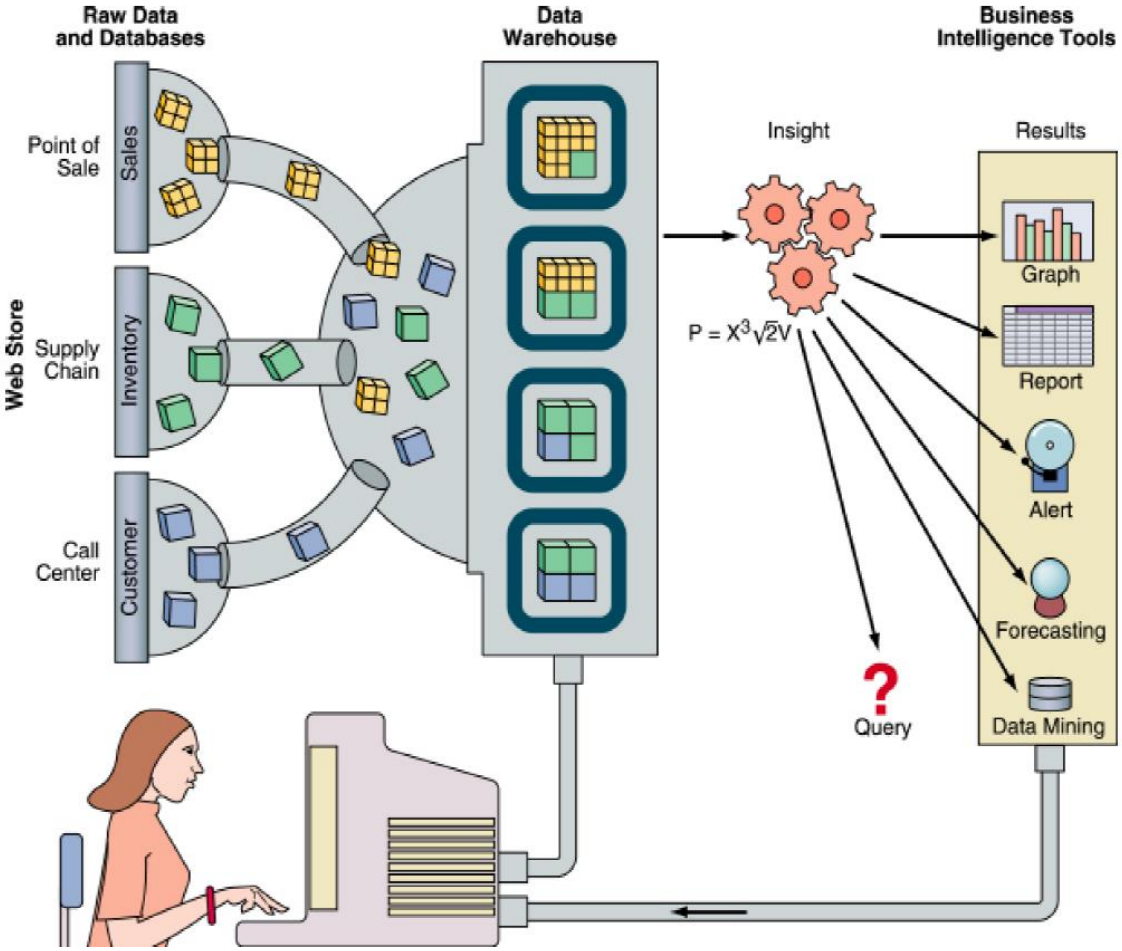


Figure 3.2 – DW/BI system (Ranjan, 2009:64)

From Figure 3.2 it can be seen that the DW database extracts data from multiple sources and transforms this data into a framework upon which BI applications/tools analyse the data to produce the required BI. Together these two components make a DW system that must gather, prepare and analyse the data. The various sources of data must be collected, transformed, cleansed, loaded and stored into the DW database and then the relevant data for a specific business operation is extracted from the DW database and analysed to produce the required reports (Ranjan, 2009:64).

For this study a system will be developed that uses both DW and BI technologies and the term DW system will be used to refer to this system. In the next section the reason and justification for the development of a DW system will be discussed.

3.5 Justification of a DW system

Justification for the development of a DW is a necessary task that is quite difficult because of the range and scope of the business solutions that a DW can bring to the table (De Silva, 2005:1.2) A DW system enables the business users to efficiently analyse large quantities of data, discover hidden opportunities and increase the organization's competitive advantage (De Silva, 2005:1.1). Information for most is considered the second most important resource an organization has second only to people and once a company can use the information to make informed decisions the company can and will increase its performance (Ranjan, 2009:64).

According to De Silva (2005:1.1), "organizations intending to deploy DW/BI can expect not only improved data access, efficiency and customer service, but also reduced costs". A DW system can remove much of the guesswork within an organization, it can improve communication among departments while coordinating activities, and equip companies with the ability to respond quickly to changes in financial conditions, customer preferences, and supply chain management (Ranjan, 2009:64).

(Kimball *et al*., 2008:28) state that a DW system has tangible and intangible benefits and tangible benefits are more critical because they are quantifiable. Below is a list of some of the tangible and intangible benefits of a DW system (Kimball *et al*., 2008:28; De Silva, 2005:1.2; BI-Insider, 2011):

Tangible benefits

- Increased revenue from new sales to new customers.
- Increased revenue from up-selling more expensive products or cross-selling additional products to existing customers.
- Increased response rate to mailings through improved targeting.
- Increased average sale per mailing through improved assortment planning.
- Elimination of low-margin products.
- Decreased raw material costs due to improved supplier management and negotiation.
- Reduced promotional spending or direct marketing costs with more timely midcourse adjustments during the campaign.
- Reduced fraudulent claim expense.
- Reduced customer churn or turnover.
- Reduced defect rates.
- Increased closure rate on first contact.
- Reduced inventory carrying costs or manufacturing overproduction.

- Reduced time to market for new products.
- Improved pricing policies based on more complete revenue and cost information.
- Eliminated cost of producing legacy reports.
- Reduced time to collect data and produce analysis.

Intangible benefits

- More timely access to information.
- Improved decision making based on facts rather than flying blind.
- Greater information standardization and consistency across the organization.
- Greater confidence in the data with fewer debates about accuracy and reduced data reconciliation effort.
- Eliminate inefficiencies by providing one version of the data.
- Faster response to changing conditions.
- Enhanced ability to analyse alternatives.
- Improved employee satisfaction and morale.
- Improved customer satisfaction.
- DW/BI is a strategic necessity to remain competitive.

From the benefits mentioned not all will be present in one organization. Some organizations will experience benefits that are not mentioned above due to the unique business environment and the way it utilizes its DW system, but it can clearly be seen that the development of a DW system is a viable and good investment that will produce a positive ROI.

In the rest of this chapter Inmon's and Kimball's approaches to the development of a DW system will be discussed.

3.6 Inmon's approach to the development of a DW

In this section Inmon's approach will be explained using Inmon's book, *Building The Data Warehouse. 3rd Ed.*

According to Inmon, the development of the operational environment is supported by the traditional system development lifecycle (SDLC) where the development of a DW system requires a different lifecycle known as the CLDS (reverse of the SDLC) (Inmon, 2003:21). The traditional SDLC is a requirements-driven methodology that requires the project team to first and foremost identify the requirements after which the flow progresses to design and then development. The CLDS (Inmon's recommended lifecycle for DW development) is almost the exact reverse of the SDLC, it is a data-driven methodology that once the data is obtained starts with the integration and testing of the data. The CDLC then moves towards the writing

of programmes against the data, which is then analysed and lastly, the requirements of the DW system are understood. (Inmon, 2003:21)

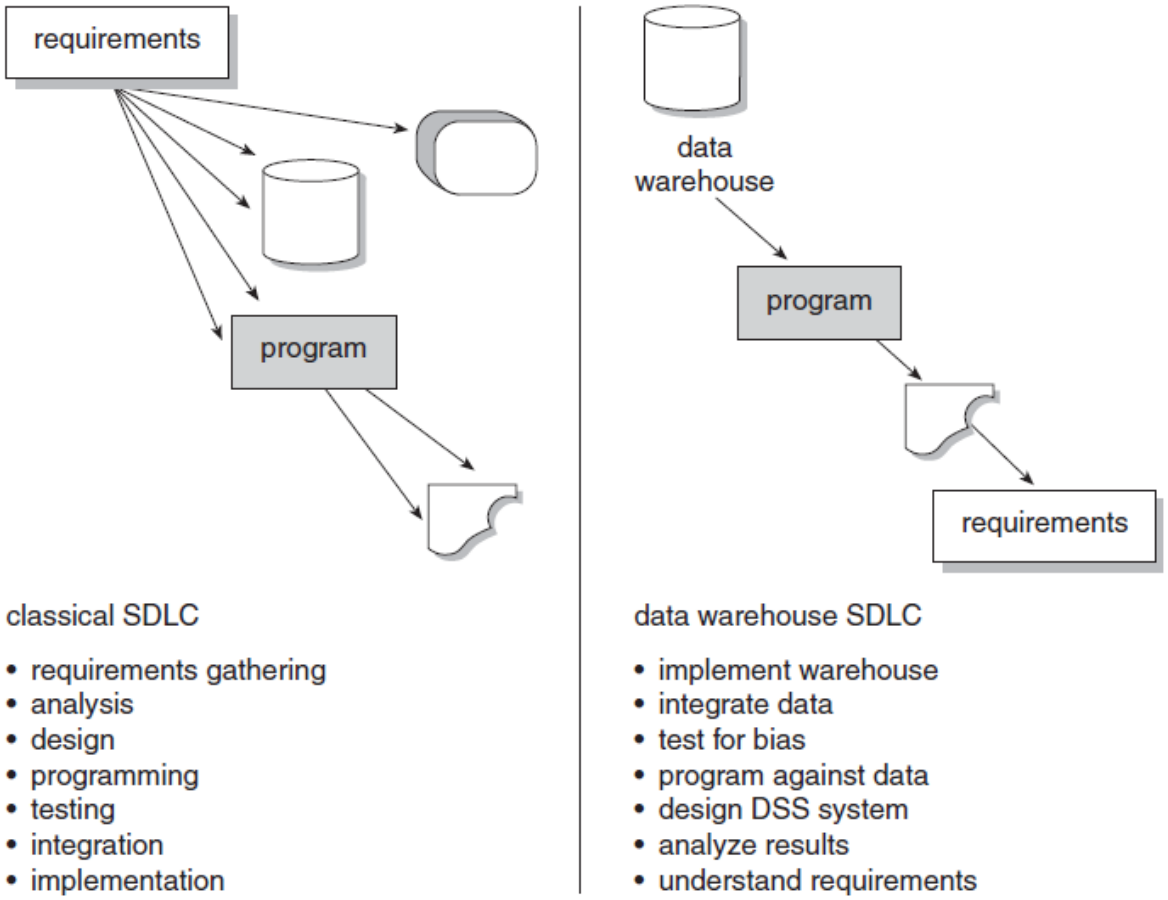


Figure 3.3 – Difference between the operational and DW development lifecycles (Inmon, 2003:22)

3.6.1 Inmon's lifecycle

Inmon's approach introduces an architecture environment that defines the entire database environment of the organization (Inmon, 2003:17):

- **Operational level** – Contains application-oriented primitive data that allows for high-performance transaction processing.
- **Atomic/data warehouse level** – Consists of integrated, historical primitive data that cannot be updated.
- **Departmental level (also referred to as data-mart level)** – Stores derive data that is shaped by end-user requirements for a specific department.
- **Individual level** – This performs heuristics analysis.

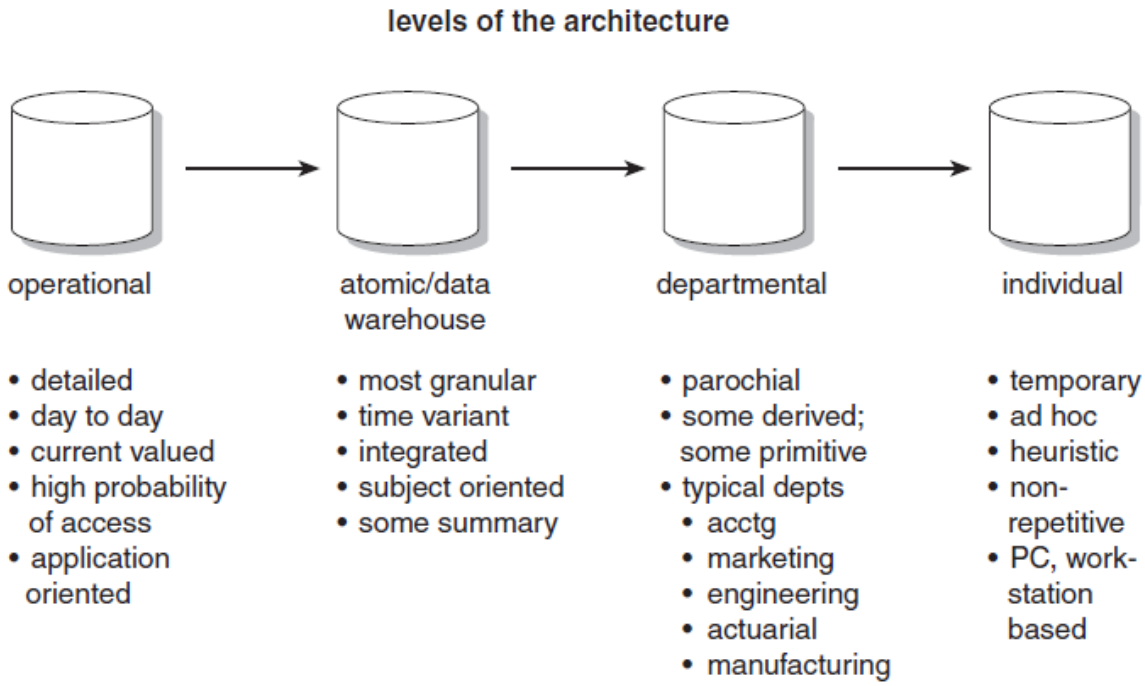


Figure 3.4 – Corporate information factory (Inmon, 2003:16)

Inmon's methodology aims to develop the entire database environment of the organization, including the operational database. Inmon breaks his methodology into a number of methods that flow sequentially from one another in order to create the organizational architecture environment: (Inmon, 2003:343-369)

- **Meth 1** – Developing the operational systems
- **Meth 2** – Data warehouse development
- **Meth 3** – Heuristic processing
- **Meth 4** – Heuristic DSS development

This study is only concerned with the development of the DW system and therefore will briefly discuss Meth 2 (DW database), Meth 3 (departmental BI applications) and meth 4 (individual BI applications).

3.6.1.1 Data warehouse development – Meth 2

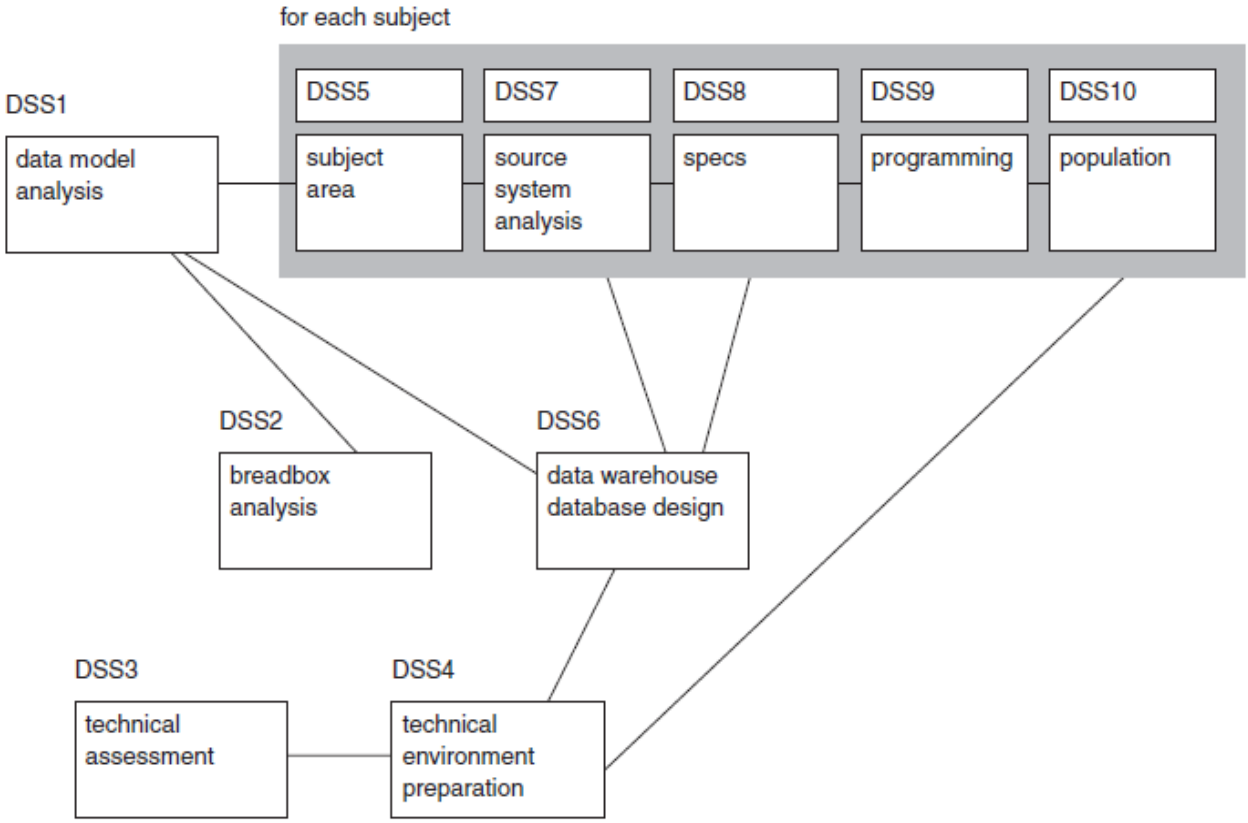


Figure 3.5 – Data warehouse development – Meth 2 (Inmon, 2003:365)

Flow of Meth 2

From Figure 3.5, a relationship can be distinguished between the steps, but a clear flow cannot be determined. In order to clearly demonstrate the flow of meth 2 the predecessor of each step is given below:

- DSS1 - Organization's commitment to develop a DW system.
- DSS2 - Data model analysis step (DSS1).
- DSS3 - Organization's commitment to develop a DW system.
- DSS4 - Technical assessment step (DSS3).
- DSS5 - Data model analysis step (DSS1).
- DSS6 - Data model analysis (DSS1), source system analysis (DSS7) and breadbox analysis steps (DSS2).
- DSS7 - Subject area analysis (DSS5).
- DSS8 - Source system analysis (DSS7) and data warehouse database design (DSS6).
- DSS9 - Specification step (DSS8)
- DSS10 - Programming (DSS9) and technical environment preparation (DSS4) steps.

This description does not provide a decisive flow from DSS1 to DSS10 because of the fact that the project team is burdened with the responsibility of choosing the flow of meth 2 while adhering to the predecessor constraints mentioned above.

DSS1 – Data model analysis

The development of the DW database starts with data model analysis and this step is triggered by the organization's commitment to develop a DW system. This step confirms that the organization has built a solid data model and the model should define the following information (Inmon, 2003:358):

- Major subject area
    - Subtypes of data
    - Attributes of data
    - Clearly defined relationships of data
    - Defined groupings of data
    - Defined keys
- Clear boundaries of the model
- Separate primitive from derived data

The time required to complete this step is dependent on the status and quality of the data model and this step is only performed once. If the data model does not meet the above-mentioned criteria, then the model must be updated to adhere to the criteria before development can continue. (Inmon, 2003:358)

DSS2 – Breadbox analysis

This activity is preceded by the data model analysis activity and is estimated to take from one day to two weeks depending on how well the data model has been defined. This step is concerned with determining the amount of data in terms of number of rows on both a year horizon and a five-year horizon. Using these estimates, the project team must decide whether different levels of granularity will be needed. If this is the case, the project must also define exactly what the levels of granularity are. This activity is only performed once in the development of the DW system. (Inmon, 2003:358-359)

DSS3 – Technical assessment

This step follows the organization’s commitment to develop a DW system and is estimated to take one week to complete. The technical requirements associated with a DW system differ from those of the operational environment. Therefore, an assessment of the technical

environment for the DW system must be performed to ensure that the DW environment satisfies the following aspects (Inmon, 2003:359-360):

- Ability to manage large amounts of data.
- Ability to allow data to be accessed flexibly.
- Ability to organize the data according to a data model.
- Ability both to receive and to send data to a wide variety of technologies.
- Ability to have data periodically loaded *en masse*.
- Ability to access data a set at a time or a record at a time.

DSS4 – Technical environment preparation

This activity follows the technical assessment step and is estimated to be completed in one week to one month and will only be performed once. This step aims to prepare the technical environment so that there will be no technical barriers to the successful development of the DW system. Once this step is complete, the following technical components should be installed, allocated, “burned in” and ready to receive data (Inmon, 2003:360):

- The network.
- Direct access storage device (DASD).
- The operating system managing the DASD.
- The interface to and from the data warehouse.
- The software used to manage the data warehouse.
- The data warehouse.

DSS5 – Subject area

This step is preceded by the data model analysis step and is estimated to take one day and will only be performed once per population project. During this step the project team selects the subject area that must be populated. The development of the first subject area in the DW system must be large enough to make sense and small enough to be implemented. (Inmon, 2003:361)

DSS6 – Data warehouse database design

This step is preceded by the data model analysis, source system analysis and breadbox analysis steps and is estimated to take one to three weeks and will only be performed once. This step uses the data model to design the DW database and the design must include the following characteristics (Inmon, 2003:361-362):

- An accommodation of different levels of granularity if multiple levels exist.
- An orientation of data to the major subjects of the organization.
- The presence of only primitive and publicly derived data.
- The absence of non-DSS (decision support system) data.
- Time variance of every record of data.
- Physical de-normalization of data where applicable
- Creation of data artefacts that move data from the operational environment to the DW.

DSS7 – Source system analysis

The source system analysis is preceded by the subject area analysis and is estimated to take one week per subject area and is performed once for each subject area. During this step, the project team must identify the source data for the selected subject from the existing source system. During this process integration from among different sources must be discussed. (Inmon, 2003:363)

DSS8 – Specifications

This step is preceded by the source system analysis and data warehouse database design steps and is estimated to take one week per extract/integration program and must be executed once for each program that needs to be written. Once the interface between operational and BI environments has been outlined, the project team must formalize the interface by identifying the specifications for the interface. This step involves the following issues that must be addressed by the project team (Inmon, 2003:363):

- How do I know what operational data to scan?
    - Is the operational data time stamped?
    - Is there a delta file?
    - Are there system logs/audit logs that can be used?
    - Can existing source codes and data structures be changed to create a delta file?
    - Do before and after image files have to be rubbed together?
- How do I store the output, once scanned?
    - Is the DSS data pre-allocated, preformatted?
    - Is data appended?
    - Is data replaced?
    - Are updates in the DSS environment made?

DSS9 – Programming

Programming follows the specification step and is estimated to take one week per extract/integration program and must be performed once for each program. This step is concerned with creating the program that performs the extract, transform and load of the data and will consist of all the standard activities associated with programming. Once this step is complete the code that is generated out of this step must be efficient, documented, adaptable, accurate, and complete. (Inmon, 2003:364)

DSS10 – Population

The population step is preceded by the programming and technical environment preparation steps and does not have an estimated time as well as a number of times it must be executed since this step will continue to be performed as long as the DW system continues to grow. This step is concerned with the execution of the DSS programs that have been developed in previous steps and the following issues must be addressed (Inmon, 2003:364-365):

- Frequency of population
- Purging populated data
- Aging populated data
- Managing multiple levels of granularity
- Refreshing living sample data (if applicable)

The completion of this step results in an accessible, comprehensible DW database that serves the needs of the BI community.

This concludes the development of the DW database and the next phase of Inmon's approach will be discussed in the next section.

3.6.1.2 Heuristic processing – Meth 3

The third phase of development for Inmon's architected environment focuses on using the DW database developed in phase two for the purpose of analysis. Therefore, this phase is concerned with the BI aspect of a DW system.

It is important to note that there are a number of differences for the development of the BI layer compared to the DW database layer of the DW system. The first main difference is that the development process always starts with the data in the DW database. Secondly, requirements are not known at the start of the development process. The third and last difference is that processing is done in a very iterative, heuristic fashion. (Inmon, 2003:365)

The steps performed in the BI development layer can be divided into two categories – the repetitively occurring analysis (also referred to as the departmental analysis) and the true heuristic processing (also referred to as the individual level) (Inmon, 2003:366).

Inmon does not discuss this phase further and moves on to the fourth and final phase, which will be described in the next section.

3.6.1.3 Heuristic DSS development – Meth 4

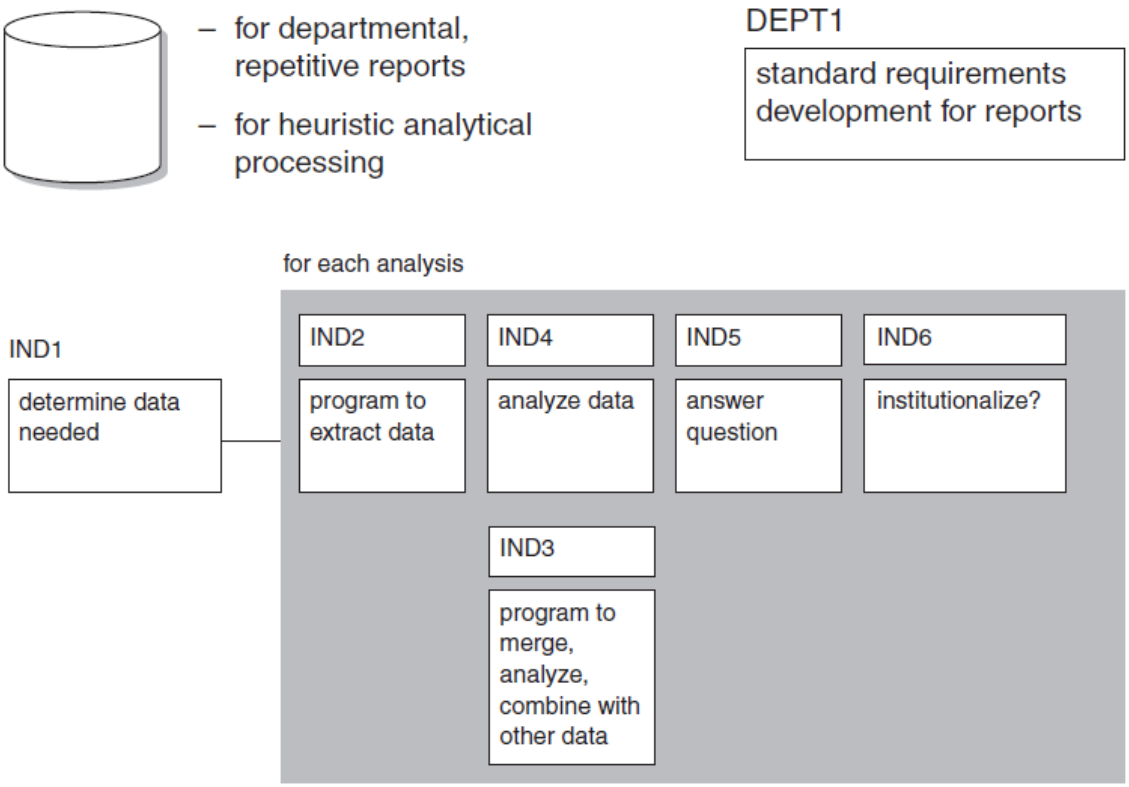


Figure 3.6 – Heuristic DSS development – Meth 4

DEPT1 – Standard requirements development for reports

During this step the repetitive analytical processing is performed using the standard requirements-driven processing used for the development of the operational system from phase 1 (outside the scope of this study). This step aims to create general reports that meet the needs of standards reporting that usually include the following reports (Inmon, 2003:367):

- Accounting reports
- Sales reports
- Key factor indicator reports
- Regulatory reports
- Marketing reports

IND1 – Determine data needed

During this stage of the BI layer development, data from the DW database that may prove to be potential candidates for the satisfaction of reporting requirements are selected. During this step, the developer will work from an educated-guess perspective and for the first two or three iterations only some of the necessary data will be extracted. (Inmon, 2003:368)

IND2 – Program to extract data

Once the project team has decided upon which data is needed for reporting, the next step is to develop a program that can access and strip the data in the DW database. The developed program must be easy to change since it is expected that the initial program will be executed, modified and executed again for a number of times.

IND3 – Combine, merge, analyse

After the selection of the necessary data, it must be prepared for analysis. This usually involves editing the data, data integration and refining the data. As is the case with all heuristic processes, the program developed must be adaptable to change since it is anticipated that the program will be modified and executed a number of times. Upon completion of this step, the project team will have data for the purpose of analysis. (Inmon, 2003:368)

IND4 – Analyse data

This step analyses the prepared data and asks the question, “Do the results obtained meet the needs of the analyst?” (Inmon, 2003:368). If the results do not meet the analyst’s needs, then a new iteration is triggered in order to get results that meet the analyst’s needs. If the results do, however, meet the needs of the analyst, the final reporting preparation is started. (Inmon, 2003:368)

IND5 – Answer question

This step is self-explanatory and simply just answers the question posed in the previous step. The final report that is developed is usually the end product of many iterations of processing. It rarely occurs that the final report is the result of a single iteration of analysis. (Inmon, 2003:368)

IND6 – Institutionalization

The final step in the development of the BI layer of the DW system is to decide whether the final report developed should be institutionalized. If there is a need to repetitively run the final report, it will be best to submit the report as a set of requirements. Once it has been selected

as a requirement, the report must be re-developed as a regularly occurring operation. (Inmon, 2003:369)

This concludes the lifecycle of Inmon's methodology. Since it is one of the objectives of this study to compare Inmon's approach to Kimball's approach, a more detailed description of Inmon's and Kimball's approaches will be discussed in regard to the four main stages associated with DW system development. These stages are: collecting requirements, data modelling, ETL (extract, transform and load) and data access and deployment (Grey, 2006:122). The reasons behind choosing these four stages are based on the fact that they are the only phases/stages involved in DW development that are present in both Inmon's and Kimball's approaches and therefore form the foundation of the evaluation between Inmon and Kimball's approaches (Grey, 2006:122). In the next section, Inmon's approach will be discussed in more detail in regard to collecting requirements.

3.6.2 Collecting requirements

Inmon's approach is a data-driven methodology that starts with the available data and works its way down the development process to the last phase where requirements are identified. This approach differs greatly from traditional requirements-driven methodologies where requirements are identified and a system is developed to satisfy the identified requirements. The reason for Inmon's radical approach to the development of a DW system is due to the fact that Inmon (2003:81) believes "The requirements for the data warehouse cannot be known until it is partially populated and in use and design approaches that have worked in the past will not necessarily suffice in subsequent data warehouses".

Inmon's approach first populates one portion of the DW database after which the data is used and scrutinized by the DSS analyst. Only after the DSS analyst has analysed the data does Inmon's approach attempt to collect requirements using a feedback loop (Figure 3.7). Based on the requirements collected during the feedback loop, data is modified and/or additional data is added. Inmon's approach then starts a new iteration where another portion of the DW system is developed. This feedback loop continues throughout the entire life of the DW. (Inmon, 2003:81)

According to Inmon (2003:86), the feedback loop is at the heart of success in the long-term development of a DW system and failing to implement the feedback loop greatly decreases the probability of success. The feedback loop occurs between the DSS analysts and the data architect and the following observations in regard to the feedback loop are of vital importance to the success of the DW project (Inmon, 2003:287):

- The DSS analyst operates – quite legitimately – in a “give me what I want, then I can tell you what I really want” mode.
- The shorter the cycle of the feedback loop, the more successful the data warehouse effort.
- The larger the volume of data that has to be changed, the longer the feedback loop takes.

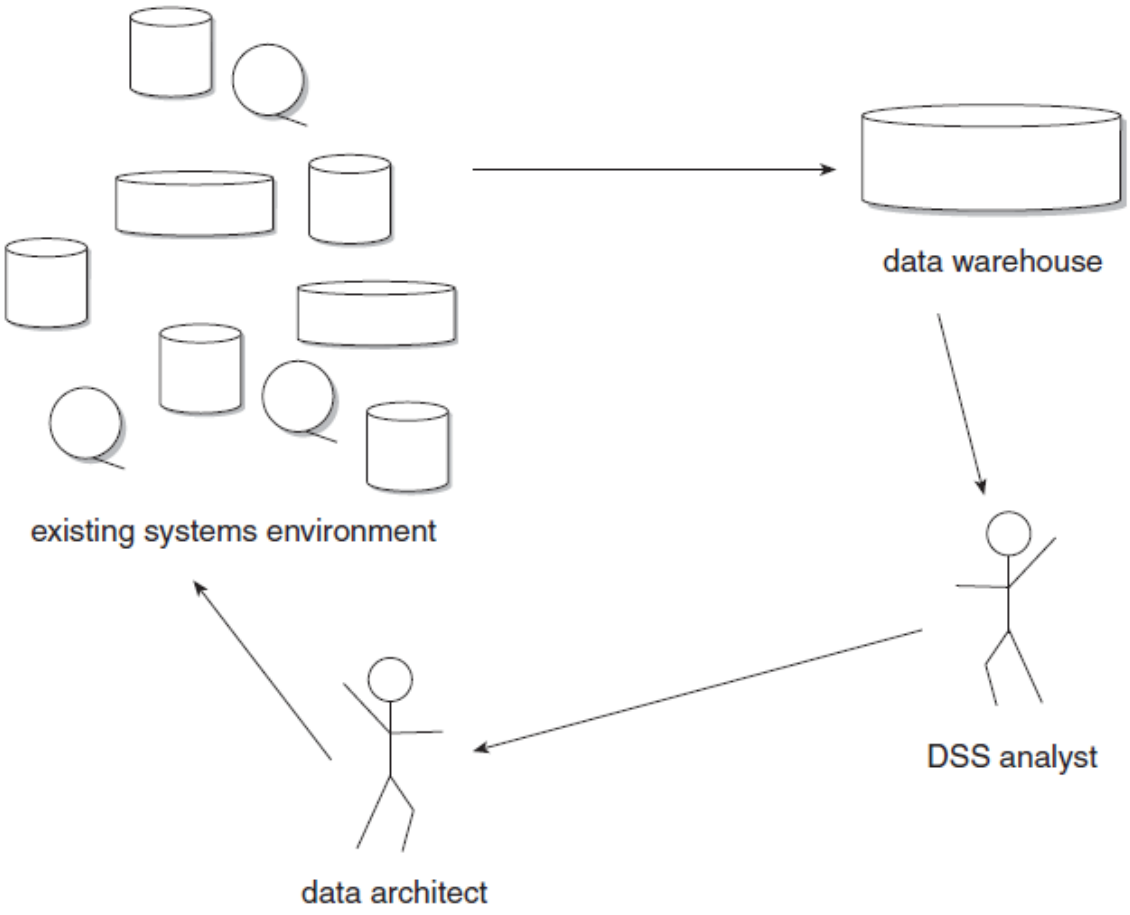


Figure 3.7 – Feedback loop between the DSS analyst and the data architect (Inmon, 2003:286)

3.6.3 Data modelling

As previously mentioned, Inmon’s approach uses an architecture environment that envelops the entire information track of the organization. Therefore, Inmon’s data model applies to both the operational and DW environments of the organization (see Figure 3.8) (Inmon, 2003:89). Inmon’s data model begins by creating a corporate data model that focuses on and presents only primitive data and serves as the starting point for the construction of separate existing data models (Inmon, 2003:89). When developing the operational data model, very few changes are made to the corporate data model, but there are a number of changes made to

the corporate data model when applied to the DW environment. The transformation of the corporate data model into the DW data model involves the following changes: (Inmon, 2003:89-90)

- Remove data that is only relevant in the operational environment.
- Enhance key structures of the corporate data model with the element of time.
- Add derived data that is publicly used and calculated only once.
- Change data relationships in the operational environment into artefacts.
- Perform stability analysis – This involves the grouping of data attributes based on their natural tendency for change.

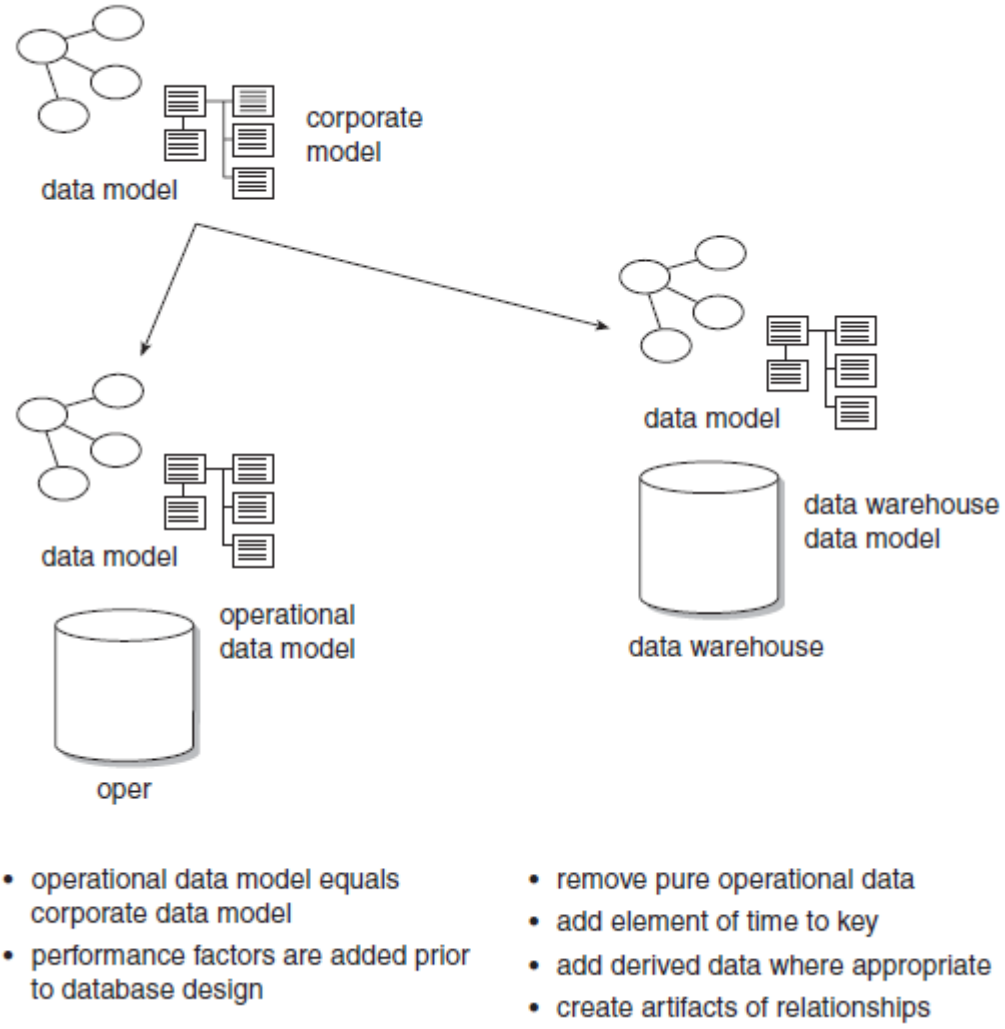


Figure 3.8 – Corporate data model (Inmon, 2003:90)

This study is only interested in the creation of the DW data model and therefore will only discuss the DW data model. Inmon's approach involves three levels of data modelling for the development of the DW data model. These levels are (Inmon, 2003:92):

- High-level modelling - Entity relationship level (ERD).
- Mid-level modelling – Data item set (DIS).
- Low-level modelling – Physical model.

The high-level modelling is performed by creating an ERD with entities and relationships. The name of each entity must be surrounded by an oval and relationships are shown with arrows. The direction and the number of arrowheads indicate the cardinality of direct relationships. The entities included in the ERD level must be at the highest level of abstraction. The data modelling team must perform “scope of integration” to determine which entities must be included in the current DW data model. (Inmon, 2003:92-94)

Once the high-level modelling has been completed, the modelling team moves towards the mid-level model where a model is created for each major subject area (entity) identified in the high-level data model. During the mid-level modelling, the project team will focus on a single entity at a time where the selected entity is expanded and fleshed out while the other entities remain static until the focus is directed towards them. The mid-level modelling consists of four constructs that make up the mid-level model. These constructs are shown below in Figure 3.9. (Inmon, 2003:94-95)

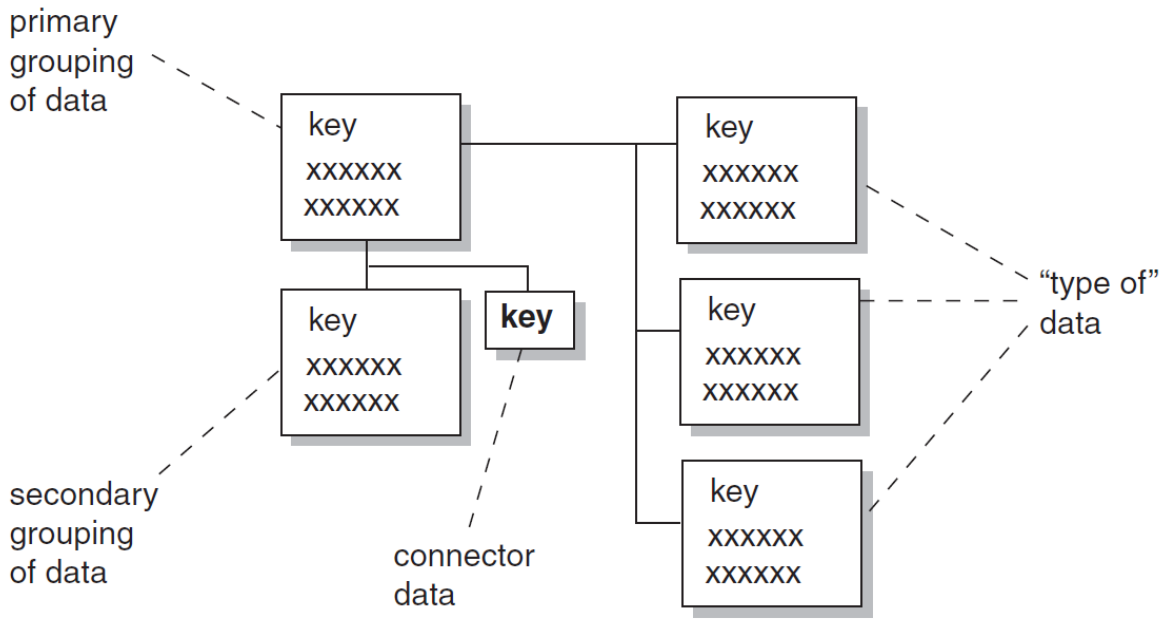


Figure 3.9 – The four constructs that make up the midlevel data model. (Inmon, 2003:95)

As can be seen from Figure 3.9, the four constructs that make up the mid-level data model are (Inmon, 2003:95-96):

- **The primary grouping of data** – The primary grouping of data consists of attributes and keys for each entity that exists only once.

- **The secondary grouping of data** – The secondary grouping contains attributes for each entity that may exist more than once. This grouping is indicated by the emanating downward line from the primary grouping as seen in Figure 3.9. In the mid-level data model there may be as many secondary groupings as distinct groups of data that can appear more than once.
- **A connecter -** As the name implies, this construct simply connects data from one grouping to another. This is indicated with a foreign key that is underlined.
- **"Type of" data –**The fourth construct is the "Type of" data that is indicated by a line towards the right side of the model that is a grouping of sub-type data for the grouping on the left (super-type data), which indicates the type of data.

These four constructs are used to identify the attributes and the relationship between the attributes based on the ERD developed in the high-level model.

After the development of the midlevel data model, the project team can begin with the low-level data model that is developed from the mid-level model by simply extending the mid-level model to include keys and physical characteristics. Once this is done the low-level (physical) model looks like a group of tables that is sometimes referred to as relational tables. At this point, it is tempting to declare the model as complete and to begin with the physical development of the database, but Inmon (2003:99) says, "one last design step remains – factoring in the performance". This last step is performed by deciding on the granularity and partitioning of the data along with the addition of the element of time. (Inmon, 2003:98-99)

3.6.4 Extract, transform and load (ETL)

According to Inmon (2003:118), the ETL process "first appears to be nothing more than the movement of data from one place to another" but this assumption "quickly turns into a large and complex task – far greater and more complex than the programmer thought". The ETL process is not just complex and time consuming but also requires that three types of load be performed to populate the DW system (Inmon, 2003:94):

- Initial load of archival data.
- Loading of data currently found in the operational environment.
- Loading of on-going changes to the DW system from changes that have occurred in the operational environment since the last refresh.

Inmon does not provide a step-by-step process in order to perform the ETL process, but mentions that the following necessary functionality is required for the ETL of data from the legacy environment to the DW environment (Inmon, 2003:118-121):

- **Change in technology** – The extraction of data from the legacy system towards the DW system requires a shift in technology to meet the specifications for a DW system.
- **Data selection** – Data selection from the legacy system can prove to be a complex activity that requires several coordinated lookups, keypad reads, connecting logic, and so forth.
- **Restructure and convert operational input keys** – In almost all cases, the operational input keys need to be changed when loaded into the DW system. In simple cases an element of time is added to the structure of the key but there are cases that require the entire input key to be restructured.
- **Transformation of non-key data** – As data is moved from the legacy system towards the DW environment, non-key data must be reformatted. A simple example can be the transformation of the default date YYYY/MM/DD in the legacy system to DD/MM/YYYY. Transformation of the operational data is usually much more complex than the example given in the previous sentence.
- **Data cleansing** – During the ETL process data must not just be transformed but also cleansed, which most often includes domain checking, cross-record verification and simple formatting verification.
- **Multiple input sources integration** – Data from multiple sources must be integrated when extracted from the operational sources and loaded into the DW system.
- **Key resolution** – In the case of multiple sources, key resolution must be performed before the data can be integrated.
- **Re-sequencing of input files** – In the case of multiple sources (almost always the case), the sequence of the different files may not always be the same or even compatible. Therefore, the input files need to be re-sequenced.
- **Different levels of summarization** – During the ETL process, data can be produced at different levels of summarization that results in multiple outputs by the same ETL process.
- **Assign default values** – In some instances, the data in the DW system will have no source and result in a NULL value. In order to produce more meaningful data, a default value must be specified for these occurrences.
- **Input data efficiency**–The efficiency of the selected input data for extraction usually becomes a problem that must be addressed. There are cases where the necessary operational data needed for extraction cannot be identified, which results in a complete file read that affects the efficiency of the DW system. This must be considered and dealt with accordingly.

- **Data summarization** – Multiple operational records must be combined into a single DW record and thus require data summarization.
- **Data elements renaming** – Data elements are usually given new names as they are loaded into the DW system and these changes must be tracked and documented.
- **Input records conversion** – Usually, input records have non-standard formats that must be converted to a standard format before being loaded into the DW system.
- **Deciphering of data relationships** – Data relationships in the operational system must be understood and unravelled before the data can be used as an input for the DW system.
- **Data format conversion** – Some cases will require data format conversion, such as EBCDIC to ASCII, which must be spelled out.
- **Account for large volumes** – In the event of large volumes of input, special design options may have to be used.
- **Conform to corporate data model** – The design of the DW must conform to a corporate data model which leads to order and discipline in the design and structure of the DW.
- **Addition of the element of time**–The DW system represents the historical need for information and might require an element of time to be added as data moves from the operational environment into the DW environment.
- **Address informational needs** – While the operational system meets the clerical needs of an organization, the DW system must address the informational needs.
- **Transmission of output file** – The transmission of newly developed output files that will be loaded into the DW system must be accounted for. In some cases, this will not be an issue, but there are cases where the operational system and the DW system are hosted on different operating systems. In this case, issues, such as where the transformation will take place must be addressed.

Although no step-by-step process is provided to perform the ETL process, Inmon does recommend the use of ETL software that automates the process of converting, reformatting and integrating data from multiple operational sources (Inmon, 2003:122). Inmon (2003:122) declares that “Only under very unusual circumstances does attempting to build and maintain the operational/data warehouse interface manually make sense”.

3.6.5 Data access and deployment

According to Inmon (2003:29), the data in a DW system is either accessed through direct or indirect access.

Direct access

Direct access to the data in a DW system occurs when the operational environment requests data that is located in the DW system. The request is shifted from the operational environment to the DW system where the data is located and transferred to the operational environment. According to Inmon (2003:129), there is a number of serious limitations in the event of direct access to the DW system. Some of the main limitations are (Inmon, 2003:129):

- The response must be casual in regard to response time.
- The request must be for a minimal amount of data.
- The technology managing the DW system must be compatible with technology managing the operational environment in terms of capacity, protocol and so forth.
- The formatting of data after it is retrieved from the data warehouse in preparation for transport to the operational environment must be non-existent (or minimal).

These limitations prevent most data from being directly transferred from the DW system to the operational environment and therefore there is a minimal amount of data retrieval from the DW system in respect of direct access (Inmon, 2003:130).

Indirect access

Indirect access of the data located in the DW system by the operational environment is one of the most effective uses of a DW system (Inmon, 2003:130). Indirect access is performed by periodically analysing the DW database using a program that examines relevant characteristics and criteria. Using the analysis, the program then creates a small file in the on-line environment that contains clear and precise information about the business of the organization. The small on-line file is then used to perform analysis efficiently and timely in the operational environment. (Inmon, 2003:136)

According to Inmon, the following element for the indirect use of data in the DW database must be considered (Inmon, 2003:136-137):

- The analysis program
    - consists of many characteristics of artificial intelligence;
    - has access to run on all the available data in the DW database;
    - must be run in the background so that processing time is not a big issue.
- The periodic refreshment
    - occurs infrequently;
    - must operate in a replacement mode;

    - handles the shift of data from the technology supporting the DW to the technology that supports the operational environment.
- The on-line pre-analysed data file
    - must contain only a small amount of data per unit of data;
    - must be able to contain a large collection of data (there may be a large number of units of data);
    - must contain exactly what the online clerk needs;
    - must not be updated, but must be periodically refreshed on a wholesale basis;
    - must be part of the online high-performance environment;
    - must be efficient to access;
    - must be designed for access of individual units of data, not massive sweeps of data.

In Inmon's lifecycle (discussed in section 3.5.1), unit testing must be performed before the DW database is populated and deployed. Once the DW database is deployed and the first iteration of the BI layer is developed, the feedback loop (discussed in section 3.5.2) is triggered to identify requirements and may also serve as acceptance testing to ensure the users get what they want (Inmon, 2003:287). Inmon's lifecycle also mentions that a user manual should be created as well as the actual training of users to ensure the users understand and will be able to use the DW system (Inmon, 2003:357).

In the next section Kimball's approach will be discussed in a similar manner as Inmon's approach.

3.7 Kimball's approach to the development of a DW

In this section Kimball's approach will be explained using the Kimball group's book, *The data warehouse lifecycle toolkit. 2nd ed.*

The Kimball lifecycle methodology began to take shape in the 1980's at Metaphor Computer Systems (Kimball *et al*., 2008:5). At the time there was no formal methodology or best practice for developing decision support systems, but among the employees working at Metaphor, the sequential steps for decision support systems were as obvious then as they are now; the training manual at Metaphor described these steps as *extract, query, analysis* and *presentation* (Kimball *et al*., 2008:5). In 1998 the Kimball group introduced their methodology in the first edition of their book and decided that the official name of the methodology be the Business Dimensional Lifecycle but this ended up being a mouthful and people in the industry simply referred to the methodology as the Kimball approach (Kimball *et al*., 2008:6).

The Kimball group felt strongly and still feel that successful DW systems depend on three fundamental concepts (Kimball *et al*., 2008:6):

- Focus on the business.
- Dimensionally structure the data that's delivered to the business via *ad hoc* queries or reports.
- Iteratively develop the overall data warehouse environment in manageable lifecycle increments rather than attempting a galactic big bang.

Regardless of substantial advancements in technology and understanding, the basic components of the Kimball lifecycle have remained surprisingly constant (Kimball *et al*., 2008:6). The Kimball group explicitly states that "Our approach to designing, developing, and deploying DW/BI solutions is tried and true. It has been tested with projects across virtually every industry, business function, and platform." (Kimball *et al*., 2008:6). The Kimball group's motto is "practical techniques, proven results" and the methodology has proven to work again and again (Kimball *et al*., 2008:6).

The Kimball approach will be explained by using Kimball's lifecycle diagram in the following section.

3.7.1 Kimball's lifecycle

The successful development of a DW system depends on the integration of a number of necessary tasks and components (Kimball *et al*., 2008:6-7). Kimball's lifecycle diagram shows the flow of high level tasks that are necessary for the successful design, development and deployment of a DW system. The diagram shows the overall roadmap of Kimball's approach and each box serves as a guidepost or milestone in the development of a DW system (Kimball *et al*., 2008:7). Each milestone will briefly be described.

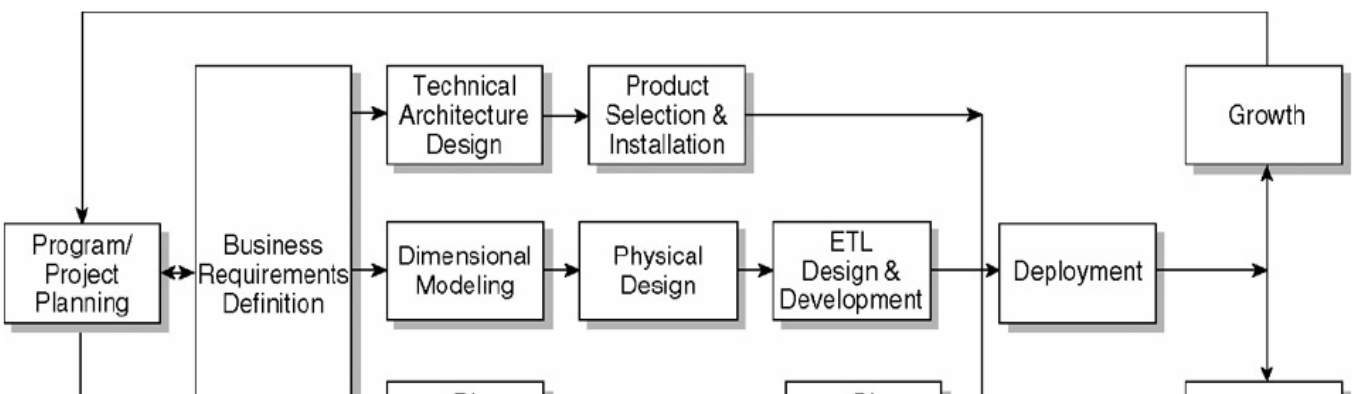


Figure 3.10 – Kimball's lifecycle diagram (Kimball *et al*., 2008:7)

Program/project planning

Kimball's lifecycle begins with program and project planning and project refers to a single integration of the lifecycle from the initial start all the way to deployment. Program refers to a broader, on-going coordination of resources, infrastructure, timelines and communication across a number of projects; it is the big picture view of the DW system and encompasses more than one project. Projects have a finite start and end where a program should continuously renew itself allowing the DW system to continue to grow and should rarely have an abrupt end. (Kimball *et al*., 2008:7)

This milestone begins with creating the scope of the DW project, but in order to make suitable scope decisions, a basic understanding of the business's requirements is needed. Therefore, there is a dependency between the program/project planning and business requirements definition milestones, which can be seen by the bi-directional arrow in Figure 3.10. In addition, project planning includes resource staffing, project task identification, assignment, duration and sequencing. In the end, this milestone produces a project plan that identifies all the tasks associated with Kimball's lifecycle and those responsible for each task and serves as the cornerstone for continuous management of the DW system. (Kimball *et al*., 2008:7-8)

Program/project management

The program/project management milestone, as one might have guessed, ensures that Kimball's lifecycle tasks stay on track and in sync. It is concerned with monitoring project status, issue tracking, and change control to maintain the scope boundaries. Continuous management includes the development of a comprehensive communication plan in order to address both the business and information technology constituencies. Kimball's lifecycle believes that continuous communication is a critical success factor to managing expectations.

Program/project planning is an on-going process and encompasses the entire lifecycle from planning to deployment. (Kimball *et al*., 2008:8)

Business requirements definition

The success of the DW project is directly linked to a sound understanding of the business users and their requirements. Without a clear understanding of the requirements the development of the DW system will be futile for the project team. Kimball's approach is a requirements-driven methodology that differs greatly from traditional data-driven requirement-analysis approaches. The DW analysis must understand the main driving factors of the business in order to successfully implement business requirements into design consideration. Kimball's approach aims to meet user satisfaction by focusing on the requirements set in place by the users to develop a DW system that will provide value to the business. The business requirements definition forms the foundation for all downstream lifecycle tasks and therefore, is a crucial milestone that must executed effectively. Requirements gathering will be discussed further in section 3.7.2. (Kimball *et al*., 2008:8)

Technology track

Once a clear understanding of the lifecycle is complete, the lifecycle flows into three concurrent tracks focusing on technology, data and business intelligence applications. The lifecycle diagram (Figure 3.10) directs the flow of activities along each of the parallel tracks; the three tracks contain some dependencies which are illustrated by the vertical alignment of the milestone boxes. (Kimball *et al*., 2008:8)

The technology track is the first of the three tracks and introduces the technical architecture concepts and is concerned with the process of designing the architecture and selecting products to instantiate it (Kimball *et al*., 2008:8).

Technical architecture design

A DW system requires the integration of a number of technologies. This milestone is concerned with defining the overall architectural framework and vision of the DW system. In order to establish a suitable DW system technical architecture design, three factors, namely the business requirements, current technical environment and planned strategic technical directions must be considered. The project team must steer clear of focusing on technology in isolation. (Kimball *et al*., 2008:8-9)

During this milestone the enterprise DW bus matrix is created from the preliminary DW bus matrix that was created in the business requirements definition milestone. The bus matrix

document is the main business processes and their functionality. The DW bus diagram is created from the bus matrix and is a visual interpretation of the matrix that represents the organization's key business processes and their associated dimensionality.

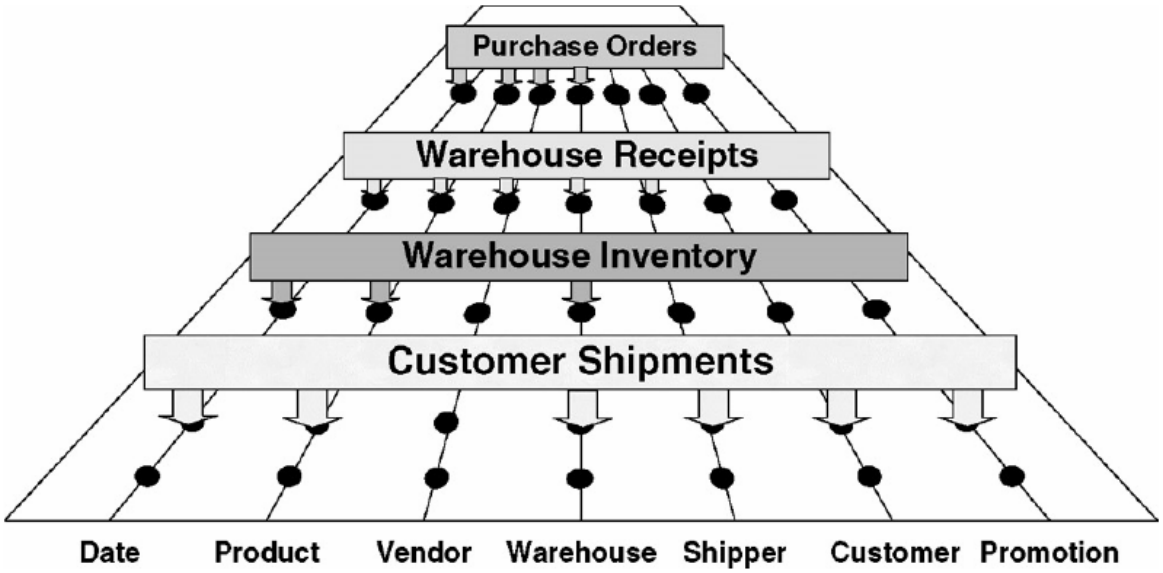


Figure 3.11 – Enterprise wide DW bus diagram (Kimball *et al*., 2008:217)

Each row in the bus diagram represents a business process and each column is the dimensions of the DW system. The downward arrows in the bus diagram indicate the relationship between the business process and the dimensions for the overall DW system. During the dimensional modelling milestone, one row at a time will be modelled and developed using incremental development until the entire DW system is complete (development of all the rows in the bus diagram). (Kimball *et al*., 2008:215-220)

Product selection and installation

In this milestone the evaluation and selection of specific architectural components, such as the hardware platform, database management system, ETL tool or data access query, and reporting tool must be performed by using the technical architecture as a virtual shopping list of needed components. Once the project team has decided upon products, these products must be installed and tested to verify end-to-end integration within the DW system environment. (Kimball *et al*., 2008:9)

Data track

The second parallel track that flows from the business requirements definition milestone is the data track. The data track consists of the design of the target dimensional model, physical instantiation of the model, and the extraction, transformation, and load into the target models.

Dimensional modelling

During the business requirements definition milestone, the business's data needs are identified and documented into a preliminary enterprise DW bus matrix that relates to the organization's main business processes and their related dimensionality. The bus matrix provides a data architecture blueprint to ensure that the DW system can be integrated and extended across the organization over time.

Dimensional models that support the organizations reporting and analytical needs require a different method than the methods used for transaction processing design. Dimensional modelling focuses on more detailed data analysis of a single business process (row in the enterprise DW bus matrix). During this analysis, the grain of the fact table, related dimensions and attributes, and numeric facts are identified. Dimensional modelling is further discussed in section 3.7.3. (Kimball *et al*., 2008:9)

Physical design

This milestone aims at defining the physical structure, setting up the database environment, and defining the necessary security. Although the physical data model of the relational database is logically identical to the dimensional model, there are additional issues to confront, such as the preliminary performance tuning strategies, from indexing to partitioning and aggregation. If necessary, OLAP databases are also designed during this milestone.

ETL design and development

This milestone is one of the most frustrating challenges tackled by the DW project team; even if all the previous milestones have been well planned and executed. This milestone contains 70% of the risk and effort associated with DW system development. This milestone is further discussed in section 3.7.4. (Kimball *et al*., 2008:10)

Business intelligence application track

The last of the concurrent tracks is the business intelligence application track and focuses on the BI application.

BI application design

While other team members are working on the technology and/or data tracks, members involved in the BI application track should work with the organization to determine suitable BI application candidates as well as to identify necessary navigation interfaces that will address the business users' needs and capabilities. BI applications are the demonstration layer of the DW system that provides value to the business. BI applications serve as the bridge between the users and the data in the DW database and allow business users to query, analyse, and generate reports on the data. (Kimball *et al*., 2008:10)

BI application development

Following the BI application design specification, BI application development focuses on configuring the business metadata and tool infrastructure as well as constructing and validating the specified analytic and operational BI applications, along with the navigational portal. (Kimball *et al*., 2008:10)

Deployment

The three parallel tracks meet at the deployment milestone and extensive planning must take place to ensure that the three tracks are tested and fit together properly. Extensive planning must also be carried out to determine the necessary education and support infrastructure. It is a critical success factor during this milestone that deployment be well orchestrated and deployment must be delayed if all the pieces, such as training documentation and validated data, are not ready for release. (Kimball *et al*., 2008:10)

Maintenance

Once the DW system has been deployed and delivered, technical tasks are required to keep the system performing at its maximum potential, which includes usage monitoring, performance tuning, index maintenance, and system backups. Additionally, the project team must provide the business users with on-going support, education and communication. (Kimball *et al*., 2008:10-11)

Growth

The final milestone in Kimball's lifecycle is the growth milestone that allows the DW system to grow which triggers a new iteration of the lifecycle where the entire lifecycle for the new business process is run from program/project planning milestone to the growth milestone, which builds upon the existing DW system. Kimball *et al*. (2008:11) state that if the DW system is successful, it is bound to expand and evolve to provide additional value to the business.

This concludes the general explanation of Kimball's lifecycle and as is the case with most approaches one may need to customize Kimball's lifecycle to address the unique development environment for the particular DW system (Kimball *et al*., 2008:11). Kimball's approach attempts to address all the activities associated with DW system development in a broader sense and not all DW projects will require every task and subtask to be performed in order to successfully complete a DW project (Kimball *et al*., 2008:11).

Similar to the discussion of Inmon's approach, Kimball's approach will be discussed in regard to the four stages (discussed in section 3.6.1) of DW development.

3.7.2 Collecting requirements

The collection of requirements is one of the most crucial phases and impacts on almost every decision made during the design and implementation of a DW system (Kimball *et al*., 2008:57). Kimball *et al*. (2008:57) claim that "From our perspective, business requirements sit at the center of the universe, because they are so critical to successful data warehousing". The development team's understanding of the user requirements will affect most lifecycle choices, from establishing the right scope, modelling the right data, selecting the right tools, applying the right transformation rules, building the right analyses, and providing the right deployment support.

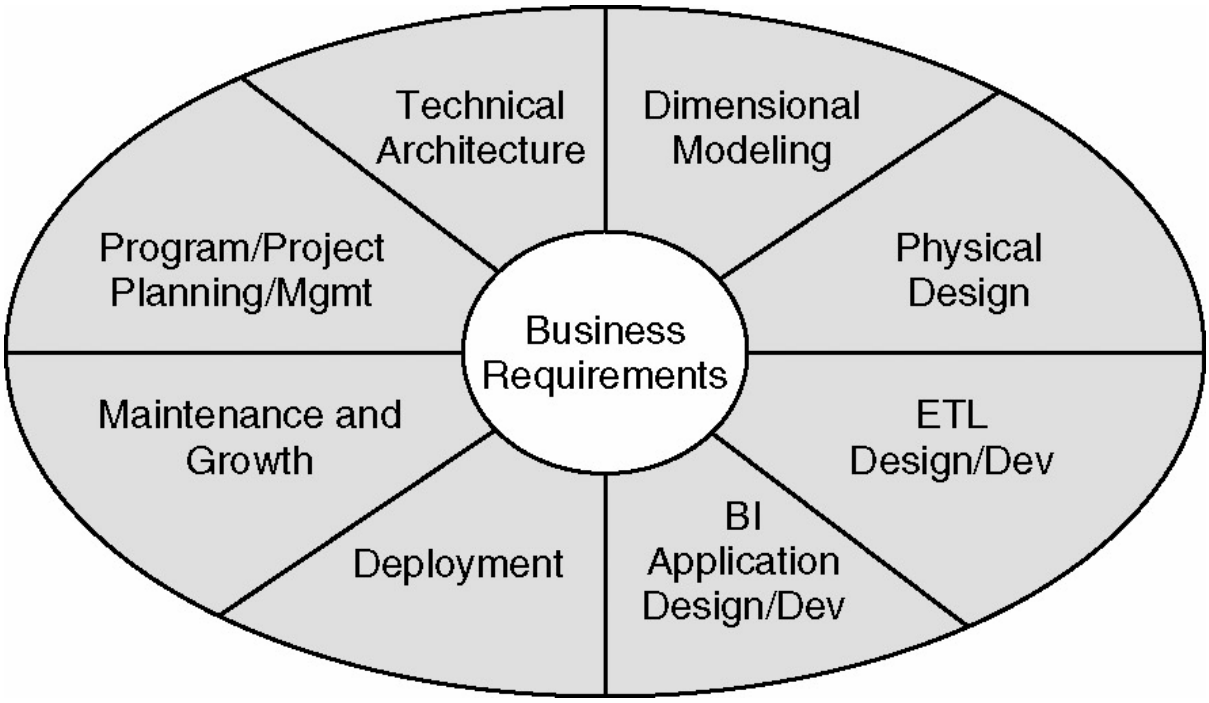


Figure 3.12 – Aspects of the project the business requirements affect (Kimball *et al*., 2008:57)

Kimball's approach performs requirements gathering at two distinct levels. The first level is the macro-level, where the business's needs and priorities that are relative from a program's perspective are understood. This may be an overview of the entire enterprise or just across

multiple projects. The second level is the micro-level where the project team investigates the user's needs and desires in the context of a single, narrowly defined project.

Whether micro- or macro-requirements are gathered, the general process and best practices recommended by Kimball's approach remain the same. According to Kimball *et al*. (2008:57-58), "All requirements gathering initiatives begin with a preparation phase, followed by interactive face-to-face elicitation meetings with the business, and are then brought to an orderly conclusion with documentation and determination of next steps". The only difference between the macro- and micro-level requirements gathering is the participant selection, breadth and depth of the information collection, and resulting documentation.

In order to better understand the business's requirements, one must begin by talking to the business users (Kimball *et al*., 2008:58). This sounds like stating the obvious but Kimball *et al*. (2008:58) state that they have seen many organizations try alternative approaches, such as analysing the operational source data layouts, and reviewing existing reports. One must not just simply ask users what data they want in the DW but one also needs to talk to the users about their jobs, their objectives, and their challenges in order to understand what they do and how they make decisions for the DW system to effectively address their needs (Kimball *et al*., 2008:58).

When gathering requirements from the business users, one must mix some data reality into the process by first interviewing key IT personnel in order to ensure that the data can support what the business users are requesting. The project will inevitably fail if the requirements are gathered without first identifying the available data. DW systems perform end-user reports from historical data but if the data does not exist to support the user-desired report the requirement will not be a possibility.

Face-to-face sessions with the business stakeholders are an important ingredient to accurately understand their requirements. Additionally they build positive working relationships and tighten the bond between business representatives and the DW system (Kimball *et al*., 2008:59). There are two basic interactive techniques for gathering requirements: interviews and facilitated sessions. Kimball's approach suggests that interviews be conducted with individuals or small manageable groups. Interviews allow for a high level of participation from the interviewees, which leads to highly detailed data. Interviews ensure that every voice is heard.

The alternative to interviews is to schedule a larger group session that is led by a facilitator who prevents extroverts from taking over. Facilitated sessions encourage brainstorming and reduce the total time required to gather information compared to interviews. Kimball *et al*.

(2008:60) state that “facilitation is more appropriate after you have gathered enough baseline information to generally understand the business and its vocabulary”.

Kimball’s approach suggests that one use interviews and facilitated meetings but it does not stop there. The approach suggests that one start by preparing for the interview (Kimball *et al*., 2008:61-63). During this step one must identify the interview team that includes a lead interviewer, someone who leads the interview; a scribe who takes notes as the interview progresses; and observers, people who are interested in participating in the interview as bystanders. Next, organizational research must be performed, followed by selecting the interviewees. When selecting interviewees one must work in correlation with business and IT management sponsors to determine who should be interviewed. Once the interviewees have been selected, questionnaires should be developed that “should be structured to align your intended interview flow” (Kimball *et al*., 2008:65). The next step is to schedule the interviews followed by sequencing the interviews, starting with the business drivers and sponsors. It is important to specify a time and place for the interviews that suit the interviewees. The last step for the prerequisite interview activities is to prepare interviewees by emailing them a briefing letter about the process and importance of their participation and contribution (Kimball *et al*., 2008:69).

An important point to remember when conducting interviews is the roles that every person presents in the interview. The lead interviewer asks the questions, the interviewee answers, the scribe takes notes and the remaining participants should listen.

When conducting interviews, it is very important to verify that communication took place and that one understands what the interviewee is saying. Terminology should be defined to ensure that everyone clearly understands what is meant and vocabulary should not be viewed as unimportant. The interview team should establish a peer base where every user is seen as an equal. Keep in mind that the expectations must continuously be managed without overselling the DW system. (Kimball *et al*., 2008:71-73)

3.7.3 Data modelling

The Kimball group has concluded from their collective experience that dimensional modelling is the most viable method to model data for BI. This is because dimensional modelling promotes business understandability and fast query performance (Kimball *et al*., 2008:203). Dimensional modelling, according to Kimball *et al*. (2008:204), “is a logical design technique for structuring data so that it's intuitive to business users and delivers fast query performance”.

Dimensional modelling breaks the data into measurements and context. Measurements are usually numeric values and Kimball's approach refers to them as facts. Facts are encircled by

largely textual context, which is divided into independent logical clusters called dimensions. Dimensions explain the “who, what, when, where, why and how” context of the fact or measurement. Every business process of the organization can be describe by a dimensional model that contains a single fact table made up of numerical measurements encompassed by a circle of dimension tables containing the textual context. This star-like structure is usually called a star join, a term that dates back to the earliest days of relational data. Star schemas are dimensional models stored in relational database environments and cubes refer to dimensional models stored in multidimensional on-line analytical processing (OLAP) structures. The Kimball group recommends populating OLAP cubes from dimensional star schemas. (Kimball *et al*., 2008:204-205)

Third normal form (3NF), which is the standardized form for operational databases, is different from dimensional modelling. 3NF divides the data into a large number of distinct entities, each of which becomes a table in the relational database that aims to reduce data redundancies. 3NF allows for faster and simpler transactional loading and updating and is therefore highly beneficial for transactional processing. Since 3NF stores the micro-relationships amongst data elements, even a simple business process produces a large number of tables that is linked by a confusing web of joins. 3NF and dimensional models can both be represented in an entity-relationship diagram (ERD) since both consists of joined relational tables; the main difference between the two is the degree of normalization. The process of converting the 3NF structures into dimensional models is a relatively simple process that consists of the following two steps (Kimball *et al*., 2008:206):

- Step one – Identify the many-to-many relationships in the normalized model containing numeric and additive non-key metrics as fact tables. Fact tables are usually in 3NF form in a dimensional model because the related context is stored in the dimensional tables.
- Step two – De-normalize the remaining tables into flat dimensional tables that contain single-part keys that relate directly to values in the fact table. “Dimension tables are usually in second normal form with many low cardinality descriptors” (Kimball *et al*., 2008:206).

There are a number of benefits to using dimensional modelling for the development of a DW system, which are (Kimball *et al*., 2008:206-207):

- Understandability – One of the primary reasons why dimensional modelling is accepted as the best practice for the desired structure for presenting the data to the business intelligence layer is because of its understandability. Dimensional models are

easier for business users to understand than the classic normalized source system because the data is categorized into coherent business dimensions that make more sense to the typical business user.

- Query performance – The predictable structure of dimensional models allows the database to make relevant assumptions about the data that aid the performance of the database.
- Query flexibility – Each dimension resembles an entry point into the fact table, which allows dimensional models to confront unexpected changes in query patterns. This allows the dimensional model to gracefully handle whatever is thrown at it.
- Widely accepted – Dimensional modelling has become the accepted standard for the best-of-breed ETL and BI vendors. Most BI tools require an underlying dimensional model in order to perform at their highest potential.
- Flexibility – Dimensional models accommodate unexpected change to the model without performing an excessive reload or reprogramming the BI applications.

The logical design of a dimensional model according to Kimball *et al*. (2008:214), is driven by four key decisions. The four steps mentioned by the Kimball group will be briefly discussed and explained using an example.

Step 1 – Choose the business process

The first step in the design of the dimensional model is to identify the business process or measurement event to be modelled.

For our example let us consider the development of a DW system for a DVD rentals company. In the example the business process that will be used for our dimensional model example is the rentals business process.

Step 2 – Declare the grain

Once the business process has been selected, the design team moves onto declaring the grain of the fact table. It is very important to define precisely what a fact table row is in the selected business process. Usually the fact table grain is as atomic as possible, which allows for the database to respond to both unexpected new queries and unexpected new data elements. Let us define the grain to be “DVD(s) rented and returned by customers”. The grain specifically states that each row in the fact table relates to a customer renting and returning DVDs. Although this might have been obvious the grain could have been “DVD rentals issued by staff". In the defined grain the customer’s detail will be disregarded as it holds no value in the defined grain as well as whether the DVDs where returned. Therefore, accurately defining

the grain of the fact table will have a major impact on what measures will be included in the fact table and what the dimensions will be.

Step 3 – Identify the dimensions

The next step is identifying the dimensions and if the grain of the fact table has been firmly defined the selection of the dimensions is fairly straightforward. At this point the design team can start to think about foreign keys. The grain of the fact table will usually predict a minimal set of dimensions. The identification of the dimension table can be assisted by answering the following question about the grain of the fact table:

- Who - Who is involved in the event?
- Why - Why will the event occur?
- Where - Where will the event occur?
- When - When will the event occur?
- What - What event occurs?

In our example the following dimensions where identified for “DVD(s) rented and returned by customers”.

- Who – Customer, Staff.
- Where – Store.
- When – Date, Time
- What – DVD/film

Please note that the “Why” question is irrelevant in this scenario as why a customer rented or returned a DVD is not captured since it is irrelevant to the DVD rental store.

Step 4 – Identify the facts

The final step in the design of the dimensional model is to carefully select the facts or metrics that are relevant to the selected business process. Each fact must be relevant to the grain of the fact table and the design team must make sure that it does not mix facts from other time periods or other levels of detail that are not relevant to the selected grain.

In our example the following facts have been identified:

- Number of rentals
- Number of returns

The dimensional model will then look something like the image in Figure 3.13.

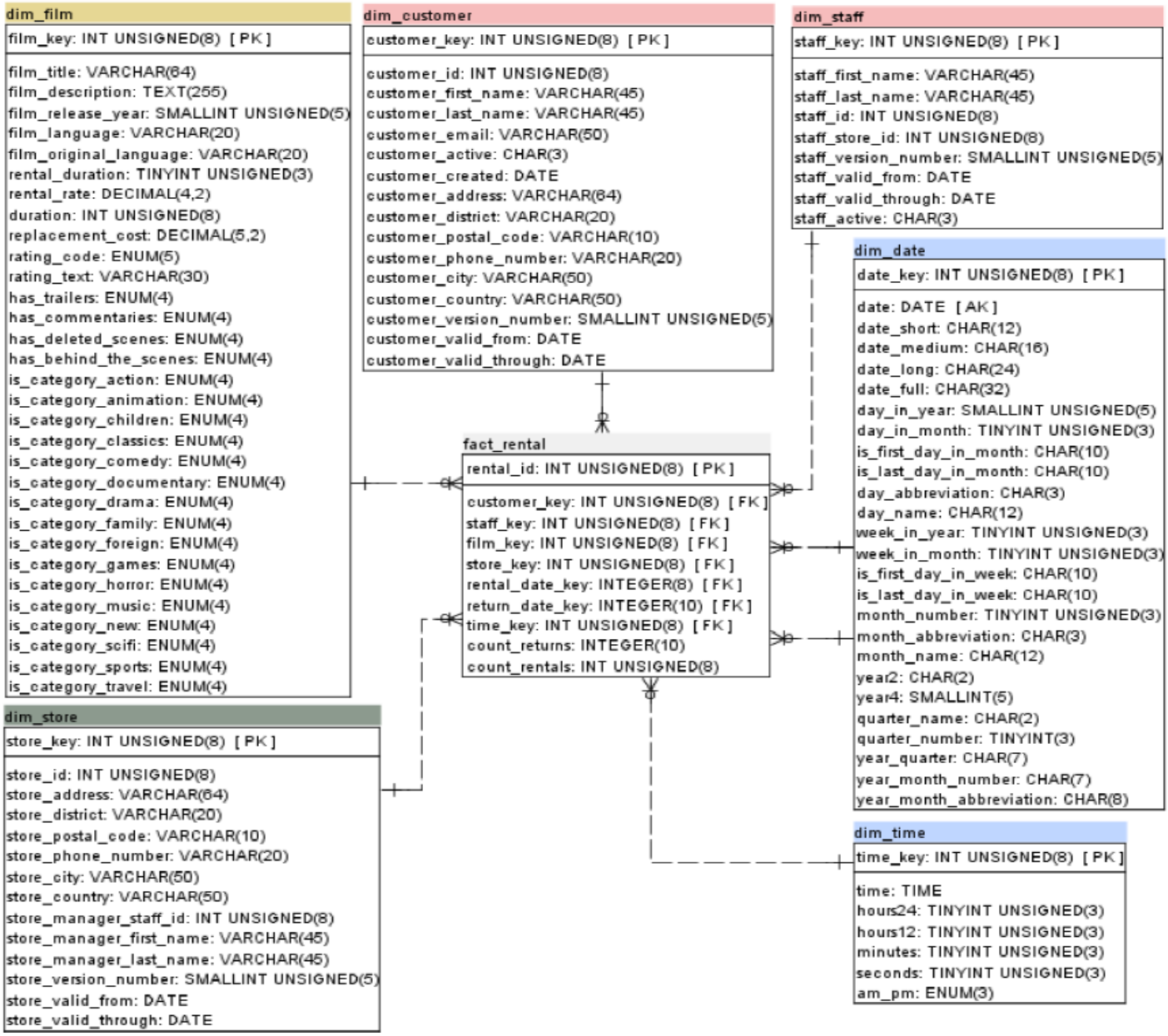


Figure 3.13 – Dimensional model for DVD(s) rented and returned by customer example

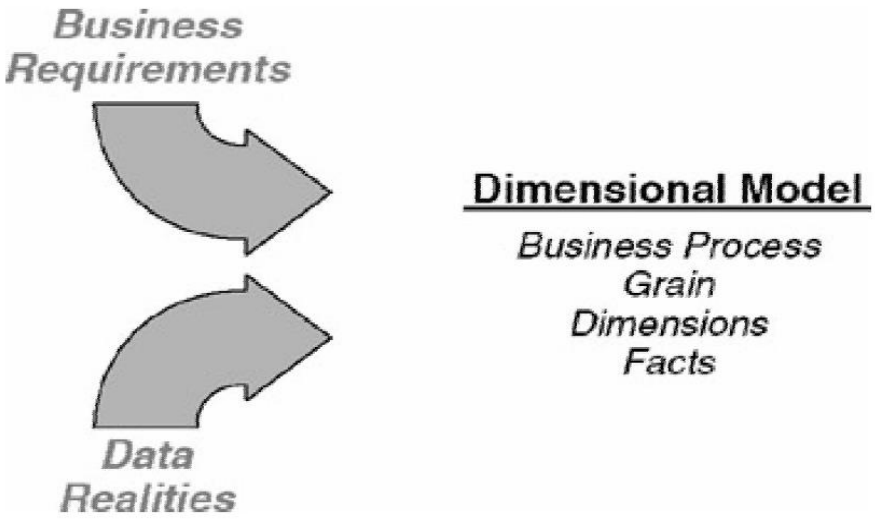


Figure 3.14 – Main driving force for the dimensional model (Kimball *et al*., 2008:215)

It is important to remember that the main driving force of the design of the dimensional model comes from a sound understanding of the business requirements and the realities of the available data source (Figure 3.14).

3.7.4 Extract, transform and load (ETL)

According to Kimball *et al*. (2008:321), "the back room's extract, transformation, and load (ETL) system is often estimated to consume 70 percent of the time and effort of building a DW/BI environment". Kimball *et al*. (2008:321) state that "the ETL system is the foundation of the DW/BI project; its success makes or breaks the data warehouse".

In short the ETL process performs the following tasks (Kimball *et al*., 2008:321):

- Extract (E) – Extract the data out of the original source system location.
- Transform (T) – Do something to the extracted data.
- Load (L) – Load the transformed data into a final set of tables for the user to query.

The ETL process described by Kimball's approach uses a ten-step process to perform the ETL process.

Step 1 – Draw the high-level plan

The ETL process starts with the high-level plan that is not affected by any specific technology or approach. Start the design process with the sources and targets pieces to create a simple schematic. Keep the design very high level, highlighting in one or two pages the data sources and indicating the major challenges in the data that have been identified at this point. As the team develops the detailed ETL system specifications, the high-level view will grow with additional details and the high-level plan must be updated frequently.

Step 2 – Choose an ETL tool

There is a wide variety of available ETL tools in the DW marketplace. ETL tools are specifically developed to perform the necessary extracts, transformation and loads on the original data to achieve the required data presented to the BI layer of a DW system. During this step, the project team must decide upon which ETL tool to use or whether hard coding will be more appropriate for the specific development environment.

Step 3 – Develop default strategies

Once the team has an overall idea of what needs to be done and understands the infrastructure of the DW database, the team must create a set of default strategies for the usual activities associated with the ETL process. These activities are (Kimball *et al*., 2008:373-374):

- **Extraction from each major source system** – Determine the default strategies for extracting data from each of the necessary source systems.
- **Archive extracted data** – Once the data has been extracted from the source system, it should be archived for at least a month. Some organizations prefer to archive the extracted data permanently. The reason for the archived data is to allow for system auditability and recoverability.
- **Police data quality for dimensions and facts** – Create a standardized method that will be used to examine data quality and what must be done if the data does not pass quality control.
- **Manage changes to dimension attributes** – Create the default strategy that will be used to deal with changes in dimension attributes. Most ETL tools deal with common changes in dimension attributes but some dimensions may require a unique strategy to address this issue.
- **Ensure the data warehouse and ETL system meet the requirements for system availability** – Document when each data source is available and block out high-level job sequencing. If system availability is an issue for the specific DW system environment, then the project team must evaluate architectural alternatives, such as background processing on an alternative server.
- **Design the data auditing subsystem** – The project team must tag each row in the DW database with auditing information that explains how the data entered the DW database.
- **Organize the ETL staging area** – Most ETL processes will write data to a disk for a later ETL process to use. As the team designs the ETL process, the team will obtain a

clearer understanding of how large the staging area must be and whether the ETL process will depend mainly on the file system or database to contain the staging area. These decisions are dependent on the ETL tool used.

These default strategies will become part of the table-specific modules for the ETL process. During the development of the ETL system the team will discover that the modules for each table are structured similarly, which allows for a more maintainable system.

Step 4 – Drill down by target table

Once the project team has developed the general game plan for the usual ETL tasks, the team must start drilling into the specific transformations needed to populate each target table in the DW database. Start by completing the source-to-target mappings and the data profiling. Data profiling has been done earlier in the lifecycle at a high level. However, the ETL team will be needed to perform additional data profiling and investigation to completely understand the required data transformations for each table and column. The data profiling done by the ETL team will consist of the following tasks (Kimball *et al*., 2008:374-379):

- Ensure clean hierarchies
- Develop detailed table schematics
- Develop the ETL specification document
- Develop a sandbox source system

Step 5 – Populate dimension tables with historic data

During this step the ETL dimension processing is performed starting with the simplest scenario. Dimension attributes are classified according to two types, namely type 1 (update in place) or type 2 (track history by adding new rows to the dimension table). Dimension processing consists of the following tasks (Kimball *et al*., 2008:380-388):

- **Populate type 1 dimension tables** – The easiest type of table to populate is a type 1 dimension table. A type 1 dimension requires that the team extract the present value for each attribute from the source system.
- **Dimension transformation** – Even the simplest dimension table can require a considerable amount of data cleanup and will definitely require surrogate key assignment.
    - **Simple data transformation** – At the simplest level, data that is extracted from the source system must be converted from the EBCDIC character set to ASCII. In addition, null values must be replaced with either a default value or an empty string.

    - **Combine from separate sources** – Dimensions are usually derived from multiple sources and all the data from multiple sources must be transformed to form a unified format in order to be combined into a single dimension table.
    - **Decode production codes** – Perform a lookup for production codes using text equivalents.
    - **Validate many-to-one and one-to-one relationships** – Some tables will most probably contain one or more rollup paths, such as subcategory and category. It is important to ensure that these hierarchical rollups are perfectly clean.
    - **Dimension surrogate key assignment** – Once the team is certain that its version of a dimension table contains a row for each unique dimension value, surrogate keys must be assigned to the table.
- **Dimension table loading –** Once the transformation for a dimension has been completed, the load is fairly straightforward. All that needs to be done is to load the transformed data into the target table.
- **Load type 2 dimension table history -** Most of the work associated with type 2 dimensions takes place in the incremental processing ETL system. This usually updates the dimension tables with new information as time progressed from the initial load of the DW database.
- **Populate date and other static dimensions –** Each DW system will contain a date dimension, typically at the granularity of one row for each day. Most projects create the date dimension by hand, usually in a spreadsheet. Additional dimensions can be created in a similar manner, for example a budget scenario dimension that holds the values “Actual” and “Budget”, which will most probably be done in a spreadsheet as well.

Step 6 – Perform the fact table historic load

In most cases, the fact data is relatively clean. The most usual transformations of facts include the transformation of null values, pivoting or un-pivoting the data, and pre-computing the derived calculations. Once the necessary transformations on metrics have been performed, the fact table data must enter the surrogate key pipeline to exchange the actual keys for the dimension surrogate keys that are managed by the DW. In short, this step will populate the fact table with the extracted and transformed metrics and surrogate keys from each dimension table that relates to the specific row in the fact table.

Step 7 – Dimensional table incremental processing

This step performs the update function on the dimension tables to ensure that dimensions are up to date in order to provide business users with accurate analysis. This step is similar to historic processing and will most likely require the same transformations performed during the historic processing of dimensions.

Process changes to dimension attributes will require that the following two tasks be performed (Kimball *et al*., 2008:397-399):

- Identify new and changed rows.
- Process changes to dimension attributes.

Step 8 – Fact table incremental processing

Most DW databases are too large to replace the central fact table in a single load window and therefore new and changed fact rows are handled incrementally. The fact table incremental processing differs from the historic load because it must be fully automated. During this step the fact table is updated just like the dimension tables to ensure for an up to date DW system that provides accurate reports.

Step 9 – Aggregate table and OLAP loads

Aggregate tables are logically easy to create and are the results of a large aggregate query that is stored as a table. Most DWs do not perform aggregate table management directly, but use technology, such as a relational database index or materialized views (OLAP data store).

Step 10 – ETL system operation and automation

In a perfect world the ETL system will run the regular load process without any human intervention. This, however, is difficult to achieve but it is possible to come close. This step requires the following tasks (Kimball *et al*., 2008:406-408):

- Schedule jobs.
- Handle predictable exceptions and errors automatically.
- Handle unpredictable errors gracefully.
- Maintain database objects.
- Develop and test ETL automation.

Developing the ETL system is one of the toughest challenges in the development of a DW system. According to Kimball *et al*. (2008:410), “the overall estimate of effort required to build the ETL system is the single largest source of uncertainty and risk in planning a DW/BI system”. It cannot be determined how complex the ETL process will be until the programmer

tackles the actual data sources in microscopic detail. Therefore, the Kimball group recommends allocating six work months of development labour for each major data source.

3.7.5 Data access and deployment

At this stage in the DW system development the backroom of the DW has been completed and a way must be found for business users to access the data in the DW database and to allow them to perform analyses on the data that are of value. Access applications are developed to provide business users with the ability to speedily and easily locate the necessary data in order to perform analyses. OLAP cubes must be created during this phase in order for BI applications to perform analyses on the DW database. During this phase the project team must choose from a wide variety of available BI applications and tailor it to meet the organization's need or develop a unique BI application from scratch.

Once the DW system has been completely developed (including BI application) deployment takes place. It is important to include the business users from the beginning of the development of the DW system to allow them to take ownership, which will ensure that the business users understand the power and effectiveness of the DW system in order to fully grasp its value.

During the deployment of the DW system, it is very important to train and educate the business users so that they can harvest the maximum potential of the system. The Kimball group emphasizes the importance of effective training and also requires that a detailed user manual be available for business users to follow. The acceptance of the DW system is dependent on the business users' effective use of the system. Regardless of how great the DW system may be, if the business users are not trained to effectively use the system it will eventually lead to discarding of the system.

Another task that must be performed during this phase using Kimball's approach is to create and implement a business end-user strategy to support business users after the deployment of the DW system. In order to deploy the complete DW system, a framework must be built that encapsulates all the pieces involved in the deployment process. During deployment the DW system must undergo an alpha test period (internally test all the components of the DW system), followed by a beta test period (allow access to a small number of business users) before the DW system is made available to all users.

Maintenance and management of the DW system after deployment as well as planning for growth of the DW system, is an on-going process. Once the single business process has been successfully deployed the lifecycle returns to the start to expand the DW system from what has already been developed focusing on new requirements and recommendations.

3.8 Conclusion

In this chapter the origin of data warehouses (DW) was briefly discussed followed by what is meant by the term business intelligence (BI) and what it comprises. The relationship between DW and BI was then discussed and it revealed that the two components together make up the DW system. The relationship revealed that the DW database serves as the foundation upon which analysis is performed by the BI application. The chapter then discussed the justification of a DW system and revealed that there are tangible and intangible benefits to a DW system and that a DW system will lead to a positive return on investment.

The approaches of the two giants of DW development (Inmon and Kimball) were introduced and discussed. Although both approaches set out to develop a successful DW system, the approaches are quite different:

Collecting requirements

Inmon uses a data-driven methodology where data is viewed as the starting point for development. Inmon then uses the data-driven approach to develop a working DW system and attempts to collect requirements at the end of the development cycle using a feedback loop.

Kimball's approach is a requirement-driven methodology where the project begins with collecting requirements, which are viewed as the core driving force of the DW project. Kimball collects requirements by performing interviews and facilitated meetings.

Data modelling

Inmon performs data modelling by using a top-down approach where the corporate data model is first developed and used to drill down into the DW data model and the operational data model. Inmon also models the data according to database standards by normalizing the tables.

Kimball performs data modelling by using a bottom-up approach by starting with a single business dimensional model that eventually evolves into the larger corporate data model. In addition, Kimball uses the dimensional modelling approach that de-normalizes the tables from the operational system.

ETL

Inmon and Kimball have a similar viewpoint in regard to ETL; both agree that it is a large and complex process that requires the full attention of the project team. In both Inmon's and Kimball's approaches it is viewed as the process of extracting data from the source systems, performing some kind of transformation and cleansing on the source data, and integrating and loading it into a central DW system. In addition, both recommend the use of ETL tools to assists the project team in successfully completing the ETL process.

Data access and deployment

Both Inmon and Kimball agree that the data in the DW system needs to be presented to the business users in a manner that provides them with valuable information. Both perform this by using an OLAP environment that allows business users to analyse the data and generate valuable business intelligence. The only notable difference is that while Inmon attempts to define and understand the user requirements during this phase of development; Kimball is concerned with ensuring that the user requirements have been met and live up to the user's expectations.

This concludes the formal discussion of the literature study (Chapters 2 - ASDM and 3 - DW) and signals the end of the awareness stage of the research methodology. In section 1.3 of this study, the identified problem was that DW development is fraught with uncertainty and changing requirements. In order to address this problem, this study aims to develop the ADW methodology using Inmon's or Kimball's approach as the foundation along with best practices of XP, SCRUM, and FDD. In regard to the aim of this study the awareness stage revealed the following knowledge that will form the driving force of the suggestion stage of the research methodology:

- ASDMs were developed to deal with uncertainty and changing requirements. Therefore, the issues faced by DW development can be addressed by implementing best practices associated with ASDMs into DW development. XP, SCRUM, and FDD are ASDMs that hold potential for DW development. This revealed that XP, SCRUM, and FDD must be further analysed for the agile development of a DW.
- Inmon and Kimball's DW development approaches are the most popular in the DW industry. The two approaches differ in a number of ways (discussed above) and will require further analysis to determine which one is best suited for agile development.

In the next chapter the research methodology used in this study will be explained as well as how it was applied.

Chapter 4: Research methodology

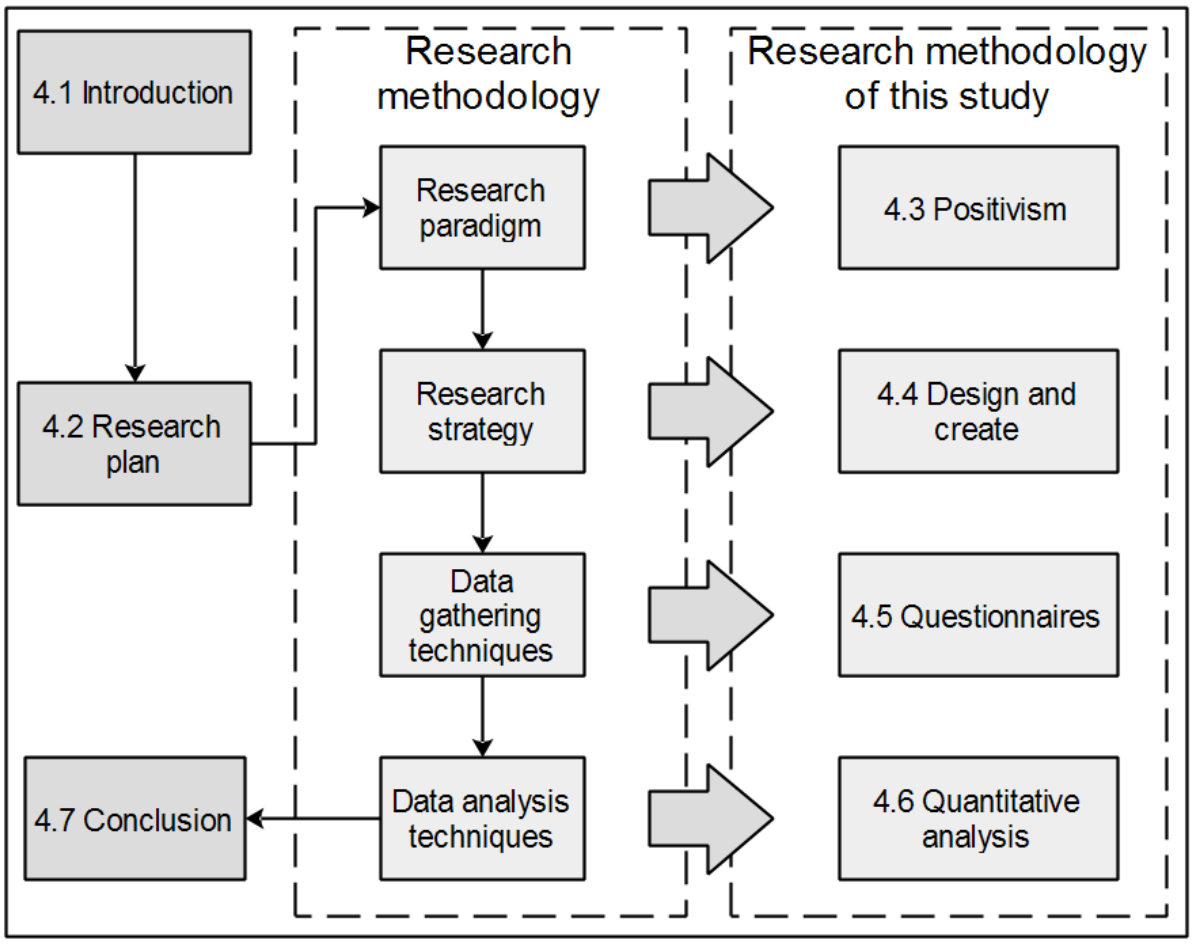


Figure 4.1 – Overview of Chapter 4

4.1 Introduction

In this chapter the research methodology used in the study will be explained starting with the research plan. The research plan will discuss the components of the research methodology and how they were applied in the study to develop the ADW methodology. The research plan is followed by a brief discussion of the components relevant to the study. The first component discussed will be the positivism paradigm followed by the design and create research strategy. The design and create section is followed by the questionnaire data generation method and how the questionnaires were developed. The questionnaire section is then followed by the quantitative analysis section and how it was performed on the data generated by the questionnaires. The chapter then ends with a summary of the research methodology of this study.

4.2 Research plan

The research methodology of this study uses the positivism paradigm as the underlying philosophy of the research methodology that aims to perform empirical research that can be generalized. Based on the underlying philosophical viewpoint, the research methodology implements the design and create research strategy as the foundation of the research. The research uses the design and create process steps moving from awareness all the way towards the conclusion step in order to introduce an empirical methodology artefact that aims to support a project team for the agile development of a DW system. With the design and create research strategy as the foundation, the research methodology uses questionnaires for data generation and quantitative analysis to analyse the generated data.

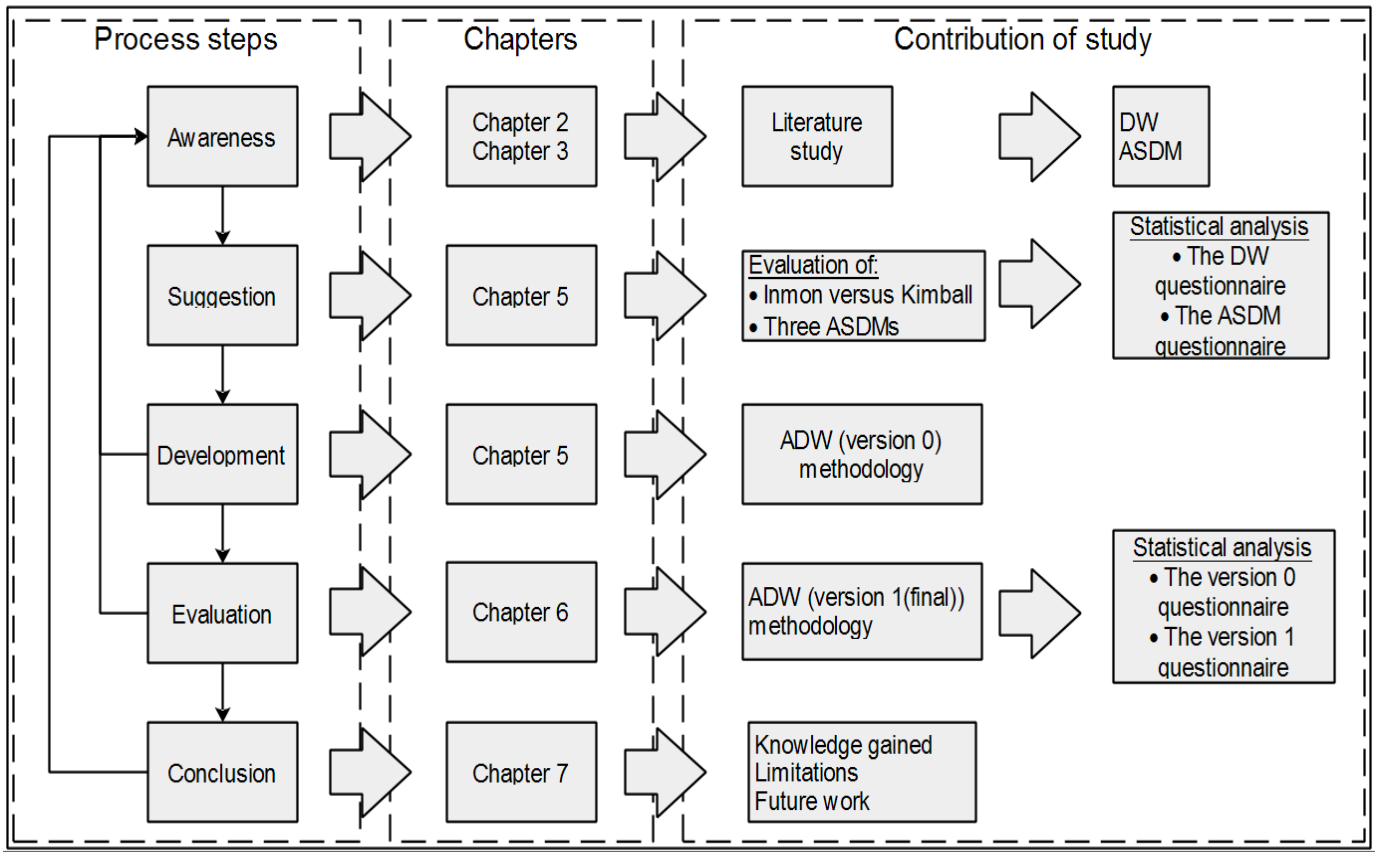


Figure 4.2 – The research plan

Awareness – The awareness stage revealed that DW development is characterised by uncertainty and changing requirements and that ASDMs are well-suited in dealing with these issues. It also revealed that an ASDM cannot address the unique technical aspects associated with DW development. Therefore, Chapter 2 performed a literature study on ASDMs, which discussed ASDMs in general and then introduced and discussed three ASDMs (XP, SCRUM, and FDD) that hold potential for the development of a DW. The literature study on ASDMs revealed that XP, SCRUM, and FDD are potential candidates and will require further analysis. In addition to ASDMs, Chapter 3 performs a literature study on DW and BI and how these two

terms come together to form a complete DW system. Chapter 3 then introduces and explains the two most popular DW development approaches (Inmon and Kimball). The literature study on DW development revealed that both Inmon's and Kimball's approaches are viable candidates for the foundation of the ADW methodology and will require further analysis. Therefore, the awareness stage identified viable candidates for further analysis to suggest the ingredients for the development of the ADW (version 0) methodology which in turn aims to address the problems faced by DW development.

Suggestion – The suggestion step in the design and create research strategy was implemented in Chapter 5. The chapter starts by performing an evaluation of Inmon versus Kimball in regard to the suitability for agile development. The evaluation process was conducted by developing the DW questionnaire based on the knowledge gained from the awareness step. The DW questionnaire was used to generate data to determine which DW development approach between Inmon and Kimball is best suited for the agile development of a DW system. Once the best-suited DW approach has been determined, an evaluation of the three ASDMs in regard to the selected DW approach was performed to determine which ASDM (XP, SCRUM, or FDD) is best suited for each phase of the selected DW approach. In order to determine this, the ASDM questionnaire was developed and distributed. Therefore, the suggestion step proposes which DW approach is best suited for the agile development of a DW system and which of the three ASDMs is best suited for each phase of the selected DW approach.

Development – The development stage was carried out in Chapter 5 and aims to develop a methodology artefact. Based on the knowledge gained during the suggestion stage of the research methodology, the initial version (version 0) of the ADW methodology was developed and explained.

Evaluation – The evaluation stage was handled in Chapter 6. The evaluation process started with the evaluation of version 0 of the ADW methodology. This evaluation was performed by developing a DW system in collaboration with the industry partner. The evaluation aimed to develop a DW system with real-world users using real-world data. The development of the DW system used the developed ADW (version 0) methodology in order to evaluate the effectiveness of the ADW (version 0) methodology in a real-world implementation. The development of the DW system followed the ADW (version 0) methodology to assist the project team in the agile development of a DW system. Apart from the knowledge gained during the actual development of the DW system (the researcher was part of the development team), the version 0 questionnaire was developed and distributed to the business users to evaluate the degree of agility (based on the Agile Manifesto) of the development process and

the success of the DW system. Using the knowledge gained from the implementation and evaluation of version 0 of the ADW methodology, version 1 of the ADW was developed and explained. The evaluation of version 1 (final) was performed by developing and distributing the version 1 questionnaire to determine the relevance of the final version of the ADW methodology from the viewpoint of experienced DW developers. The results of the evaluation stage produced a DW system that is still being used by the industry partners as well as the final version of the ADW methodology artefact.

Conclusion – The conclusion stage is discussed in Chapter 7 and includes the findings, limitations and future work of the research. The findings include the knowledge gained during the research and summarize the relevance of the ADW methodology for the agile development of a DW system. The limitations section mentions and explains the limitations encountered during the research and the development environment and the chapter concludes with the recommendation for future work to improve or further verify the usefulness of the ADW (version 1) methodology.

In the rest of the chapter each of the components that make up the research methodology of the study will briefly be explained followed by a discussion of how they were applied in the research.

4.3 Positivism

According to Oates (2006:282), “A paradigm is a set of shared assumptions or ways of thinking about some aspect of the world”, which is concerned with a communities-shared viewpoint on how research is conducted in order to contribute or create knowledge. Different philosophical paradigms have different viewpoints on how the world is perceived (ontology) and how knowledge can be acquired is this world (epistemology) (Smith, 1996:12; Oates, 2006:282; Grey 2011:143). This study will use the positivistic research paradigm and therefore it will be the only paradigm discussed.

Positivism is one of the oldest philosophical paradigms and focuses on knowledge acquired from facts, experience and evidence (Smith, 1996:14; Grey, 2011:144). The positivism paradigm aims to generalize to an entire population based on the findings of the research being conducted. The paradigm assumes that the research setting is ordered and regular, not random and that this setting can be investigated objectively (Oates, 2006:283). The shared worldview of researchers working in the positivism paradigm has identified the following characteristics (Oates, 2006:287; Trochim, 2006: Grey, 2011:144-145):

- **The world, including the social world, exists independently from humans** – There is a physical and social world to be investigated, studied, captured, and measured. This world exists and remains the same regardless of human interaction.
- **Measurements and modelling** – The researcher explores this world through observations, measurements and models (hypotheses, theories) that explain how the world works.
- **Universal law** – The researcher aims for generalizations, where patterns, universal laws or irrefutable facts are proven and declared true regardless of the research setting and the researcher.
- **Objectivity** – The researcher must remain neutral and objective, an impartial observer. Facts about the research setting can be explored independently of the researcher's personal opinions.
- **Quantitative data analysis** – Positivism often leans towards a strong preference for mathematical modelling, proofs and statistical analysis.
- **Hypothesis testing** – The research must be based on empirical testing of theories and hypotheses, which either leads to verification or refutation.

Although positivism has been around for 400-500 years (Oates, 2006: 283), it focuses solely on studying aspects of the natural world and is therefore not well-suited for researching the world of people (social world), organizations and group structures built by people, cultures developed by people and the meanings they impose on something (Oates, 2006:288). Therefore, positivism has the following criticism (Oates, 2006:288-289; Grey, 2011:145):

- Reductionism, the art of breaking down complex things into smaller and simpler components, cannot be done when studying the social world.
- Repetition is not always possible, especially when studying the social world.
- Generalization is not always desired.
- Everyone perceives and interprets the world differently.
- Regular laws and patterns may turn out to be observable in the social world that is created by people.

In conclusion, positivism is a well-established paradigm that focuses on studying the natural world, not suited for studying the social world, in order to provide empirical research that generalizes the findings to the world as a whole. Positivism serves as the underlying philosophical paradigm of the research methodology used in this study that aims to develop an empirical ADW methodology for the agile development of a DW system. In the next section the design and create research strategy will briefly be discussed as well as how it was applied in the study.

4.4 Design and create

The design and create research strategy aims at developing new IT products that are known as artefacts. These artefacts are (Oates, 2006:108; March & Smith, 1995:251):

- **Constructs** – This involves new concepts or vocabulary in a specific IT-related environment, such as entities, objects, or data flows.
- **Models** – This refers to a combination of constructs that constitutes a specific environment in order to assists in problem understanding and solution development. Examples include a data-flow diagram, use-case scenario, or a storyboard.
- **Methods (methodologies)** – This artefact aims to provide guidance or assistance on the models to be developed and usually includes a development process to follow to solve problems using IT. Methodologies typically include formal, mathematical algorithms, commercialized and published methodologies, organizations' in-house procedure manuals and informal explanations of practice derived from experience.
- **Instantiations** – This refers to a working system that reveals that constructs, models, methods, ideas, genres and/or theories can be applied to a computer-based system.

The design and create research strategy is usually a problem-solving approach that involves five steps in an iterative process. These steps are (Oates, 2006:111; Vaishnavi & Kuechler, 2004):

- **Awareness** –This step is concerned with identification and elaboration of a problem, by studying the literature where previous work reveals areas for further research, or from reading about new findings in another discipline, or from practitioners or users revealing the need for something, or from filed research or from new development in technology.
- **Suggestion** – During this step the research approach jumps from an interest about the problem to introducing an unconfirmed idea on how the problem might be solved.
- **Development** – In this step the unconfirmed idea is implemented in order to test whether it is indeed a viable candidate for addressing the problem.
- **Evaluation** – This step is focused on examining the developed artefact and determines whether the proposed idea is of value as well as deviations from expectations.
- **Conclusion** – In this step the results from the design process are strengthened and documented along with the identification of the knowledge gained. This step may also include the identification of any unexpected or anomalous results that cannot be explained, which may lead to the subject of further research.

These steps are usually not followed in a fixed, step-wise manner, but rather, in a more flexible, iterative cycle (Oates, 2006:112).

Like all research strategies, design and create has a number of advantages and disadvantages. Below are a number of advantages and disadvantages associated with the design and create research strategy (Oates, 2006:121-122):

Advantages

- Instead of just producing abstract theories or other knowledge, the strategy produces a tangible IT artefact.
- The strategy is attractive to researchers who enjoy/prefer technical and creative development work.
- For areas like computer science and software engineering it is considered the expected norm for research.
- Due to the fact that the use of IT and computers is relatively new in many areas as well as the rapid advances in technology, there is a large pool of potential for proposing and developing new IT artefacts thereby leading to a contribution to knowledge.

Disadvantages

- The researcher is challenged to justify why the research is not just normal design and create opposed to design and create research. In other words, does the research contribute knowledge or does it just develop a computer-based system.
- The research may be difficult and risky if the researcher does not have the necessary technical or artistic skills.
- It can be difficult to generalize the results to different settings from the use of an IT artefact in a single situation.
- The successful implementation of an IT artefact may depend on the involvement of the researcher and fail or not be as effective when the researcher is not present.

The research methodology of this study uses the process steps of the design and create research strategy as its foundation. The research performed in this study starts with the awareness stage and progresses all the way to the conclusion stage to develop the ADW methodology artefact. The application of the design and create strategy was discussed in section 4.2. In the next section the questionnaire data generation method will be explained as well as how it was applied in the study.

4.5 Questionnaires

According to Oates (2006:219), “A questionnaire is a pre-defined set of questions (sometimes called *items*), assembled in a pre-determined order”. Once a questionnaire has been created, it is distributed to a targeted number of people asked to answer the questionnaire, thus providing the researcher with valuable data that can be analysed and interpreted. Questionnaires are usually associated with the survey research strategy but can be used within other strategies, such as case studies, action research, or design and create.

When creating questionnaires, the researcher must design and construct the questionnaire so that valid, reliable data will be generated at an affordable cost. The questions asked must be written so that all respondents understand what is asked and generate the intended answers. According to Peterson (2000:12), “Simply stated, the quality of the information obtained from a questionnaire is directly proportional to the quality of the questionnaire, which in turn is directly proportional to the quality of the construction process”. When planning and designing a questionnaire it is important to consider the following aspects of the questionnaire (Oates, 2006:221-228; Grey, 2011:167-168):

- **Question content and wording** – When creating a questionnaire, one must be certain that the questions asked will generate relevant data for the research being conducted. Each question asked must be brief, relevant, unambiguous, specific and objective.
- **Question types** – Question can generate two kinds of data: factual data (date of birth) or opinions (what does the respondent think about the interface of the system). In addition, questions can be broken down into two kinds of questions:
    - Open questions – Questions that allow the respondents to answer in their own words – this involves leaving a blank space for respondents to fill in as they deem fit.
    - Closed questions – provide a multiple-choice answer to the asked question so that respondents can only choose between a set of pre-defined answers.
- **Format of questions and responses** – When creating a questionnaire there are a number of different kinds of question where each question generates a different kind of response. These are, for example yes/no answers, quantity questions, agree/disagree with a statement, degree of agreement or disagreement, etc.
- **Layout and structure** – A well-structured questionnaire must contain an introduction that explains the purpose, the sponsor of the research, the return address and return date. Respondents must be assured that their answers will be kept confidential and that the questionnaire is voluntary. The questionnaire must provide clear instructions on how to complete the questionnaire.

- **Pre-test and pilot** – Once the questionnaire has been completed, it is recommended to ask experts in the field of research to evaluate the questionnaire before the final version is distributed to the targeted audience.
- **Validity and reliability** – A well-structured questionnaire will contain the following components:
    - Content validity – The questions asked must be relevant to the research objectives.
    - Construct validity – This is concerned with whether the questionnaire measures what the researcher intended it to measure, therefore, the researcher must verify that the questionnaire is measuring, validating or evaluating what the researcher intended.
    - Reliability – Reliability ensures that the data is genuine and that given the same respondents, the same data will be generated.

In this study four questionnaires were developed and distributed to the relevant target groups.

The DW questionnaire

The DW questionnaire was researcher-administered and the target group consisted of six developers with experience in Inmon's approach, Kimball's approach, and ASDMs. The aim of the DW questionnaire was to determine which DW development approach (Inmon's or Kimball's approach) is best suited for the agile development of a DW system. The DW questionnaire used the four major stages of DW development (requirements gathering, data modelling, ETL, and data access and deployment) and the twelve principle of the Agile Manifesto (section 1.2.2) as the evaluation criteria.

The ASDM questionnaire

The ASDM questionnaire was also researcher-administered and was distributed to the same target group as the DW questionnaire. The ASDM questionnaire aims to determine which of the three ASDMs (XP, SCRUM and FDD) is best suited for each phase of the best-suited DW development approach (Inmon's or Kimball's approach). The ASDM questionnaire will use the development phases of the selected DW development approach as the evaluation criteria.

The version 0 questionnaire

The version 0 questionnaire was self-administered and was distributed to five relevant stakeholders and business users involved and affected by the development of the DW system

that implemented the ADW (version 0) methodology. The version 0 questionnaire aimed at determining whether the ADW (version 0) methodology assisted and provided support in the agile development of a DW system and whether the developed DW system is viewed as a success. The version 0 questionnaire consisted of two sections; the first determined whether the ADW methodology laid the foundation for an agile development environment based on the principles of the Agile Manifesto (section 1.2.2), and the second section determined whether business users view the project as a success based on a number of universal success factors (will be discussed in section 6.2.3.2)

The version 1 questionnaire

The version 1 questionnaire was self-administered and was distributed to 24 developers with experience in DW development using Kimball's approach as well as knowledge and/or experience with XP, SCRUM, and FDD. The version 1 questionnaire aimed at determining the usefulness of the best practices introduced in each major phase of development as well as the ADW (final version) methodology as a whole.

4.6 Quantitative analysis

Quantitative data refers to data or evidence based on numbers and quantitative data analysis is concerned with looking for patterns in the data in order to draw conclusions (Oates, 2006:245). Quantitative data can be analysed in a number of ways and the simplest analysis uses visual aids where the data is presented in tables and graphs in order to explore and identify values and patterns (Oates, 2006:249)

Although tables and charts are useful in identifying values and patterns, they are dependent on the researcher's interpretation of the data. Statistical analysis, another quantitative data analysis technique, brings a more universal technique to the table and aims to evaluate key points and to make generalized assumptions based on the evidence (Oates, 2006:254). Statistical analysis can provide the researcher with the necessary information to determine whether the relationship portrayed in tables and graphs is based on sound data or whether it is just a coincidence. Some statistical analysis is rather easy to calculate while others are complicated and require a computer to perform the calculations (Oates, 2006:245). Some of the most popular statistical analysis techniques are: mean, median, mode and standard deviation.

This study uses four different questionnaires to gather data and in order to interpret the data, quantitative data analysis was performed on the data. Each of the questionnaires was tested using Krippendorff's Alpha-Reliability (Krippendorff, 2011) in order to determine the reliability of the data generated by the questionnaires. Krippendorff's Alpha (α) is a reliability coefficient

that measures the agreement amongst observers, coders, judges, raters or measuring instruments (in this case respondents' answers to a questionnaire). The mathematical equations used to determine Krippendorff's Alpha is beyond the scope of this study but the results determining the reliability of the data will briefly be discussed.

When the reliability coefficient (α) equals 1 then data represents perfect reliability and the opposite is true for a value of 0. In order to validate and trust the data generated by any method, α needs to be far away from these two extreme conditions and an acceptable range for α is (Krippendorff, 2011:1)

$$1 \geq \alpha \geq 0 \begin{cases} -\text{Systematic disagreement} \\ \pm\text{Sampling errors} \end{cases}$$

Using Krippendorff's alpha, the minimum acceptable alpha must be chosen depending on the conclusions to be derived from imperfect data. Therefore, this study will define the acceptable range of reliable data as (also the range mostly used by social scientists): (Krippendorff, 2011:5)

$$0.800 \geq \alpha \geq 0.667$$

Once the data generated by the questionnaires had been proven to be reliable, basic statistical analysis was performed on the data to determine the total (in some cases also the percentage of the totals) and/or the mean (average). The reason only these basic statistical analyses were performed, was due to the fact that all the questionnaires aimed to either compare different approaches or to determine the effectiveness of the ADW methodology. Therefore, the total (including percentages) and average is more than sufficient to draw the necessary conclusions from these basic values.

The statistical values for each questionnaire will be presented in a table to represent the data and easily show how the conclusions were interpreted.

4.7 Conclusion

This chapter introduces the research methodology used by this study and the components that make up this methodology are the positivism paradigm, design and create research strategy, questionnaire data generation method, and quantitative data analysis technique. The chapter starts off by introducing the research plan that explains how the research methodology was implemented in the study to produce an empirical methodology artefact that aims to assist projects teams in the agile development of a DW system. The research plan demonstrates the process steps followed to develop the desired artefact as well as the knowledge each step

contributes to the field of IT in regard to DW development. Following the research plan, each component of the research methodology is briefly discussed; including the role it played in the research methodology.

In the next chapter the suggestion and development stages of the research will be performed and explained.

Chapter 5: Proposed Framework

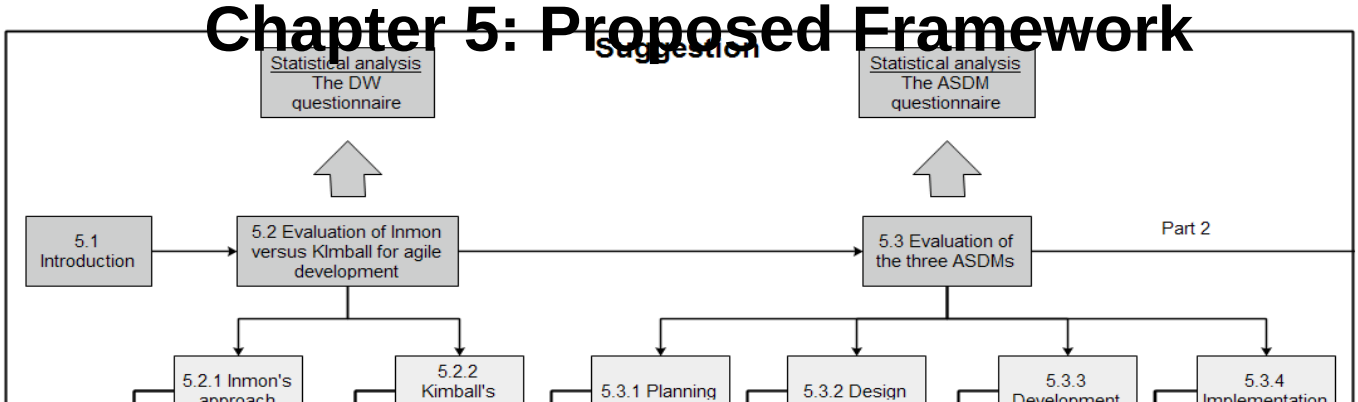


Figure 5.1 – Overview of Chapter 5 part 1

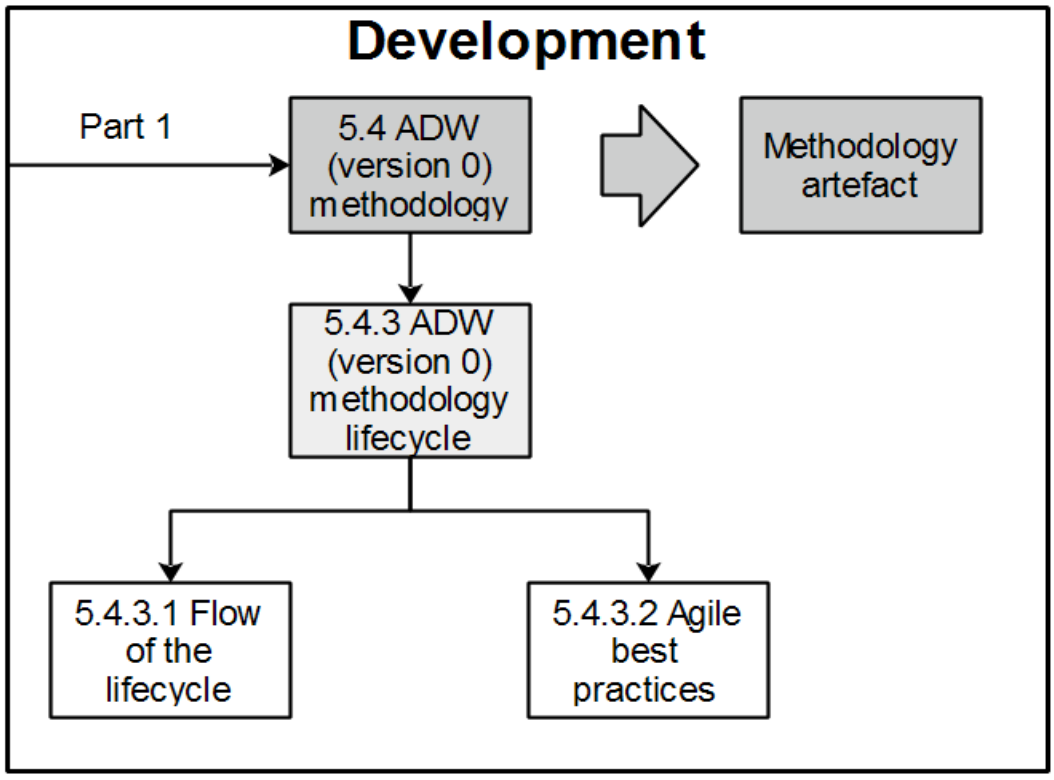


Figure 5.2 – Overview of Chapter 5 part 2

5.1 Introduction

In this chapter the evaluation to determine whether Inmon's or Kimball's approach is best suited for agile DW development is performed. Once the best-suited approach for agile development has been identified, another evaluation will be conducted to determine which ASDM of the three used in this study (XP, SCRUM, and FDD) is best suited for each stage of Inmon's or Kimball's approach (the one that is best suited for agile development).

Once these evaluations have been performed the ADW methodology will be introduced that will either use Inmon's or Kimball's approach as the foundation of the ADW methodology. Agile best practices from XP, SCRUM or FDD will be introduced that build upon the foundation provided by either Inmon's or Kimball's approach in regard to the major phases of DW development. The structure of the ADW methodology will be explained by using a metaphor and the lifecycle, flow and best practices of the ADW methodology will be explained.

After reading this chapter, the reader will know whether Inmon's or Kimball's approach is best suited for agile development and why, which ASDM is best suited for each major phase of Inmon's or Kimball's approach for DW development. The reader will be introduced to a new DW development methodology for the agile development of a DW system.

5.2 Evaluation of Inmon versus Kimball for agile development

In this section Inmon's and Kimball's approaches will be evaluated to determine which one is best suited for agile development in regard to the four primary stages of DW system development: requirements gathering, data modelling, ETL, data access and deployment. Since the purpose of the evaluation is to determine which of the two DW development approaches is best suited for agile development, the evaluation will be based on the twelve principle of the Agile Manifesto (section 1.2.2) in regard to the four stages of DW development. The best suited DW development approach will then be evaluated in regard to the three ASDMs (XP, SCRUM and FDD) discussed in Chapter 2 in order to determine which of the three will be the best fit for each phase of the selected DW approach.

The evaluation of Inmon versus Kimball in regard to the suitability of agile development was performed by developing and distributing the DW questionnaire (see Appendix A) to six experienced developers. The target group has experience with Inmon's approach, Kimball's approach, and ASDMs. The target group consisted of post-graduate students (Masters and higher) in the field of IS development (also referred to as information technology) with the majority of the respondents being male.

The DW questionnaire was researcher-administered and used an agree-disagree scoring table using the four major stages of DW development and the twelve principles of the Agile Manifesto (section 1.2.2) as the evaluation criteria. The scoring system and the data generated by the questionnaire are presented in Tables 5.1 and 5.2.

Score	Description
1	Unsuitable
2	Low suitability

3	Adequate suitability
4	High suitability
5	Very high suitability

Table 5.1 – Scoring system

Inmon's approach: Respondents 1-6				
Agile principles	**Requirements gathering**	**Data modelling**	**Extract, transform and load (ETL)**	**Data access and deployment**
Principle 1 - Early delivery	3;2;3;2;3;3	1;1;2;1;2;1	3;3;4;3;5;4	3;2;3;4;3;3
Principle 2 - Welcome change	1;1;2;3;1;1	1;1;1;1;1;1	1;1;2;1;3;3	4;4;4;3;4;3
Principle 3 - Deliver frequently	1;1;1;2;1;2	2;1;2;1;2;1	3;4;5;4;4;5	4;3;4;4;3;5
Principle 4 - Collaboration	2;2;2;3;3;3	2;3;2;2;2;3	1;1;2;1;2;1	3;3;3;4;4;4
Principle 5 - Motivated individuals	2;3;2;3;2;2	3;4;4;4;3;3	4;5;4;4;3;4	3;3;4;3;3;3
Principle 6 - Face to face communication	3;3;4;4;4;3	3;3;3;2;2;2	3;2;3;2;1;2	3;4;4;4;3;3
Principle 7 - Working software = success	3;3;4;3;3;4	2;3;3;2;2;2	5;5;5;5;4;5	5;5;5;5;5;5
Principle 8 - sustainable development	3;3;3;3;4;3	4;3;3;2;4;4	4;4;5;4;4;3	4;4;3;4;4;4

Agile principles	**Requirements gathering**	**Data modelling**	**Extract, transform and load (ETL)**	**Data access and deployment**
Principle 9 - Technical excellence	4;4;3;4;3;4	5;5;4;5;4;5	3;4;3;4;3;3	4;5;3;4;4;5

Principle 10 - Simplicity	1;2;2;1;2;2	1;1;1;1;2;1	4;4;4;4;3;4	4;3;3;3;3;3
Principle 11 - Self-organizing	1;1;3;3;2;2	2;3;3;2;3;2	2;2;3;4;3;3	3;3;4;3;4;3
Principle 12 - Reflect and adjust	4;4;3;4;4;4	3;3;3;2;1;1	4;4;4;3;4;3	4;4;5;5;5;4
Total	**28;29;32;35;32;33**	**29;31;31;25;28;26**	**37;39;44;39;39;40**	**44;43;45;46;45;45**

Table 5.2 – Inmon's scores from the DW questionnaire

Kimball's approach: Respondents 1-6

Agile principles	Requirements gathering	Data modelling	Extract, transform and load (ETL)	Data access and deployment
Principle 1 - Early delivery	4;4;5;4;4;4	3;4;3;3;3;3	3;3;3;3;3;3	5;4;4;5;5;5
Principle 2 - Welcome change	4;4;4;5;5;4	4;4;4;4;4;4	3;3;3;4;3;3	4;4;4;4;4;4
Principle 3 - Deliver frequently	4;5;5;4;4;4	4;3;4;4;3;4	3;4;4;4;3;4	4;5;5;4;4;5
Principle 4 - Collaboration	4;4;4;4;4;4	3;3;4;3;4;3	2;2;2;2;2;2	4;4;4;4;5;4
Principle 5 - Motivated individuals	3;3;3;3;3;3	2;2;2;3;3;2	3;4;3;4;4;3	3;3;4;3;3;3
Principle 6 - Face to face communication	5;5;5;4;5;5	4;3;3;4;3;4	3;3;3;3;3;2	5;5;4;4;4;4

Agile principles	Requirements gathering	Data modelling	Extract, transform and load (ETL)	Data access and deployment
Principle 7 - Working software = success	4;5;5;5;4;5	3;3;3;3;3;4	5;5;5;5;5;5	5;5;5;5;5;5

Principle 8 - sustainable development	4;3;3;4;4;4	4;4;4;5;5;4	4;4;5;4;4;4	4;4;3;4;4;4
Principle 9 - Technical excellence	4;4;3;4;4;4	5;5;5;5;4;5	3;3;3;3;3;3	4;5;4;4;4;5
Principle 10 - Simplicity	4;4;4;3;4;3	4;4;4;4;4;4	4;5;5;4;4;4	5;4;4;5;4;4
Principle 11 - Self-organizing	4;3;3;4;3;3	3;3;3;3;3;3	4;4;4;4;4;3	4;4;4;4;4;4
Principle 12 - Reflect and adjust	4;3;3;3;4;3	4;4;4;4;5;4	4;4;4;3;4;3	4;4;5;5;5;4
Total	**48;47;47;47;48; 46**	**43;42;43;45;44; 44**	**41;44;44;43;42; 39**	**51;51;50;51;51; 51**

Table 5.3 – Kimball's scores from the DW questionnaire

The data in the tables (both Inmon's and Kimball's evaluation scores) contains the scores of all six respondents and is separated by a semi-colon. In order to analyse and interpret the data generated by the questionnaire, a reliability test was performed using Krippendorff's Alpha (discussed in section 4.6) for each of the tables presented. In section 4.6 the acceptable range of reliability was defined as $0.800 \geq \alpha \geq 0.667$. Below is the reliability alpha for both Inmon's and Kimball's scores from the DW questionnaire:

Inmon = 0.732

Kimball = 0.692

Given the above reliability alphas, the data generated by the DW questionnaire can be viewed as reliable and acceptable data. In order to clearly draw a sound conclusion from the generated data the total score for each major phase of DW development was calculated as well as the average score. The statistical summary of the data is presented in Table 5.4.

Inmon versus Kimball

DW development stages	Inmon's approach	Kimball's approach
Requirements gathering	Total score = 189 Average score = 2.625	Total = 283 Average score = 3.931
Data modelling	Total = 170	Total = 261

	Average score = 2.361	Average score = 3.625
ETL	Total = 238	Total = 253
	Average score = 3.306	Average score = 3.514
Data access and deployment	Total = 268	Total = 305
	Average score = 3.72	Average score = 4.24
Total	**Total = 865**	**Total = 1102**
	Average score = 3.00	**Average score = 3.83**

Table 5.4 – Statistical evaluation of the DW questionnaire

It can clearly be concluded from the summary of the data that Kimball scored higher than Inmon in every criteria of the evaluation. Therefore, Kimball's approach is best suited for the agile development of a data warehouse. In the rest of this section a discussion based on the knowledge gained from the awareness stage of the research will support and explain the possible reasons behind the scores generated by the DW questionnaire. This discussion will aim to validate the claim that Kimball's approach is best suited for the agile development of a DW system.

5.2.1 Inmon's approach

Inmon's approach to the development of a DW will be discussed according to the four stages of DW development. Each phase will be discussed for the general characteristics of ASDMs in regard to suitable and unsuitable characteristics.

5.2.1.1 Requirements gathering

Inmon (2003:144) states "requirements for a data warehouse cannot be known *a priori*". Inmon's approach does not collect requirements before the development of the DW, but rather requirements gathering take place after deployment by using the "feedback loop". Inmon's approach follows a data-driven methodology where data is the most important. This approach begins with integrated and tested data, whereafter programs are developed that use this data (Inmon, 2003:25). The results of the programs are analysed to identify and understand the requirements. Inmon believes that requirements can only be understood once the first iteration of the DW has already been developed. The identified requirements are then implemented into the existing DW by the data architect in order to keep the DW up to date and of value to the users.

Suitable characteristics

Even though Inmon's approach only collects requirements at the end of the development process, the main goal is still to satisfy the users, which are the same for ASDMs. Inmon states that users do not know what they want and therefore the available data is all that is needed

for the development process. Once the development process is complete the data architect will let the users know what their requirements are based on the developed DW.

ASDMs are suitable in regard to the fact that they aim to please the customers as the primary objective. ASDMs also agree that users rarely know exactly what they want.

Unsuitable characteristics

Even though ASDMs agree with Inmon that users rarely know what they want and that requirements gathering is bound to change. ASDMs do not agree with the approach Inmon suggests where requirements are only gathered at the end of the development process. Instead ASDMs emphasize the gathering of requirements before the development process with a flexible approach that allows users to continuously add and change requirements until a satisfactory set of requirements is identified that will result in a DW that will be of value to the users.

Therefore Inmon's approach in regard to requirements gathering will not be a suitable fit for ASDMs. Additionally, ASDMs review practices ensure that requirements are met in the learning phase where the direction of the development process can be redirected early in the development process, whereas Inmon's approach can only be justified after the development process, which may result in something that the users never wanted.

5.2.1.2 Data modelling

Inmon's approach uses an ERD (entity relationship diagram) data model for the data modelling phase. A data model can be created by using the corporate model as a starting point. The corporate ERD is a combination of ERDs with each ERD reflecting the different views of people in an organization.

Suitable characteristics

ASDMs do not uniquely specify which data model should be used when modelling the raw data since ASDMs are process-oriented models that focus on customer involvement and iterative development, not a data modelling technique. ASDMs do, however, reflect that IS projects are unique and that each project has different characteristics that must be taken into account and a DW system requires data model(s). Using an ERD data model to model the raw data will allow the project team and users to empower themselves to make their own decision without the explicit approval of higher management. This technique therefore allows for self-organization, which is one of the philosophies of ASDMs (Section 2.3.1). Therefore ASDMs allow for the project team to select its own data modelling approach (which can be Inmon's ERD data model approach) for the unique environment in which it finds itself.

Unsuitable characteristics

ASDMs provide the ability to deliver a system (this also includes DW systems) in a constantly changing environment by using incremental and iterative development with the focus on communication and requirements satisfaction. Inmon's approach aims to perform the data modelling phase as one big bang, modelling the entire data structure of the organization and therefore does not allow for incremental development during the data modelling phase. ASDMs also emphasize that new or changed requirements must be incorporated in the design as development progresses. Since Inmon only gathers requirements after the development process, it will only be possible to implement the requirements in the design of the DW after the development of the DW is completed. Therefore the use of ASDMs with Inmon's approach in regard to the data modelling phase is not possible.

5.2.1.3 Extract, transform and load (ETL)

During the ETL process, data cleansing is performed by extracting the required data from trusted sources, and is then transformed using tools and loading the transformed data into a central DW. Inmon's approach is a data-driven methodology and therefore the data is viewed as the most important factor.

Suitable characteristics

In order to perform the ETL process of Inmon's approach, a program has to be developed to extract and transform the necessary data for the DW. The data should then be analysed (for example combined, merged and redefined) and if the analysed data does not satisfy the specified needs, a new iteration is started with redefined and new technical and development requirements. This will cause the developed program to be changed and modified to meet the new development requirements, which can lead to unwanted delays.

The ETL process can be performed using the lifecycle of any ASDM where the ETL process is performed using iterative development. In regard to requirements gathering, which is performed at the start of an ASDM lifecycle, new development requirements of Inmon's ETL process are used instead of user-defined requirements. Inmon's approach promotes the use of iterative development, which is one of the philosophies of ASDMs (section 1.2.2).

Unsuitable characteristics

Inmon's approach encourages the use of iterative DW development. Inmon's approach does, however, have a disadvantage in the ETL process due to the fact that users are not involved in the process. ASDMs are people-oriented and require that users be involved in each phase of development including the ETL process in order to ensure that requirements are met and

developed as specified. Since Inmon's approach only identifies the requirements during the "feedback loop", only a fraction of the necessary data will be retrieved from the data warehouse during the first few iterations of the ETL process. For each iteration, the program that performs the ETL process has to be re-coded, which leads to a waste of time. Therefore, Inmon's approach to the development of a DW does not provide enough advantages to be considered a viable choice.

5.2.1.4 Data access and deployment

In Inmon's approach there are two kinds of data access in a DW, which are direct access and indirect access (2003:128). Once alpha and beta testing has been performed the DW is deployed and populated from the existing system. After the deployment process, new requirements are identified and implemented using the "feedback loop" between the DSS (decision support system) analyst and the data architect.

Suitable characteristics

Inmon (2003:96) stresses the importance that "in all cases the data warehouse is best built iteratively", which also includes iterative deployment. This allows for new requirements identified during the "feedback loop" to be implemented by the data architect into the existing DW. Iterative and incremental delivery and deployment are key to acceptable business solutions (Grey, 2006:165) which is one of the nine principles that reflect the core values of agile development (section 1.2.2).

Inmon performs alpha and beta testing and all the ASDMs perform excessive testing in order to ensure the quality of the delivered product as well as to ensure that user requirements are met. Using Inmon's "feedback loop" and testing to ensure the implementation of the user's requirements have been met will lead to satisfied users, which is one of the core values of the agile movement.

This phase of Inmon's approach has the most appealing reasons for using ASDMs in the development of a DW system using Inmon's approach because of the unique "feedback loop". During the "feedback loop" requirements are identified and added to the DW which is a fundamental approach used in all ASDMs. This also allows for user involvement, which is one of the principles of the agile movement.

Unsuitable characteristics

According to Inmon (2003:66), the project team must include an "official organizational explanation (standards manual) and description of the data warehouse". ASDMs are considered lightweight methodologies and tend to keep documentation to a bare minimum

and the documentation required by Inmon's approach makes it difficult to adjust it to a suitable ASDM. ASDMs do not provide specific guidelines on how to perform data access and deployment for a DW system. They also do not provide best practices and tools on reporting.

Although ASDMs do not provide these technical aspects, such as specific methods to perform data access and deployment and reporting, they focus more on the management of successful projects and provide techniques that will assist useful approaches to develop a system that is accepted by the users and insist that the project team choose technical techniques for projects based on the uniqueness of each project. Therefore it will be possible to perform data access and deployment with Inmon's approach using an ASDM.

5.2.2 Kimball's approach

Just as for the evaluation of Inmon's approach, Kimball's approach will be evaluated in regard to the general characteristics of ASDMs and the four stages of DW system development. The suitable and unsuitable characteristics in regard to each stage will be discussed.

5.2.2.1 Requirements gathering

Kimball's approach follows a requirements-driven methodology for the development of a DW system. This means that the DW system requirements must first be gathered in order to understand what the DW system sets out to achieve. The design and development of the DW system is then performed around the specified requirements and the success of the system depends on the successful implementation of the requirements.

Kimball *et al*. (2008:xxxiii) state that "we have learned to be humble in the presence of business users. It's not our opinion that matters; it's theirs". From this statement it can be concluded that Kimball's approach uses a requirements-gathering approach not because the requirements/goals of the system are the driving force of the project but rather that customer satisfaction and the usefulness of the system to the customer are the main focus of the DW system. Therefore Kimball's approach is requirements-driven in order to gather and understand what the customers deem valuable and then aim to develop a DW system that meets the requirements of the customers.

A well-developed DW system is continuously evolving and dynamic as new business requirements are discovered, and new managers and new executives place unexpected demands on the DW system or new data sources become available (Kimball *et al*., 2008:xxxii). Therefore the development process of a DW system requires techniques that are flexible and adaptable (Kimball *et al*., 2008:xxxii).

Suitable characteristics

One of the main concerns of ASDM development is customer satisfaction. In section 1.2.2 the agile manifesto states that the highest priority of ASDMs is to satisfy the customer through early and continuous delivery of valuable software as well as welcoming changing requirements throughout the entire development process. In order to successfully satisfy customers, the project team needs to identify and understand the customers' requirements to develop a DW system that will provide value to the customers.

Kimball's approach suggests that interviews or facilitated sessions be used to gather the necessary information and requirements to develop a useful DW system. Although interviews are time consuming, they lead to detailed data and participation from different levels in an organization. During interviews, high-level requirements will be identified, which will become the primary requirements of the DW system and as development progresses, extra requirements can be added by the users. This allows for high-level requirements to be frozen early in the project development to allow for detailed investigation procedures to determine the consequences of the primary requirements (one of the values of agile processes).

Using an ASDM for the development of a DW system will provide for early customer identification, involvement and requirements-gathering, allowing for users to be actively involved throughout the development process. Additionally, the general characteristics of ASDMs allows for the use of any requirements-gathering techniques, including facilitated sessions and interviews, which are the suggested techniques of Kimball's approach. ASDMs aim to frequently deliver fragments of the system in increments for feedback on the developed requirements, which forces continuous customer involvement and emphasizes the fact that frequent delivery has the highest priority. Additionally, ASDMs promote the idea of having users involved at all times during the development process to keep developers on track to satisfy requirements.

Unsuitable characteristics

The primary concern of ASDMs is to focus on the development process by managing the process and developing a system that will satisfy user requirements, rather than the gathering of requirements. Kimball emphasizes focus on requirements-gathering for the sole purpose of developing a system that satisfies the user's requirements. Therefore there are not really any unsuitable characteristics in using ASDMs with Kimball's approach in regard to requirements gathering.

In conclusion, the main driving force of using a requirements-driven approach is to identify and satisfy the user's needs and therefore ASDMs are a perfect match for Kimball's approach in regard to requirements-gathering.

4.2.2.2 Data modelling

Once the high-level requirements have been identified, Kimball's approach moves to the data modelling stage. In this stage the requirements identified must be modelled into diagrams. Kimball suggests a star schema that represents a dimensional model.

Suitable characteristics

ASDMs do not uniquely specify which data model should be used when modelling requirements, since ASDMs are process-oriented methodologies that focus on customer involvement and iterative development, not a data modelling technique. ASDMs do, however, reflect that IS projects are unique and that each project has different characteristics that must be taken into account and a DW system requires data model(s). Using a star schema to model requirements will allow the project team and users to empower themselves to make their own decision without the explicit approval of higher management. This technique therefore allows for self-organization, which is one of the philosophies of ASDMs (Section 2.3.1). Therefore using ASDMs in the data modelling stage of a DW system development would indeed be possible when using Kimball's approach.

Using Kimball's star-schema approach for the data modelling stage has two strengths in regard to using ASDMs for the development of a DW system. Kimball *et al*. (2008:148) state that the first strength of using a star schema is to "withstand unexpected changes in user behavior". The second strength is that a star schema "is gracefully extensible to accommodate unexpected new data elements and new designs" (Kimball *et al*., 2008:148). Adaptability to change has the advantage where no reporting or query tools need to be re-developed to accommodate for changes made by the customers or developers. Therefore, "old applications can continue to run without yielding different results" (Kimball *et al*., 2008:148). The main objective of implementing ASDMs in an organizational setting is to deliver IS quickly, to quickly adapt to change and to allow for change as frequently as possible (Highsmith, 2002a:4). Therefore the use of Kimball's star-schema approach can easily adapt to change, which is one of the main reasons for implementing an ASDM.

Unsuitable characteristics

ASDMs do not provide explanations of star schemes and ERDs as data modelling techniques; in fact they do not provide any techniques for data modelling.

In conclusion, ASDMs do not place any focus or provide any techniques for data modelling, but ASDMs understand the uniqueness of each IS project and instructs that the appropriate techniques for each IS project be used. Therefore, Kimball's star-schema method is supported

by all ASDMs. Additionally, star schemes can gracefully adapt to change, which makes it suitable for ASDMs.

5.2.2.3 Extract, transform and load (ETL)

During the ETL stage, data is extracted from the data source system(s), transformed to meet DW standards and cleansed before it is loaded into the DW database. In section 3.5.5 the ten steps of Kimball's approach to the ETL process is explained.

Suitable characteristics

Kimball *et al*. (2008:630) mention that the ETL process can be done iteratively. Therefore the ETL process of a DW system can be performed using an ASDM, since ASDM uses iterative development and can be viewed as the development process of every ASDM. In step two of the ETL process, Kimball *et al*. (2008:612) agree that tools can be used to clean the data during the data-staging process and when using an ASDM, the project team may use its own tools and techniques to perform a specific task in the development process (one of the values of agile, which is discussed in section 1.2.2). Therefore, the project team may use any tool that empowers it to perform the ETL process when using an ASDM.

ASDMs use incremental development for IS development and each increment can be developed iteratively so that changed or new requirements can be effectively managed. The iterative and incremental development process used by ASDMs can be applied to the ETL process where a business process dimensional model can be seen as an increment or a set of increments that needs to be developed. Each business process dimensional model is developed and deployed incrementally and rapidly, placing emphasis on the idea that rapid iteration and incremental delivery stimulate user satisfaction, which is one of the values of agile development as discussed in section 1.2.2. New and changing requirements can be added as the development progresses for a particular business process dimensional model, which allows for the development process to implement the same concept of ASDMs.

Therefore, the ETL process for each business process dimensional model using Kimball's approach can be divided into iterations. This means that the ETL process is completed iteratively to guarantee that altered technical and user requirements are successfully completed during each increment. Using incremental and iterative development allows for new and modified requirements to be added at any stage of development.

Unsuitable characteristics

ASDMs do not provide a list of tools and techniques that can be used for the ETL process.

ASDMs allow the team to select tools and techniques that are appropriate for the unique development environment of each IS project. Therefore, Kimball's ETL process is supported by all ASDMs.

5.2.2.4 Data access and deployment

Data access refers to the applications that are developed that provide business users with the ability to access the relevant data as speedily and easily as possible in order to analyse the data. OLAP is used to develop applications that give business users access to the data in order to perform queries on the DW database. Before the project team can begin the deployment of the DW system, alpha and beta testing must be performed.

The deployment process takes place when the DW system has been completely developed and at this stage the end users can begin to use the DW system and benefit from the system as well as evaluate whether the project team has developed a DW system that is of value to them. During the deployment process, business users must be trained to use the DW system in order for them to reap the full benefits of the system. After successful deployment of the DW system, it is important to manage and maintain the DW system and to prepare it for future growth.

Suitable characteristics

When using an ASDM in DW development and deployment, a small release or artefact can be seen as a business process dimensional model that is delivered or integrated into the main DW database. ASDMs use incremental development, which makes it possible to add new iterations for further growth of the DW system.

ASDMs aim to satisfy the users and to provide them with a system that is of value to them. This entails that users must be trained to effectively use the DW system (also performed by Kimball's approach). If new requirements surface after a business process dimensional model has been deployed, the new requirements can be added in the next iteration. This allows for backtracking or reconstruction of previous versions of business process dimensional model development (one of the values of agile, which was discussed in section 1.2.2).

Unsuitable characteristics

ASDMs do not specify helpful techniques for data access, which reporting tools to use, and how users should be educated in using the DW system.

Kimball's approach does not have any major conflict to that of ASDMs in regard to data access and deployment and it will be possible to use Kimball's approach with an ASDM to perform data access and deployment.

In conclusion, it is clear that most of the unsuitable characteristics (especially those identified in Kimball's evaluation) are associated with the fact that ASDMs do not provide support for the technical and unique aspects associated with DW development. These shortcomings can be viewed as reasons to discard the idea for the agile development of a DW system, but when considering a hybrid approach that uses a DW development approach, such as Kimball's approach and ASDMs, the hybrid approach will equip the project team with the tools, techniques, and best practices that deal with the unique environment of a DW system as well as provide the team with flexibility and adaptability to deal with uncertainty and changing requirements.

This concludes the possible explanation behind the data generated from the DW questionnaire and it can be clearly seen from the explanation that the data generated by the questionnaire is not just the opinions of six respondents, but the knowledge gained from the literature study in Chapters 2 and 3 agrees with and supports the results of the DW questionnaire and the claim that Kimball's approach is best suited for the agile development of a DW system.

5.3 Evaluation of the three ASDMs

In this section the three ASDMs (XP, SCRUM, FDD) discussed in section 2.4 will be evaluated for their suitable and unsuitable characteristics for implementation with Kimball's DW development approach. In the previous section the evaluation of Inmon's and Kimball's approaches were evaluated in regard to the four primary stages of DW development. In this section each of the three ASDMs will be evaluated according to the entire lifecycle of Kimball's approach rather than just the four primary stages of DW development. The most suitable ASDM for a particular stage in Kimball's approach will be selected and used to develop the ADW (version 0) methodology using Kimball's approach as the foundation. Kimball's lifecycle has 13 stages but using a different ASDM for each stage will just make the hybrid ASDM framework messy and complicated. Rather, the 13 stages of Kimball's lifecycle will be categorized into four primary phases with the 13 stages of Kimball's lifecycle as sub-phases. The most suitable ASDM for each of the primary phases will then be selected and used to develop the ADW (version 0) methodology. The four primary phases with their sub-phases are:

- **Planning**
    - Program/project planning

- Program/project management
- Business requirements definition
- Technical architecture design
- Product selection and installation

- **Design**
  - Dimensional modelling
  - Physical design
  - BI application design
- **Development**
  - ETL, Design and Development
  - BI application development
- **Implementation**
  - Deployment
  - Growth
  - Maintenance

Using the above categorization of the 13 stages of Kimball's approach, the ASDM questionnaire was developed and distributed to the same target group as the DW questionnaire (section 5.2).. The target group has experience in using Kimball's approach for the development of a DW system as well as experience in using XP, SCRUM, and FDD in the development of ISs. The ASDM questionnaire was researcher-administered and used the same agree-disagree scoring system presented in Table 5.1 (section 5.2). The ASDM questionnaire used the four major phases of Kimball's lifecycle and their associated sub-phases (discussed above) as the evaluation criteria. The data generated by the ASDM questionnaire is represented in a manner similar to that in Tables 5.2 and 5.3 and is presented below.

Evaluation of the three ASDMs: Respondents 1-6 (Data generated)

Kimball's DW development stages	Extreme programming (XP)	SCRUM	Feature driven development (FDD)
Planning	**15;16;15;16;15;16**	**21;20;18;20;19;19**	**12;12;12;12;11**

Program/project planning	3;3;3;4;3;3	5;5;4;4;4;4	4;3;3;3;3;3
Program/project management	3;2;3;3;2;3	5;5;5;5;5;5	2;2;2;3;2;3
Business requirements definition	4;5;4;4;5;4	5;4;4;5;5;5	3;3;3;2;2;1
Technical architecture design	3;4;3;2;2;3	3;4;2;3;3;3	2;2;2;1;2;2
Product selection & installation	2;2;2;3;3;3	3;2;3;3;2;2;	1;2;2;3;2;2
Design	**10;11;11;9;12;9**	**8;8;7;8;7;7**	**12;11;12;12;11;12**
Dimensional modelling	4;3;3;3;4;3	2;3;2;4;3;2	4;3;4;4;3;4
Physical design	3;4;4;3;4;3	3;2;1;1;2;2	4;3;4;5;4;4
BI application design	3;4;4;3;4;3	3;3;4;3;2;3	4;5;4;3;4;4
Development	**8;8;7;9;9;9**	**6;5;5;4;6;4**	**9;7;7;7;7;7**
ETL design & development	4;3;4;5;4;4	3;2;3;2;3;1	4;3;4;2;3;3
BI application development	4;5;3;4;5;5	3;3;2;2;3;3	5;4;3;5;4;4
Implementation	**13;12;13;11;13;12**	**14;14;14;11;14;14**	**10;12;11;10;11;11**
Deployment	5;4;5;4;5;4	4;5;5;4;5;5	3;4;3;3;4;3
Growth	5;5;5;4;5;5	5;5;5;4;5;5	5;5;5;4;5;5
Maintenance	3;3;3;3;3;3	5;4;4;3;4;4	2;3;3;3;2;3

Table 5.5 – Data generated by the ASDM questionnaire

Similar to the DW questionnaire, a reliability test was performed using Krippendorff's Alpha and the data produced a reliability alpha of 0.715. Therefore, the data generated by the ASDM questionnaire is reliable and trustworthy. Using the generated data, basic statistical analysis was performed in order to clearly interpret and determine which ASDM is best suited for each phase of development. The statistical analysis consisted of calculating the total and average score for each evaluation criteria and is presented below in Table 5.6.

Evaluation of the three ASDMs: Respondents 1-6 (Analysis)			
Kimball's DW development stages	**Extreme programming (XP)**	**SCRUM**	**Feature driven development (FDD)**

Planning	**Total score = 93** **Average score = 3.1**	**Total score = 117** **Average score = 3.9**	**Total score = 71** **Average score = 2.37**
Program/project planning	Total score = 19 Average score = 3.17	Total score = 26 Average score = 4.33	Total score = 19 Average score = 3.17
Program/project management	Total score = 16 Average score = 2.67	Total score = 30 Average score = 5	Total score = 14 Average score = 2.33
Business requirements definition	Total core = 26 Average score = 4.33	Total score = 28 Average score = 4.67	Total score = 14 Average score = 2.33
Technical architecture design	Total score = 17 Average score = 2.83	Total score = 18 Average score = 3	Total score = 11 Average score = 1.83
Product selection & installation	Total score = 15 Average score = 2.5	Total score = 15 Average score = 2.5	Total score = 12 Average score = 2
Design	**Total score = 62** **Average score = 3.44**	**Total score = 45** **Average score = 2.5**	**Total score = 70** **Average score = 3.89**
Dimensional modelling	Total score = 20 Average score = 3.33	Total score = 16 Average score = 2.67	Total score = 22 Average score = 3.67
Physical design	Total score = 21 Average score = 3.5	Total score = 11 Average score = 1.83	Total score = 24 Average score = 4
BI application design	Total score = 21 Average score = 3.5	Total score = 18 Average score = 3	Total score = 24 Average score = 4
Development	**Total score = 50** **Average score = 4.17**	**Total score = 30** **Average score = 2.5**	**Total score = 44** **Average score = 3.67**
ETL design & development	Total score = 24 Average score = 4	Total score = 14 Average score = 2.33	Total score = 19 Average score = 3.17

Kimball's DW development stages	**Extreme programming (XP)**	**SCRUM**	**Feature driven development (FDD)**
BI application development	Total score = 26 Average score = 4.33	Total score = 16 Average score = 2.67	Total score = 25 Average score = 4.17
Implementation	**Total score = 74** **Average score = 4.11**	**Total score = 81** **Average score = 4.5**	**Total score = 65** **Average score = 3.61**
Deployment	Total score = 27 Average score = 4.5	Total score = 28 Average score = 4.67	Total score = 20 Average score = 3.33
Growth	Total score = 29 Average score = 4.83	Total score = 29 Average score = 4.83	Total score = 29 Average score = 4.83

Maintenance	Total score = 18 Average score = 3	Total score = 24 Average score = 4	Total score = 16 Average score = 2.67

Table 5.6 – Statistical summary of the data generated by the ASDM questionnaire

The purpose of the evaluation was to determine the best-suited ASDM for each major phase rather than for each sub-phase in order to prevent the developed ADW methodology from being overly complex and messy. Therefore, the ASDM that scored the highest in each major phase was selected regardless of whether another ASDM had the same score for one of the sub-phases (product selection and installation, and growth) in that particular major phase. It must also be noted that some of the selected ASDMs for each major phase scored lower than desired (3 and lower) in some of its associated sub-phases. This is due to the fact that ASDMs are generalized IS development approaches and do not aim to address unique technical aspects associated with DW development. This however does not raise any concern since the ADW methodology will use Kimball's approach as the foundation to deal with the technical aspects associated to DW development. Given the summarization in Table 5.6, it can be concluded that the best-suited ASDM for each major phase of Kimball's DW development approach is

- Planning – SCRUM
- Design – FDD
- Development – XP
- Implementation – SCRUM

In the rest of this section (similar to section 5.2), a possible explanation that supports and validates the data generated and conclusions drawn from the data will be discussed based on the knowledge gained through the literature study performed in Chapters 2 and 3. The explanation will discuss the suitable and unsuitable characteristics of each of the three ASDMs in regard to the phases of Kimball's approach.

5.3.1 Planning

In this section the three ASDMs will be evaluated for their suitability in regard to the planning phase for the development of a DW system using Kimball's lifecycle as the basis. The sub-phases that make up the planning phase are (Kimball *et al*., 2008:7-9)

- **Program/project planning** - The program/project planning stage of Kimball's approach requires that the scope, resource staffing, coupled with project task identification, assignment, duration and sequencing be performed. It also includes data analysis and justification of the project (referring to time and cost of the project compared to the value the DW will provide).

- **Program/project management** – This stage ensures that Kimball's lifecycle activities remain on track and focus on monitoring project status, issue monitoring and change control.

- **Business requirements definition –** During this stage requirements are gathered and Kimball's approach suggests using interviews and facilitated meetings as the desired techniques for gathering the requirements.

- **Technical architecture design** – The technical architecture design establishes the overall architectural framework and vision and consists of three factors – business requirements, current technical environment and planned strategic technical directions.

- **Product selection and installation** – this stage uses the technical architecture plan as a virtual shopping list to evaluate and select the appropriate software and tools for each stage of development of the DW. The selected software is then installed and tested.

5.3.1.1 Extreme programming (XP)

The XP methodology is explained in section 2.4.1 and will not be discussed again. The suitable and unsuitable characteristics with reference to Kimball's approach in regard to the sub-phases of the planning phase will be discussed below.

Suitable characteristics

- **Program/project planning** – XP suites this stage of Kimball's lifecycle since the metaphor core practice serves as a guideline for the development process and explains the systems behaviour (this can include resource staffing, task identification, duration and sequencing) and defines the scope and purpose of the project. User stories are used to determine the cost and time of the project, which can be used to justify the DW system.

- **Program/project management** – XP requires that there be a customer on site at all times to ensure the development stays on track. This also makes the issue of change control easier. Communication and feedback (values of XP) assist in the management of the project to ensure that the project manager is well aware of how the DW project is progressing.

- **Business requirements definition –** Although XP uses story cards to identify the user requirements, these story cards are obtained through interviews or facilitated meetings, which correspond to the techniques suggested by Kimball's approach.

- **Technical architecture design** – The metaphor practice describes the complete vision of the project.

- **Product selection and installation –** XP, like all ASDMs, allows the team to select software and tools that meets the unique environment of the IS project.

Unsuitable characteristics

- **Program/project planning** – XP does not specify that data analysis be conducted. The time and cost of each story are performed to justify the system but XP does not explicitly state that additional cost be included in the cost analysis (i.e. software cost, hardware cost, etc.). The purpose of the metaphor core practice is to have a complete and simple way to describe the IS project's vision but does not specify which documents must be created to achieve this. XP does not explicitly state that resource staffing, task identification, duration and sequencing be documented and how to achieve this. XP performs iteration planning, which can relate to program planning in DW system development but does not mention performing planning of the entire project (program planning).

- **Program/project management** – XP is mostly development focused and does not provide techniques to perform successful project management.

- **Business requirements definition –** XP allows for the use of interviews and facilitated meeting but includes a twist (the use of “user stories”) that is not supported by Kimball's approach.

- **Technical architecture design** – XP explains that the metaphor core practice must explain the IS project’s vision that defines the scope and purpose of the project but Kimball's approach requires that much more be analysed and included in the technical architecture.

- **Product selection and installation –** XP leaves the selection of products to the project team but does not provide any techniques or approaches on how to perform this.

In conclusion XP states that the entire IS system be explained using the metaphor practice, which can mean that for the development of a DW system the necessary documentation

mentioned by Kimball's approach can also be included. XP is a universal SDM that does not cater for a specific IS project (such as a DW system) and therefore does not explicitly mention tools, techniques and required documentation that will assist in the development of a DW system. Regardless of all this, XP provides a foundation for the project team and expects the team to make use of any tools and techniques and to include documentation it believes is required for its unique project. Therefore XP is a viable choice for the planning phase of a DW system but may prove to be inferior to SCRUM and FDD.

In the next section SCRUM will be evaluated in the same way that XP was evaluated in regard to the planning phase of a DW project.

5.3.1.2 SCRUM

The SCRUM methodology is explained in section 2.4.2 and will not be discussed again. The suitable and unsuitable characteristics with reference to Kimball's approach in regard to the sub-phases of the planning phase will be discussed below.

Suitable characteristics

- **Program/project planning** – The product backlog of SCRUM is part of the planning phase and contains all the work required during the entire project based on the identified requirements. The product backlog also includes the definition of the system being developed, the project team, training needs and verification of management's approval. The product owner is responsible for ensuring that the necessary funding is available. Before the start of each sprint, pre-sprint planning is performed which is similar to program planning

- **Program/project management** – SCRUM's primary concern rests with a few key management tasks rather than with the development process. The product backlog also includes risk assessment and controlling issues. The SCRUM master is responsible for the success of the entire project. SCRUM requires that pre- and post-sprint meetings be conducted to determine what needs to be completed for each sprint and analyses the results of each sprint. In addition, each morning a short fifteen-minute meeting is held to keep track of development. These meeting help to identify issues encountered and solutions to these issues are discussed.

- **Business requirements definition –** SCRUM states that any means necessary can be used to collect requirements from the software developers, users and experts. This includes interviews and facilitated meetings.

- **Technical architecture design** – Apart from what was mentioned above, the product backlog also contains tools and other resources. SCRUM mentions an architecture phase that is completed along with the planning phase, which includes any changes made and problems these changes may cause when implementing the product backlog.

- **Product selection and installation –** The project team is responsible for the development of the actual system and it can choose tools it feels will best suit its project.

Unsuitable characteristics

- **Program/project planning** – SCRUM does not focus solely on DW system development and therefore does not specify that data analysis be performed to ensure that the necessary data is available to meet user requirements.

- **Program/project management** – SCRUM is primarily focused on the management tasks and techniques required to develop an IS and therefore does not have any unsuitable characteristics in this area.

- **Business requirements definition –** SCRUM allows for the requirements to be gathered by any means possible and therefore has no unsuitable characteristics to Kimball's approach, since Kimball's suggested approach to requirements gathering can be used as the means to acquire the requirements when using SCRUM.

- **Technical architecture design** – SCRUM does not mention what tasks and activities need to be conducted in the architectural phase or what documentation must be performed.

- **Product selection and installation –** SCRUM is more concerned with the management aspect of the IS system and leaves the development to the project team. This includes the software, tools and hardware selection and does not provide assistance in selecting the most suitable products.

In conclusion, SCRUM is a very promising ASDM to use when implementing the sub-phases of the planning phase for Kimball's approach. SCRUM provides methods for most of the crucial sub-phases and the product backlog serves to support most of the documentation required by Kimball's approach.

5.3.1.3 Feature-driven development (FDD)

The FDD methodology is explained in section 2.4.3 and will not be discussed again. The suitable and unsuitable characteristics with reference to Kimball's approach in regard to the sub-phases of the planning phase will be discussed below.

Suitable characteristics

- **Program/project planning** – During the “develop an overall model” the domain experts become familiar with the scope and context of the IS. Once the domain experts are familiar with the idea that the users have for the IS, a walk-through is performed where the chief architect informs the team members of the main requirements and the system definition. One of the tasks required by FDD in the “develop an overall model” process is to study the relevant documents and in regard to DW system development, the documents mentioned by Kimball's approach can be studied and the preferred documents can be included in the overall model. Once the feature list is complete it must be reviewed by the users to verify the project.

- **Program/project management** – FDD focuses on feature development and places management focus on the successful development of features. Monitoring of the progress of feature development is performed.

- **Business requirements definition –** During the “develop an overall model” the domain experts become familiar with the requirements of the IS. FDD uses features that represent requirements obtained and during the development process the focus is placed on the features and development orbits around the features. This ensures that the user requirements are the primary driving force for development.

- **Technical architecture design** – FDD is primarily concerned with the way IS are developed and plans according to the development of features. Therefore FDD does not specify or include analyses of the technical architecture.

- **Product selection and installation –** FDD allows for the project team to select the products it feels will best suit the development environment for the IS project.

Unsuitable characteristics

- **Program/project planning** – FDD is development-driven and does not mention what activities and documents needs to be performed for the planning of the IS system.

- **Program/project management** – FDD only focuses on the management of feature development and neglects to mention techniques and approaches to manage the project team.

- **Business requirements definition –** FDD does not mention what approach or techniques must be used to acquire requirements.

- **Technical architecture design** – FDD only mentions the development of a framework that consists of the feature lists and does not mention the analysis of the technical architecture or what is needed for the DW system to function on the current system. FDD does not plan for the technical aspect of DW system development.

- **Product selection and installation –** FDD does not provide assistance in selecting appropriate products for DW development.

FDD places considerable emphasis on the design and development of features and fails to mention requirements-gathering approaches. FDD does not seem to be a very good fit for the planning phase of DW development.

In this section XP, SCRUM and XP were evaluated for their suitability for implementation with Kimball's approach in regard to the planning phase of DW system development. From the evaluation it seems that SCRUM is the most suited for the planning phase.

In the next section the evaluation performed in this section will be repeated in regard to the design phase for the development of a DW system.

5.3.2 Design

The sub-phases and their short descriptions that make up the design phase are

- **Dimensional modelling -** This phase focuses on modelling the identified user requirements into diagrams. The requirements are modelled into dimensional models which are also known as star schemes, which differ from the traditional entity-relationship diagrams for operational databases. During dimensional modelling, the data is divided into facts and dimension tables to support BI applications and each dimensional model represents a single business process. The collection of multiple models relates to each other and forms the complete DW database. Dimensional modelling adapts easily to new and changing requirements at any stage of development.

- **Physical design –** This phase defines the physical structures necessary to support the logical DW design. Indexing, partitioning and aggregation strategies are discussed

and a design is created to implement the appropriate indexing, partitioning and aggregation solutions for the DW system that will allow for increased performance and integration across departments. Naming standards are defined and setting up the DW environment is performed. If required by the specific DW project, OLAP databases are also designed during this phase of development.

- **BI application design –** This phase requires that team members work with the business to identify candidate BI applications along with an appropriate navigation interface (GUI) to address the users' needs and requirements.

5.3.2.1 Extreme programming (XP)

Suitable characteristics

- **Dimensional modelling -** Simple design (core practice) can be achieved by using the star schema that models the requirements into a simple and easy to understand framework. Dimensional modelling focuses on a single business process at a time, which corresponds to XP's simple design. The refactoring core practice requires that the dimensional model be re-designed once it deviates from the user's requirements, which also allows for flexibility when requirements are changed. Star schemes are adaptable to change and allow for new star schemes to be added as development progresses, which makes the incremental and iterative development process of XP suitable for the design of star schemes. XP adapts to changing requirements and requires that a user be on site at all times (a core practice of XP) allowing for new requirements to be implemented into the star schema.

- **Physical design -** The metaphor core practice can include the physical design to support the logical DW design in regard to DW projects since XP allows for the project team to be courageous and to use techniques it believes are necessary for the successful completion of the IS. XP sees design as a continuous and on-going process throughout the development process, which will allow for changes to the physical design to be added or for changes at any stage in the development process. Coding standards (core practice) require that the project team define naming standards.

- **BI application design –** The project team is responsible for the development of the system and can choose tools and techniques it believes are appropriate for the IS project. This also includes the selection of appropriate BI applications. The simple design core practice instructs the team to keep the navigational interface as simple as possible while still meeting the needs of the users. This allows for an easy to

understand and simplified development of the interface. It also allows users to not be overwhelmed by the interface.

Unsuitable characteristics

- **Dimensional modelling -** XP does not provide support on how to perform dimensional modelling or any assistance on how to design the fact and dimension tables.
- **Physical design –** XP is not a data-focused methodology and therefore does not provide tools, techniques and best practices to create the physical design for DW projects.
- **BI application design –** XP is focused on best practices associated with the development process for any IS project (not just for DW systems) and does not include in-depth detail on how to evaluate appropriate BI application and how to design the navigational interface.

The sub-phases that make up the design phase are focused on DW development where XP aims to provide best practice across a wide variety of IS projects and therefore does not provide best practice on how to perform the unique tasks associated with DW development. XP does, however, allow for the project team to adjust to each unique project, including tools, techniques and documentation that apply to project environments, which will make XP an acceptable choice for the design phase of DW development.

5.3.2.2 SCRUM

Suitable characteristics

- **Dimensional modelling -** Since SCRUM believes that requirements are most likely to change and implements a flexible and adaptable approach, it will be easy to remodel the star schemes as requirements are changed and added. During sprints in SCRUM, sectioned tasks must be performed to develop deliverables that meet the requirements of the users. Star schemes are formulated around the user requirements and can count as one of the sectioned tasks for each sprint when using the SCRUM methodology. Star schemes are adaptable to change and allow for new star schemes to be added as development progresses, which makes the incremental and iterative development process of SCRUM suitable for the dimensional modelling sub-phase. The independent project team is responsible for developing quality deliverables and the project may choose the methods and techniques in Kimball's lifecycle to perform dimensional modelling in regard to the development of a DW system.

- **Physical design –** The pre-sprint planning may include identification of what physical structures will be necessary to accomplish the current sprint goals. The sprint backlog may include the physical design necessary to implement the sprint goals. The short daily meeting aims to address issues encountered, which will also include issues encountered with the physical structure of the DW and how these issues can be addressed.

- **BI application design –** The project team is responsible for the development of the system and can choose tools and techniques it believes appropriate for the IS project. This also includes the selection of appropriate BI applications.

Unsuitable characteristics

- **Dimensional modelling -** SCRUM does provide any techniques to be used for dimensional modelling. SCRUM focuses more on the management methods and techniques and does not provide techniques on the design phase of IS development.

- **Physical design –** SCRUM is not focused on providing tools and techniques for a specific IS (such as a DW system) but aims to address issues encountered across a number of unique IS projects. Therefore, SCRUM does not provide approaches and techniques to perform the physical design of a DW project. SCRUM does not mention how to perform indexing, partitioning and aggregation to increase performance. SCRUM does not provide tools and technique to design the OLAP database and does not mention defining naming standards.

- **BI application design -** SCRUM does not mention how one must go about identifying candidate BI applications and does not mention how to design the navigation interface associated with DW systems or any interface design for that matter.

SCRUM places emphasis on management and what methods and techniques will assist the team in successfully completing the IS project. SCRUM leaves the responsibility of the actual design and development of the IS to the project team and does not discuss any approaches to follow that will assist in the design of the IS. If the project team is experienced in DW development then SCRUM may be a suitable methodology for DW development. SCRUM may prove to be inferior to XP and FDD in regard to the design phase of DW development.

5.3.2.3 Feature-driven development (FDD)

Suitable characteristics

- **Dimensional modelling -** Domain object modelling (best practice of FDD) defines the problem domain and investigates to provide a framework where features can be added. If the user requirements are seen as features then this best practice can be used to perform the star schemes. Like XP and SCRUM, FDD uses incremental and iterative development processes, which allow for new features to be added and included in the star schemes. One of the core values of FDD is that a simple, well-defined process works best and this suits dimensional modelling, since dimensional modelling aims to simplify the ERD model of the operational data into a simpler and easier to understand star schema. In the "develop an overall model" phase of FDD, the entire IS domain is divided into a number of smaller domains, which is similar to dimensional modelling where the entire DW database can be broken down focusing on a single business process and further breaking it down into dimensions and facts. The "develop an overall model" has a task that requires the team to develop the model, which may also include the development of the dimensional models in regard to DW development. The feature list of FDD is a promising component that can assist in dimensional modelling by viewing the user requirements as the highest level of the feature list that is broken down into single business processes that relate to the specific business process. The feature list can then be broken down further into the fact table, which is then broken down into dimensions to support BI applications that will meet the user requirements. By using the feature list in this manner, the entire DW project can be mapped across all business processes and the necessary dimensions and facts that make up the entire DW database. If necessary the dimension tables can further be broken down into the columns that they are made up of. This process performs dimensional modelling while mapping the entire DW database with all its dimensions and fact tables. The feature list will serve as a guideline for the ETL process and the smallest feature set can be viewed as a star schema.

- **Physical design –** One of the tasks of "develop an overall model" is to conduct external and internal assessments, which will also include the evaluation of the physical system, which will lead to the definition of the physical structure to support the logical DW design. Chief programmers in FDD are experienced developers who lead small teams in the analysis, design and development of new features, which will also include the design of the physical structures of the DW system.

- **BI application design –** During the "plan by feature" phase a development plan must be created, which will include the selection of BI applications for DW development. The chief programmer who is experienced and responsible for analysis, design and

development of new features will provide the team with candidate BI applications as well as the design of the navigational interface.

Unsuitable characteristics

- **Dimensional modelling -** FDD's feature list can easily be implemented into dimensional modelling but is not focused on DW development and therefore does not provide methods to achieve dimensional modelling.
- **Physical design –** In FDD, the feature owner is responsible for the design, development and implementation of individual features but it does not mention defining naming standards, which may cause issues when integration of different classes takes place. FDD is not a methodology that focuses on database development and therefore does not provide design approaches for query performance (indexing, partitioning and aggregation of the data).
- **BI application design –** FDD does not provide assistance in identifying candidate BI applications and does not mention or provide best practices for interface design. FDD leaves the chief programmer in charge of analysis, design and development of new features but the methodology itself does not provide best practices for BI application design.

FDD is focused on the design and building phases of IS development, using the feature list to perform dimensional modelling and depending on the experience of the chief programmer to assist the team in performing successful physical and BI application design, FDD becomes a promising candidate for the design phase when using Kimball's approach as the foundation.

XP, SCRUM and FDD have been evaluated in regard to the design phase of DW development and FDD is the most suitable candidate for this phase.

In the next section the evaluation process performed above will be repeated in regard to the development phase of DW development.

5.3.3 Development

The sub-phases and their short descriptions that make up the development phase are:

- **ETL, design and development –** This phase requires that the necessary data be extracted from the source systems, transformed according to DW standards and cleansed before being loaded into the DW database using the dimensional models for guidance. Two ETL processes need to be put in place, one for the initial population of the DW database and the other one for regular incremental loads.

- **BI application development –** BI application development includes the configuration of business metadata and tool infrastructure, construction and validation of the identified analytics and operational BI application as well as the development of the navigational portal.

5.3.3.1 Extreme programming (XP)

Suitable characteristics

- **ETL, design and development –** XP uses incremental development focusing on a small release (core practice of XP) at a time, which can relate to a single dimensional model. This fits perfectly with Kimball's approach where he believes one should start the ETL process by focusing on a single and easy to complete dimensional model. Incremental development allows for the ETL process to be performed one dimensional model at a time and at the end of each iteration another dimensional model is added to the DW that will eventually lead to the complete DW database. Continuous integration (core practice of XP) allows for new dimensional models that have been developed to be integrated into the current DW database. Coding (one of the four activities of XP) is viewed as a learning process where requirements are coded and transformed to satisfy the users need. Designing (one of the four activities of XP) allows for new ETL designs to be implemented as the project team continues to learn. XP uses pair programming (core practice) in the development phase of IS projects which increases the quality and eventually leads to more productive development process. Pair programming will lead to faster and higher quality ETL development as errors can be identified earlier and complex extractions, transformations and loads can be tackled by two programmers. Graziano (2005:5) mentions that pair programming can be used during the ETL process. Simple design (core practice) ensures that the ETL process is kept as simple as possible to make the tedious ETL process easier to take on. Collective ownership increases quality, reduces bugs and removes duplicates during the ETL process. A forty-hour week (core-practice) ensures that programmers remain fresh and prevents them from burning out, which leads to sloppy ETL development. On-site clients make it easy for the programmers to meet and to discuss issues encountered in the ETL development with the client. Coding standards (core practice) allow for new dimensional models to easily be integrated into the current DW database.

- **BI application development -** XP uses incremental development focusing on a small release (core practice of XP) at a time, which can relate to a single report that meets the user requirements. Using the completed ETL development that focuses on a single

dimensional model that aims to meet the reporting requirements of the users as the foundation for the BI application development, the BI application can be developed in increments where one user-requirements report is developed at a time. Continuous integration (core practice of XP) allows for new reports that have developed to be integrated into the current BI application. Coding (one of the four activities of XP) is viewed as a learning process where user-requirements reports are coded and transformed to satisfy the users' needs. Designing (one of the four activities of XP) allows for new BI reports to be implemented as the project team continues to learn. XP uses pair programming (core practice) in the development phase of IS projects which increases the quality and eventually leads to a more productive development process. Pair programming will lead to faster and higher quality BI application development as errors can be identified earlier and complex reports and the navigational portal can be tackled by two programmers. Simple design (core practice) ensures that the navigational portal is as simple as possible, which will make the development process easier to perform as well as easier for users to understand. Collective ownership increases quality, reduces bugs and removes duplicates during the development of the BI applications. A forty-hour week (core-practice) ensures that programmers remain fresh and prevents them from burning out, which leads to sloppy application development. An on-site client makes it easy for the programmers to meet and to discuss issues encountered in the BI application development with the client. It also assists in the validation of the identified analytics and operational BI application. Coding standards (core practice) allow for new reports to easily be integrated into the current BI application.

Unsuitable characteristics

- **ETL, Design and Development –** XP provides a number of best practices for development but does not specifically focus on DW development and therefore does not provide assistance for the practical side of ETL development.
- **BI application development –** XP does not provide assistance for configuring the business metadata and tool infrastructure and does not provide best practices that focus on BI application development.

XP places considerable emphasis on the development of IS and provides best practices that can easily be applied to DW development. XP is a good candidate for the development phase of DW development.

5.3.3.2 SCRUM

Suitable characteristics

- **ETL, Design and Development –** Sprints can be used to perform the ETL development of a single dimensional model and the increment developed during the sprint can be integrated into the current DW database. Daily SCRUM meetings will aid in the management of ETL development to ensure the team stays on track and that the ETL process is completed on time. SCRUM believes that every IS project has its own characteristics and uncertainties and allows the project team to use ETL development tools of its choice for the development of a DW system.

- **BI application development -** Sprints can be used to perform the BI application development of a single user-requirements report and the increment developed during the sprint can be integrated into the current BI application. Daily SCRUM meetings will aid in the management of BI application development to ensure the team stays on track and that the BI application development is completed on time. SCRUM believes that every IS project has its own characteristics and uncertainties and allows the project team to use BI application development tools of its choice for the development of a DW system.

Unsuitable characteristics

- **ETL, Design and Development –** SCRUM is focused on the management rather than the development side of IS development and does not provide best practices that focus on the development phase. SCRUM leaves the responsibility of development with the SCRUM master and does not provide any tools or techniques for ETL development in DW development.

- **BI application development –** The SCRUM master is burdened with the responsibility of the actual development of the system, which includes the BI application development for DW systems and therefore the methodology does not provide any tools or techniques to assist the project team in BI application development.

SCRUM focuses on the management side associated with IS development and does not provide best practices for the development phase but leaves this responsibility with the SCRUM master and project team. The management techniques of SCRUM will assist the

team to stay on track during the development phase and with an experienced project team, the use of SCRUM for the development of a DW project will lead to the successful development of the DW system. SCRUM does, however, fall short in comparison to XP in regard to the development phase of DW development.

5.3.3.3 Feature-driven development

Suitable characteristics

- **ETL, Design and Development –** FDD performs development by focusing on a feature set (dimensional model in regard to ETL development). Developing by feature sets includes incremental development where a single dimensional model (feature sets) can be used to perform ETL development and be integrated into the current DW database. This allows for incremental ETL development. Developing by feature will assist the team in monitoring the progress of the ETL process. Inspection (one of FFD's best practices) is performed by using the best-known defect-detection mechanisms, which will relate to ETL tools to detect any defects in the transformation and load steps in the ETL process. Regular builds (best practice of FDD) allows for new dimensional models to be added to the current DW database after their ETL process has been completed. “A simple, well-defined process works best” (one of the core values of FDD) will allow for the team to simplify the ETL process and to focus on a single dimensional model at a time. During the “design by feature and build by feature” phase in FDD the chief programmer selects a small group of features, which can relate to ETL development as the features that make up a single business process. This allows for the team to perform ETL development focusing on one business process at a time, thus leading to a simplified ETL process.

- **BI application development –** FDD performs development by focusing on a feature set (user reports in regard to BI application development). Developing by feature sets includes incremental development where a single user-defined report (feature sets) is developed and integrated into the current BI application. This allows for incremental BI application development. Developing by features will assist the team to monitor the progress of the BI application development. Inspection (one of FFD's best practices) is performed by using the best-known defect-detection mechanisms which will help to identify bugs and errors in the development of the BI application. Regular builds (best practice of FDD) allows for new user-defined reports to be developed and added to the current BI application. “A simple, well-defined process works best” (one of the core values of FDD) will allow for the project team to develop a simple BI application and navigational portal while still meeting the needs of the business users. This will allow

for the business users to more easily understand and use the BI application and navigational portal. During the “design by feature and build by feature” phase in FDD the chief programmer selects a small group of features, which can relate to the reports that are generated from a single business process, which will simplify the development of the BI application.

Unsuitable characteristics

- **ETL, Design and Development –** FDD does not recommend tools and techniques that are unique to ETL design and development.
- **BI application development –** FDD does not provide aid in configuring the business metadata and tool infrastructure and does not provide best practices that are unique to BI application development.

FDD is focused on the design and development of IS projects and performs development by feature. FDD provides a number of best practices to assist the team in the development of IS and therefore is a good candidate for the development phase for DW development.

SCRUM has some shortcomings when compared to XP and FDD in regard to the development phase of DW development. Although FDD is a good candidate, it cannot compete with the values that XP brings to the table. Therefore, XP is the most suitable candidate for the development phase for DW development.

In the next section the evaluation process performed above will be repeated in regard to the implementation phase of DW development.

5.3.4 Implementation

The sub-phases and their short descriptions that make up the implementation phase are:

- **Deployment –** The technology, data and BI application tracks integrate at the deployment phase. This phase requires extensive planning to ensure that the pieces are tested and fit together properly. It also requires that the necessary training and support infrastructure be put in place. Deployment should be delayed if all the pieces, such as training, documentation and validated data are not ready for release.
- **Growth –** If the DW system is successful, it is bound to grow and evolve to prove more value to the organization. This phase allows for the DW system to grow and the development cycle returns back to the beginning of the lifecycle and builds the new requirements upon the existing DW system.

- **Maintenance -** This phase includes technical operational tasks to ensure the system continuous to perform at optimal level. This includes usage monitoring, performance tuning, index maintenance and system backup. The project team must also provide on-going support, training and communication for the business users.

5.3.4.1 Extreme programming (XP)

Suitable characteristics

- **Deployment –** Testing (one of the core practices of XP) performs unit and acceptance testing. Unit testing ensures that the system does what it is intended to do. Acceptance testing ensures that the system developed corresponds with the user requirements and meets the expectations of the user. Unit testing can be applied to the technology, data and BI application tracks to ensure the DW system does what developers intended for the system. Continuous integration (core practice) makes the integration of the architecture, data and BI application tracks easy, since integration takes place on a daily basis during the development process and prevents the integration of large chunks of the system, which leads to time-consuming and tedious integrations. Coding standards (core practice) make the integration process easier as all the code classes appear to be developed by a single programmer. Simple design (core practice) makes it easier for business users to understand the system and makes the training process easier. Incremental and iterative development allows the DW system to be deployed in increments, eventually leading to a highly sophisticated DW system as the number of iterations increases.

- **Growth –** XP uses incremental and iterative development and welcomes the growth of the current DW system by returning to the start of the development process as a new iteration. XP delivers the system in increments called small releases (core practice).

- **Maintenance –** XP is user-focused and in order to keep the business users satisfied, it will require that maintenance, backups, usage monitoring, performance tuning and index maintenance be performed. This may also include on-going support to ensure users are satisfied and benefit from the system. Communication (one of the four values of XP) will allow for the project team to stay in contact with the business users and to provide additional training if needed.

Unsuitable characteristics

1. **Deployment –** XP is focused on the development of IS projects and does not provide techniques or approaches to put a support infrastructure in place. XP believes that the business users must be trained to use the system instead of providing the users with a user manual.

- **Growth -** XP does not have any unsuitable characteristics in regard to growth.

- **Maintenance –** XP does not particularly have a maintenance and training plan in place. XP does not provide best practices on dealing with usage monitoring, performance tuning, index maintenance and backups for DW systems.

XP meets the most crucial needs of the implementation phase of DW development and this phase will definitely be possible to complete using the XP methodology.

5.3.4.2 SCRUM

Suitable characteristics

- **Deployment –** The product backlog in SCRUM contains training needs and verification management approval. This ensures that the necessary training needs are met and that a support infrastructure be put in place (this may include a user-manual). Verification approval tests whether the DW system is accepted by the end users and the product owner is responsible for this and must sign off on all deliverables that have been completed. During pre-sprint planning, the team will discuss the integration of the technology, data and BI application integration and create an integration plan to use during the deployment step. The SCRUM master is responsible for the delivery of quality products and therefore will ensure that integration testing be performed in order to achieve this. The post-sprint meetings ensure that all the requirements have been met and that the DW system is functioning and performs as desired. The post-game phase, which is a built-in phase of the SCRUM methodology performs maintenance and implementation tasks.

- **Growth -** SCRUM uses incremental and iterative development and welcomes the growth of the current DW system by returning to the start of the development process as a new iteration.

- **Maintenance -** SCRUM is user-focused and in order to keep the business users satisfied, it will require that maintenance, backups, usage monitoring, performance tuning and index maintenance be performed. This may also include on-going support to ensure that users are satisfied and benefit from the system. The post-game phase,

which is a built-in phase of the SCRUM methodology performs maintenance and implementation tasks.

Unsuitable characteristics

- **Deployment -** SCRUM is mostly concerned with management tasks and leaves the team in charge of deployment. Therefore SCRUM does not provide any tools or techniques for implementing deployment in DW projects.
- **Growth –** SCRUM does not have any unsuitable characteristics in regard to growth.
- **Maintenance -** SCRUM does not explicitly state that it supports on-going maintenance and does not provide best practices on dealing with usage monitoring, performance tuning, index maintenance and backups for DW systems.

The only concern is that SCRUM does not cater for the unique characteristics associated with DW development. Although SCRUM does not cater for the unique characteristics associated with DW development, it does, however, provide management techniques to assist the project team in the successful deployment of a DW system and therefore is a suitable choice for the implementation phase of DW development.

5.3.4.3 Feature-driven development (FDD)

Suitable characteristics

- **Deployment –** Inspection (one of the best practices of FDD) ensures that testing be performed, which allows for a smoother integration process. During the "design and build by feature" process the feature owner is responsible for design, development and implementation. This includes testing and integration of the technology, data and BI application tracks. According to FDD, each iteration contains coding, code inspection, testing and integration. Therefore FDD tests the code and performs testing and integration.
- **Growth -** FDD uses incremental and iterative development and welcomes the growth of the current DW system by returning to the start of the development process as a new iteration.
- **Maintenance –** The best practices of FDD encompass incremental development to manage and monitor the development process after deployment (Abrahamsson *et al.*, 2002:47; Hislop *et al*., 2002:175). Progress reporting (best practice of FDD) requires that reporting be done at an organizational level on all completed features. This will

allow for the identification of maintenance requirements, which will lead to the effective maintenance of the DW system.

Unsuitable characteristics

- **Deployment –** FDD is concerned with the design and development of IS projects and does not mention developing a training plan or user support plan.
- **Growth –** FDD does not have any unsuitable characteristics in regard to growth.
- **Maintenance –** FDD does not mention the development of a training plan and does not provide best practices on dealing with usage monitoring, performance tuning, index maintenance and backups for DW systems.

In conclusion FDD is a suitable methodology for the implementation of a DW system and only falls short in regard to user training and that it is not focused on the unique characteristics associated with DW system development. SCRUM does, however, provide the most benefits for the implementation phase of DW development.

Most of the reasons that fuel the unsuitable characteristics for the three ASDMs in regard to Kimball's approach come from the fact that they aim to satisfy a broad range of IS projects and does not specifically address issues associated with DW development. Using Kimball's approach as the foundation and using the best practices from XP, SCRUM and FDD, it will be possible to create a hybrid methodology that address the technical issues associated with DW development as well as to implement agile best practices to improve and ensure the success rate of DW projects.

This concludes this section and it can be seen from the explanations provided above that the literature interpretations supports and agrees with the data generated by the ASDM questionnaire as well as the claims made from the interpretation of the data. In the next section the ADW (version 0) will be introduced and explained.

5.4 Version 0 of the agile data warehouse (ADW) methodology

In this section the ADW (version 0) methodology for the development of a DW system will be introduced. The ADW methodology will use Kimball’s approach as the foundation and a combination of the best practices of XP, SCRUM and FDD will be built on the foundation to provide an agile approach for DW development.

The layout of the ADW methodology can be explained by using a metaphor. Consider that a DW system is a pizza and the ADW methodology is the recipe for creating the pizza (DW system). It is common knowledge that the difference between a chicken barbeque and four

seasons pizza is in the toppings and the base of the pizzas is exactly the same. Therefore, the recipe for the base of a pizza remains the same regardless of what kind of pizza is being created. The new pizza (DW system) recipe (ADW methodology) will use Kimball's approach for the creation of the base of the pizza. Although each DW system is unique and has a unique development environment, there are certain components that will always be present regardless of the environment. These DW standard components can be seen as the base of the pizza and Kimball's approach will be used as the foundation for the development of the DW system. Using Kimball's approach as the recipe for the base of the pizza, a hybrid recipe containing the best practices of XP, SCRUM and FDD will be used to add flavour to the pizza. In the hybrid recipe, XP may represent the approach to prepare the meat, whereas FDD is used to create the sauce and SCRUM can, for instance represent the optimal heating temperature as well as the required time to bake the pizza. In addition, the ADW methodology uses incremental and iterative development. This implies that rather than creating one large pizza that takes a long time to create, mini pizzas are created instead. Creating mini pizzas allows for faster delivery that sates the appetite of the clients. It also allows for the clients to address issues with the pizza that must be addressed in the creation of the next mini pizza.

Therefore the ADW methodology uses Kimball's approach to address structure, technical aspects, and unique characteristics associated with DW development and XP's, SCRUM's and FDD's best practices to add flavour to the development process of a DW system. The ADW methodology will implement the best practices associated with agile development on the foundation provided by Kimball's approach to allow for the agile development of a DW system. Since Kimball's approach is the underlying approach of the ADW methodology, the ADW methodology does not attempt to modify or tailor Kimball's approach but rather introduces agile best practices upon the already laid concrete foundation. Therefore, it is necessary that one must be familiar with Kimball's approach if one wishes to implement the ADW methodology in DW development. Kimball's approach is a highly technical approach that aims to cover all the aspects that might be present in any DW project. It is recommended that the project team that wishes to implement the ADW methodology use Kimball's approach as the foundation and choose what components of Kimball's approach to include in its development process in regard to the unique environment of its DW project. In addition to using Kimball's approach as the foundation, the project team must adhere to the agile best practices introduced in the ADW methodology for each phase of development to fully benefit from the ADW methodology for DW development.

5.4.1 ADW (version 0) methodology lifecycle

In this section the lifecycle of the ADW (version 0) methodology for the development of a DW will be introduced and used to explain the flow, best practices and approaches associated with it. The new lifecycle can best be described by using a diagram, which is given in Figure 5.3.

The lifecycle consists of four major phases, which are planning, design, development and implementation. These major phases are each made up of a number of sub-phases that have been adopted from Kimball's approach. Kimball's approach consists of 13 sub-phases, which are identical to the ADW methodology. The only difference is that the 13 sub-phases have been categorized into one of the four major phases and the flow of the lifecycle has minor changes in regard to the original lifecycle of Kimball's approach. For each of the major phases, the best practices of XP, SCRUM or FDD (can be seen in the lifecycle diagram) will be adopted to assist in the successful completion of the phase.

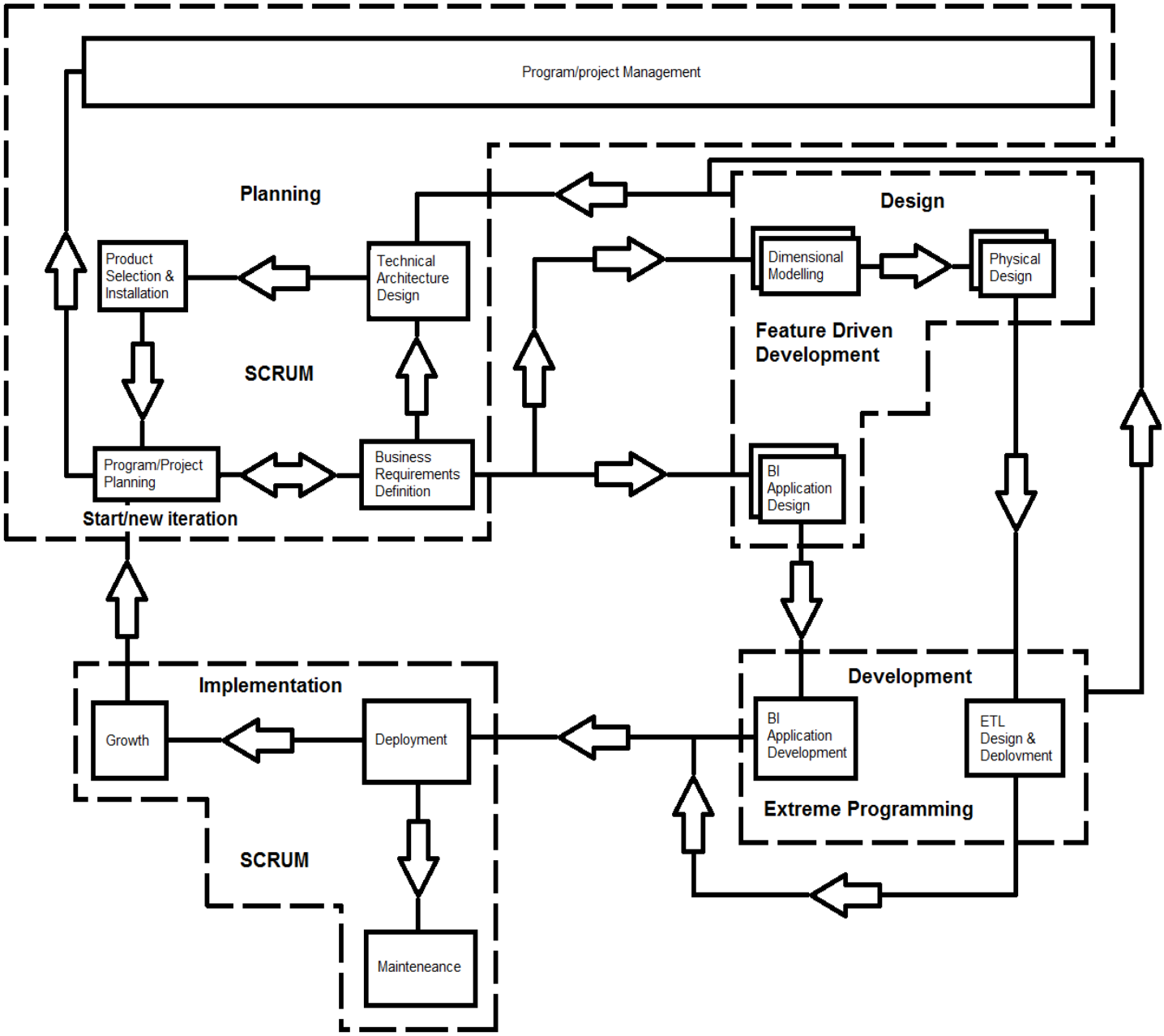


Figure 5.3 – ADW (version 0) methodology lifecycle diagram

5.4.1.1 Flow of the lifecycle

Taking a quick glance at the framework it might seem like a complex web of intertwined tasks, but it is quite simple and similar to the easy to understand lifecycle of Kimball's approach.

Similar to Kimball's approach, the ADW (version 0) methodology lifecycle starts with the project/program planning sub-phase, which leads to the business requirements definition sub-phase. These two phases are multi-directional as is the case with Kimball's approach because in order to ensure that the user requirement can be met, a data analysis must first be performed (task in the project/program planning sub-phase). During the gathering of requirements, one would typically switch between the program/project planning sub-phase and the business requirements definition sub-phase. Once the project team is satisfied with the project/planning and business requirements definition sub-phases, the lifecycle moves on

to the technical architecture design. Once the project team has completed the technical architecture design sub-phase the lifecycle flows into the product selection and installation sub-phase. Upon completion of the product selection and installation sub-phase the lifecycle goes back to the program/project planning phase where the planning of the DW system is completed with the additional information gathered from the technical architecture design and product selection and installation sub-phases. This loop is repeated until all the stakeholders are satisfied with the initial requirements, product selection, cost and time estimates of the project. Additionally, the program/project planning sub-phase flows into the program/project management sub-phase. The program/project management sub-phase is present during all stages of the lifecycle and enforces proper management of the project to ensure successful completion of the DW project

Upon approval from the stakeholders, and using the business requirements definitions as the driving force, the lifecycle moves on to the design phase. The business requirements definition sub-phase flows into the dimensional modelling and BI application sub-phases. The dimensional modelling and BI application sub-phases are the beginning of the data and BI application tracks respectively. Once the project team has completed the dimensional modelling sub-phase of the data track the lifecycle moves on to the physical design sub-phase.

The physical design and BI application design sub-phases both lead to the development phase. The physical design sub-phase of the data track flows into the ETL design and development sub-phase and the BI application design sub-phase flows into the BI application development sub-phase. The data track is completed once the ETL design and development sub-phase has been completed and the BI application track ends with the BI application development sub-phase. The data and BI application track may run parallel depending on what the project team prefers.

Once the data and BI application tracks have been completed the lifecycle flows into the implementation phase. The implementation phase begins with the deployment sub-phase that integrates, tests and deploys the data and BI application tracks. Once the deployment sub-phase is complete the lifecycle can take one of two directions. The first, is the successful completion of the DW project and leads to the maintenance sub-phase and the second is the growth sub-phase that triggers another iteration of development, which directs the flow of the lifecycle back to the program/project planning phase.

It is important to note that the design and development phases have a back loop that directs them to the technical architecture design sub-phase. This is due to the fact that during the design and development phases, new information may come to light that was not present in the initial technical architecture design. When the design or development phases trigger the

back loop to the technical architecture design sub-phase the new information must be taken into account, which may result in a re-design of the technical architecture, which can either cause the flow of the lifecycle to follow the initial flow from the technical architecture design sub-phase (flow into the product selection and installation sub-phase) or to just continue from where the flow of the lifecycle was when the back loop was triggered. The flow from a back loop is determined by the project team based on the new information that has come to light. If the new information has a minor impact on the technical architecture design, the project team can just update the technical architecture and continue with the development process before the back loop was triggered. If the new information leads to major changes to the technical architecture, new products might be needed which will affect the cost of the project and the flow of the lifecycle must be followed from the technical architecture design sub-phase.

5.4.1.2 Agile best practices

In this section the best practices that make up the ADW (version 0) methodology will be introduced and discussed. As mentioned earlier, the ADW (version 0) methodology uses Kimball's approach as the foundation and the project team is responsible for deciding to what degree they choose to use Kimball's approach for its unique development environment. This includes what documentation to produce, and which tools, techniques and approaches are appropriate for their project. Using Kimball's approach as the foundation, this section will introduce the agile best practices to assist and aid the project team to successfully complete DW system development. The agile best practices of the ADW methodology will be discussed according to the overall lifecycle, planning phase, design phase, development phase, and implementation phase associated with the ADW (version 0) methodology.

Lifecycle agile best practices

Even though either XP, SCRUM, or FDD best practices will be implemented in one of four major phases of DW development, there is a number of agile best practices that encompasses the entire lifecycle. This is due to the fact that XP, SCRUM and FDD all belong to the agile family, which involves an underlying philosophy that is true for each of them. Therefore, the ADW (version 0) methodology will adopt best practices from this agile philosophy that are true for the entire lifecycle regardless of whether XP, SCRUM or FDD is being used at a certain phase of the lifecycle.

In order to comply with the agile development environment, the ADW (version 0) methodology must adhere to the purposes and principles mentioned in the agile manifesto (section 1.2.2). Although Kimball's approach serves as the foundation, the ADW methodology uses the Agile Manifesto as the underlying philosophical view and if Kimball's approach suggests a large amount of documentation that contradicts the agile philosophy; the team must attempt to

minimize the documentation as much as possible in order to make the development process agile while still performing the necessary tasks for successful development of the DW system. An example of this would be that agile recommends minimum documentation and face-to-face communication, whereas Kimball's approach suggests that a communication plan be created to assist with communication. Applying the agile philosophy to Kimball's approach in this regard may involve that the creation of a communication plan be excluded from the development process and that face-to-face communication amongst stakeholders be promoted. Therefore, Kimball's approach serves as the foundation but is not set in stone and the agile philosophy must be used to determine what documentation, tools, techniques and approaches must be included for the DW project in regard to the unique development environment.

In addition, the ADW (version 0) methodology uses incremental and iterative development processes with the focus on a single business process per iteration (depending on what the team prefers) in order to promote simplicity, rapid development and adaptability. When in doubt about the philosophical view of the ADW (version 0) methodology, refer to the agile manifesto and choose the appropriate components suggested by Kimball's approach that do not contradict this philosophy and meets the needs of the unique development environment of the DW project.

Planning phase

During the planning phase, best practices from the SCRUM methodology will be used to successfully complete the planning phase of a DW project. Since SCRUM is the most suited for the planning phase as well as the implementation phase, the names mentioned by SCRUM to refer to IS development processes will be used across the entire lifecycle to reduce confusion. Therefore, a sprint will be used to refer to an iteration in the ADW (version 0) methodology. Different names that refer to the same roles (SCRUM owner – SCRUM, chief architect – FDD) will also be adopted from the SCRUM ASDM in order to reduce confusion and misunderstanding.

The planning phase will make use of the following SCRUM best practices to complete appropriate tasks and documentation suggested by Kimball's approach:

- **SCRUM roles (product owner, SCRUM master, team)** – In addition to the DW-specific roles mentioned by Kimball's approach, SCRUM-specific roles must be included when roles are being assigned in order to identify who will be responsible for SCRUM-related activities. This will usually just involve the re-naming of some of the

roles identified by Kimball's approach to adhere to the unique names associated with SCRUM.

- **Product backlog** – The product backlog will contain all the DW-specific documentation (refer to Kimball's approach) that is relevant for the unique development environment. Additionally, the product backlog must be handled as described by the SCRUM methodology (section 2.4.2). Any documentation produced during the other phases must be included in the product backlog.

- **Sprint backlog** – Using the product backlog, the sprint backlog is populated with the requirements, after they have been prioritized, to be completed in the next sprint.

- **Pre-sprint planning** – Before the start of each sprint, a pre-sprint planning meeting must be held to identify the tasks necessary, using Kimball's approach as a guideline to DW development.

- **Daily SCRUM meeting** – Although the SCRUM ASDM is only used for the planning and implementation phases, the sub-phase program/project management (sub-phase of planning) is present during the entire lifecycle and therefore a daily SCRUM meeting must be held throughout the entire development process.

This concludes the SCRUM best practices that are recommended during the planning phase of DW development using the ADW (version 0) methodology.

Design phase

During the design phase, best practices from the FDD methodology will be used to successfully complete the design phase of a DW project. The best practices adopted from FDD for the design phase of DW development are:

- **Design by feature** – The identified requirements are broken down into manageable features to perform the design of the DW system.

- **Inspection** – Perform inspection on the dimensional model, physical design and BI application design to ensure that features (requirements) are the main focus and that the designs will address the features that must be completed.

- **Feature teams** – Perform the design phase in small, dynamic teams that compliment and aid each other to perform the most optimal design of the DW system.

- **Simplicity** – Keep the design of the DW system as simple as possible to assist rather than complicate the development process. This also ensures that the final product is easy to use and understand.

- **Eliminate process pride** – Focus on completing the design of the DW system; do not let team members' pride stagnate the design of the DW just to prove their approach was possible.

- **Feature list** – Use the feature-list method introduced in FDD to completely map the entire DW system starting with a single business process to assist in the design of the DW system. The feature list will assist particularly with the dimensional modelling sub-phase of the design phase to map the entire DW system and will assist in identifying the grain of the fact table as well as the dimensions needed to meet the requirements of the user.

This concludes the suggested best practices of FDD to assist in the design phase of DW development using the ADW methodology.

Development phase

During the development phase, best practices from the XP methodology will be used to successfully complete the development phase of a DW project. The best practices adopted from XP for the development phase of DW development are:

- **Small releases** – The ADW (version 0) methodology implements iterative development and focuses on one business process at a time and this must stay true during the development phase. Perform the ETL process focusing on a single business process and expand the DW database as the iterations increase. Develop the BI application only meeting the needs of the current business process to avoid being overwhelmed by the complexity and size of the final system.

- **Metaphor** – Any documentation performed during the development phase must be included in the product backlog.

- **Testing** – Perform unit tests to ensure the ETL process produced desired data and handled all the necessary transformations to get the data as clean as possible. Unit testing must also be performed on the BI application to remove bugs and to ensure the application performs as expected. Acceptance tests must be performed on the ETL results and BI application to ensure the development phase meets the requirements and developed something of value to the users.

- **Simple design** – During the ETL process and BI application development, the project team must stay clear of complexity and keep it as simple as possible.
- **Refactoring** – Once the project team identifies anything during development that is of no value the project team must return to the design phase and re-evaluate to design a DW system that will be of value to the users.
- **Pair programming** – The ADW (version 0) methodology recommends pair programming during the ETL process and BI application development.
- **Collective ownership** – The ETL process and BI application development responsibility belong to the entire team and each member has access and ownership of the activities involved in the development phase.
- **Continuous integration** – The ETL process and BI application development must be integrated as frequently as possible. New transformations must be integrated and loaded into the database as frequently as possible. New components of the BI application must be added to the BI application as soon as they have been tested.
- **Forty-hour week** – The development phase of a DW project is very demanding and it is important that developers are fresh and rested to prevent careless mistakes. A project team that has a positive moral will lead to faster and increased productivity. Overtime is allowed but only for a short duration at a time.
- **On-site client** – During the development phase it is very helpful to have a business representative present at all times. This is particularly important during the ETL process to assist the project team to make sense of some of the issues encountered with the operational data. This will help the team to quickly define why certain data is captured in a certain manner and how to deal with the cleaning of these issues.
- **Coding standards**- During the development phase it is very important to use coding standards with the ETL process and BI application development. This ensures that the integration of the ETL process is easier to perform and brings clarity amongst developers. If coding standards have been incorporated in the ETL process the BI application development will be easier to perform.

This concludes the best practices of XP that must be implemented during the development phase of DW development using the ADW (version 0) methodology.

Implementation

During the implementation phase, best practices from the SCRUM methodology will be used to successfully complete the implementation phase of a DW project. The best practices adopted from SCRUM for the implementation phase of DW development are:

- **Product backlog** – The project team must refer to the product backlog to see that the training plan that has been put in place and to update the plan if necessary.
- **Sprint results** – The sprint results document will contain the components and issues discussed in the post-sprint meeting. The sprint results document must be added to the product backlog.
- **Post-sprint meeting** – After each sprint a post-sprint meeting must be held to evaluate the completed sprint and to ensure the sprint produced a system that is of value to the users. During this meeting it must be decided if a new sprint must be initiated for further growth of the DW system. This meeting must also identify what the training needs and support infrastructure of the users are. The maintenance needs of the users must initiate a new sprint that is only focuses on the maintenance and improvement of the current DW system.

This concludes the best practices of SCRUM that must be implemented during the development phase of DW development using the ADW (version 0) methodology.

In conclusion, it can be seen that the proposed framework does not attempt to introduce a new methodology for the development of a DW system, but introduces agile best practices from XP, SCRUM, and FDD that build upon the foundation provided by Kimball's approach. Kimball's approach provides support for the technical and unique aspects associated with DW development and incorporates best practices from agile to assist in the management, planning, design, development, and implementation tasks associated with Kimball's approach. The agile best practices introduced are not primarily focused on aspects associated with DW development but are best practices that aim to assist in IS development from a broad perspective. With Kimball's approach as the foundation for DW-specific tasks and the agile best practices that aims to provide an agile development environment, the ADW (version 0) methodology may prove to be an effective methodology for the agile development of a DW system.

In the next chapter the evaluation stage of the research will be performed in order to test version 0 of the ADW methodology. Based on the evaluation of version 0, version 1 (final) will be developed and evaluated.

Chapter 6: Evaluation

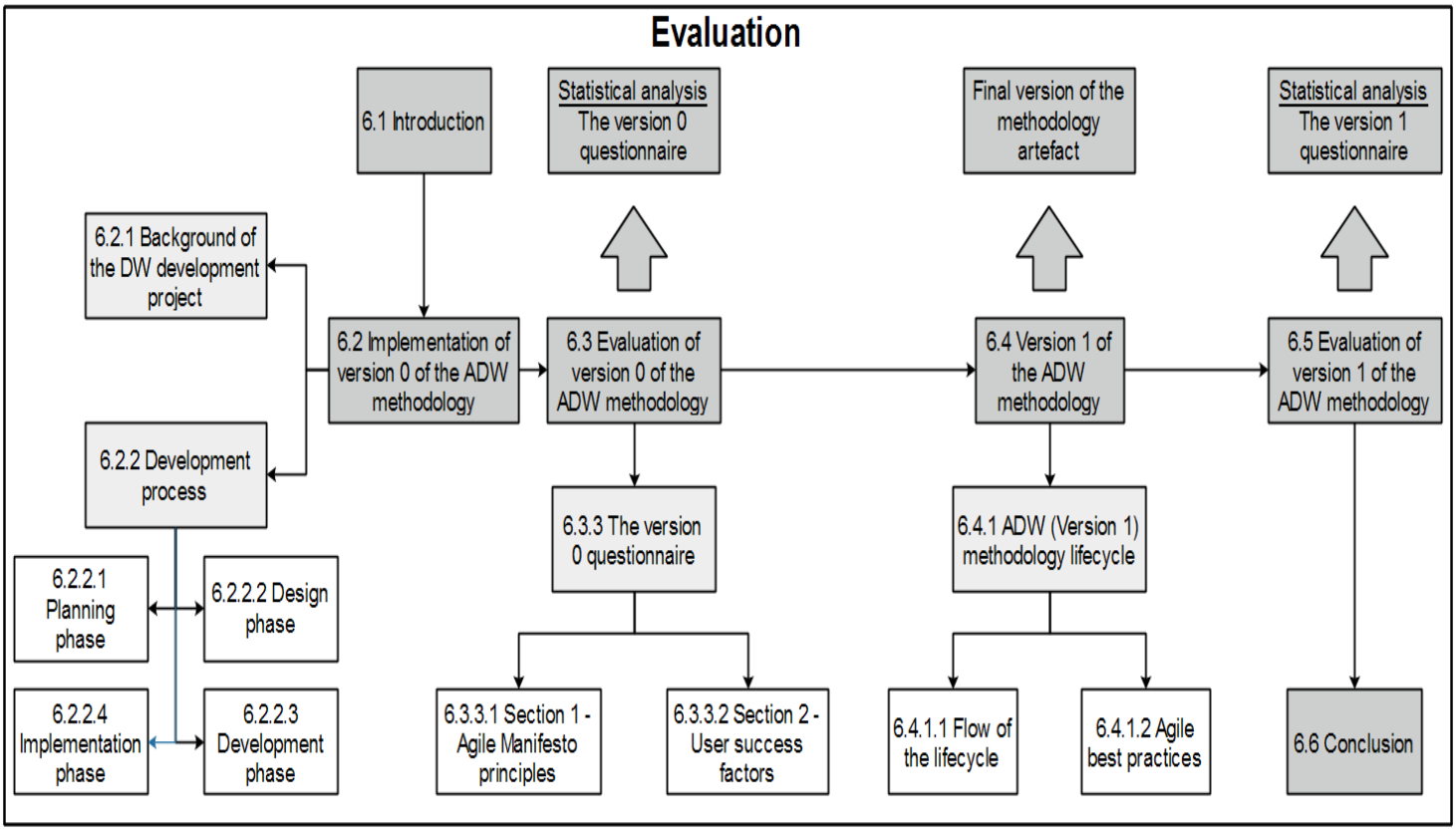


Figure 6.1 - Overview of Chapter 6

6.1 Introduction

In this chapter the evaluation stage of the research methodology is executed and aims to develop and verify the usefulness of version 1 (final) of the ADW methodology for the agile development of a DW system. In order to achieve this aim, the chapter starts off with the testing and evaluation of version 0 of the ADW methodology for the actual development of a DW system. The testing of version 0 is first discussed by explaining the development environment and process of the implementation of the ADW (version 0) methodology.

The discussion on the implementation of version 0 in the development of a DW system is followed by the evaluation of version 0 of the ADW methodology. The evaluation of version 0 of the ADW methodology is concerned with determining whether version 0 of the ADW methodology was successful in the agile development of a DW system. The evaluation of version 0 will be based on the implementation of the ADW (version 0) methodology and will be performed by developing and distributing the version 0 questionnaire to the relevant stakeholders involved in the development process.

Once the evaluation of version 0 has been completed, the chapter moves towards the development of version 1 of the ADW methodology that is based on the knowledge gained during the implementation and evaluation of version 0. Version 1 of the ADW methodology will be introduced and discussed in regard to the changes and the reasons behind the changes performed on version 0 in order to result in version 1 of the ADW methodology.

Once version 1 of the ADW methodology has been introduced and the reasons behind the transformation of version 0 into version 1 have been discussed, the evaluation of version 1 will be performed in order to verify the usefulness of the ADW (version 1) methodology. This evaluation will be conducted by developing the version 1 questionnaire that aims to determine whether the best practices introduced in each major phase and whether the ADW (version 1) methodology are considered useful for the agile development of a DW system from the viewpoint of a number of experienced developers. The statistical analysis of the data generated by the version 1 questionnaire will determine the usefulness of version 1 (final) of the ADW methodology which will in turn provide the answer to whether the ADW (version 1) methodology will provide support and guidance for the agile development of a DW system.

6.2 Implementation of version 0 of the ADW methodology

At its core, the primary aim of this study is to develop a methodology that recommends guidelines to assist and support a project team in the agile development of a DW system. Therefore, the aim of the ADW (version 0) methodology is to assist a project team in the agile development of a DW system. The best approach to test and evaluate whether the ADW methodology can achieve the desired aim is to use the methodology as a guideline in the actual development of a DW system in a real-world environment where change is bound to happen. Therefore, the ADW (version 0) methodology was tested and evaluated by implementing version 0 in the actual development a DW system in collaboration with an industry partner.

Since the study aims to develop a methodology that recommends guidelines to assist and support a project team in the agile development of a DW system, the study is concerned with the development process rather than the developed DW system. Therefore, the implementation of the ADW (version 0) methodology will be described by discussing the background of the DW development project and the development process in regard to the four major phases and their associated sub-phases of the ADW (version 0) methodology.

6.2.1 Background of the DW development project

In this section the background of the DW development project will be described by briefly discussing the industry partner and the stakeholders involved in the project.

Industry partner

The industry partner is a small company situated in the Pretoria region of South-Africa. Due to a NDA (non-disclosure agreement) contract with the industry partner, not much can be said about the industry partner or the developed DW system. The company is a relatively new company with a growing interest in the internal management of quality data and data warehousing. The company is currently busy with a data value chain project in order to increase the quality and value of the company's current data and database. The development of the DW system formed part of this project as a sub-project to provide quality data that can be used to perform business intelligence.

Stakeholders

The stakeholders involved in the project consisted of the researcher and a small number of employees involved in the data value chain project. The stakeholders involved in the DW development project were

- CEO
- IT department manager
- Data analyst
- Database manager
- Junior developer
- Researcher

Therefore the project consisted of six stakeholders including the researcher.

6.2.2 Development process

In this section the development process implementing the ADW (version 0) methodology will be described by discussing how each major phase and associated sub-phases were performed.

6.2.2.1 Planning phase

The planning phase will be discussed by explaining how each sub-phase in the planning phase was performed while pointing out the best practices recommended in the planning phase of the ADW (version 0) methodology.

Program/project planning

The program/project planning sub-phase started with data analysis in order to better understand the available data for development. The data analysis was performed by discussing the available data with the data analyst in an informal interview. In preparation to the interview, an in depth analysis of the available data was performed. During this sub-phase the project team was formed and each member was assigned a role according to the naming standards of SCRUM (**SCRUM roles**) and the project team with their roles were

- IT department manager (product owner)
- Researcher (SCRUM master)
- Data analyst (team)
- Database manager (team)
- Junior developer (team)

During this sub-phase, the scope of the project was agreed upon (after the business definition sub-phase, see section 5.4.3) and documented in the **product backlog**. Unfortunately, most of the details contained in the **product backlog** is confidential, but relevant information to the understanding of the development process (without revealing confidential information) will be mentioned. Due to the limited available data and time, the DW development project only completed one iteration of the ADW (version 0) methodology lifecycle. Therefore the developed DW system consists of only a single business dimensional model.

Once the initial product backlog document was created, the **sprint backlog document** was developed from the **product backlog** to identify the relevant requirements for the current sprint (in this case all the requirements were added since the scope consisted of only one iteration). Upon completion of the **sprint backlog**, a **pre-sprint meeting** was held and identified that each sub-phase of the ADW (version 0) methodology will be required for the current sprint with extra attention to the technical architecture design and product selection and installation.

Program/project management

The program/project management sub-phase was performed throughout the entire development process. Due to the size of the project and busy schedules of the industry partner employees, **daily SCRUM meetings** were held at the start of each week as opposed to the start of each day (recommended). The responsibility of the SCRUM meeting and the successful management of the project team was given to the product owner and the SCRUM master.

Business requirements definition

The business requirements definition sub-phase was performed after the data analysis performed in the program/project planning sub-phase and was performed with interviews and facilitated sessions. The researcher served as the facilitator for all the interviews and facilitated sessions. The initial requirements were gathered with a facilitated session with the CEO, data analyst, and database manager. The initial requirements definition was added to the **product backlog** and the development process moved forward with the technical architecture design sub-phase using the initial requirements as the **sprint backlog.** After two weeks into the development process, the IT department manager came back from a paid vacation and an interview with the IT department manager resulted in changes to the initial requirements and resulted in the final requirements definition for the current iteration. The **product and sprint backlog** were updated with the latest requirements. Three weeks into the development process small changes were made to two of the identified requirements and updated in the relevant documents.

Technical architecture design

Analysis of the current technical architecture revealed that one desktop computer be purchased to serve as the server of the DW system along with the network already in place.

Product selection and installation

One of the technical requirements for the project was that open source software be used for development since the industry partner does not have the necessary funding to purchase high end software for development. After extensive research on the available open source software necessary for the development of the DW system, the following software was selected and installed

- Linux Debian 8 "Jessie" - Operating system.
- MySQL - Platform for the DW database (database management software).
- Pentaho 5.3 Data Integration (Kettle) Community Edition - ETL tool.
- Pentaho 5.3 BI Server Community Edition - Platform for the BI application (OLAP software).

This concludes the planning phase of the development process of the DW development project, it must be mentioned that although not explicitly stated, the **product backlog** was continuously updated as new information relevant to the scope of the project was identified (i.e. purchase of the desktop computer and installed software). The implementation of the ADW (version 0) methodology in regard to the planning phase, revealed no difficulties concerning the sub-phases, flow of the planning phase, and the recommended best practices.

Therefore the implementation of the ADW (version 0) methodology in regard to the planning phase recommends that no changes be made to the planning phase for the development of version 1 of the ADW methodology.

6.2.2.2 Design phase

The design phase will be discussed in a similar manner to the discussion of the planning phase.

Dimensional modelling

The dimensional modelling sub-phase was performed by the researcher and the data analyst (**feature teams**). The dimensional model focused on a single business process (**design by feature**) and designed a single dimension business model (**simplicity**) by using the **feature list** best practice to break the selected business process into the necessary components needed for the dimensional model. Table 6.1 consists of the incomplete feature list to reveal how the feature list best practice was implemented to assist in the design of the dimensional model.

Feature list

Business Process	Grain (Fact table)	Dimension tables	Fields
Challenge responses	Responses per challenge (Reponses_Fact)	Response_Dim Challenge_Dim Contact_Dim Date_Dim	**Response_Dim** • Responses_Key (PK)(SK) • Response_Id (AK) • ... **Challenge_Dim** • Challenge_Key (PK)(SK) • ... **Contact_Dim** **Date_Dim**

Table 6.1 - Incomplete feature list used in the implementation of version 0

Upon completion of the feature list, the dimensional model was created and is presented in Figure 6.2. During the design of the dimensional model, the researcher believed that a platform dimension must be added to the dimensional model, but this prevented the completion of the design and was discarded (**eliminate process pride**). In the end, the develop DW system performed as desired without the addition of the platform dimension in the dimensional model.

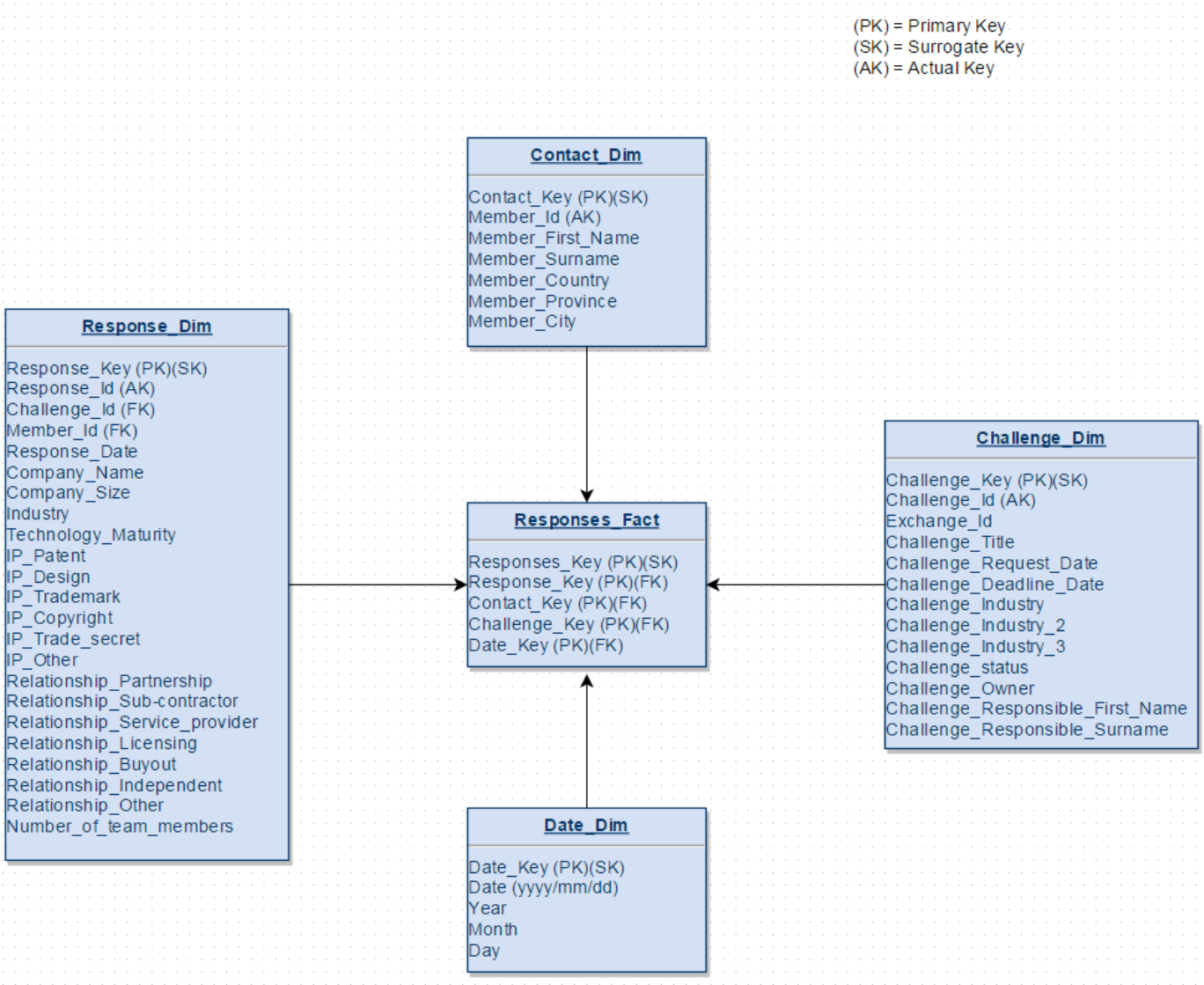


Figure 6.2 - Dimensional model created during the implementation of version 0

The created dimensional model was presented to the entire team for **inspection**.

Physical design

The physical design sub-phase consisted of setting up the DW database environment in MySQL by creating the necessary database with its associated tables (fact and dimensions) and fields. This included defining the metadata of each field (defining the size, type, null status, and key reference of the field). The DW system consisted of only one user (**simplicity**) protected by a password that was given to the relevant business users.

BI application design

The BI application design sub-phase was performed by learning to use the Pentaho 5.3 BI server software in order to select the most effective approach for the selected OLAP software in the development of the BI application. After a number of youtube tutorials, the project team decided to use Pentaho 5.3 BI server wizard to create the cube for the BI application and use

the Saiku analytics add-on of the BI software to perform analysis on the developed cube using the easy drag and drop interface of Saiku analytics.

This concludes the design phase of the development process of the DW development project. During the design phase, the **product backlog** was continuously updated as new information relevant to the current sprint was identified as well as the addition of the documents created during the design phase (i.e. feature lists and dimensional model). The implementation of the ADW (version 0) methodology in regard to the design phase, revealed no difficulties concerning the sub-phases and the recommended best practices. The implementation did however, reveal difficulty in the flow of the diagram when running the data and BI tracks in parallel. The project team concluded that performing the two tracks back to back starting with the data track will lead to a better understanding of what is required by the BI application and what data is available to perform the required analysis. In order to deal with the issue of the flow of the lifecycle between the data and BI tracks, the project team recommended merging the design and development phases into a single phase (discussed in section 6.4). Therefore, the implementation of the ADW (version 0) methodology in regard to the design phase suggests that no changes be made to the recommended best practices but suggests that the design and development phases be merged into one phase to lead to an improved version of the ADW methodology.

6.2.2.3 Development phase

The development phase will be discussed in a similar manner to the discussion of the design phase.

ETL, design and development

The ETL, design and development sub-phase used the design of the dimensional model (**small releases**) as guidance during the ETL process. The ETL design and development sub-phase was performed by the researcher and the junior developer and implemented **pair programming** to perform the ETL process. Each individual involved in the ETL process was allowed permission to edit the ETL program at any time (**collective ownership**). The ETL process used Pentaho 5.3 Data Integration to assist in the ETL process and focused on a single dimension at a time (**small releases**). The ETL for each dimension was performed by extracting the necessary data from the source system and identifying the necessary transformations needed to clean the data. The transformation process for each dimension was designed (**simple design**) by creating a data staging document (**coding standards**) (confidential) revealing the quality of the data in each field and defining the needed transformations. During the transformation process, new information was identified a number of times and required that changes be made to the data staging document (**refactoring**).

During the ETL process, **testing** was continuously performed to check whether the current transformation was performed as desired after which it was loaded into the DW database (**continuous integration**).

BI application development

The BI application was develop by using Pentaho 5.3 BI server software to assist in the development and hosting of the application. The BI application focused on the requirements identified for the current sprint (**small releases**) and used the DW database created during the ETL process. The OLAP cube was easily created using the Pentaho software cube creation wizard. Upon completion of the cube, the final interface used by the business users was created in the Saiku analytics add-on. The interface consisted of a simple drag and drop interface (**simple design**) to perform analysis on the data in the DW database. Each of the fields in the dimension were renamed (**coding standards**) to increase understandability amongst business users. A number of queries were performed to **test** the accuracy of the analysis.

This concludes the development phase of the development process of the DW development project. During the development phase, the **product backlog** was continuously updated (**metaphor**) with documents created during the development phase (i.e. data staging document). The development phase, like all the other phases were performed at the industry partner and allowed clients to be present at all times (**on-site client**) which provided great support during the entire development process, specifically during the ETL process that required explanations behind issues encountered with the data. During the entire development process, each project member worked an average of **forty-hours** per week with an exception of the last week where the researcher worked an additional eight hours of overtime in order to complete the project on-time. In regard to the best practices recommended in the development phase, all except for pair programming proved to be useful and effective. The ETL process was performed using an ETL tool and the entire ETL process produced only a single line of code where the rest of the process was performed with the simple built-in functions of the ETL tool that is code free. Therefore pair programming proved ineffective for DW development. The rest of the development phase had no difficulties during the implementation of version 0 of the ADW methodology except for the recommendation of the project team to merge the design and development phases into one (discussed in section 6.2.2.2). Therefore, the implementation of the ADW (version 0) methodology in regard to the development phase suggests that pair programming be removed from the recommended best practices and that the design and the development phases be merged into one major phase.

6.2.2.3 Implementation phase

At the start of the implementation phase, before any of the sub-phases, a **post-sprint meeting** was held and the project team agreed that the completed sprint was successful and resulted in a DW system that will provide value to the business users. The **post-sprint meeting** also identified factors that relate to training, maintenance and growth which will be discussed in the deployment, maintenance and growth sub-phases.

Deployment

During the deployment sub-phase, the developed DW system was delivered to the industry partner by creating a user manual and training each of the relevant business users in an interactive training environment. Below are a number of screenshots of the BI application of the developed DW system.

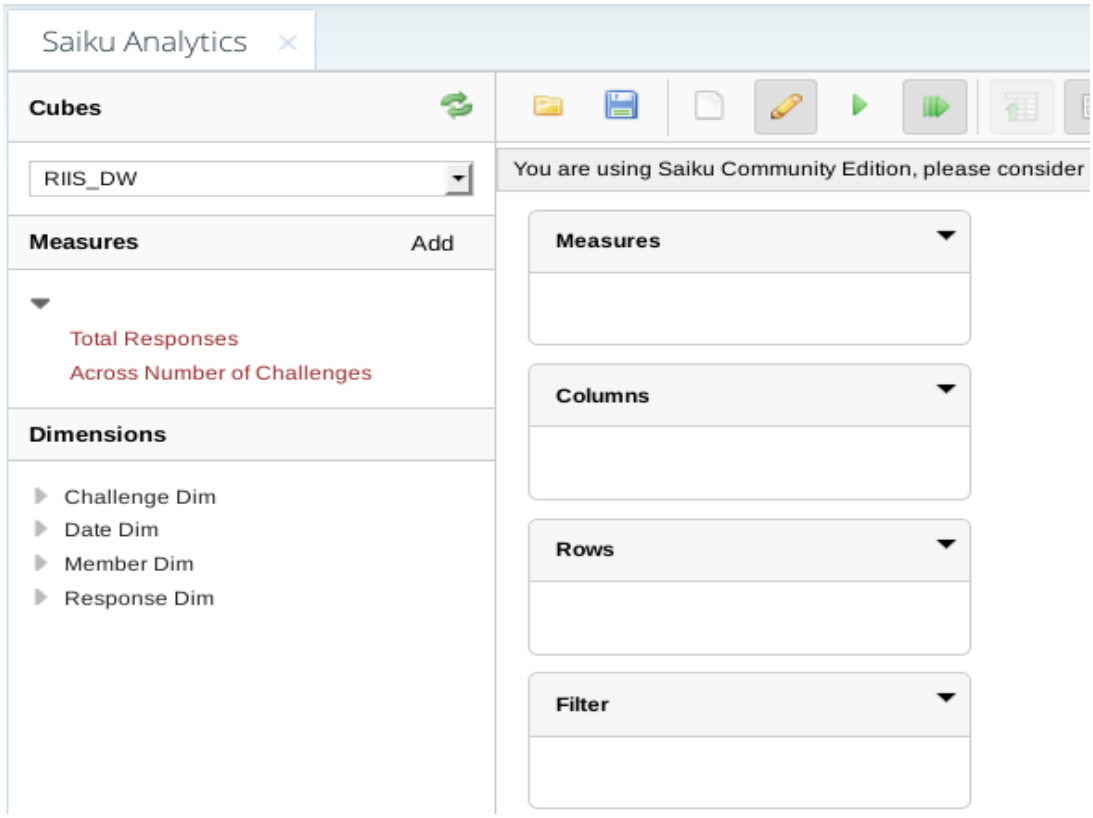
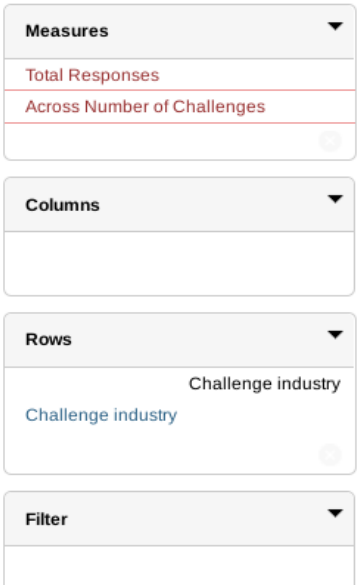


Figure 6.3 - BI application cube



Challenge industry	Total Responses	Across Number of Challenges
Agriculture, forestry and fishing	48	2
Chemicals and materials	3	1
Construction	4	1
Electronics, semiconductors, sensors	7	1
Environment, conservation	33	1
Financial and insurance	82	2
Information technology, communications	106	5
Manufacturing	5	2
Mining and quarrying	9	1
Other	10	1
Pharmaceutical	17	1
Water supply, sewerage, waste management	29	1

Figure 6.4 - Simple drag and drop query

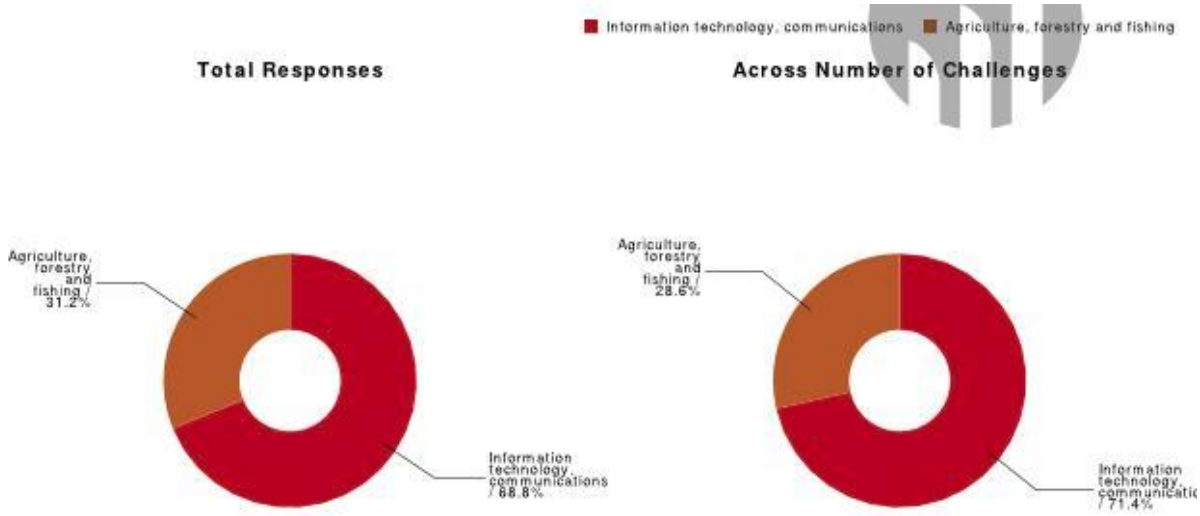


Figure 6.5 - Pie chart of the query presented in Figure 6.4

Maintenance

A maintenance manual was created and given to the database manager along with an interactive training session on how to maintain and manage the developed DW system.

Growth

The **post-sprint meeting** revealed that expansion of the developed DW system is feasible and highly recommended but, not pursued at the moment due to limit resources.

This concludes the implementation phase of the development process of the DW development project. Upon completion of the implementation phase, the **sprint results** document was created and consisted of the components (user manual and maintenance manual) and issues (limited resources for growth) discussed during the **post-sprint meeting** and was added to the **product backlog**. The implementation of the ADW (version 0) methodology in regard to the implementation phase, revealed no difficulties concerning the sub-phases, flow of the implementation phase, and the recommended best practices. Therefore the use of the ADW (version 0) methodology in regard to the implementation phase recommends that no changes be made to the implementation phase for the development of version 1 of the ADW methodology.

In conclusion to the implementation of version 0 of the ADW methodology, the development team recommends that the following be considered for the development of version 1 (final) in order to strengthen the effectiveness of the ADW methodology for the agile development of a DW system:

- Keep the planning and implementation phases as is.
- Increase the effectives of the BI track by first completing the data track. In order to achieve this without causing difficulty in the flow of the lifecycle, the project team recommended merging the design and development phases into one phase.
- Remove the pair programming best practice.

In the next section the evaluation of version 0 of the ADW methodology will be discussed.

6.3 Evaluation of version 0 of the ADW methodology

This section is concerned with the evaluation of version 0 which revolves around the aim and purpose of the ADW (version 0) methodology which is to provide a project team with guidance and support in the successful and agile development of a DW system. The evaluation of version 0 of the ADW methodology will be focused on determining whether the ADW (version 0) methodology achieved the aim and purpose it set out to achieve. Therefore, the version 0 questionnaire was developed and the objective of the questionnaire was to determine whether the ADW methodology assisted and provided support in the agile development of a DW system and whether the developed DW system is viewed as a success. The version 0 questionnaire is divided into two sections: the first section determines whether the ADW methodology laid the foundation for an agile development environment based on the principles of the Agile Manifesto (section 1.2.2), the second section determines whether the developed DW system is considered as a success based on a number of user success factors (identified during the implementation of version 0) (will be discussed in section 6.2.3.2).

The version 0 questionnaire is self-administered and the questions are multiple-choice questions presented as statements that require the respondents to provide an agree-disagree answer based on the scoring system presented in Table 6.2.

Score	Description
1	Completely disagree
2	Disagree
3	Neutral
4	Agree
5	Completely agree

Table 6.2 – Scoring system

The version 0 questionnaire (see Appendix A) was distributed to five relevant stakeholders and business users involved in the developed DW system (section 6.2):

- CEO (business user-male)
- IT department manager (team member and business user-female)
- Data analyst (team member and business user-male)
- Database manager (team member and business user-male)
- Junior developer (team member-male)

The version 0 questionnaire generated the data shown in Table 6.3.

Evaluation of version 0 of the ADW methodology - Data generated

Evaluation criteria	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5
Section 1					
Principle 1 - Early delivery	5	5	4	5	5
Principle 2 - Welcome change	5	5	5	5	5
Principle 3 - Deliver frequently	5	5	4	5	5

Evaluation criteria	Respondent 1	Respondent 2	Respondent 3	Respondent 4	Respondent 5

Principle 4 - Collaboration	5	5	5	5	5
Principle 5 - Motivated individuals	4	4	4	4	4
Principle 6 - Face to face communication	4	5	4	4	4
Principle 7 - Working software = success	4	5	4	4	4
Principle 8 - sustainable development	2	3	3	3	3
Principle 9 - Technical excellence	4	4	3	4	4
Principle 10 - Simplicity	5	4	5	5	4
Principle 11 - Self-organizing	5	5	5	5	5
Principle 12 - Reflect and adjust	4	4	3	4	4
Section 2					
User success factor 1 - Met requirements	4	4	4	4	4
User success factor 2 - Acceptable time frame	5	5	5	5	5

Evaluation criteria	**Respondent 1**	**Respondent 2**	**Respondent 3**	**Respondent 4**	**Respondent 5**

User success factor 3 - ease of use and training	5	5	5	5	5
User success factor 4 - Provide value	5	4	4	4	4

Table 6.3 - Data generated from the version 0 questionnaire

Similar to the DW and ASDM questionnaires, Krippendorff's alpha (discussed in section 4.6) reliability test was performed on the generated data in order to determine whether the data can be trusted. In section 4.6 the acceptable range of reliability was defined as $0.800 \geq \alpha \geq 0.667$. The data generated by the version 0 questionnaire produced a reliability alpha of 0.728. Therefore, the data generated by the version 0 questionnaire can be viewed as reliable and trustworthy.

For the statistical analysis of the DW and ASDM questionnaires, the total scores and average scores were calculated in order to easily interpret the data and to determine the best-suited approaches between two or more choices for a particular criterion. The version 0 questionnaire, however, does not aim to determine which is the best choice between a number of choices, but aims to determine whether the ADW (version 0) methodology succeeded in the agile development of a DW system and whether the system is viewed as successful. Therefore, the statistical analysis of the version 0 questionnaire calculated the total number of responses for each available answer (Table 6.2) for every statement followed by the percentage of responses for each available answer (Table 6.2) for every statement. In addition, the statistical analysis also included the calculation of the average score for each statement (evaluation criteria). These calculations will reveal how the respondents viewed the development process of the DW system and will be used to interpret whether the ADW (version 0) methodology succeeded in achieving its aim and purpose. The statistical analysis of the version 0 questionnaire is presented in Table 6.4.

Evaluation of version 0 of the ADW methodology - Statistical analysis						
Evaluation criteria	**Completely disagree**	**Disagree**	**Neutral**	**Agree**	**Completely agree**	**Average score**

Section 1	**0** **0%**	**1** **1.67%**	**6** **10.00%**	**25** **41.67%**	**28** **46.66%**	**4.33**
Principle 1 - Early delivery	-	-	-	1 20%	4 80%	4.8
Principle 2 - Welcome change	-	-	-	-	5 100%	5
Principle 3 - Deliver frequently	-	-	-	1 20%	4 80%	4.8
Principle 4 - Collaboration	-	-	-	-	5 100%	5
Principle 5 - Motivated individuals	-	-	-	5 100%	-	4
Principle 6 - Face to face communication	-	-	-	4 80%	1 20%	4.2
Principle 7 - Working software = success	-	-	-	4 80%	1 20%	4.2
Principle 8 - Sustainable development	-	1 20%	4 80%	-	-	2.8
Principle 9 - Technical excellence	-	-	1 20%	4 80%	-	3.8
Principle 10 - Simplicity				2 40%	3 60%	4.6
Principle 11 - Self-organizing	-	-	-	-	5 100%	5
Principle 12 - Reflect and adjust	-	-	1 20%	4 80%		3.8

Evaluation criteria	**Completely disagree**	**Disagree**	**Neutral**	**Agree**	**Completely agree**	**Average score**
Section 2	**0** **0%**	**0** **0%**	**0** **0%**	**9** **45%**	**11** **55%**	**4.55**

User success factor 1 - Met requirements	-	-	-	5 100%	-	4
User success factor 2 - Acceptable time frame	-	-	-	-	5 100%	5
User success factor 3 - ease of use and training	-	-	-	-	5 100%	5
User success factor 4 - Provide value	-	-	-	4 80%	1 20%	4.2

Table 6.4 - Statistical analysis of the data generated by the version 0 questionnaire

Section 1 focused on determining whether the ADW (version 0) methodology succeeded in the agile development of a DW system. From Table 6.4, it can be seen that 88.33% of the responses are found in the agree (41.67%) and completely agree (46.66%) categories for section 1 and for each individual principle, the majority of responses are located in the agree and completely agree categories with an exception for only one principle (principle 8). Additionally, section 1 has an average score of 4.33 (maximum = 5) and each individual principle has a positive average score with the exception of principle 8 (discussed in section 6.2.3.1). Therefore, excluding principle 8, it can be concluded that the ADW (version 0) was successful in the agile development of a DW system.

Section 2 aimed to determine whether the developed DW system is viewed as a success based on four user success factors. From Table 6.4 it can be seen that 100% of the responses fall in the agree (45%) and completely agree (55%) categories for section 2 and for each individual user success factor, the responses either fall in the agree or completely agree categories. Additionally, the average score for section 2 is 4.55 (maximum = 5) and each individual user success factor has a positive average score. Therefore, the developed DW system, using the ADW (version 0) methodology, is viewed as successful and valuable by the relevant stakeholders.

Therefore, ADW (version 0) methodology succeeded in the aims and objectives it set out to achieve (successful and agile development of a DW system). In the next section, the results of the version 0 questionnaire will be discussed in order to provide a possible explanation that supports and validates the data generated and conclusions interpreted from the data of the

version 0 questionnaire based on the implementation of version 0 of the ADW methodology (section 6.2).

6.3.3 The version 0 questionnaire

In this section the questions asked by the version 0 questionnaire will be mentioned, followed by a brief discussion of how the DW development project accomplished the principle/success factor in question to reveal the possible reasons behind the respondents' answers.

6.3.3.1 Section 1 - Agile Manifesto principles

This section consists of twelve statements (questions) that are based on the principles of the Agile Manifesto (section 1.2.2). The twelve statements of the first section of the questionnaire will be provided exactly as they were presented to the respondents along with the average score of the principle (see Table 6.4) followed by a discussion of the possible reasons behind the respondents answers.

Principle 1 - The project placed the highest priority on satisfying the customer/client through early delivery of valuable software. **4.8/5**

The first iteration of the lifecycle was completed in six weeks and resulted in a single business dimension DW system that can be queried to generate BI for a particular business process.

Principle 2 - The project welcomed changing requirements, even late in the development process. **5/5**

The initial requirements were gathered using interviews and facilitated group sessions from the relevant stakeholders. During the development of the project, requirements were added and changed at two different points. At two weeks into the initial start of the project, some requirements were removed and new ones added and at three weeks into the project there were small changes made to two of the identified requirements. The project, using the ADW (version 0) methodology, gladly welcomed these changes and handled them gracefully.

Principle 3 - The project delivered working software from a couple of weeks to a couple of months. **4.8/5**

As previously mentioned, the project completed the first iteration of the ADW methodology lifecycle within six weeks. This resulted in a working DW system that consisted of a single business dimensional model that can be used to query and report on the data relevant to the specific business process that relate to the dimensional model. In addition, four weeks after

the initial start of the project, the first fragments of the ETL program was delivered that performed valuable transformations on the current data to result in higher quality data.

Principle 4 - The project allowed for business users to work together with the development team on a daily basis. **5/5**

The development team set up at the industry partner's offices and developed the DW system on-site. This allowed the development team and the business users to work together on a daily basis.

Principle 5 - The project was developed around motivated individuals, giving them the environment and support they needed and trusting them to get the job done. **4/5**

The project team as well as the involved stakeholders were excited and motivated about the DW project. During development, responsibilities were assigned to individuals along with the environment, support, and trust needed for them to complete the task(s). Each individual successfully completed his/her responsibilities.

Principle 6 - The project focused more on face-to-face communication as opposed to e-mails and other means of media communication. **4.2/5**

During the development of the DW system, face-to-face communication was the primary source of communication. E-mails were used in cases where face-to-face communication was not possible. In addition, e-mails were used to transfer digital documents used in the development process.

Principle 7 - The primary measure of success for the project was working software. **4.2/5**

The project team and the relevant stakeholders agreed that working software is the primary measure of success, followed by ease of use and maintainability. Upon completion of the project, the DW system was demonstrated, resulting in a working DW system, The business users found the system easy to use, and maintenance was easy to implement.

Principle 8 - The project promoted sustainable development and the project team maintained a constant pace. **2.8/5**

During the planning phase of the DW development project, the project team agreed to change the daily SCRUM meetings from the start of every day to the start of every week due to the demanding work schedule of the industry partner employees/project team. This caused the reasons behind implementing the daily SCRUM meeting to lose its intended impact (ensure a constant pace in order to stay on track with the planned timeline) and caused the project team

to function under an unhealthy pace (high work output at the start of the week that slowly withers away towards the ends of the week).

The data generated by the version 0 questionnaire revealed a low score for this particular principle and countermeasures must be taken into account for the development of version 1 of the ADW methodology to prevent this from reoccurring.

Principle 9 - The project focused attention on technical excellence and good design. **3.8/5**

The business partners requested that the development team create a data-staging document that reveals the issues encountered during the ETL process. This document revealed the data integrity, redundancies, and gaps in the data so that the industry partners could focus on enforcing better data-capturing methods. The industry partners requested that certain technical requirements be met; some of these requirements were that open source software (Linux, etc.) be the only software used by the development team and the DW system.

During the design phase, an in-depth analysis of the available data was performed followed by a very detailed dimensional model. The design of the dimensional model was close to perfect and required that only one small component be re-designed for the final product of the DW system.

Principle 10 - The project implemented simplicity. **4.6/5**

The project focused on a single business process per iteration, which allowed for small bite-sized chunks at a time. During the ETL process, only one dimension was worked on at a time. In addition, the DW system focused on working software rather than appealing interfaces that are difficult to use. The project team constantly reminded itself to keep things simple and to focus on what is important for the successful development of the DW system.

Principle 11 - The project allowed for self-organizing teams. **5/5**

As previously mentioned, responsibilities were assigned to individuals, which in turn led to individuals and teams being self-organizing. Teams/individuals responsible for certain tasks were trusted to deal with any issues and support and guidance was provided if requested.

Principle 12 - The project allowed the project team to reflect on how to become more effective at regular intervals. **3.8/5**

Every Monday, the project team met and discussed what was accomplished in the previous week (weekly SCRUM meetings), what issues were encountered, how the issues were addressed, and how the process could be improved for future iterations/projects.

6.3.3.2 Section 2 - User success factors

Section 2 of the questionnaire aimed at determining whether the business users viewed the DW system as a success based on four user success factors identified during the implementation of version 0 of the ADW methodology. Similar to the discussion of section 1 of the questionnaire, the discussion of section 2 will present the statements as they were given to the respondents along with the average score of the of the user success factor (see Table 6.4) followed by a discussion of the possible reasons behind the respondents answers

User success factor 1 - Does the developed system meet the needs specified by the business users? **4/5**

The DW development project identified two categories of requirements: critical requirements and optional requirements. The critical requirements referred to the requirements that will lead to the successful development of the DW system. Optional requirements were additional features to be developed if there remained sufficient resources after the completion of the critical requirements. The development of the DW system successfully completed all the critical requirements but could not develop the optional requirements due to limited resources (time and team members).

User success factor 2 - Was the developed system completed in an acceptable time frame? **5/5**

The project only completed one iteration, but this iteration was completed within only six weeks. According to (Kimball *et al*., 2008:108), the project team must assign at least six months for the ETL process for each dimensional model. Therefore, the developed DW system finished the first iteration in a very short timeframe.

User success factor 3 - Is the system easy to use and was the necessary training provided? **5/5**

The developed DW system uses OLAP to query and analyse the data and the OLAP interface focuses on simplicity, functionality, and ease of use with a simple drag and drop interface. After the deployment of the system, face-to-face training with the relevant business users was conducted in order to ensure the business users understand and manage to use the system to its full potential. In addition, a user manual was developed to assist business users in learning to use the system.

User success factor 4 - Does the developed system provide value to the business? **4.2/5**

The DW system provides value to the business by allowing the relevant users to query and analyse data of the applicable business process. The analysed data and reports can be used by the business to make informed decisions to further improve business operations. In addition, the DW system revealed a large number of issues and gaps in the current data-capturing methods that the business now aims to address in order to generate higher quality data.

This concludes the discussion of the version 0 questionnaire and it can be seen from the discussion that the data generated as well as the conclusions are based on supporting evidence that reveals how the DW project implemented the principles of the Agile Manifesto under the guidance of the ADW (version 0) methodology. With the recommendations identified in section 6.2 for the development of version 1 and the countermeasure to address the low score of principle 8 discovered from the evaluation of version 0, version 1 will be developed to evolve version 0 into a more effective and successful DW development approach for agile development. In the next section, version 1, which is also the final version of the ADW methodology will be introduced and explained.

6.4 Version 1 of the ADW methodology

In this section version 0 of the ADW methodology will be improved in order to develop version 1, which is also the final version of the methodology. The improvements on version 0 will be performed based on the knowledge gained during the implementation of version 0 in the actual development of a DW system (section 6.2). The researcher played a major role in the development team and the knowledge gained from the implementation of version 0 is based on the constructivism and empirical data obtained by the researcher. Constructivism is an epistemology (theory of knowledge) that states that knowledge and truth are constructed by people through actively reflecting on their experiences (Grey, 2011:156). Therefore, version 1 is based on the knowledge gained by the researcher through the experience of using version 0 in the actual development of a DW system and the identified gaps in version 0 (principle 8) revealed by the evaluation of version 0. Version 1 of the ADW methodology will only be discussed by introducing and explaining the changes made to version 0. Therefore, in order to fully understand the ADW (version 1) methodology, it is recommended to first read section 5.4 (version 0) in order to understand the predecessor of version 1 and to understand how version 0 evolved into the final version.

6.4.1 ADW (version 1) methodology lifecycle

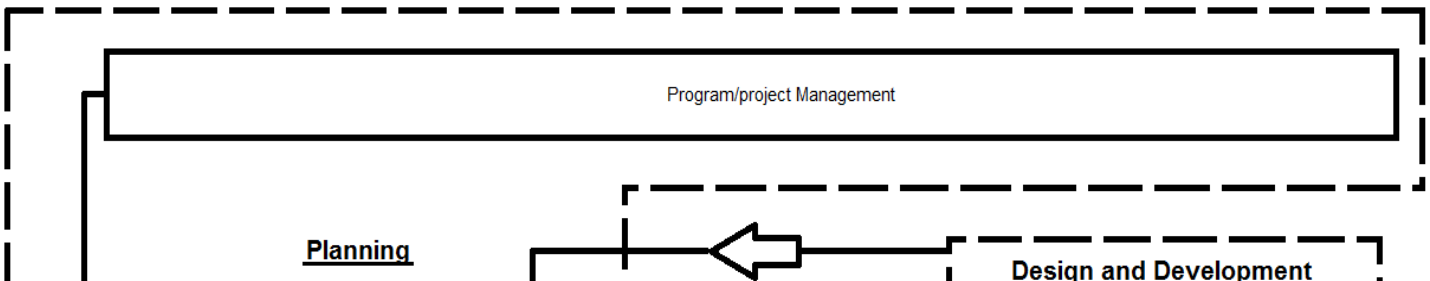


Figure 6.6 - ADW (version 1) methodology lifecycle diagram

Version 1 of the ADW methodology differs slightly from version 0, but the most prominent difference between the two versions lie in the lifecycle. Version 1 of the methodology improved upon version 0 in regard to the lifecycle by merging the design and development phases into one phase. Therefore version 1 consists of three major phases which are planning, design and development, and implementation. The reasons behind this change will be discussed in the next section. The sub-phases of the methodology remain the same with the exception of the design and development sub-phases coming together into one major phase. Other than that there are no changes to the sub-phases.

6.4.1.1 Flow of the lifecycle

The difference between version 1 and 0 of the lifecycle diagram can be seen with the merger of the design and development phases into one phase, but this merger also affected the flow of the lifecycle. To be more precise, the flow of the diagram is the primary reason behind the merger. During the development of the DW system using version 0 of the ADW methodology, the knowledge gained (section 6.2.2.2) from experience revealed that it makes more sense to first finish the data track before starting with the BI track. This allows for a clearer understanding of what is expected by the BI application and what data is available to accomplish the desired result. This is because the data track, at this point, has been completed, which means the ETL process for the current dimensional model has been performed, thus allowing for the BI track to better understand the available data and how to manipulate it to produce meaningful analysis. Therefore, it is recommended that the two (data and BI) tracks be completed back to back with the data track being completed first. This change in the flow of the lifecycle does, however, create some difficulty in regard to version 0. In version 0, the data track is completed in the development phase and the BI track is started in the design phase and implementing the above-mentioned flow causes the lifecycle to flow from design to development and then back to design followed by development before finally flowing into the implementation phase. This back and forth jump from design to development causes difficulty with the unique best practices introduced in both phases and therefore inspired the merging of the two major phases into one phase. In addition, the development process revealed that both the design and development phases and sub-phases associated with them can benefit from the best practices of both XP and FDD. Therefore, the merger of the two phases is supported in order to provide a more meaningful flow between the data and BI tracks as well the idea that both these phases can benefit from the best practices associated with XP and FDD.

Apart from the changes in the flow between the data and BI tracks, the flow of the lifecycle for version 1 is similar to that of version 0.

6.4.1.2 Agile best practices

The agile best practices for version 1 are close to identical to that of version 0 with very few changes. The lifecycle agile best practices adopted from the Agile Manifesto for version 0 remain unchanged in version 1 and were discussed in section 5.4.1.2.

In regard to the gap (principle 8 - sustainable development) revealed by the evaluation of version 0 of the ADW methodology, it is believed that the daily SCRUM meeting best practice introduced in the planning phase will effectively address the identified gap if implemented effectively. Therefore, to remind the project team that the daily SCRUM meeting must be

present throughout the entire lifecycle along with the program/project management sub-phase, the best practice will be mentioned in each major phase and not just in the planning phase. In addition, version 1 of the ADW methodology highly recommends that the daily SCRUM meeting best practice be performed on a daily bases to ensure sustainable development.

The agile best practices adopted from either XP, SCRUM, or FDD for each of the major phases will be mentioned below (discussed in section 5.4.1.2) followed by a short discussion of the changes made and why these changes were made.

Planning phase

The implementation of version 0 during the planning phase produced no difficulty and resulted in providing support and assistance to the development team in the successful completion of the planning phase in an agile environment. Therefore, the best practices for the planning phase of version 1 will remain the same as for version 0 with the reminder to perform the daily SCRUM meetings on a daily bases and these best practices are (section 5.4.1.2)

- **SCRUM roles**
- **Product backlog**
- **Sprint backlog**
- **Daily SCRUM meeting (execute on a daily bases)**
- **Pre-print planning**

This concludes the recommended best practices adopted from the SCRUM methodology for version 1 of the ADW methodology.

Design and development phase

In version 0 of the methodology the design phase adopted best practices from FDD and the development phase from XP. In the combined phase (design and development), best practices from both XP and FDD will be recommended for the successful completion of the design and development phase for the agile development of a DW system. The recommended best practices will simply be the merger of the best practices of these two phases from version 0 with the exclusion of the pair programming best practice.

During the development of the DW system, it was identified that pair programming is not a suitable fit for the development of a DW system and the reasoning behind this is based on the fact that pair programming shines in a strenuous programming environment by increasing the accuracy, effectiveness, and quality of the code. The development process of the DW system revealed that with the aid of ETL tools and the ease of use and effectiveness of these tools;

the ETL process can be completed without constructing a single line of code and therefore, refutes the idea of implementing pair programming to amplify the effectiveness and accuracy of the ETL process during development. Therefore, the pair programming best practice resulted in the ineffective use of resources (developers). In addition to the exclusion of the pair programming best practice, simplicity (FDD) and simple design (XP) has been combined into one best practice as both these best practices aim to achieve the same thing (simplicity). Therefore, the best practices for the design and development phase of version 1 with the addition and reminder of the importance of the daily SCRUM meeting best practice introduced in the planning phase of the ADW methodology are

- **Daily SCRUM meeting (execute on a daily bases)**
- **Design by feature (FDD)**
- **Inspection (FDD)**
- **Feature teams (FDD)**
- **Simplicity/simple design (FDD/XP)**
- **Eliminate process pride (FDD)**
- **Feature list (FDD)**
- **Small releases (XP)**
- **Metaphor (XP)**
- **Testing (XP)**
- **Refactoring (XP)**
- **Collective ownership (XP)**
- **Continuous integration (XP)**
- **Forty-hour week (XP)**
- **On-site client (XP)**
- **Coding standards (XP)**

Each of the above-mentioned best practices was first introduced in section 5.4.1.2 (version 0) where a more detailed description of each of the best practices is given. This concludes the recommended best practices for the design and development phase of version 1 of the ADW methodology.

Implementation phase

During the implementation of version 0, the best practices of the implementation phase proved to be very successful and aided the project team in the successful completion of the implementation phase. Therefore, the best practices recommended for the implementation phase will remain the same for version 1 with the addition and reminder of the importance of

the daily SCRUM meeting best practice introduced in the planning phase and these best practices are

- **Daily SCRUM meeting (execute on a daily bases)**
- **Product backlog**
- **Sprint results**
- **Post-sprint meeting**

These best practices, as for version 0, are adopted from the SCRUM methodology and are explained in more detail in section 5.4.1.2.

This concludes the introduction and discussion of version 1 of the ADW methodology and it must be noted that the changes imposed on version 0 in order to arrive at version 1 are a very small number of changes. Regardless of the number of changes involved during the transformation of version 0 into version 1, it is strongly believed (from the knowledge gained during the development process) that these changes that led to version 1 will strengthen the impact and support of version 1 in comparison to version 0 for the agile development of a DW system. In the next section, the evaluation of version 1 of the ADW methodology will be discussed.

6.5 Evaluation of version 1 of the ADW methodology

During the evaluation of version 0, it was concluded that the ADW methodology will provide a project team with support and contribute to the successful agile development of a DW system. Therefore, the evaluation of version 1 is not aimed at determining whether the ADW methodology can be used for the successful agile development of a DW system, but rather the evaluation is aimed at verifying the usefulness of version 1 of the ADW methodology. This evaluation is based on the opinion of experienced developers in order to generate the necessary data to determine from an agree-disagree perspective how they view each of the best practices introduced in each major phase of the ADW methodology as well as the methodology as a whole.

In order to perform the evaluation spoken of in the previous paragraph and to generate the desired data, the version 1 questionnaire was developed and distributed to a target group that consisted of 24 developers with experience in DW development using Kimball's approach as well as knowledge and/or experience with XP, SCRUM, and FDD. The target group consisted of post-graduate students (Honours and Masters) in the field of IS development (also referred to as information technology) with the majority of the respondents being male.

The version 1 questionnaire aims at determining the opinions of the target group in regard to each of the best practices introduced in each major phase of development as well as the ADW (version 1) methodology as a whole. The version 1 questionnaire performs this by introducing and explaining the ADW (version 1) methodology in regard to the lifecycle and each major phase of development with its associated best practices. After the introduction and in-depth explanation/discussion of the ADW (version 1) methodology, the target group was asked to provide an opinion on the usefulness of each of the best practices for the agile development of a DW system. In addition, the version 1 questionnaire finishes off with a question that requests the target group to provide its opinion on the usefulness of the ADW (version 1) methodology as whole to provide support and guidance for the agile development of a DW system. Therefore, the version 1 questionnaire produces data that will be analysed to determine how the target group views the usefulness of the best practices introduced in each major phase of development as well as how useful it views the methodology as a whole (lifecycle phases and sub-phases, lifecycle flow, and the integration of the major phases and their associated best practices).

The version 1 questionnaire is self-administered and the questions are multiple-choice questions presented as statements that require the respondents to provide an agree-disagree answer based on the scoring system presented in Table 6.2. The version 1 questionnaire (see Appendix A) was distributed to the above mentioned target group and generated the data shown in Table 6.5.

Evaluation of version 1 of the ADW methodology - Data generated

Evaluation criteria	Respondent 1-6	Respondent 7-12	Respondent 13-18	Respondent 19-24
Planning phase				
SCRUM roles	3;4;4;4;4;4	4;4;4;4;4;4	4;4;4;3;4;4	4;4;4;4;4;4
Product backlog	4;4;3;4;4;4	4;3;3;4;4;4	4;4;4;4;4;4	4;4;3;4;4;4
Sprint backlog	4;4;4;4;5;4	4;4;4;5;4;5	5;4;4;4;4;4	4;4;4;4;5;4
Daily SCRUM meeting	4;5;4;5;5;5	5;4;4;5;5;5	5;5;5;4;5;5	5;4;5;5;4;5
Pre-sprint planning	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5
Design and development phase				

Design by feature	4;4;4;4;4;4	4;4;4;4;4;3	4;4;4;4;4;4	4;3;4;4;4;4
Inspection	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4
Feature teams	3;4;4;4;4;3	4;4;4;3;4;4	4;4;4;4;4;4	4;4;4;4;4;3
Simplicity/simple design	4;3;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4
Eliminate process pride	4;4;4;3;4;4	3;3;4;4;4;4	4;4;4;4;3;4	4;3;4;4;3;4
Feature list	4;4;4;4;4;4	4;4;4;4;4;4	3;4;3;4;4;4	4;4;4;4;4;4
Small releases	5;5;5;5;5;5	5;5;5;4;5;5	4;5;5;5;5;5	5;5;5;5;5;5
Metaphor	4;4;4;4;4;3	4;4;3;4;4;4	3;3;4;4;4;4	4;4;3;4;4;4
Testing	5;4;4;4;5;4	4;4;4;4;4;4	4;4;4;5;4;5	4;4;4;4;4;4
Refactoring	4;4;4;5;4;4	4;4;4;4;4;4	4;4;4;4;5;5	4;4;4;4;4;4
Collective ownership	4;5;5;4;4;4	4;5;4;4;4;4	4;5;5;5;4;4	4;4;4;4;4;4
Continuous integration	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5
Forty-hour week	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4
On-site client	3;4;4;3;3;4	3;3;3;3;3;3	4;3;3;3;3;4	3;3;4;4;3;3
Coding standards	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5	5;5;5;5;5;5
Implementation phase				
Product backlog	4;4;4;4;4;4	4;4;3;3;4;4	3;4;4;4;4;4	4;4;4;4;3;4
Sprint results	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4	4;4;4;4;4;4
Post-sprint meeting	4;5;4;4;4;4	4;4;4;4;4;4	4;4;5;4;4;4	4;4;4;4;4;4
ADW methodology as a whole	**4;4;5;4;4;5**	**5;5;5;5;4;5**	**4;5;4;5;4;4**	**4;4;4;5;4;5**

Table 6.5 - Data generated from the version 1 questionnaire

Just as with all the questionnaires before the version 1 questionnaire, Krippendorff's alpha (discussed in section 4.6) reliability test was performed on the data in order to determine whether the data can be trusted. In section 4.6 the acceptable range of reliability was defined as $0.800 \geq \alpha \geq 0.667$. The data generated by the version 1 questionnaire produced a reliability alpha of 0.668. Although on the edge of the defined acceptable range, the data still falls between the acceptable ranges and therefore the data generated by the version 1 questionnaire can be viewed as reliable and trustworthy. Similar to the version 0 questionnaire, the version 1 questionnaire does not attempt to determine the best option between a number of options, but aims to determine the usefulness of each agile best practice and the ADW (version 1) methodology as a whole. Therefore, the statistical analysis for the version 1 questionnaire will be similar to that of the version 0 questionnaire (section 6.3) and these calculations will reveal how the target group (24 experienced developers) viewed the best practices of each major development phase and the ADW (version 1) methodology as a whole. The statistical analysis of the version 1 questionnaire is presented in Table 6.6.

Evaluation of version 1 of the ADW methodology - Statistical analysis

Evaluation criteria	Completely disagree	Disagree	Neutral	Agree	Completely agree	Average score
Planning phase	**-**	**-**	**7 5.84%**	**67 55.83%**	**46 38.33%**	**4.33**
SCRUM roles	-	-	2 8.33%	22 91.67%	-	3.92
Product backlog	-	-	4 16.67%	20 83.33%	-	3.83
Sprint backlog	-	-		19 79.17%	5 20.83%	4.21
Daily SCRUM meeting	-	-	-	7 29.17%	17 70.83%	4.71
Pre-sprint planning	-	-	-	-	24 100%	5
Design and development phase	**-**	**-**	**38 10.56%**	**239 66.39%**	**83 23.05%**	**4.13**
Design by feature	-	-	3 12.50%	21 87.50%	-	3.88
Inspection	-	-	-	24 100%	-	4
Feature teams	-	-	4 16.67%	20 83.33%	-	3.83

Evaluation criteria	Completely disagree	Disagree	Neutral	Agree	Completely agree	Average score
Simplicity/simple design	-	-	1 4.17%	23 95.83%	-	3.96
Eliminate process pride	-	-	6 25%	18 75%	-	3.75
Feature list	-	-	2 8.33%	22 91.67%	-	3.92
Small releases	-	-	-	2 8.33%	22 91.67%	4.92
Metaphor	-	-	5 20.83%	19 79.17%	-	3.79

Testing	-	-	-	20 83.33%	4 16.67%	4.17
Refactoring	-	-	-	21 87.50%	3 12.50%	4.13
Collective ownership	-	-	17 70.83%	7 29.17%	-	3.29
Continuous integration	-	-	-	24 100%	-	4
Forty-hour week	-	-	-	-	24 100%	5
On-site client	-	-		18 75%	6 25%	4.25
Coding standards	-	-	-	-	24 100%	5
Implementation phase	**-**	**-**	**4 5.55%**	**66 91.67%**	**2 2.78%**	**3.97**
Product backlog	-	-	4 16.67%	20 83.33%	-	3.83
Sprint results	-	-	-	24 100%	-	4
Post-sprint meeting	-	-	-	22 91.67%	2 8.33%	4.08

Evaluation criteria	Completely disagree	Disagree	Neutral	Agree	Completely agree	Average score
ADW methodology as a whole	**-**	**-**	*****	**14 58.33%**	**10 41.67%**	**4.42**

Table 6.6 - Statistical analysis of the data generated by the version 1 questionnaire

The version 1 questionnaire aimed at determining whether the target group agree-disagree with the usefulness of the best practices introduced by each major phase of development of the ADW (version 1) methodology in order to verify the usefulness of these best practices. Therefore, if the target group viewed the best practices as useful, then the best practices of version 1 of the ADW methodology will be verified by the target group of the version 1 questionnaire. The same counts for whether the target group viewed the ADW (version 1) methodology as useful for the agile development of a DW system.

Taking a look at Table 6.6, it can be seen that 94.16% of the responses are found in the agree (55.83%) and the completely agree (38.33%) categories for the best practices of the planning phase and for each best practice of the planning phase, the majority of responses are found in the agree and completely agree categories. Additionally, the planning phase contains an average score of 4.33 (maximum = 5) and each best practice has a positive average score. Therefore, the best practices of the planning phase are viewed as useful for the successful completion of the planning phase in the agile development of a DW system.

For the best practices of the design and development phase, 89.44% of the responses are found in the agree (66.39%) and the completely agree (23.05%) categories and the majority of responses for each best practice is located in the agree and completely agree categories with an exception for the collective ownership best practice. Additionally, the design and development phase contains an average score of 4.13 (maximum = 5) and each best practice except for the collective ownership best practice has a positive average score. It is believed (the researchers opinion and not the actual opinion of the respondents) that the reason behind the undesired (leans toward the neutral category) scores of the collective ownership best practice is fuelled by the idea that developers do not want other developers to play around with their code/work. This is based on the fear that another developer may make things worse rather than improve upon the current code/work. For the implementation of the ADW (version 1) methodology, the use of the ADW methodology and its components is dependent on the culture of the unique development environment and the project team may therefore choose to remove the collective ownership best practice. The reasons why the use of the collective ownership is left to the project team rather than being removed from the ADW methodology is based on the belief that DW system development environments will exist that can benefit from this best practice. Therefore, the collective ownership best practice will remain a recommended best practice in the design and development phase of the ADW (version 1) methodology. Based on this belief, it can be concluded that the best practices of the design and development phase are viewed as useful for the successful completion of the design and development phase in the agile development of a DW system.

From Table 6.6 it can be seen that 94.45% of the responses are found in the agree (91.67%) and the completely agree (2.78%) categories for the best practices of the implementation phase and for each best practice of the implementation phase, the majority of responses are located in the agree and completely agree categories. Additionally, the implementation phase contains an average score of 3.97 (maximum = 5) and each best practice has a positive average score. Therefore, the best practices of the implementation phase are viewed as useful for the successful completion of the implementation phase in the agile development of a DW system.

In addition to the questions (statements) that aimed to determine the usefulness of the best practices of each major phase, the usefulness of the ADW (version 1) methodology as a whole was also determined. This is different from just simply calculating the total scores of all the best practices of the planning, design and development, and implementation phases because it is based on the ADW (version 1) methodology as a whole. This refers to the entirety of the ADW (version 1) methodology, which (apart from the best practices already evaluated) also included the phases and sub-phases associated with them, the flow of the phases and sub-phases and how they come together to form the lifecycle of the ADW (version 1) methodology, and philosophical view of the methodology (agile). From Table 6.5, the statistical analysis reveals that 100% of the responses are found in the agree (58.33%) and the completely agree (41.67%) categories for the evaluation of the entirety of the ADW (version 1) methodology and that there is an average score of 4.42. Therefore, the ADW (version 1) methodology is viewed as useful for the successful agile development of a DW system.

In conclusion, the best practices associated with the planning, design and development, and implementation phases as well as the ADW (version 1) methodology as a whole are viewed as useful for the successful and agile development of a DW system. Therefore, the data generated by the version 1 questionnaire verifies the usefulness of the ADW (version 1) methodology as a whole as well as each of the best practices introduced in all of the major phases of the ADW (version 1) methodology.

6.6 Conclusion

This chapter started off with the evaluation of version 0 of the ADW methodology by first discussing the implementation of version 0 in the actual development of a DW system in an uncontrolled development environment. The discussion on the implementation of the ADW (version 0) methodology consisted of explaining the background and development process of the DW development project. The discussion on the implementation of version 0 of the ADW methodology was followed by the evaluation of version 0 of the ADW methodology which aimed to determine whether version 0 succeeded in the successful agile development of a DW system. The evaluation of version 0 was performed by developing and distributing the version 0 questionnaire, which was focused on determining whether the ADW (version 0) methodology assisted and provided support in the agile development of a DW system and whether the developed DW system was viewed as a success. The evaluation of version 0 (using the version 0 questionnaire) revealed that the ADW (version 0) methodology succeeded, with the exception of principle 8, in the initial aim and objective it set out to achieve (successful agile development of a DW system) and the data that led to this conclusion was supported by discussing how the ADW (version 0) methodology was implemented during the

development of the DW system in regard to the questions asked by the version 0 questionnaire.

Although the evaluation of version 0 of the ADW methodology revealed that the implementation of the ADW (version 0) methodology provided support and assistance in the successful agile development of a DW system, the knowledge gained during development revealed that version 0 can be improved to create a more effective DW development approach. The introduction and explanation of the ADW (version 1) methodology followed the evaluation of version 0 and was performed by only discussing the changes and the reasons behind the changes that led to version 1 of the ADW methodology. The evolution of version 0 into version 1 consisted of a very small number of changes and the most influential difference between version 1 and version 0 of the ADW methodology was the combination of the design and development phases into one phase. Regardless of the number of changes made on version 0, version 0 blossomed into version 1 (final) of the ADW methodology and it is believed that version 1 will strengthen the impact and support of the ADW methodology in comparison to version 0.

This chapter finishes off with the evaluation of version 1 of the ADW methodology in order to verify the belief that version 1 will strengthen the impact and support of the ADW (version 1) methodology for the agile development of a DW system. This evaluation was performed by developing and distributing the version 1 questionnaire and it aimed at determining the usefulness of the best practices introduced in each major phase of the ADW (version 1) methodology as well as the methodology as a whole. The analysis of the data generated by the version 1 questionnaire revealed that all the best practices for each major phase as well as the ADW (version 1) methodology as a whole were considered useful in the agile development of a DW system. Therefore, the evaluation of version 1 resulted in the verification of the usefulness of the ADW (version 1) methodology to provide support and guidance to a project team for the agile development of a DW system.

This concludes the evaluation stage of the research methodology and the end product of this stage as well as the final artefact of the study produced a verified version 1 (final) of the ADW methodology for the agile development of a DW system. In the next chapter the conclusion stage of the research methodology will be discussed in terms of the objectives (section 1.4) of the study in order to recap how the objectives were addressed and to highlight key findings.

Chapter 7: Conclusion

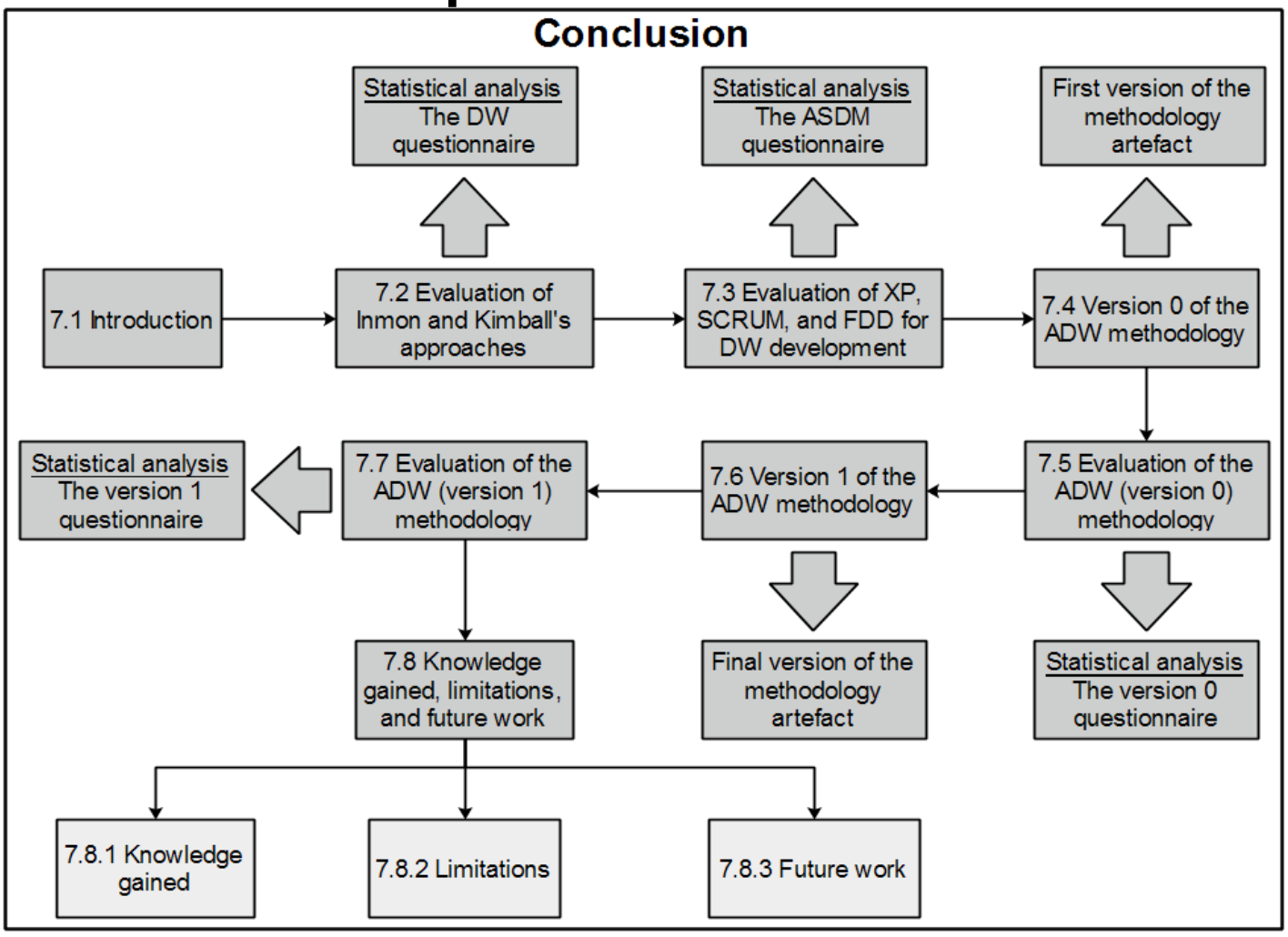


Figure 7.1 - Overview of Chapter 7

7.1 Introduction

In section 1.3 the problem statement of this study pointed out that DW system development is plagued by uncertainty and changing requirements and the aim of the study was to address the issues encountered by DW development by adopting best practices from ASDMs.

Therefore, the aim and purpose of this study was to develop the ADW methodology that uses either Inmon's or Kimball's approach as the foundation and to implement the best practices from XP, SCRUM, and FDD upon this foundation. The purpose and goal of the ADW methodology was to provide guidelines and support to a development team that deals with the unique technical aspects of DW development along with the issues associated with DW development using agile best practice in order to develop a DW in an agile environment. In other words, this study aimed to develop the final version of the ADW methodology in order to assist and support a project team in the successful and agile development of a DW system. In section 1.4 a number of objectives are mentioned in order to achieve the aim of this study and they are

- Evaluate Inmon's and Kimball's approaches and determine which is best suited for agile development.
- Evaluate XP, SCRUM, and FDD in order to determine which of the three is best suited for each major phase of the best-suited DW development approach (either Inmon or Kimball).
- Develop version 0 of the ADW methodology.
- Evaluate the ADW (version 0) methodology by developing a DW in collaboration with the industry partner.
- Develop version 1 (final) of the ADW methodology based on the evaluation of version 0.
- Evaluate the final version of the ADW methodology using a questionnaire that is targeted at experienced DW developers.
- Present the knowledge gained, limitations, and future work.

In this chapter, a brief summary of how each of the objectives have been achieved will be discussed in order to reveal how the study succeeded in accomplishing the aims and objectives it set out to achieve in order to address the problem statement introduced in section 1.3.

7.2 Evaluation of Inmon's and Kimball's approaches

The evaluation of Inmon's (section 3.6) and Kimball's (section 3.7) approaches formed part of the suggestion stage of the research methodology and was discussed in section 5.2. The evaluation of Inmon's and Kimball's approaches aimed to determine which one of the two approaches is best suited for the agile development in regard to the four main stages of DW system development: requirements gathering, data modelling, ETL, and data access and deployment. The two approaches were evaluated by the general characteristics of ASDMs using the twelve principle of the Agile Manifesto (section 1.2.2) in order to determine which of

these two approaches would be the most suited approach in regard to the four major stages of DW development to form the foundation for version 0 of the ADW methodology for the agile development of a DW system.

The evaluation of Inmon's and Kimball's approaches was performed by developing and distributing the DW questionnaire to six developers with experience in both Inmon's approach, Kimball's approaches and ASDMs. The DW questionnaire aimed at determining which of the two DW development approaches is best suited for the agile development of a DW system. The DW questionnaire used an agree-disagree scoring table (Table 5.1) and used the four major stages of development and the twelve principles of the Agile Manifesto as the evaluation criteria which revealed that Kimball's approach is best suited for agile development compared to Inmon's approach. Therefore, the evaluation of Inmon's and Kimball's approaches pointed out that Kimball's approach is best suited for the agile development of a DW system and thus formed the foundation for further evaluation in order to develop version 0 of the ADW methodology. In the next section a summary of the evaluation of the three ASDMs (XP, SCRUM, and FDD) will be discussed.

7.3 Evaluation of XP, SCRUM, and FDD for DW development

The evaluation of XP (section 2.4.1), SCRUM (section 2.4.2), and FDD (section 2.4.3) formed part of the suggestion stage of the research methodology and was discussed in section 5.3. Based on the evaluation of Inmon's and Kimball's approaches (summarized in section 6.2), Kimball's approach (best suited for agile development) was selected to form the foundation of the evaluation of the three ASDMs (XP, SCRUM and FDD). Using Kimball's approach as the foundation of the evaluation, the 13 stages of development in Kimball's lifecycle were categorized into four primary phases (planning, design, development, and implementation) with the 13 stages of Kimball's lifecycle as sub-phases. The aim and purpose of the evaluation of the three ASDMs was to determine the best suited ASDM for each of the primary phases of Kimball's lifecycle.

The evaluation of XP, SCRUM, and FDD was performed by developing and distributing the ASDM questionnaire to six developers who have experience in using Kimball's approach for the development of a DW system as well as experience is using XP, SCRUM, and FDD in the development of ISs. The ASDM questionnaire focused on revealing which of the three ASDMs is best suited for each primary phase of Kimball's lifecycle and used an agree-disagree scoring table (Table 5.1) with the four primary phases of Kimball's lifecycle and their sub-phases as the evaluation criteria. The statistical analysis of the data generated by the ASDM questionnaire revealed that the following ASDMs are best suited for each primary phase:

- Planning – SCRUM
- Design – FDD
- Development – XP
- Implementation – SCRUM

Therefore, the evaluation of XP, SCRUM, and FDD revealed that SCRUM is best suited for the planning and implementation phases, FDD is best suited for the design phase, and XP is best suited for the development phase for the agile development of a DW system using Kimball's approach as the foundation. In the next section a summary of version 0 of the ADW methodology will be discussed.

7.4 Version 0 of the ADW methodology

The development and introduction of version 0 of the ADW methodology formed part of the development stage of the research methodology and was discussed in section 5.4. Based on the evaluation of Inmon's and Kimball's approaches (summarized in section 7.2) and the evaluation of the three ASDMs (summarized in section 7.3) version 0 of the ADW methodology was developed. The ADW (version 0) methodology used Kimball's approach as the foundation to address the unique technical aspects associated with DW development and used the best practices of XP, SCRUM, and FDD to build upon this foundation to develop an approach for the agile development of a DW system. The ADW (version 0) methodology recommends best practices from SCRUM for the planning and implementation phases, best practices from FDD for the design phase and best practices from XP for the development phase. The lifecycle and best practices for each primary phase of the ADW (version 0) methodology are the important factors to highlight about the ADW (version 0) methodology and will be mentioned but not discussed (discussed in detail in section 5.4).

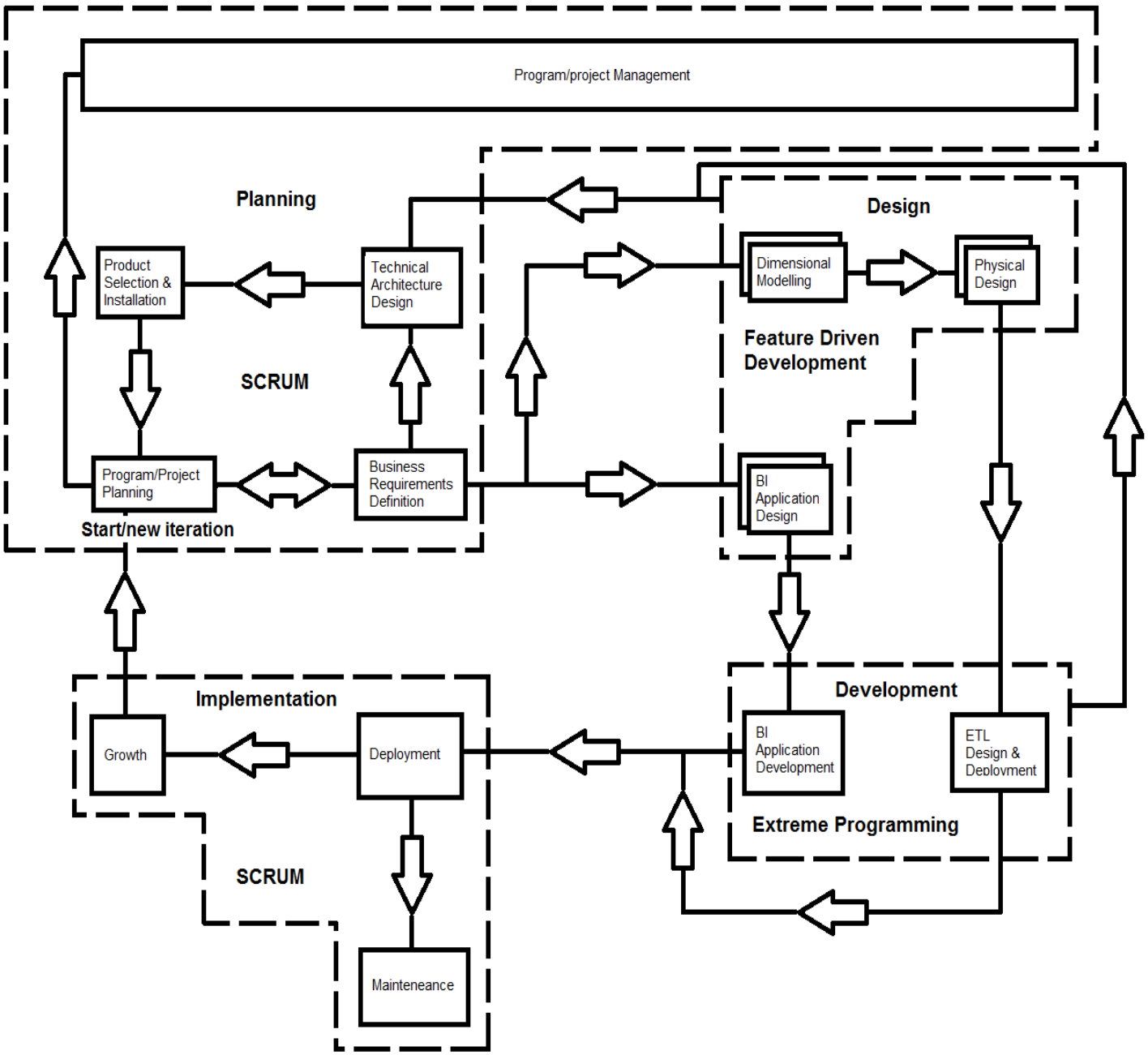


Figure 7.2 - Recap of the ADW (version 0) methodology lifecycle diagram

Figure 7.2 illustrates the lifecycle of the ADW (version 0) methodology and the best practices introduced in each primary phase of development for version 0 are

Planning phase

- **SCRUM roles**
- **Product backlog**
- **Sprint backlog**
- **Daily SCRUM meeting**
- **Pre-print planning**

Design phase

- **Design by feature**
- **Inspection**
- **Feature teams**
- **Simplicity**
- **Eliminate process pride**
- **Feature list**

Development phase

- **Small releases**
- **Metaphor**
- **Testing**
- **Simple design**
- **Refactoring**
- **Pair programming**
- **Collective ownership**
- **Continuous integration**
- **Forty-hour week**
- **On-site client**
- **Coding standards**

Implementation phase

- **Product backlog**
- **Sprint results**
- **Post-sprint meeting**

The best practices mentioned above are the recommended best practices of the ADW (version 0) methodology adopted from either XP, SCRUM, or FDD to assists and provide support to a project team in the successful agile development of a DW system. In the next section the summary of the evaluation of the ADW (version 0) will be discussed.

7.5 Evaluation of the ADW (version 0) methodology

The evaluation of version 0 of the ADW methodology formed part of the evaluation stage of the research methodology and was discussed in section 6.3. The aim of version 0 of the ADW methodology was to assist and support a project team in the successful agile development of a DW system. The aim of version 0 was tested by implementing version 0 of the ADW

methodology in the actual development of a DW system in collaboration with a business partner in a real-world environment where change is bound to happen (reason behind the agile development of a DW system). The implementation of version 0 was discussed in section 6.2 and revealed that the following recommendations be considered for the development of version 1 (final) of the ADW methodology:

- Keep the planning and implementation phases as is.
- Increase the effectives of the BI track by first completing the data track. In order to achieve this without causing difficulty in the flow of the lifecycle, the project team recommended merging the design and development phases into one phase.
- Remove the pair programming best practice.

In addition to the constructivism and empirical knowledge gained during the testing of version 0, version 0 was also evaluated in order to determine whether the ADW (version 0) methodology achieved the aim and purpose it set out to achieve. Therefore, the evaluation of version 0 of the ADW methodology was to determine whether the ADW (version 0) methodology can assist and provide support to a project team for the agile and successful development of a DW system.

The evaluation of version 0 of the ADW methodology was performed by developing and distributing the version 0 questionnaire to five relevant stakeholders and business users. The version 0 questionnaire aimed at determining whether the implementation of version 0 of the ADW methodology provided support and assistance to the project team for the agile development of a DW system and whether the developed DW system was viewed as a success.

The version 0 questionnaire was self-oriented and the questions were multiple-choice questions presented as statements that required the respondents to provide an agree-disagree answer based on the scoring system presented in Table 6.1. The version 0 questionnaire revealed that version 0 of the ADW methodology succeeded in providing support and assistance to the project team for the successful and agile development of a DW system. In the next section a summary of version 1 of the ADW methodology will be discussed

7.6 Version 1 of the ADW methodology

Although the evaluation of version 0 of the ADW methodology proved to assist and support a project team in the successful and agile development of a DW system, it also revealed that the implementation of version 0 lacked sustainable development (principle 8 of the Agile manifesto). In addition to the countermeasure to address the gap revealed by the evaluation of version 0, the implementation of version 0 identified certain aspects (mentioned above in

section 7.5) of the methodology that can be improved upon in order to strengthen and increase the impact of version 0 of the ADW methodology. Therefore, based on the knowledge gained during the implementation and evaluation of version 0 of the ADW methodology, version 1 (final) was developed and formed part of the evaluation stage of the research methodology and was discussed in section 6.4. Version 1 of the ADW methodology is the final version of the ADW methodology and was developed by performing changes to version 0 (will be discussed as the components of version 1 are introduced) based on the knowledge gained during the implementation and evaluation of version 0 to deliver an improved and final version of the ADW methodology. The lifecycle and best practices for each primary phase of the ADW (version 1) methodology are the important factors to highlight about the ADW (version 1) methodology and will be mentioned but not discussed (discussed in detail in section 6.4).

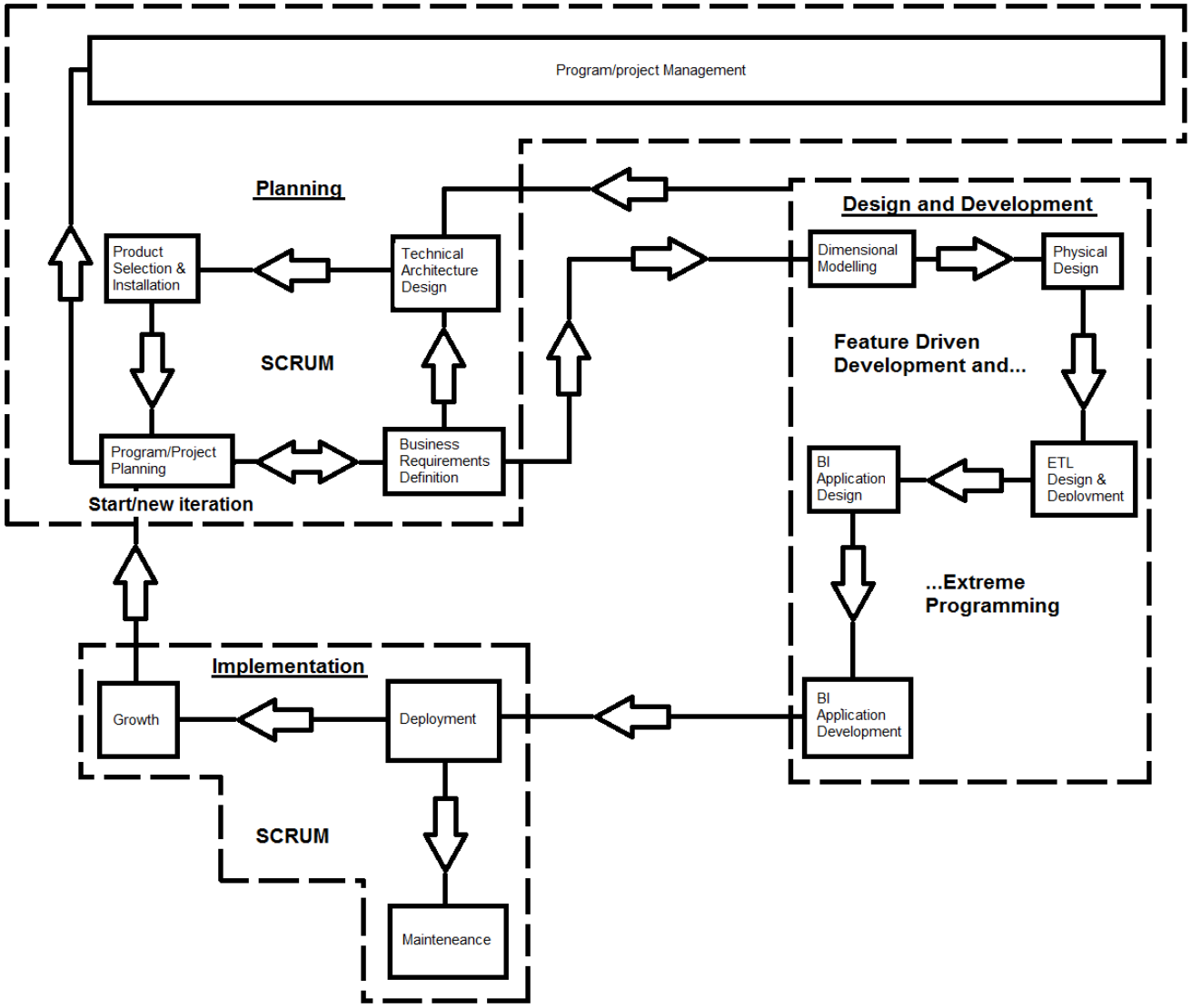


Figure 7.3 - Recap of the ADW (version 1) methodology lifecycle diagram

Version 1 of the ADW methodology differs slightly from version 0, but the most prominent difference between the two versions lie in the lifecycle with the merger of the design and the

development phases into one phase. This affected the overall flow of the lifecycle, which is believed will provide for an improved flow between the data and BI tracks (reason behind the merger). Taking a look at both Figure 7.2 (version 0 and presented in section 7.4) and Figure 7.3 (version 1) the difference caused by the merger can be seen and how it affected the flow of the diagram. These were the only differences associated with the lifecycle of version 1 in comparison to version 0 and the recommended best practices for version 1 are

Planning phase

- **SCRUM roles**
- **Product backlog**
- **Sprint backlog**
- **Daily SCRUM meeting (complete on daily bases)**
- **Pre-print planning**

Development phase

- **Daily SCRUM meeting (complete on daily bases)**
- **Design by feature (FDD)**
- **Inspection (FDD)**
- **Feature teams (FDD)**
- **Simplicity/simple design (FDD/XP)**
- **Eliminate process pride (FDD)**
- **Feature list (FDD)**
- **Small releases (XP)**
- **Metaphor (XP)**
- **Testing (XP)**
- **Refactoring (XP)**
- **Collective ownership (XP)**
- **Continuous integration (XP)**
- **Forty-hour week (XP)**
- **On-site client (XP)**
- **Coding standards (XP)**

Implementation phase

- **Daily SCRUM meeting (complete on daily bases)**
- **Product backlog**
- **Sprint results**
- **Post-sprint meeting**

The only differences between version 0 and version 1 in regard to the recommended best practices are the merger of XP and FDD best practices from the design and the development phases of version 0 and the removal of the pair programming best practice. Additionally, the daily SCRUM meeting best practice introduced in the planning phase was added to each major phase to address the issue of sustainable development (principle 8). In the next section the summary of the evaluation of the ADW (version 1) will be discussed.

7.7 Evaluation of the ADW (version 1) methodology

The evaluation of version 1 of the ADW methodology formed part of the evaluation stage of the research methodology and was performed in section 6.5. The evaluation of version 1 of the ADW methodology was concerned verifying the effectiveness of the ADW (version 1) methodology for the successful and agile development of a DW system.

The evaluation of version 1 of the ADW methodology was performed by developing and distributing the version 1 questionnaire to 24 developers with experience in using Kimball's approach for DW development as well as knowledge and/or experience with XP, SCRUM, and FDD. The version 1 questionnaire was concerned with identifying the opinions of the target group in regard to each of the best practices introduces in each major phase of development as well as the ADW (version 1) methodology as a whole.

The version 1 questionnaire was self-oriented and the questions were multiple-choice questions presented as statements that required the respondents to provide an agree/disagree answer based on the scoring system presented in Table 6.1. The version 1 questionnaire revealed that the respondents viewed the best practices introduced in each major phase and the ADW (version 1) methodology as a whole as useful for the successful and agile development of a DW system. Therefore, the version 1 questionnaire verified the usefulness of version 1 of the ADW methodology and the study can confidently state that the ADW (version 1) methodology will provide assistance and support to a project team for the agile development of a DW system in a changing environment. In the next section the key point of the study will be highlighted, limitations will be discussed, and future work to promote the evolution of the ADW methodology will be mentioned.

7.8 Knowledge gained, limitations and future work

In this section the study will be brought to a close by highlighting the knowledge gained, mentioning and discussing the limitations encountered, and recommending future work.

7.8.1 Knowledge gained

The general aim of the study was to address the uncertainty and changing requirements issues facing DW development. From the literature it was identified that ASDMs are focused on dealing with the same issues encountered in DW development but fall short in regard to the technical aspects associated to DW development. Therefore, a more specific aim of the study was to develop a new innovative methodology (ADW methodology) that uses Inmon's or Kimball's approach as the foundation to address the technical aspects associated with DW development. Building upon this foundation the best practices from three ASDMs (XP, SCRUM, and FDD) were added to address the uncertainty and changing requirements issues plaguing DW development. In order to achieve this aim the study discovered the following key factors that had a direct influence on the development of the ADW methodology.

- Kimball's approach is best suited for agile development compared to Inmon's approach.
- XP is concerned with and places emphasis on best practices that focus on improving the development aspects (coding, integration, testing, etc.) of IS development and is the best fit from the three ASDMs for the development phase of Kimball's approach.
- SCRUM is concerned with and places emphasis on best practices that focus on improving the management aspect of IS development and is the best fit from the three ASDMs for the planning and implementation phase of Kimball's approach.
- FDD is concerned with and places emphasis on best practices that focus on improving the design aspect of IS development and is the best fit from the three ASDMs for the design phase of Kimball's approach.

These factors formed the building blocks of the ADW (version 0) methodology to address the issues facing DW development while still addressing the unique technical aspects associated to DW development. Using these building blocks, the ADW (version 0) was developed, but the newly developed methodology consisted of no substance/value without evidence to support the purpose of the ADW (Version 0) methodology (aim of the study). Therefore, the ADW (version 0) was tested and evaluated by implementing the methodology in the actual development of a DW system in a real-world environment where uncertainty and changing requirements are bound to be encountered during development. This revealed the value of version 0 as a useful agile DW development approach to assist and support a project team for the successful development of a DW system in an environment characterised by uncertainty

and changing requirements. Regardless of the value revealed by the testing and evaluation of version 0, version 0 was improved based on the constructivism and empirical knowledge gained during the implementation of the ADW (version 0) methodology to result in version 1 (final) of the ADW methodology. The value of version 1 of the ADW methodology is directly related to the value of version 0 which was further strengthened during the evaluation of version 1 of the ADW methodology. Therefore, based on the contribution verified by the evaluation of version 0 and version 1, the final verdict of the study is - **The ADW (version 1) will provide assistance and support to a project team in the successful and agile development of a DW system in a development environment characterised by uncertainty and changing requirements.**

Therefore, the study contributed an empirical agile DW development approach (ADW (version 1) methodology) that uses Kimball's approach (high roller in DW development) as the foundation combined with the benefits and best practices of three ASDMs (XP, SCRUM, and FDD) suitable for DW development. Therefore, the ADW (version 1) methodology can do everything Kimball's approach can and then some.

In conclusion, the ADW (version 1) methodology introduces a new innovative approach to DW development to provide assistance and support to project teams for the successful and agile development of a DW system. Therefore, the ADW methodology is recommended to project teams who desire to shift their DW development process towards a more agile environment or for DW development projects faced with uncertainty and changing requirements.

In the next section the limitations encountered during the research will be discussed.

7.8.2 Limitations

During the research carried out in the study a number of limitations was encountered and these will briefly be discussed:

- The evaluation of Inmon versus Kimball only collected data from a small target group (six developers) and may prove to generate different results in a larger group.
- The above mentioned might have been influenced by the context.
- The evaluation of the three ASDMs only collected data from a small target group (six developers) and may prove to generate different results in a larger group.
- Although the implementation of version 0 of the ADW methodology was performed in collaboration with an industry partner in a real-world setting, the following limitations were encountered in the implementation of the ADW (Version 0) methodology due to the size of the company and the limited time of this study:
    - Only one iteration of the ADW (version 0) methodology could be performed.

- The project team was very small and could not extensively test the effectiveness of the ADW (version 0) methodology in larger teams faced with more complex and challenging issues.

This concludes the limitations encountered during the research. In the next section the future work will be discussed.

7.8.3 Future work

The study verified the usefulness of the ADW (version 1) methodology using the version 1 questionnaire, but for future work to further strengthen the claim of this study or to improve upon the ADW (version 1) methodology to result in a more effective DW development approach, it is suggested that the ADW (version 1) methodology be implemented in a number of organizations involved in DW development projects to test, evaluate and answer the following questions:

- How does the ADW (version 1) methodology perform in a large DW project (longer than 1 year)?
- How does the ADW (version 1) methodology perform in a large project team?
- How does the ADW (version 1) methodology perform across a number of different organizational cultures and can a trend be identified?

This concludes the future work, this chapter as well as this study.

References

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J. 2002. Agile software development methods: Review and analysis. Espoo: VTT publications.

Almeida, M.S., Ishikawa, M., Reinschmidt, J. & Roeber, T. 1999. Getting started with data warehouse and business intelligence. San Jose, CA: IBM coporation, International Technical Support Organization.

Ambler, S.W. 2013. Agile best practices for data warehousing (DW)/ business intelligence (BI) projects. http://agiledata.org/essays/dataWarehousingBestPractices.html. Date of access: 23 October 2014.

Avison, D. E. & Fitzgerald G. 2003. Where Now for Development Methodologies? *Communications of the ACM,* 46(1):79-82.

Avison, D. E. & Fitzgerald G. 2006a. Information systems development: Methodology, techniques, and tools. 4th ed. Maidenhead: McGraw Hill Publishing Company.

Avison, D., & Fitzgerald, G. 2006b. Methodologies for developing information systems: A historical perspective. (*In* Avison, D. *et al*., *eds*. The Past and Future of Information Systems: 1976–2006 and Beyond. Santiago, Chile: Springer. p. 27-38).

Aydin, M. N., Harmsen, F., Van Slooten, K., & Stegwee, R. A. 2005. On the adaptation of an agile information systems development method. *Journal of database management*, *16*(4):24-40.

Baird, S. 2002. Sams teach yourself extreme programming in 24 hours. Indianapolis, IN: Sams.

B*ălăceanu, D.2007. Components of a business intelligence software solution. Informatica Economică,* 2(42), 67-73.

BI-Insider. 2011. Benefits of a data warehouse. http://bi-insider.com/portfolio/benefits-of-a-data-warehouse/. Date of access: 24 October 2014.

Boehm, B. 2002. Get ready for agile methods, with care. *IEEE Computer,* 35(1):64-69.

Boehm, B.W. & Turner, R. 2003. Balancing agility and discipline: a guide for the perplexed. Boston, MA: Addison-Wesley.

Breslin, M. 2004. Data warehousing battle of the giants: Comparing the basics of the Kimball and Inmon models. *Business intelligence journal*, 7:6-20.

Burke, E.M. & Coyner, B.M. 2003. Java extreme programming cookbook. Sebastopol, CA: O'Reilly Media.

Chin, G. 2003. Agile project management: How to succeed in the face of changing project requirements. http://www.powells.com/cgi-bin/biblio?inkey=62-0814471765-0. Date of access: 23 October 2014.

Conboy, K. & Fitzgerald, B. 2004. Towards a conceptual framework of

agile methods. (*In* Zannier, C. *et al*., *eds*. XP/Agile universe 2004. Berlin, Heidelberg: Springer. p. 105-116.)

Chromatic. 2003. Extreme programming pocket guide. Sebastopol, CA: O'Reilly Media.

Crispin, L. & House, T. 2003. Testing extreme programming. Boston, MA: Addison-Wesley.

De Luca, J. 2005. Feature driven development overview. http://www.nebulon.com/articles/fdd/download/fddoverview.pdf. Date of access: 11 March 2015.

De Silva, F. 2005. Data warehousing & business intelligence ROI. Proceedings of the international conference on computer and international management, ICIM, October 29-30, 2005, Bangkok, Thailand.

Fowler, M. 2001. Put your process on a diet. www.sdmagazine.com/documents/s=737/sdm0012a/0012a.htm Date of access: 4 March 2014.

Fowler M. & Highsmith. J. 2001. The Agile Manifesto. Software development magazine, August. http://www.sdmagazine.com/documents/s=844/sdm0108a/0108a.htm. Date of access: 1 October 2014.

Good, J.M. 2003. A pragmatic approach to the implementation of agile software development methodologies in plan-driven organisations. Canterbury: Lincoln University. (Dissertation-Hons. BSC).

Graziano, K. 2005. Agile methods and data warehousing. http://download-east.oracle.com/oowsf2005/628 wp.pdf. Date of access: 23 October 2014.

Grenning, J. 2002. Extreme programming and embedded software development. Embedded Systems Conference 2002, Chicago.

Grey, J. 2006. An investigation of the suitability of agile system development methodologies for the development of data warehouses. Potchefstroom: NWU. (Dissertation – MSC).

Grey, J. 2011. The development of a hybrid agile project management methodology. Potchefstroom: NWU. (Thesis – PhD).

Guerra, J. & Andrews, D. 2013. Why You Need a Data Warehouse. 1ste ed. Cheshire: RapidDecision.

Haugan, G.T. 2011. Project management fundamentals: key concepts and methodology. Vienna, VA: Management Concepts.

Highsmith, J. 2002a. What is agile software development? Crosstalk:4-9, Oct. http://www.adaptivesd.com/articles/cross_oct02.pdf. Date of access: 23 March 2015

Highsmith, J. 2002b. Extreme programming: Agile project management advisory white paper service. http://rockfish.cs.unc.edu/COMP290-agilelxp- highsmith.pdf. Date of access: 23 October 2014.

Highsmith, J. 2002c. Agile software development ecosystems. Boston, MA: Addison-Wesley.

Highsmith, J. & Cockburn, A, 2001. Agile software development: the business of innovation. *IEEE Computer,* 34(9):120-122.

Hislop, G.W., Lutz, M.J., Naveda, J.F., McCracken, W.M., Mead, N.R. & Williams, L.A. 2002. Integrating agile practices into software engineering courses. *Computer science education*, 12(3):169-185.

Holcombe, M. & Holcombe, W.M.L. 2008. Running an agile software development project. Hoboken, NJ: Wiley.

Huijbers, R., Lemmens, F., Senders, B., Simons, S., Spaan, B., van Tilburg, P. & Vossen, K. 2004. Software Project Management: Methodologies & Techniques. http://paul.luon.net/essays/SEP-essay-final.pdf. Date of access: 4 May 2015.

Huisman, H.M. 2014. Information systems engineering. Potchefstroom: NWU, Potchefstroom Campus. (Study guide ITRI 624).

Huisman, M. & Iivari, J. 2002. The individual deployment of systems development Methodologies. (*In* Banks Pidduck, A. *et al*., *eds*. Advanced information systems engineering. Berlin, Heidelberg: Springer. p.134-150).

Huisman, M. & Iivari, J. 2006. Deployment of systems development methodologies: Perceptual congruence between IS managers and systems developers. *Information and management 43*:29-49

Inmon, W.H. 2003. Building the data warehouse. 3rd ed. Indianapolis: Wiley Publishing Inc.

Informatica. 2001. Agile methodology for data warehouse and data integration projects. Karthik Kannan, Informatica Professional Services white paper. https://www.informatica.com/downloads/1749_Agile_Methodology_wp_web.pdf. Date of access: 7 October 2015.

Jukic, N. 2006. Modeling strategies and alternatives for data warehousing projects. *Communications of the ACM*, 49(4):83-88.

Kammermeier, M. 2009. Agile project management in IT development projects with a focus on team performance. Norderstedt: GRIN.

Kimball, R., Reeves, L., Ross, M. & Thornthwaite, W. 2008. The data warehouse lifecycle toolkit. 2de ed. New York: John Wiley & Sons.

Krawatzeck, R., Dinter, B. & Thi, D.A.P. 2015. How to make business intelligence agile: The agile BI actions catalog. 48th Hawai International Conference on System Sciences. p:4762-4771.

Krippendorff, K. 2011. Computing Krippendorff's Alpha-Reliability. http://repository.upenn.edu/asc_papers/43. Date of access: 2 Augustus 2015.

Larman, C. 2004. Agile and iterative development: a manager's guide. Boston, MA: Addison-Wesley.

Lindstrom, L. & Jeffries, R. 2004. Extreme programming and agile software development methodologies. *Information systems management*, 21(3):41-52.

Lindvall, M., Basili, V., Boehm, B., Costa, P., Kathleen, D., Shull, F., Tesoriero, R., Williams, L.A. &Zelkowitz, M.V. 2002. Empirical findings in agile methods. (*In* Wells, D. & Williams, L., *eds*. Lecture notes in computer science: Extreme Programming and agile methods – XP/Agile Universe 2002. Vol. 2418. Berlin: Springer. p. 81-92.).

March, S. & Smith, G. 1995. Design and natural science research on information technology. *Decision support systems*, 15:251–266.

Mishra, J. & Mohant, A. 2011. Software Engineering. 1ste ed. India: Pearson Education.

Nguyen, P. V. 2011. Using data warehouse to support building strategy or forecast business tend. *arXiv preprint arXiv:1205.0724*.

Oates, B.J. 2006. Researching information systems and computing. London: SAGE Publications Ltd.

Palmer, S.R. &Felsing, J.M. 2002. A practical guide to feature driven development. Upper Saddle River, NJ: Prentice-Hall.

Pearman, G. & Goodwill, J. 2006. Pro .NET 2.0 extreme programming. Berkeley, CA: Apress.

Peterson, R.A. 2000. Constructing effective questionnaires. London: Sage, Straub.

Quarles, D. 2002. Business intelligence: The umbrella term. (In BWI – Werkstuk. Universiteit Amsterdam. Nov. 2002. http://www.few.vu.nl/stagebureau/werkstuk/werkstukken/werkstuk-quarles.doc. Date of access: 23 October 2014).

Qumer, A. & Henderson-Sellers, B. 2008. An evaluation of the degree of agility in six agile methods and its applicability for method engineering. *Information and software technology*, 50:280-295.

Rainardi, V. 2006. Differences between data warehousing and business intelligence. https://dwbi1.wordpress.com/2011/05/31/differences-between-data-warehousing-and-business-intelligence/. Date of access: 6 June 2015.

Ranjan, J. 2009. Business intelligence: concepts, components, techniques and benefits. *Journal of theoretical and applied information technology,* 9(1):060-070.

Rosencrance, L. 2011. Top five benefits of a data warehouse. Spotfire blogging team, 10 Augusts 2011. http://spotfire.tibco.com/blog/?p=7597. Date of access: 15 October 2014.

Rouse, M. 2007. Agile software development (ASD). *TechTarget*, February. http://searchsoftwarequality.techtarget.com/definition/agile-software-development. Date of access: 11 March 2015.

Rychly, M. &Ticha, P. 2008. A tool for supporting feature-driven development. p. 196–207. (*In* Meyer, B., Nawrocki, J.R. & Walter, B., *eds*. Balancing agility and formalism in software engineering: Second IFIP TC 2 Central and East European Conference on Software Engineering Techniques, CEE-SET 2007, Poznań, Poland, Oct. 10–12, 2007: revised selected papers. Berlin: Springer. p. 196–207.)

Schuh, P. 2004. Integrating agile development in the real world. Hingham, MA: Charles River Media, Inc.

Schwaber, K. 2001. Will the real agile process please stand up? *E-Project Management Advisory Service, Cutter Consortium*, 2(8).

Schwaber, K.& Beedle, M. 2002. Agile software development with SCRUM. Upper Saddle River, NJ: Prentice-Hall.

Schwaber, K. & Sutherland, J. 2013. The scrum guide – The definitive guide to scrum: The rules of the game. http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf. Date of access: 8 March 2015.

Slazinski, E.D. 2004. Structured query language (SQL). (*In* The internet encyclopedia. Volume 3. 353-364).

Smith, S. 1996. Positivism and beyond. (*In* Smith, S., Booth, K. & Zalewski, M. *eds*. International theory: positivism & beyond. Cambridge, Cambridge university press. p. 11-46).

Stackowiak, R., Rayman, J. & Greenwald, R. 2007. Oracle data warehousing and business intelligence solutions, Indianapolis, NI, Wiley Publishing, Inc.

Stober, T. & Hansmann, U. 2010. Agile software development: best practices for large software development project. Berlin, Heidelberg: Springer.

The business intelligence blog. 2008. Agile development for BI. https://biguru.wordpress.com/2008/04/06/agile-development-for-bi/#comments. Date of access: 7 October 2015

TDWI (The Data Warehousing Institute). 2004. TDWI business intelligence fundamentals: From data warehousing to business impact. http://www.tdwi.org/files/pub/tdwi/BI_Fundamentals_Preview_1.pdf. Date of access: 23 October 2014.

Toflo, C. & Wazlawick, R.S. 2008. The influence of organizational culture on the adoption of extreme programming. *Journal of systems and software*, 81(11):1955-1967.

Trochim, W.M.K. 2006. Positivism & post-positivism. http://www.socialresearchmethods.net/kb/positvsm.php. Date of access: 20 October 2015.

Truex, D., Baskerville, R. & Travis, J. 2000. Amethodical systems development: the deferred meaning of systems development methods. *Accting., Mgmt. & Info. Tech*, 10:53-79.

Vaishnavi, V. & Keuchler, B. 2004. Design research in information systems, AISWorldnNet. http://desrist.org/design-research-in-information-systems/. Date of access: 21 August 2015.

Walker, D.M. 2009. Data warehousing – Change management in a challenging environment. SYBASE white paper.

Westfall, L. 2009. The certified software quality engineer handbook. Milwaukee, Wisconsin: ASQ Quality Press.

Wu, H.B. 2011. On software engineering and software methodologies a software developer's perspective. 2011 international conference on information science and technology, ICIST, March 26-28, 2011, Nanjing, China.

Zaied, A.N.H., Aal, S.I.A., & Hassan, M.M. 2013. Rule-based expert system for selecting information systems development methodologies. *I.J intelligent systems and applications*, 09:19-26.

Zelkowitz, M.V. 2004. Advances in computers: advances in software engineering. Amsterdam: Academic Press.

Appendix A

The DW questionnaire

The DW questionnaire was researcher-administered and therefore the researcher arranged a group meeting with the entire target group. The researcher explained the aim of the questionnaire and provided the target group with a copy of the Agile Manifesto (section 1.2.2) the scoring system, and the following tables to complete the questionnaire. Questions were welcomed.

Score	Description
1	Unsuitable
2	Low suitability
3	Adequate suitability
4	High suitability
5	Very high suitability

Inmon's approach				
Agile principles	**Requirements gathering**	**Data modeling**	**Extract, transform and load (ETL)**	**Data access and deployment**
Principle 1				
Principle 2				
Principle 3				
Principle 4				
Principle 5				
Principle 6				
Principle 7				
Principle 8				
Principle 9				
Principle 10				
Principle 11				
Principle 12				

Kimball's approach				
Agile principles	**Requirements gathering**	**Data modeling**	**Extract, transform and load (ETL)**	**Data access and deployment**
Principle 1				
Principle 2				
Principle 3				
Principle 4				
Principle 5				
Principle 6				
Principle 7				
Principle 8				
Principle 9				
Principle 10				
Principle 11				
Principle 12				

The ASDM questionnaire

The DW questionnaire was researcher-administered and therefore the researcher arranged a group meeting with the entire target group. The researcher explained the aim of the questionnaire and answered any questions the respondents had in regard to the questionnaire. The questionnaire consisted of the following tables to capture the data from the target group.

Score	Description
1	Unsuitable
2	Low suitability
3	Adequate suitability
4	High suitability
5	Very high suitability

Kimball's DW development stages	Extreme programming (XP)	SCRUM	Feature driven development (FDD)
Planning			
Program/project planning			
Program/project management			
Business requirements definition			
Technical architecture design			
Product selection & installation			
Design			
Dimensional modelling			
Physical design			
BI application design			
Development			
ETL design & development			
BI application development			
Implementation			
Deployment			
Growth			
Maintenance			

The version 0 questionnaire

The questionnaire was distributed electronically via e-mail and created in word using radio buttons as a multiple-choice format for respondents to easily select the desired answer.

The respondent and their answers will remain anonymous. Please complete the questionnaire as quickly as possible and send the completed questionnaire to: 22357467@nwu.ac.za

Aim and goal of the questionnaire

The questionnaire aims to determine the degree of agility of the development process for the developed data warehouse (DW) project from the viewpoint of the business users. The first twelve questions (principle 1-12) are based on the principles of the Agile Manifesto and the questionnaire aims to determine how the business users viewed the development process and to what degree they agree with the statements made (questions). The last four questions (criteria of success 1-4) aim to measure the degree of success of the developed system based on a few basic and standard measures of success.

Instructions

For each of the statements (questions), please provide an answer that reveals to what degree the respondent agrees-disagrees with the statement. Please answer each question to the best of your ability and if the respondent has no opinion or idea on a specific statement, please use the bracket section to make an educated guess.

Note – If the document does not allow you to select an answer then you need to enable macros for the document and can usually be performed by just clicking on the enable button that appear as a notification under the ribbon menu.

Principle 1-The project placed the highest priority on satisfying the customer/client through early delivery of valuable software.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 2 - The project gladly welcomed changing requirements, even late in the development process.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 3 - The project delivered working software from a couple of weeks to a couple of months.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 4- The project allowed for business users to daily work together with the development team.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 5 - The project was developed around motivated individuals, giving them the environment and support they need and trust them to get the job done.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 6 - The project focused more on face-to-face communication opposed to e-mails and other means of media communication.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 7 - The primary measure of success for the project was working software.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 8 - The project promoted sustainable development; the project team maintained a constant.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 9 - The project placed attention to technical excellence and good design.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 10 - The project implemented simplicity.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 11 - The project allowed for self-organizing teams.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Principle 12 - The project allowed the project team to reflect on how to become more effective at regular intervals.

- ○ Completely disagree
- ○ Disagree
- ○ Neutral
- ○ Agree
- ○ Completely agree

Criteria of success 1 - Does the developed system meet the needs specified by the business users.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Criteria of success 2 -Was the developed system completed in an acceptable time frame (6 weeks for a DW system, even small ones, is extremely fast (Kimball states that the first iteration of a DW system usually takes at least 6 months)).

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Criteria for success 3 - Is the system easy to use and was the necessary training provided.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Criteria for success 4 - Does the developed system provide value to the business.

- ○ Completely disagree
- ○ Disagree
- ◉ Neutral
- ○ Agree
- ○ Completely agree

Thank you for your time.

The version 1 questionnaire

The questionnaire was developed in survey monkey and the link to the questionnaire was distributed during a class on database management for post graduate students.

Instructions

The questionnaire aims to verify the usefulness of the developed methodology. The ADW methodology aims to provide guidance and support for the agile development of a data warehouse system. The questionnaire is structured to describe a relevant aspect of the ADW (version 1) methodology followed by a multiple-choice option to select the **degree of usefulness** of the element under question. Please answer the questionnaire to the best of your knowledge. Please do not just fill in the answers randomly, read through the description of the methodology and answer the questions to the best of your ability. The layout of the questionnaire will consist of a short description of the methodology followed by a few questions.

Keywords:

Data warehouse (DW)

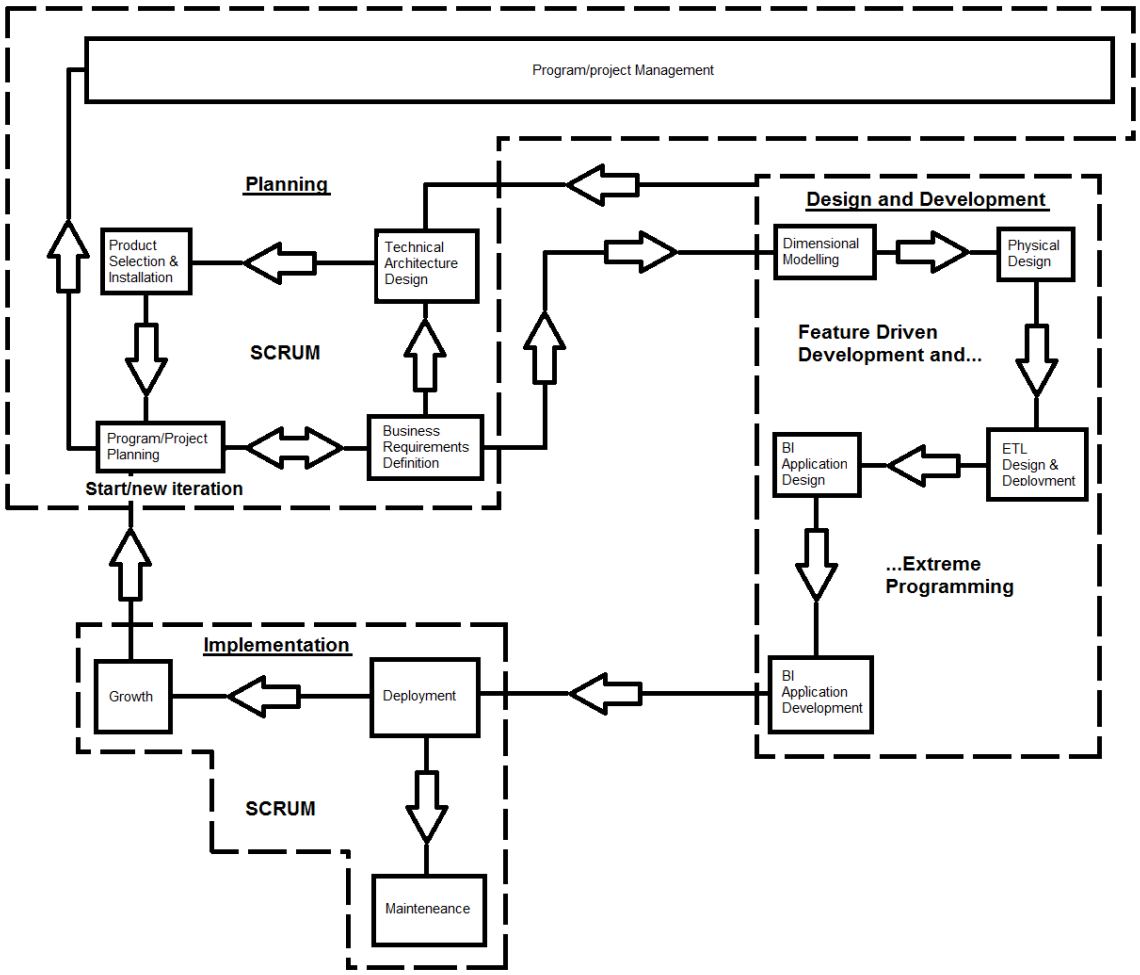
Extreme programming (XP)

SCRUM

Feature driven development (FDD)

Please contact me at 22357467@nwu.ac.za for any questions or if you wish to discuss some aspects components of the methodology. Thank you for your time.

Lifecycle



ADW (version 1) methodology

The lifecycle consists of three major phases which are planning, design and development, and implementation. These major phases are each made up of a number of sub-phases that have been adopted from Kimball's approach. Kimball's approach consists of 13 sub-phases which are identical to the ADW methodology. The only difference is that the 13 sub-phases have been categorized into one of the three major phases and the flow of the lifecycle has minor changes in regard to the original lifecycle of Kimball's approach. For each of the major phases, the best practices of XP, SCRUM or FDD (can be seen in the lifecycle diagram) will be adopted to assists in the successful completion of each phase.

Flow of the lifecycle

The flow of the lifecycle is relatively plain and simple and can easily be determined by looking at the diagram above. The only things worth mentioning is the fact that the loop in the planning phase will continue until all the stakeholders are satisfied with the requirements, scope, and cost estimates for the given iteration (cycle). Also, the start of the project triggers the program/project management sub-phase (top sub-phase of the diagram) which runs throughout the entire length of the project to ensure proper management of the project. As can be seen in the diagram, the design and development phase has an arrow that flows backwards into the technical architecture design sub-phase, this backwards flow is triggered when new information discovered during this phase will cause the architecture of the system

to change (might require additional software/hardware or different software than initially specified). In this event, the flow of the project moves back towards the technical architecture design sub-phase and moves forward from there. The rest of the lifecycle flow is plain and simple, where each task must first be completed before it can flow into the next.

The planning phase best practices

During the planning phase, best practice from the SCRUM methodology will be used to successfully complete the planning phase of a DW project. Since SCRUM is the most suited for the planning phase as well as the implementation phase, the names mentioned by SCRUM to refer to IS development processes will be used across the entire lifecycle to reduce confusion. Therefore, a sprint will be used to refer to an iteration in the ADW methodology. Also, different names that refer to the same roles will be adopted from the SCRUM ASDM in order to reduce confusion and misunderstanding.

SCRUM roles (product owner, SCRUM master, team) – In addition to the DW specific roles mentioned by Kimball's approach, SCRUM specific roles must be included when roles are being assigned in order to identify who will be responsible for SCRUM related activities. This will usually just involve the re-naming of some of the roles identified by Kimball's approach to adhere to the unique names associated with SCRUM.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Product backlog – The product backlog will contain all the DW specific documentation (refer to Kimball's approach) that is relevant for the unique development environment. Additionally, the product backlog must be handled as described by the SCRUM methodology. Any documentation produced during the other phases must be included in the product backlog.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Sprint backlog – Using the product backlog, the sprint backlog is populated with the requirements, after they have been prioritized, to be completed in the next sprint.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Pre-sprint planning – Before the start of each sprint a pre-sprint planning meeting must be held to identify the tasks necessary using Kimball's approach as a guideline to DW development.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Daily SCRUM meeting – Although the SCRUM ASDM is only used for the planning and implementation phases, the sub-phase program/project management (sub-phase of planning) is present during the entire lifecycle and therefore daily SCRUM meeting must be held throughout the lifecycle.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Design and development phase

During the design and development phase, best practice from the XP and FDD methodology will be used to successfully complete the design and development phase of a data warehouse project.

Design by feature – The identified requirements are broken down into manageable features to perform the design of the data warehouse system.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Inspection – Perform inspection on the dimensional model, physical design and BI application design to ensure that features (requirements) are the main focus and that the designs will address the features that must be completed.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Feature teams – Perform the design and development phase in small, dynamic teams that compliment and aid each other to successfully complete this phase.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Simplicity/simple design – Keep the design of the data warehouse system as simple as possible to assist rather than complicate the development process. This also ensures that the final product is easy to use and understand.

- Completely disagree

- Disagree
- Neutral
- Agree
- Completely agree

Eliminate process pride – Focus on completing the design and development of the data warehouse system, do not let team members pride stagnate the design and particularly the development of the data warehouse just to prove their approach was possible.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Feature list – Use the feature list method introduced in FDD to completely map the entire data warehouse system starting with a single business process to assist in the design of the data warehouse system. The feature list will particularly assist in the dimensional modeling sub-phase to map the entire data warehouse system and assist in identifying the grain of the fact table as well as the dimensions needed to meet the requirements of the user.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Small releases – The ADW methodology implements iterative development and focuses on one business process at a time and has the biggest impact on the design and development phase where the focus must be placed on a single business process at a time.

- Completely disagree
- Disagree

- Neutral
- Agree
- Completely agree

Testing – Perform unit tests to ensure the ETL process produced desired data and have handled all the necessary transformations to get the data as clean as possible. Unit testing must also be performed on the BI application to remove bugs and ensure the application performs as expected. Acceptance test must be performed on the ETL results and BI application to ensure the development phase meets the requirements and developed something of value to the users.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Refactoring – Once the project team identifies anything that is of no value, the project team must re-evaluate and return to the relevant sub-phases to re-design or re-develop the deliverable to produce something of value.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Collective ownership – The designs and codes created during the design and development phase belongs to the entire project team regardless of the feature team that created it and has the right to make changes as they see fit.

- Completely disagree
- Disagree
- Neutral

- Agree
- Completely agree

Continuous integration – The ETL process and BI application development must be integrated as frequently as possible. New transformations must be integrated and loaded into the database as frequently as possible. New components of the BI application must be added to the BI application as soon as they have been tested.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Forty-hour week – The design and development phase of a data warehouse project is very demanding and it is important that developers are fresh and rested to prevent tedious mistakes and errors. A project team that has a positive moral will lead to faster and increased productivity. Over-time is allowed but only for a small duration at a time.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

On-site client – During the design and development phase it is very helpful to have a business representative present at all time. This is particularly important during the ETL process to assist the project team to make sense of some of the issues encountered with the operational data. This will help the team to quickly define why certain data is captured in a certain manner and how to deal with the cleaning of these issues.

- Completely disagree
- Disagree
- Neutral

- Agree
- Completely agree

Coding standards- During the development phase it is very important to use coding standards with the ETL process and BI application development. This ensures that the integration of the ETL process is easier to perform and bring clarity amongst developers. If coding standards have been incorporated in the ETL process the BI application development will be easier to perform.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Implementation phase

During the implementation phase, best practice from the SCRUM methodology will be used to successfully complete the implementation phase of a DW project. The best practices adopted from SCRUM for the implementation phase of DW development are:

Product backlog – The project team must refer to the product backlog to see the training plan that has been put in place and update the plan if necessary.

- Completely disagree
- Disagree
- Neutral
- Agree
- Completely agree

Sprint results – Sprint results document will contain the components and issues discussed in the post-sprint meeting. The sprint results document must be added to the product backlog.

- Completely disagree
- Disagree

○ Neutral

○ Agree

○ Completely agree

Post-sprint meeting – After each sprint a post-sprint meeting must be held to evaluate the completed sprint and to ensure the sprint produced a system that is of value to the users. During this meeting it must be identified if a new sprint must be initiated for further growth of the DW system. This meeting must also identify what the training needs and support infrastructure of the users are. The maintenance needs of the users must initiate a new sprint that is only focuses on the maintenance and improvement of the current DW system.

○ Completely disagree

○ Disagree

○ Neutral

○ Agree

○ Completely agree

In conclusion, it can be seen that the ADW methodology does not attempt to introduce a new approach for the development of a data warehouse system, but introduces agile best practices from XP, SCRUM and FDD that builds upon the foundation provided by Kimball's approach. Therefore, the ADW methodology uses Kimball's approach to provides support for the technical and unique aspects associated with DW development and incorporates best practices from agile to assist in the management, planning, design and development, and implementation tasks associated with data warehouse development. The methodology only focuses on recommending the flow and agile best practices to assist a project team that is familiar with Kimball's approach (the methodology requires that one refer to Kimball's approach for the technical guidelines and techniques unique to data warehouse development).

The ADW methodology as a whole can be used to successfully develop a data warehouse system in an agile environment?

○ Completely disagree

○ Disagree

◉ Neutral

○ Agree

○ Completely agree

This serves to confirm that I, Isabella Johanna Swart, registered with and accredited as translator by the South African Translators' Institute, registration no. 1001128, language edited the following dissertation, excluding Table of Contents, Tables and Bibliography.

Inmon versus Kimball: The agile development of a data warehouse

by

Iddo-Imri Scholtz

Date: 24 November 2015

Dr Isabel J Swart

23 Poinsettia Close
Van der Stel Park
Dormehlsdrift
GEORGE
6529
Tel: (044) 873 0111
Cell: 082 718 4210
e-mail: isaswart@telkomsa.net