12-2016

Improving the Efficiency of BI Report Generation for Data Warehousing Projects

Mukarram Ali Mohammed
St. Cloud State University

Improving the Efficiency of BI Report Generation for Data Warehousing Projects

by

Mukarram Ali Mohammed

A Starred Paper

Submitted to the Graduate Faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements

for the Degree

Master of Engineering Management

December, 2016

Starred Paper Committee:
Ben Baliga, Chairperson
Hiral Shah
Balasubramanian Kasi

Abstract

This project has been implemented to replace the waterfall model of development with a new agile model of development for data warehousing projects in an organization. Some of the problems associated with waterfall model of development include high cost, late delivery of business value, inability to adapt to the changing requirements etc. The main goals of this project were to reduce the project cost, improve the quality of reports to increase user adoption, and make the process more time efficient. The stated goals were achieved by implementing agile model of development to the data warehousing projects, and were validated by the analysis of the data collected throughout the process of project execution.

Acknowledgements

I would like to take this opportunity to thank my advisor Dr. Ben Baliga, not only for his guidance and cooperation for the successful completion of my starred paper, but also for his continuous support and encouragement throughout my master's program. I would also like to extend my thanks to Dr. Hiral Shah for being an excellent mentor to me and for the guidelines she provided that helped me throughout the project. I would also like to thank Dr. Balasubramanian Kasi for serving as a committee member for this project, and the Department of Engineering Management, SCSU, for their help in providing the necessary resources.

I wish to thank my siblings Muqeet Ali Mohammed and Ruqia Maihveen for the continuous encouragement, support, love, and care they provided throughout my life. Lastly, I would like to express my deepest gratitude to my parents, Ahmed Ali Mohammed and Bilquees Begum. Their love, care and motivation at every step of my life defined the person I am today. They inspire me every day and gave me the strength that brought my Master's journey to its culmination.

Table of Contents

Chapter Page

List of Tables

List of Figures

Chapter I: INTRODUCTION

Introduction

The primary intent of this project was to mitigate the performance inefficiencies of the current data warehousing techniques within an organization, by migrating from the traditional "Waterfall" model to the new "Agile" model of software development. The goal of this project was to improve the quality of BI (Business Intelligence) reports being generated, reduce the cost of development and make the process time efficient. The migration of SDLC (software development life cycle) from "Waterfall" to "Agile" was performed in order to achieve the goals stated.

Data warehousing is the process of collecting data to be stored in a managed database in which the data are subject-oriented and integrated, time variant, and non-volatile for the support of decision-making (Inmon, 1993). Business Intelligence is a system that possess a set of tools and technologies that provide relevant information to the business users that is required to solve certain business problems and make tactical business decisions. In this era of increased competition, it is necessary to be equipped with means that are cost-effective, and ensure rapid access to the useful business information.

The company this project was undertaken at faced issues of extracting the desired data/reports on time from the data warehouse as per the requirements. The data in use by the company was stored in a typical data warehouse and "Informatica" tool was used to perform ETL (Extraction Transform Load) functions for data integration. Traditional waterfall model of software development was implemented to all the data warehouse development projects, which followed the life cycle of planning, requirements, analysis, design, construction and

deployment. As a part of this project, the new model of development impacted this life cycle to give way to a more iterative and collaborative approach of agile model of development.

Problem Statement

The traditional Waterfall model of development was implemented owing to the wide adoption and continued use of this approach of development by the software industry. The typical problems associated with implementing waterfall model for data warehousing projects were high-costs, late delivery of business value and inability to adapt to the changing business requirements. Data warehousing projects form an essential part of Customer Relationship Management (CRM), hence the inability to deliver the desired results on time affected the relationship of the organization with existing customers and made it difficult to expand the customer base, which resulted in the conceptualization of this project.

Nature and Significance of the Problem

Data warehouses store crucial data collected from a varied sources of information, including customers, vendors, markets, internal departments, etc., in a uniform format. This available data needs to be processed at the earliest to be useful to the organization, as outdated data will be of lower credibility. The resulting information aids in the process of making fact-based decisions that are of great importance to the organization.

The current waterfall model when implemented for data warehousing projects posed limitations such as inability to adapt to changes, utilizing more time and budget to secure the defined goals. The project aimed at improving the quality of generated BI reports and deliver results within a reasonable time and budget constraint, by mitigating the inefficiencies of the "Waterfall" model by replacing it with the "Agile" model of development.

Objective of the Project

The objective of this project was to improve the process of generating reports for BI purposes, expedite the processing time, and reduce the development costs comparatively by implementing agile model of development for data warehousing projects. The agile model involves incremental, iterative and collaborative development among cross-functional teams consisting of IT professionals and business users.

Project Questions

The following questions should be answered on successful execution of this project:

1. How successfully will the implementation of agile model handle the shortcomings of the current model of development?
2. Why agile model is suitable for data warehousing projects too, along with the contemporary software projects?
3. How effective has the implementation been for the core competencies of the organization?

Limitations of the Project

A few limitations were identified over the course of the project even though the agile model implementation improved the quality of reports, reduced duration and cost of execution of project successfully. The most significant hurdle in terms of project implementation was identified in the first couple of sprints when the actual transition began, where the team had to implement the agile principles learnt in training sessions to a real time project, which proved to be an overwhelming experience for the team which called for extra training sessions which caused delays in delivering the story points in the beginning.

Definition of Terms

Business Intelligence (BI): Business Intelligence is a technology-driven process for analyzing data and presenting actionable information to help corporate executives, business managers and other end users make more informed business decisions.

Data Warehouse (DW): Data Warehouses are central repositories of integrated data from one or more disparate sources. They store current and historical data and are used for creating analytical reports for knowledge workers throughout the enterprise.

Customer Relationship Management (CRM): Customer Relationship Management is a term that refers to practices, strategies and technologies that companies use to manage and analyze customer interactions and data throughout the customer lifecycle, with the goal of improving business relationships with customers, assisting in customer retention and driving sales growth.

Software Development Life Cycle (SDLC): The software development life cycle is a framework defining tasks performed at each step in the software development process. SDLC is a structure followed by a development team within the software organization. It consists of a detailed plan describing how to develop, maintain and replace specific software.

Extract, Transformation, and Load (ETL): ETL system is a set of processes that clean, transform, combine, de-duplicate, archive, conform, and structure data for use in the data warehouse.

Summary

This chapter gave a brief introduction of the project and discussed the problem statement in detail to get a better understanding of the problem at hand. It also covered the

nature and significance of the problem outlining the intent and need for this project. The objective of the project provided a subtle introduction to the primary goals of the project. The project questions stated in this chapter will be addressed throughout the course of this project, with a detailed analysis of the goals of the project that were accomplished. The limitations of the project discussed the hurdles and constraints that were faced over the course of the project. The definitions of technical terms used in the report provide a better understanding of the project. The next chapter will provide a detailed background and literature review related to the problem and the methodology used in the project.

Chapter II: BACKGROUND AND REVIEW OF LITERATURE

Introduction

This chapter will provide a detailed background and literature related to the problem undertaken, in order to provide better understanding of the goals and objectives stated earlier. This chapter also describes the literature related to the methodology that was implemented in this project.

Background Related to the Problem

The company this project was undertaken at is a global logistics company headquartered in Seattle, Washington. It provides customized solutions to the sophisticated needs of international trade through integrated information systems. The services provided by the company include, supply chain solutions, transportation, customs and compliance, distribution, order management and risk management. The customer base of the company include automotive, telecommunications, high-tech and food industries. For an organization of this stature, it is critical to have optimized functional systems in place to meet the desired requirements of the trading partners. The project scope encompasses critical functional areas within the organization such as transportation, order management, risk management logistics and supply chain.

A typical data warehouse project is expected to perform a number of tasks including creation of a data warehouse, data profiling and modelling, ETL development and unit testing, semantic layer development and QA (Quality Assurance) testing, and report builder development. Each of these tasks possess a level of dependency among themselves (Sifre, 2015). Figure 1 demonstrates Data warehousing and BI architecture that integrates the internal

and external sources of information for analytical uses. The process of data integration and homogenization typically accounts for costs greater than 50% of total project budget (70% to 80% of budget in some cases). The traditional waterfall methodology in use by the company followed data-driven approach that was time-consuming and complex in addition to being expensive. A data-driven approach almost always requires integration and homogenization of most of the data (if not all), before a single BI report can be generated. This approach hence expends majority of the project budget (up to 80%), and takes months and sometimes more than a year to complete integration and homogenization of data, without creating any business value.

This project intended to solve this problem of overly expensive and time consuming data warehouse projects by mitigating the costs associated with bulk integration and homogenization of data, by following a business-driven approach wherein data needed to answer certain business problems only is sourced rather than complete integration of data.

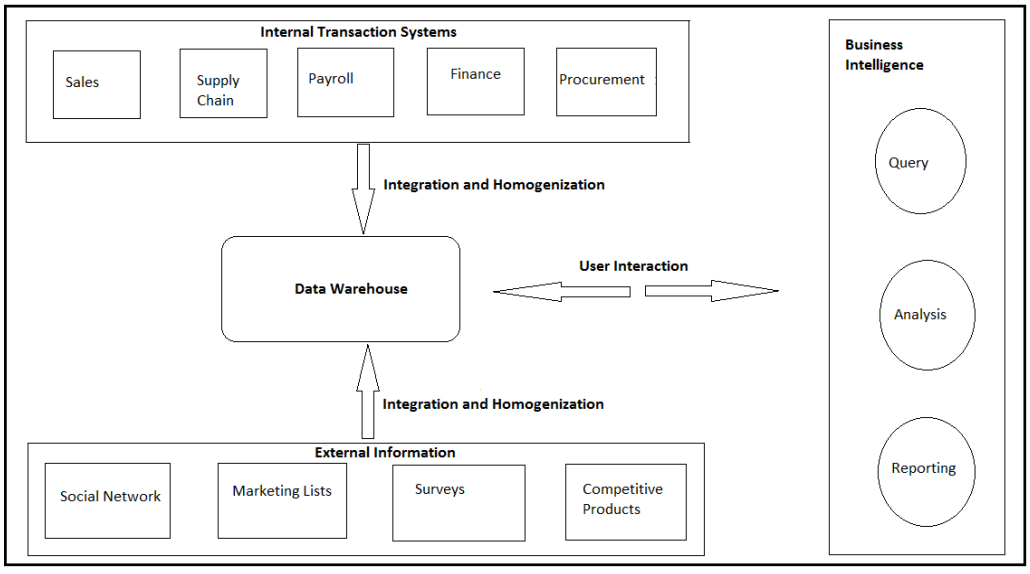

Figure 1. Data warehouse and BI architecture.

According to Hughes (2015), a data warehouse can be of extreme value to the organization if implemented quickly and at a reasonable cost, however, when the traditional waterfall approach of development is pursued, data warehouses take too long and cost too much to build. Waterfall methodologies follow the principle that each phase must be completed before the next phase begins, as the output of one phase becomes the input of next phase and so on (Öztürk, 2013). Figure 2 illustrates the order of phases followed in the waterfall approach of development (Hughes, 2015).

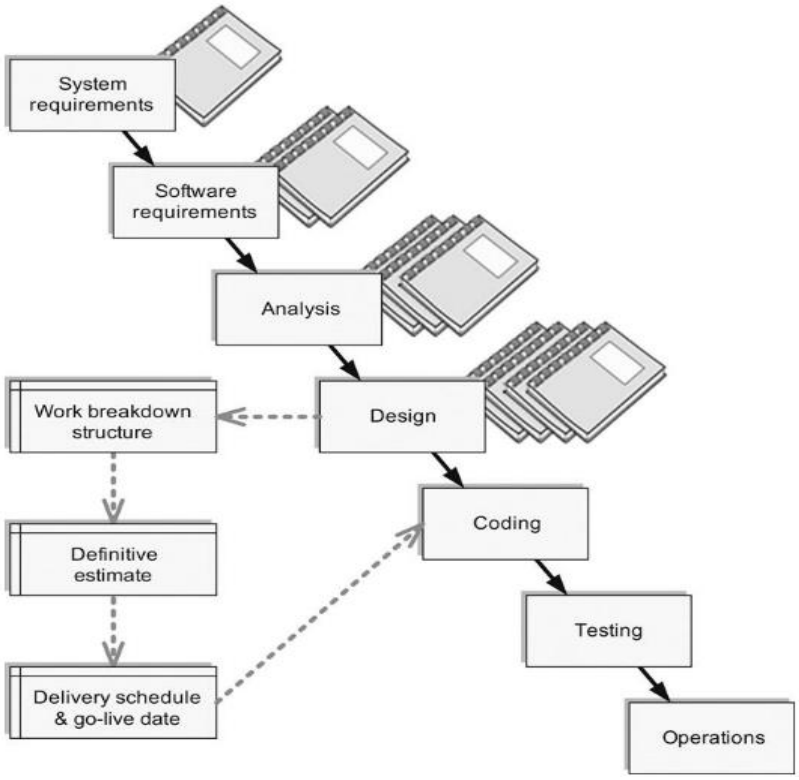

Figure 2. The traditional waterfall model.

As depicted in Figure 2, majority of the development in waterfall approach is spent on paper. The phases of creating requirements documents, external design models, internal design specifications etc. needs to be completed before the actual coding even begins, without

providing any business value to the organization. Moss (2012) states that the highly unreliable estimates and the inability of users to see their system until the phase of accepting testing is reached, contributes to the problems of this model.

Waterfall model applied to DW projects. According to Gallo (2012), the waterfall model when applied to DW projects possess a higher risk factor and the user may not be completely satisfied with the end product. It adheres to the principle of all or nothing, as requirements need to be defined completely before the design phase begins, and in order to begin coding, the design phase must be completed first. The business value is delivered only at the end of the project, making it even more risky. Figure 3 (Gallo, 2012) illustrates the increased risk of failure of waterfall model, as the end product is delivered based on assumptions and interpretations of requirements, which might fail to satisfy the business needs.

Agile model applied to DW projects. Agile model when applied to DW projects show business value with each iteration (Gallo, 2012). The most important tasks of database design (data integration and homogenization), when broken into short delivery cycles (sprints) reduces the level of risk to a great extent, as the work performed by the team is more visible to the business. The use of short sprint cycles also make it possible to deliver business value on an incremental basis. Figure 4 (Gallo, 2012) demonstrates the benefits of applying agile model of development to DW projects like reduced risk of failure and delivery of functional units on an incremental basis.

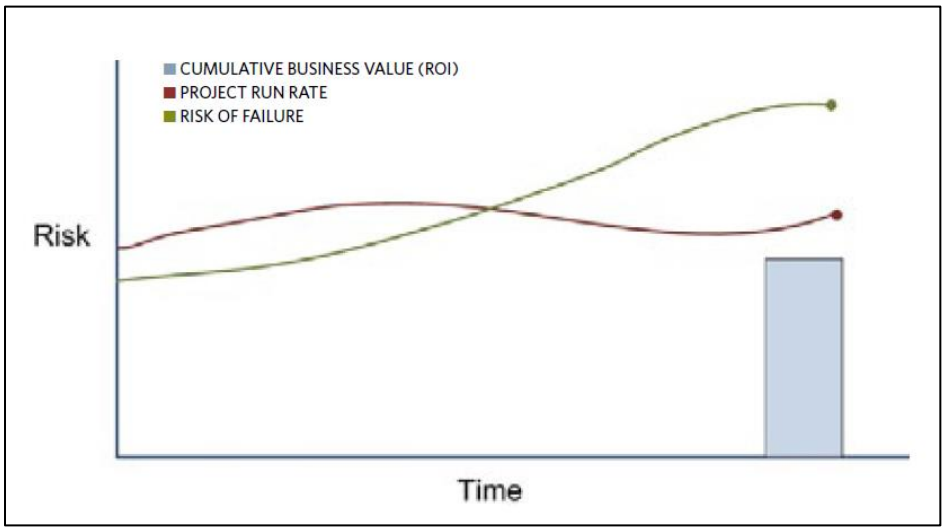

Figure 3. Waterfall model applied to DW projects.

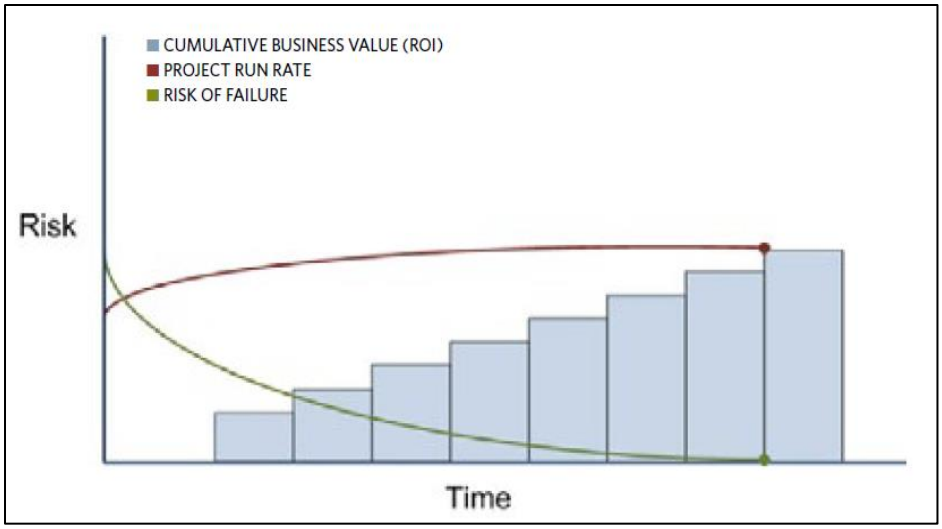

Figure 4. Agile model applied to DW projects.

Benefits of Agile model. Agile models provide flexibility to both the users and developers in many ways. The benefits of agile software development are discussed as follows (Segue Technologies, 2015):

a) *Stakeholder Engagement:* The high degree of collaboration between the project team and client provide a clear of understanding of client requirements and

increases the stakeholders' trust in the team's ability to deliver, owing to the early and frequent releases of working software.

b) *Transparency:* The client gets an opportunity to be present at each stage of software development from iteration planning to review sessions.

c) *Early and Predictable Delivery:* The fixed sprints schedule (1-4 weeks) allow for quick and early delivery of new features, making it possible to beta test the software early.

d) *Predictable costs and Schedule:* The fixed duration of a sprint limits the cost to the specific amount of work that can be performed during that sprint. The cost estimates for each sprint allow for estimating cost of each feature that help in prioritizing of features and iterations needed.

e) *Allows for change:* The team delivers subset of product at the end of each iteration, and the client can make changes in the requirements based on the feature delivered, which will be incorporated in the next iteration.

f) *Focuses on Users:* The use of user stories that focuses on needs of real users, delivering value with each feature.

g) *Focuses on Business Value:* The team understands the important features for client's business and reflects maximum business value in the features delivered.

h) *Improves Quality:* Frequent testing and reviews makes it easier to find defects and fix them quickly. It also provides an opportunity to identify mismatched expectations early.

Literature Related to the Methodology

Software Development Life Cycle (SDLC): Software Development Life Cycle is an all-inclusive working model that defines the sequence of phases and activities that will or should take place during the entire software development process (Öztürk, 2013). There can be several variations to this model based on the activities involved, number of iterations and schedule for product delivery (piece-wise or as a whole). It contains the following major phases and sequence of development activities (Hughes, 2015).

a) *Requirements:* In this phase, an appropriate level of clarity is achieved regarding the functional and performance services needed from the software along with the need for integration between applications.

b) *Analysis:* It this phase, details of application inputs, processes, outputs, and interfaces are generated.

c) *Design:* The physical characteristics of the application are specified in this phase, including major subsystems and their inputs and outputs. The subsystems are further divided into modules or units and detailed logic specifications are prepared for each module.

d) *Coding:* It this phase, the specifications prepared in the design phase are translated into executable systems of software, hardware and communications.

e) *Testing:* After the construction phase is completed, the testing phase begins that ensures the application coding is complete, correct and adhered to the pre-set quality standards.

f) *Operations:* This phase is also referred to as "production". In this stage, the system as envisioned in the requirements phase, is made available for business users.

Types of SDLC models: The SDLC models vary based on different parameters including activities involved, number of iterations and schedule for product delivery. These parameters dictate the project management methodology that will be suitable for the software development project (Tayntor, 2003). The selected model plays a crucial role in communicating a universal understanding of the steps needed for execution of the project, providing a set of milestones to quantitatively measure the project progress, defining roles and responsibilities and dealing with uncertainties (Öztürk, 2013). Some of the frequently used SDLC models are agile, waterfall, prototyping, IID (Iterative and Incremental Development) and spiral SDLCs (Öztürk, 2013).

Agile SDLC model: Collier (2011) defines agile as follows, "Agile is a reserve word meaning a collection of philosophies, practices, behaviors, and techniques that relies on discipline and rigor, but is not heavyweight or overly ceremonious, falling instead in between just enough structure and just enough flexibility." Agile software development model aims to deliver working features of an application at the end of each time-boxed iteration cycles called "sprints". The agile model adheres to the agile manifesto described as follows (Cockburn, 2001),

- Individuals and interactions over processes and tools.
- Working software over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan. (p. 175)

The general agile software approach covers the following principles (Hughes, 2015):

- Progressive decomposition of requirements to generate a simple list of the programming task.
- Co-located, self-organized teams of developers.
- Iterative programming techniques that deliver small slices of the application every couple of weeks.
- Frequent review of those small slices by one or more members of the end-user community.

Figure 5 (Rehani, 2011) illustrates a typical agile development approach where a requirements backlog received from users is divided into sprints (iterations) based on their complexity. The length of each sprint is limited to one or two weeks, during which all development processes including analysis, designing, building and user testing are completed followed by a user demo at the end of each sprint cycle to get user feedback. A production roll out is carried out after every few sprints to deploy the changes to production.

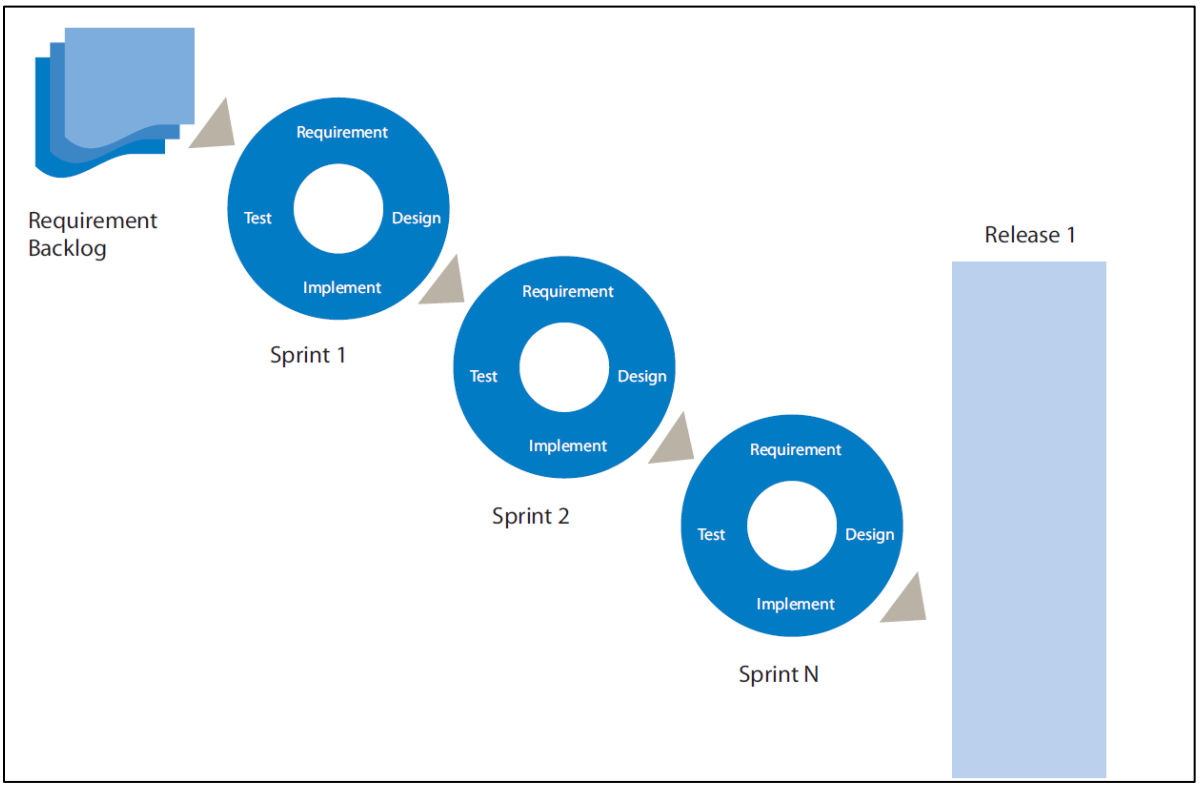

Figure 5. The traditional agile development approach.

The two most popular agile methodologies implemented to software development projects are Scrum and XP. Both these methodologies were developed to aid in delivering quality software in short intervals (Moss, 2012).

Scrum methodology: In the Scrum approach a business partner, who clearly understands the need of business, also known as "product owner" is embedded with the development team. The product owner is responsible for deciding the features required to be added into the application (Hughes, 2015). The development team also selects an individual from within the team known as "scrum master" who facilitates the steps and techniques to be pursued any given day of development. The five steps of iteration followed by scrum teams are discussed below (Hughes, 2015):

a) *Story Conference:* The iteration begins with story conference wherein the team attempts to get required details from user stories that helps estimate the amount of work required for each iteration. A user story describes the requirements in one or two sentences with three components, “Who” (stakeholder who uses the application), “What” (the work that stakeholder wants to be accomplished with the application) and “Why (the benefit derived by the organization from the usage)”.

b) *Task Planning:* In this step the team divides the user stories into developmental tasks that should be completed to add the feature to the application, and assigns labor hours required to complete each story. These developmental tasks encompass all phases of software development including requirements, analysis, design, testing and documentation.

c) *Development:* In this step the team works on the user stories to transform them into working features that are ready to be deployed into production.

d) *User Demo:* In this step the developers demonstrate the working feature to the product owner, who then decides whether to mark each user story as “accepted” or “rejected” based on the capabilities delivered. The accepted stories will then be forwarded for production usage whereas the rejected user stories may be added to the upcoming iterations or be discarded by the product owner.

e) *Iteration Retrospective:* In this step the team reflects on the policies that worked for the iteration and also the techniques that did not work and might need to be changed in the next iteration, to ensure quick delivery and high-quality of features.

Extreme Programming methodology (XP): XP is one of the agile methodology that focuses on the small team and provides various ways to give direction to the development tasks to help build a working and validated application (Hughes, 2015). The general XP engineering practices are discussed as follows (Beck 1999).

a) *Simple System Design:* This practice focuses on the designing of only what is required to support the current functionality.

b) *Test-first coding:* In this practice unit tests are written before the code is written in order to have a clear understanding of the expected functionality.

c) *Continuous integration:* In this practice the code is built and tested at regular interval to ensure software is stable and high quality.

d) *Refactoring:* This practice allows improving of class architecture and design as required to assist new functionalities.

e) *Pair programming:* Pair programming refers to the practice of doing work in teams of two or three, performing all functions including data analysis, data modelling and programming.

Scrum and XP share similar ideologies except that Scrum provides an overall managerial outlook of the product development whereas XP techniques streamline the programming tasks to deliver better quality code for features. According to Schwaber and Mar (2002), scrum and extreme are complementary practices, when combined together provide a means of delivering high quality functionality that caters to the customer's needs.

There are several metrics that can be employed to effectively measure the performance of agile methodologies:

1. *Scrum Velocity:* Scrum velocity is measured as sum of story points completed, which measures the amount of work completed in one scrum. Velocity is a very important metric as it enforces release planning and schedule updates. Figure 6 (Rehani, 2011) shows a general scrum velocity chart depicting the total number of story points completed in each sprint respectively.

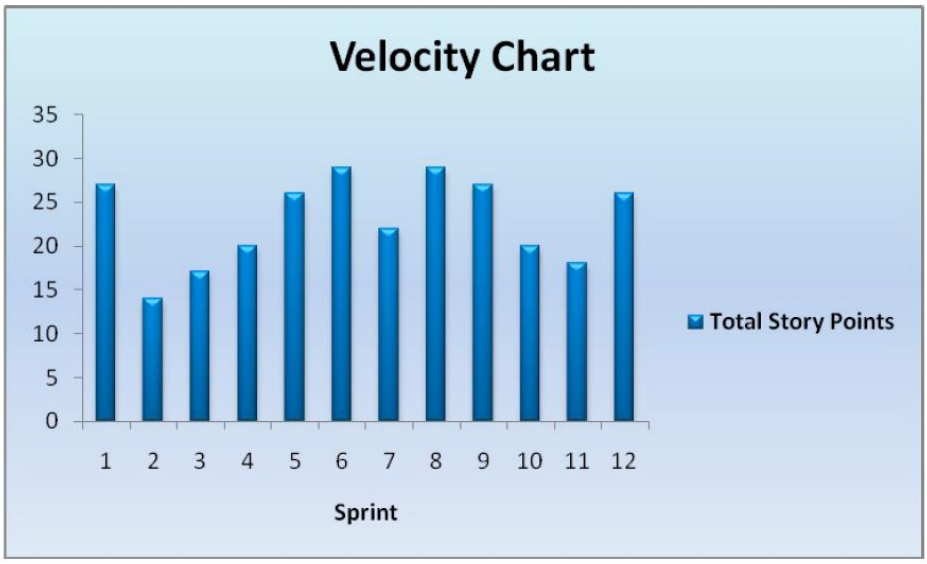

Figure 6. Scrum velocity chart.

2. *Sprint Burn Down:* The sprint burn down chart is the measure of amount of work to be completed over the sprints. It can also be calculated each day to keep track of the work to be done in each sprint individually and deduce if any changes are required in the plan of action to complete the iteration on time.
3. *Release Burn Down:* This metric illustrates the burn down across the sprints that measure the performance improvements with each sprint.

4. *Scrum Task Board:* Scrum board is created with a normal white board and index cards tacked on a large wall of the project room. It consists of horizontal swim lanes that are dedicated to each user story, wherein the task cards travel to their right as each task related to the story is completed by the developers. Figure 7 (Hughes, 2015) demonstrates a sample scrum task board as it would appear in an iteration. Each developer may take the ownership of the task card by placing their initials on it and select the task he is qualified to work on.

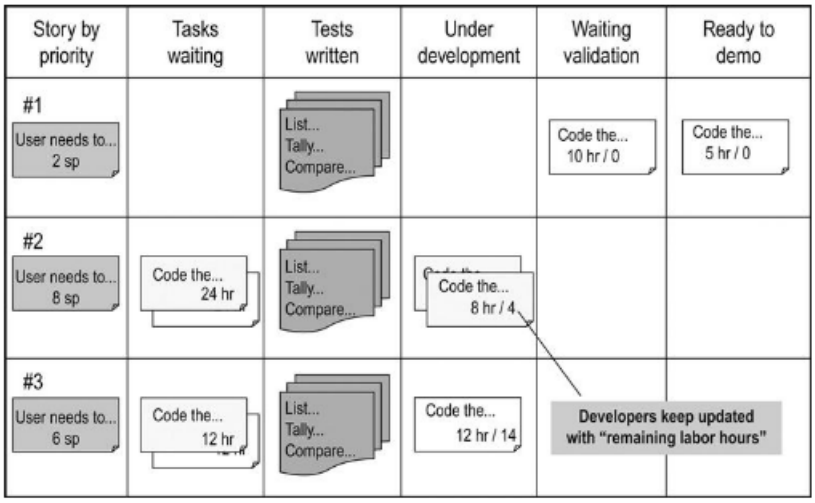

Figure 7. A sample scrum task board.

Summary

This chapter described in detail the background and literature related to the problem. It also discussed in detail the literature related to the methodology that was implemented in the project, that was a combination of both components of agile SDLC, Scrum and XP, in an to attempt to solve the problem. The next chapter will discuss in detail the project methodology specific to the project tools, timeline and budget.

Chapter III: METHODOLOGY

Introduction

This chapter discusses in detail the project methodology and design of study implemented in the project. It also describes the data collection and analysis tools used to execute this project. Also discussed are the project timeline, budget and constraints that dictated the progression of the project.

Design of the Study

The design of study followed for this project was a combination of quantitative and qualitative analysis. This approach was appropriate as the goal of this project was to improve the performance of an EDW by migrating from the traditional waterfall software development model that was being implemented, to the agile software development model, and this improvement in performance could be measured by analysis of the data collected on the execution of the project. Also, a customer satisfaction survey allowed for objective qualitative analysis of the same. The project design and structure of methodology are outlined as follows:

Pilot: In this phase a project proposal was created and presented to the higher management to obtain the approval for desired budget for the project. The document illustrated the benefits of implementing agile model to data warehousing and presented proof of concept for the same by executing a pilot test.

The pilot test involved two sprint cycles for a small business requirement by the sales department. Each sprint cycle lasted for 2 weeks and delivered the requirements at the end of each iteration. The pilot tested successfully and the required budget was approved.

A project manager with experience in working with agile model of software development was identified. The project manager was also declared the Scrum Master to ensure that the project execution followed agile principles and directed their day to day activities to follow the agile approach. The team consisted of five members including the project manager, system architect, data architect and two developers.

Requirements: The main requirement of the project was to create Business Intelligence reports for higher management for their new services. The different sources of data for the reports were identified as follows:

1. Budget data–A financial document created in Excel spreadsheets for various services by the higher management.
2. Forecast data–A document created in Excel spreadsheets by middle managers that contains data required to predict the future sales.
3. Customer Relationship Management (CRM) data–This data was stored in a cloud server.
4. Survey data–This contains the survey data collected from various customers regarding the services provided by the organization and was stored in a cloud server.

Project Life Cycle: The project started with a kick off meeting held between the stakeholders, product owner and the team to make important project decisions regarding the priority of features and order of deliverables required. After careful consideration it was decided that the project would be started with the budget data. Based on the requirements and

complexity of the project, sprint size was decided as two weeks initially, but may be increased or decreased as per the requirements.

Sprint: Each sprint began with a sprint planning meeting wherein the work was divided into tasks and pasted on the scrum task board. Each developer then added their respective initials on the specific tasks they wanted to perform and took ownership of the tasks. At the end of each sprint cycle, a user demo was held wherein the team would show the deliverable to the product owner, who would accept or reject the feature, based on its capability.

Figure 8 demonstrates a sample user story document that was used in this project that described the user stories that needed to be completed in that respective sprint (sprint 14). The template contained important details like the story description, estimated time of completion, acceptance criteria for the story to be considered as complete.

SPRINT - USER STORY DETAILS							
Sprint ID	**ID**	**Story Name/Description**	**Estimate**	**Story**	**Acceptance Criteria**	**Notes/Links**	**PWA**
14.1	249	**UCN Manager - Customer Org Loader testing**	8	As a DIJST technical manager, I want to ensure that the customer org loader checks for existing customer orgs, So that it does not constantly update the orgs with every event	*All Necessary QA activities performed *QA Approved	Done 8/19	Customer Organization -Release 1.0
14.2	257	**Plan for Customer Org deployment Activities**	2	As a migration program manager, I want to ensure that all teams involved in the customer org "go live" deployment are well aware of the tasks and dependencies, so that we can easily communicate any issues on deployment day	*Team has fully reviewed timeline and provided estimates for task duration *if needed, new tasks are added to the timeline *tasks are assigned to individuals and backups are listed *Steps to complete tasks are documented in a common place	*This deployment will happen during migration planning, some people may not be available Done 8/29	Customer Organization -Release 1.0

Figure 8. Sample user story requirements document.

Each user story was given a name based on the requirements, and is described in detail in the story column which followed the structured format that answers three main questions, “who is requesting the requirement?”, “what is the actual requirement?”, and “why should this requirement be fulfilled?” This structure simplified the requirements for the team and aided in

clear understanding of requirements. As shown in Figure 8, each user story had its own set of acceptance criteria that needed to be satisfied in order for the user story to be considered as completed.

Each story was given an estimate or story points that are Fibonacci numbers, with the lowest number assigned to a user story that has few dependencies and was considered easy by the team. The estimate/story points (Fibonacci number) increased with the increase in the number of dependencies and the complexity of the task.

Data Collection

Data collection tools and techniques act as a yard stick to evaluate the successful implementation of a project against the pre-defined performance metrics and validate that all the project objectives have been adequately filled. The data collection tools employed for this project involved tabulating the sprint burndown points in each sprint and calculating the sprint velocity which reflected the success rate of the agile implementation and predicted the number of sprint points that can be completed in the successive sprints based on the average sprint velocity. The sprint velocity is an important metric to measure the successful implementation of agile methodology as each sprint cycle is expected to deliver a feature requested by the business/product owner and sprint velocity measures the rate of delivery of the requested features.

The next phase of data collection involved tabulating the total time (in hours) dedicated by the team to successfully complete the project and compare it with the total number of hours that were required to complete a similar project by the team in the past by implementing traditional waterfall model.

The qualitative data collection tools employed were feedback surveys (see the Appendix) conducted with the various business units to gauge their satisfaction level with the performance of the agile software development approach versus the traditional waterfall development approach for EDW projects.

The sprint burn down points were tabulated using Microsoft Excel, an extremely useful spreadsheet that offers various features like graphing tools, pivot tables, etc. Figure 9 demonstrates a sample sprint burndown sheet that tabulated the number of points committed, remained, and burned on each day of the sprint. The sprint burndown sheet shown in Figure 9 was tabulated for a 2-week long sprint and illustrates the points that were burnt on each day of the sprint. The data from the sheet was then represented graphically in the form of a sprint burndown chart as shown in figure 10, which clearly represented the progress of the team as a whole by indicating the extra pull in points that were completed in a given sprint.

	Days	Total Remaining Points	Total Remaining Committed Points	Total Remaining Pull-in Points	Total Points Burned
thurs	0	62	62	0	0
fri	1	62	62	0	0
mon	2	62	62	0	0
tues	3	64	62	2	0
wed	4	72	57	15	5
thurs	5	70	57	13	7
fri	6	70	57	13	7
mon	7	70	57	13	7
tues	8	62	49	13	15
wed	9	36	23	13	41
thurs	10	31	18	13	31

Figure 9. Sprint burndown sheet.

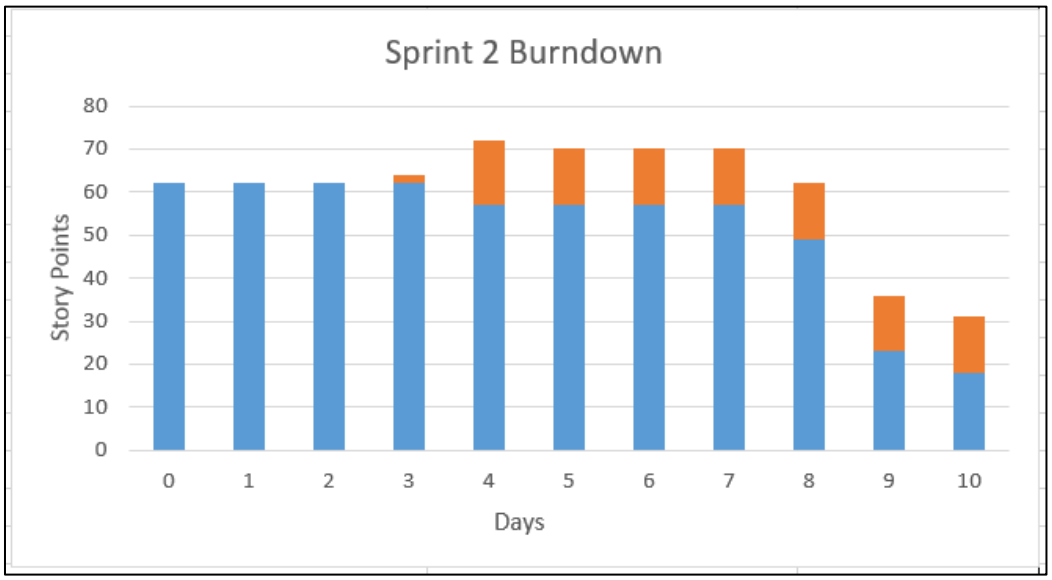

Figure 10. Sprint burndown chart.

Data Analysis

The data collected in the form of sprint burndown points was then analyzed by creating a sprint velocity chart that showed the performance of a team and predicted the scope that can be delivered by the team, making it a suitable metric to deduce the success of agile implementation. For instance, releasing a deliverable (small feature) at the end of each sprint cycle is a measure of project's value stream, and, sprint velocity validates the ability of the team to deliver the said requirements.

The second part of the data analysis was a comparison of the time taken to complete the project with agile model implementation to that of the traditional waterfall model implementation, using historical data.

The final step in the data analysis for this project involved quantifying the results obtained from the business user satisfaction/feedback surveys to obtain a holistic view of the success of the overall project from the internal customers' perspective.

Budget

The overview of the costs associated with the project are shown in Table 1. Table 2 provides the details of breakdown by work effort.

Table 1

Project Budget Overview

	Expense
Labor Cost	$308,295
Travel	$25,015
Total Cost	**$333,310**

Table 2

Project Budget–Work Effort Breakdown

Phase	**Hours**	**Cost**
Planning	24	$ 6,120
Sprint 1	80	$ 20,400
Sprint 2	80	$ 20,400
Sprint 3	80	$ 20,400
Sprint 4	80	$ 20,400
Sprint 5	80	$ 20,400
Sprint 6	80	$ 20,400
Sprint 7	80	$ 20,400
Sprint 8	80	$ 20,400
Sprint 9	80	$ 20,400
Sprint 10	80	$ 20,400
Sprint 11	80	$ 20,400
Sprint 12	80	$ 20,400
Sprint 13	80	$ 20,400
Sprint 14	80	$ 20,400
Post-deployment Support	65	$ 16,575
Total	**1,209**	**$ 308,295**

Timeline

The proposed timeline for this projects is defined as follows in Table 3.

Table 3

Project Timeline

Task	Duration	Start	Finish
Planning	3 days	02/29/2016	03/02/2016
Sprint 1	10 days	03/03/2016	03/17/2016
Sprint 2	10 days	03/17/2016	03/31/2016
Sprint 3	10 days	03/31/2016	04/14/2016
Sprint 4	10 days	04/14/2016	04/28/2016
Sprint 5	10 days	04/28/2016	05/12/2016
Sprint 6	10 days	05/12/2016	05/26/2016
Sprint 7	10 days	05/26/2016	06/09/2016
Sprint 8	10 days	06/09/2016	06/23/2016
Sprint 9	10 days	06/23/2016	07/07/2016
Sprint 10	10 days	07/07/2016	07/21/2016
Sprint 11	10 days	07/21/2016	08/04/2016
Sprint 12	10 days	08/04/2016	08/18/2016
Sprint 13	10 days	08/18/2016	09/01/2016
Sprint 14	10 days	09/01/2016	09/15/2016
Post Deployment	8 days	09/16/2016	09/27/2016
Draft Capstone Final Project Report	18 days (including weekends)	09/16/2016	10/3/2016
Capstone Project Defense	0 days	11/16/2016	11/16/2016

Summary

This chapter described the data collection tools and techniques utilized for this project, and briefly discussed the quantitative and qualitative data analysis techniques employed in this project. The details regarding the project timeline proposed for this project was also provided in this chapter to gain a better understanding of the project structure, and the project budget breakdown was included to provide an overview of the costs associated with the

project. The next chapter will focus on the presentation and detailed analysis of data that was collected as a part of this project.

Chapter IV: DATA PRESENTATION AND ANALYSIS

Introduction

This chapter will provide a structured presentation of the data that was collected as part of this project, followed by the analysis of the data by utilizing the data analysis tools and techniques discussed earlier to help answer the project question stated in the beginning of the project and to compile results and recommendations.

Data Presentation

The data was collected in each sprint in the form of sprint burndown points to determine the number of user story points covered in each sprint. The data for all the sprints was tabulated as shown in table 4, to calculate the average sprint velocity for the project. This data was then used to calculate the sprint velocity as shown in Table 5.

Table 4

Sprint Burndown Points for All the Sprints

Sprint	Thu	Fri	Mon	Tue	Wed	Thu	Fri	Mon	Tue	Wed	Thu	**Total points**
1	0	0	0	0	5	7	7	7	15	41	31	**113**
2	0	0	0	0	5	7	7	7	15	41	31	**113**
3	0	0	0	7	8	8	8	13	13	13	26	**96**
4	0	13	14	22	22	22	22	38	59	67	77	**356**
5	0	0	0	0	0	0	0	0	0	0	3	**3**
6	0	0	0	0	0	0	0	5	26	26	55	**112**
7	0	0	0	0	0	0	0	0	3	8	29	**40**
8	0	0	0	13	13	13	47	47	47	47	47	**274**
9	0	0	0	0	21	21	21	21	34	42	55	**215**
10	0	0	0	0	0	0	0	0	0	24	45	**69**
11	0	0	0	5	5	5	10	10	18	31	52	**136**
12	8	8	8	8	26	26	31	31	36	49	49	**280**
13	8	10	21	10	21	49	49	8	21	49	49	**295**
14	0	8	8	8	8	8	8	10	21	49	49	**177**

Table 5

Sprint Velocity Sheet

Sprint	**Sprint Velocity**	**Average Completed Stories Velocity**
1	27	27.0
2	46	36.5
3	27	33.3
4	79	3.0
5	3	36.4
6	55	39.5
7	29	38.0
8	47	39.1
9	55	40.9
10	45	41.3
11	52	42.3
12	47	42.7
13	55	43.6
14	21	42.0

The second set of data collected was the total project duration (in hours) to complete the project by implementing agile model of development as shown in Table 6.

Table 6

Time Taken to Complete the Project in Agile Implementation

Phase	**Hours**
Planning	24
Sprint 1	80
Sprint 2	80
Sprint 3	80
Sprint 4	80
Sprint 5	80
Sprint 6	80
Sprint 7	80
Sprint 8	80
Sprint 9	80
Sprint 10	80
Sprint 11	80
Sprint 12	80
Sprint 13	80
Sprint 14	80
Post-deployment Support	65
TOTAL	**1,209**

The timeline for a similar project conducted last year which implemented the traditional waterfall model of development was also collected from the team to compare the time frames for execution of the project using each development approach. Table 7 shows the total project duration (in hours) to complete the project by implementing waterfall model of development.

Table 7

Time Taken to Complete the Project with Waterfall Model Implementation

Phase	**Hours**
Program Management	174
Scoping	224
Design/Build	750
Test	422
Deploy	86
Post-Deployment support	76
TOTAL	**1,732**

The final set of data was collected from the feedback survey distributed to all the departments in the company that were involved or impacted in any way. The participants were given a questionnaire and were asked to rate 5 statements regarding the project performance under agile approach of development on a 5 point tiered scale ranging from strongly agree to strongly disagree. The following statements were presented in the survey:

1. "Agile model has significantly reduced the amount of time taken to generate the reports"
2. "All the business requirements have been fulfilled by using the agile model in this project"
3. "The active involvement of a business person (product owner) made it easier to understand requirements"
4. "I prefer the traditional waterfall model of development for EDW projects"
5. "Agile model allowed for quick changes in development based on the changes in requirements"

The table provides the summary of the survey results. The total number of responses received were 17 and the summary of results is provided in the Table 8:

Table 8

Summary of Survey Results

Department	Response Date	Question 1	Question 2	Question 3	Question 4	Question 5
Sales	09/25/2016	Strongly Agree	Agree	Neutral	Strongly Disagree	Agree
Business Intelligence	09/25/2016	Agree	Agree	Neutral	Disagree	Agree
Sales	09/25/2016	Agree	Neutral	Disagree	Agree	Agree
Finance	09/25/2016	Neutral	Disagree	Neutral	Disagree	Strongly Agree
Visibility	09/25/2016	Disagree	Strongly Agree	Strongly Agree	Strongly Agree	Neutral
Business Intelligence	09/25/2016	Strongly Disagree	Strongly Disagree	Disagree	Agree	Disagree
Business Intelligence	09/25/2016	Strongly Agree	Agree	Agree	Neutral	Strongly Disagree
CRM	09/25/2016	Agree	Strongly Agree	Agree	Strongly Disagree	Strongly Agree
Finance	09/26/2016	Neutral	Agree	Agree	Disagree	Strongly Disagree
Visibility	09/26/2016	Strongly Agree	Strongly Agree	Agree	Strongly Disagree	Agree
CRM	09/26/2016	Agree	Disagree	Agree	Disagree	Agree
Sales	09/27/2016	Disagree	Agree	Agree	Neutral	Agree
Visibility	09/27/2016	Neutral	Neutral	Strongly Agree	Neutral	Agree
CRM	09/27/2016	Agree	Strongly Agree	Disagree	Disagree	Agree
CRM	09/27/2016	Strongly Agree	Agree	Agree	Strongly Disagree	Agree

Data Analysis

Sprint velocity analysis: The sprint velocity chart was drafted based on the sprint velocity sheet as shown in Figure 11. The analysis is summarized as shown in Table 9.

Table 9

Summary of Sprint Velocity

Sprint velocity range (story points)	Percentage of occurrence
0-10	7.14%
11-20	0%
21-30	28.57%
31-40	0%
41-50	28.57%
51-60	28.57%
61-70	0%
71-80	7.14%

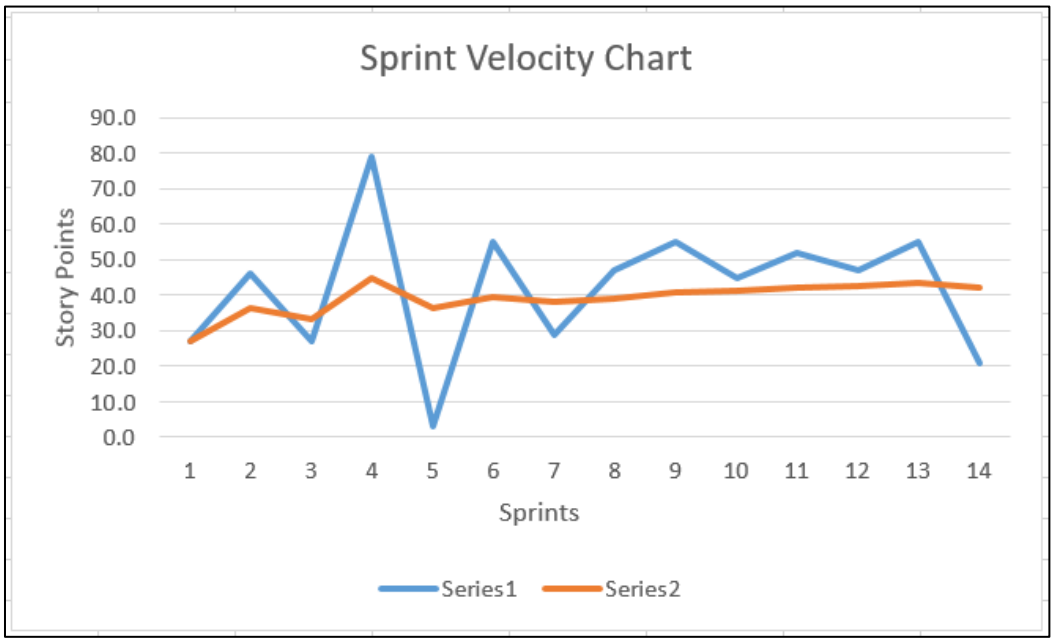

Figure 11. Sprint velocity chart.

Based on the data analysis, the following observations can be made:

1. On an average, more than 40 story points were completed in 65% of the sprint cycles.
2. 20 to 30 story points were delivered in 28.57% of the sprint cycles.
3. Less than 10 story points were completed in 7% of the sprint cycles.
4. More than 7 story points were completed in 7% of the sprint cycles.
5. The average sprint velocity of the team is 42 story points per sprint cycle.

Timeline analysis/comparison: The comparison of timelines of both waterfall and agile approach implementation on similar data warehousing projects showed a significant amount of work hours saved by the latter to complete the project, saving the organization a labor cost for those hours. The analysis can be summarized as shown in Table 10:

Table 10

Summary of Comparison of Timeline

	Waterfall Model Implementation	**Agile Model Implementation**
Total No of Hours	1732	1209
Total Labor Cost	$441,660	$308,295

Based on the data analysis the following observations can be made:

1. The project duration was reduced by 523 work hours by the implementation of agile model of development.
2. The agile model implementation saved a significant amount of $133,365 in labor cost.

User satisfaction survey: The responses received through the distribution of feedback survey questionnaires for agile model implementation to data warehouse projects can be summarized as follows in Table 11:

Table 11

Survey Response Sheet

	"Agile model has significantly reduced the amount of time taken to generate the reports"	**"All the business requirements have been fulfilled by using the agile model in this project"**	**"The active involvement of a business person (product owner) made it easier to understand requirements"**	**"I prefer the traditional waterfall model of development for EDW projects"**	**"Agile model allowed for quick changes in development based on the changes in requirements"**
Strongly Agree	26.7%	26.7%	13.3%	6.7%	13.3%
Agree	33.3%	40%	46.7%	13.3%	60%
Neutral	20%	13.3%	20%	20%	6.7%
Disagree	13.3%	13.3%	20%	33.3%	6.7%
Strongly Disagree	6.7%	6.7%	0%	26.7%	13.3%

Based on the survey results the following observations can be made:

1. 60% of the respondents either agree or strongly agree that agile approach of development has significantly reduced the time taken to generate final reports.
2. 66.7% respondents either agree or strongly agree that all the business requirements have been fulfilled by using the agile approach of development.
3. 60% of the respondents either agree or strongly agree that the active involvement of a business person is feasible (product owner) made it easier to understand requirements

4. 60% of the respondents either disagree or strongly disagree with the premise that waterfall methodology was better.
5. 73.3% of the respondents either agree or strongly agree that agile approach allowed change of requirements as and when required.

Summary

This chapter discussed in detail the data that was collected and the various tools that were used to analyze the collected data. The next chapter will focus on the results of the analysis and will conclude the project report with recommendations for similar projects in the future.

Chapter V: RESULTS, CONCLUSION, AND RECOMMENDATIONS

Introduction

This chapter discusses in detail the results obtained on execution of the project, after the data collection and data analysis was completed. It also covers the recommendations made based on the important lessons learned as part of the project execution.

Results

The overall methodology employed for the completion of this project was agile model of software development. The agile model of development follows an iterative, incremental, and collaborative development among cross-functional teams consisting of IT professionals and business users. In this model of development the tasks are broken down into user stories that are completed within a fixed-time cycle called sprint, and a sub module of the feature is delivered at the end of each sprint for the business person to assess its features. This involvement of the user throughout the execution of the project allows for generation of better quality reports and increases user adoption as the user is part of the development process and can quickly identify the changes that might be required based on the modules delivered at the end of each sprint.

Following is a summary of the results obtained from the study of this project:

1. The sprint velocity chart shows that the average number of user story points delivered by the agile development team in each sprint was more than 40 which is indicative of stable team development which resulted in a decent number of features being delivered at the end of each sprint cycle, making it possible to

generate better quality and useful business reports in a short span of time, with the involvement of a business person throughout the execution of the project.

2. The reduction in the number of work hours and labor costs associated with the project by the agile model implementation, validates the success of agile model in reducing time and budget of data warehousing projects.
3. Based on the results of the feedback survey the overall consensus was that the majority of the stakeholders were satisfied with the performance of agile model of development when applied to data warehousing project.

The project questions stated at the beginning of the project can be answered as follows:

1. **How successfully will the implementation of agile model handle the shortcomings of the current model of development?**
    a. The shortcomings of the traditional methodology, like the inability to deliver the requirements on time and not being able to update the model based on the changes in user requirements, were mitigated by the implementation of agile model, as the business received their deliverables in a short period of time (at the end of each sprint), and any changes requested by the users were incorporated in the following sprint making it possible to integrate a more flexible approach to development.
2. **Why Agile is suitable for Data warehousing projects too, along with the contemporary software projects?**

a. The agile method of development was initially designed to cater to software development projects that focused on creating stand-alone systems unlike EDW projects that are highly interdependent and have a complex architecture.

b. This project implemented the agile methodology to EDW project and successfully fulfilled the project objectives. EDW projects can be divided into a series of software releases by creating a high-level roadmap of the project that presents an overall understanding of the resources, cost, schedule, risks and assumptions for the BI application. This roadmap can thus guide the scope and sequence of the software releases proposed and determine the possibility of achieving the goals within the time-box, making agile suitable for EDW projects.

3. How effective has the implementation been for the core competencies of the organization?

a. Data warehouses encompasses all the major functional units of an organization in terms of movement of data both from internal and external resources to integrate and analyze business strategies that support the core competencies of the organization. Therefore, the improvements in the data warehousing projects in terms of faster deliverables, integration of changes in user requirements, and comparative reduction in cost can be directly correlated to the improvement of the company's core competencies.

Conclusion

Data warehouse projects often turn out to be expensive, time consuming and fail to deliver the requirements as expected. Owing to the size of the data warehouse and its complex design, it becomes difficult to model, transform and load data into a data warehouse for business utilization. Agile model of development addresses this issue by breaking the tasks into small time cycles or sprints that focus on working with the data required for the particular sprint at a time keeping a holistic view of the project requirements in its entirety.

Based on the results of the project study it can be deduced that implementation of agile approach of development to data warehousing projects reduces the time taken and costs associated to generate useful business reports significantly. It also allowed for generation of better quality of reports owing to the active involvement of a business person throughout the project life cycle, giving useful inputs and validating the deliverables at the end of each sprint, thereby mitigating the problems inherent with the traditional waterfall model of development.

Recommendations

The process of replacing a traditional model of development with a new model of development which follows contrasting principles and techniques can be quite an overwhelming experience. This is especially true when the project is associated with multifunctional units of the organization like an Enterprise Data warehouse. The following are the recommendations to improve the execution of similar projects in future:

1. Automated testing development that tests the code rigorously and highlights the problem areas in the code as early as possible should be followed, so that the bugs

are found and fixed quickly, and the features are delivered within the time-boxed sprint.

2. A clear consensus existed among the team members regarding the sprint size of 2 weeks being too small for developing and testing, hence an extension of at least one week in the sprint size (3 weeks sprint size) would have a positive impact on the execution of the project, especially for the teams implementing agile model of development for the first time.
3. It is recommended to focus on the basic agile principles at all times, and not deviate from following the agile techniques owing to the complexity of the problem. Effective communication plays an important role in any agile project and it is crucial to have clear communication between business and development at all stages of the project to prevent any complexities that tempt the team to deviate from the basic agile principle.

References

Beck, A. K. (1999). *Extreme programming explained: Embrace change.* Boston, MA: Addison-Wesley.

Cockburn, A. (2001). *Agile Software Development.* Boston, MA: Addison-Wesley.

Collier, K. (2011). *Agile analytics: A value-driven approach to business intelligence and data warehousing*. Upper Saddle River, NJ: Addison-Wesley.

Gallo, J. (2012). Applying agile methods to data warehouse projects. *BI trends+strategies,* (1), 7-11. Retrieved from http://searchbusinessanalytics.techtarget.com/feature/Applying-agile-methods-to-data-warehouse-projects.

Hughes, R. (2015). *Agile data warehousing for the enterprise.* Waltham, MA: Morgan Kaufmann Publications.

Inmon, W. H. (1993). *Building the data warehouse.* New York, NY: John Wiley & Sons.

Moss, L. (2012). An agile approach to enterprise data warehousing and business intelligence. *Technology Transfer.* Retrieved from http://www.technologytransfer.eu/article/104/2012/9/An_Agile_Approach_to_Enterprise_Data_Warehousing_and_Business_Intelligence.html.

Öztürk, V. (2013). Selection of appropriate software development life cycle using fuzzy logic. *Journal of Intelligent & Fuzzy Systems, 25*(3), 797-810.

Rehani, B. (2011). Agile way of BI implementation. In *2011 Annual IEEE India Conference*. IEEE Publications.

Schwaber, K., & Mar, K. (2002). Scrum with XP. *InformIT*. Retrieved From http://www.informit.com/articles/article.aspx?p=26057&seqNum=3.

Segue Technologies. (2015). *8 benefits of agile software development* [White Paper]. Retrieved from http://www.seguetech.com/8-benefits-of-agile-software-development/.

Sifre, W. (2015). Can data warehouse development follow an agile/scrum SDLC? [Web log post]. Retrieved from http://www.allegient.com/can-data-warehouse-development-follow-an-agilescrum-sdlc/.

Tayntor, C. (2003). *Six Sigma software development.* Boca Raton: Auerbach Publications.

Appendix: Customer Satisfaction Survey Questionnaire

User Satisfaction Survey

"Agile model has significantly reduced the amount of time taken to generate the reports"

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

"All the business requirements have been fulfilled by using the agile model in this project"

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

"The active involvement of a business person (product owner) made it easier to understand requirements"

- Strongly Agree
- Agree
- Neutral

- Disagree
- Strongly Disagree

"I prefer the traditional waterfall model of development for EDW projects"

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

"Agile model allowed for quick changes in development based on the changes in requirements"

- Strongly Agree
- Agree
- Neutral
- Disagree
- Strongly Disagree

Department:

Email:

_____

SUBMIT