Spring 2016

Information Systems Development Methodologies Transitions: An Analysis of Waterfall to Agile Methodology

Oriana Karina Eason
University of New Hampshire - Main Campus, okj2@wildcats.unh.edu

Information Systems Development Methodologies Transitions: An Analysis of Waterfall to Agile Methodology

Keywords
agile, waterfall, transition, methodology

Subject Categories
Business

Information Systems Development Methodologies Transitions: An Analysis of Waterfall to Agile Methodology

Oriana Eason

Advised By: Khole Gwebu

The University of New Hampshire

Table of Contents

1.0 Introduction

1.1 Background

There is always a newer and better way to create something. As a society, this is part of what we do; we create, we change, we evolve. Companies do the same. In order to succeed, companies constantly strive to create and change and evolve. In the information systems (IS) domain, creation and innovation are key components to success. Software Development Methodologies have been evolving since their creation. According to Hoffer, George and Vlacich (2006) software development methodology is defined as "a standard process followed in an organization to conduct all the steps necessary to analyze, design, implement, and maintain information systems [1]. This definition implies that software development is characterized by a time element in which various tasks are assigned. Consequently, there are multiple ways in which tasks can be allocated and organized over the time necessary to develop the information system. The tasks themselves can vary in time and specificity.

Since their emergence in the 1960's, software development methodologies have evolved [2]. Today, there are well over 15 software development methodologies that exist. For examples, there are Waterfall Methodology, Agile Software Development Methodology, Spiral Methodology, Dynamic Systems Development Model Methodology, Extreme Programming Methodology, Feature Driven Development Methodology, Joint Application Development Methodology, Lean Development Methodology, Rapid Application Development Methodology, and Rational Unified Process Methodology.

Perhaps one of the earliest methodologies was the Waterfall Methodology, which has been developed and adapted in many different ways. But over time some of the weaknesses of the Waterfall methodology have become apparent forcing companies to turn to or develop new

methodologies. Nevertheless, adapting or changing to a new methodology is not without problems. Oftentimes such change is time consuming, difficult and can result in software development failure.

1.2 Research Justification

This research seeks to identify some of the potential challenges associated with transitioning from one software methodology to another. Given that software development involves both technical and human aspects, these challenges are organized into two broad categories (i) technical challenges and (ii) human challenges. This research is important because by identifying and understanding these challenges companies may be better able to manage future migration to new software development methodologies which could ultimately result in significant time and cost savings. In this thesis the focus will be on the change is transitioning between Waterfall software development methodology to Agile software development methodology. The reason for focusing on the transition between these two methodologies is because there seems to be a trend occurring in the business world where companies are making this transition. There is a great amount of research on the two types of methodologies but less on the effects of transitioning between them and that is what this research is looking to explain.

1.3 Organization of the Thesis

The remainder of the thesis is organized as follows. The next section reviews the extant literature on software project success and failure. The goal of this section is to initially understand why software projects fail in general as lessons from extant studies may contribute to the understanding of the challenges associated with methodology transitions. Next, the thesis considers the role of human and technology factors in the transition between software development methodologies. Thereafter a case study a company engaged in a transition from the

waterfall methodology to an agile methodology is presented. Finally, conclusions from the case study are drawn, limitations of the research are highlighted and directions for future research are suggested.

2.0 Literature Review2.1 Software Project's Success and Failure

It is a well-researched and well-known fact that about 70% of all IT and IS projects fail [3]. Failure does not only occur when a project becomes abandoned. This research defines a project failure as a project that was not finished on time, was over budget and/or is not delivered with the functionality that was originally agreed upon when the project began. On the other hand, success is defined by a project that is on time, under budget and is fully functional will all aspects of the system working the way that was originally agreed upon. It is difficult for a project to be deemed fully successful due to the nature of the definition of success. But there are many IS projects that will are partially successful because they meet more than one of the requirements for success, but if there is more than one requirements that is not met, then the project failed.

We know projects fail for many different reasons. Field states that, "projects fail too often because the project scope was not fully appreciated and/or user needs not fully understood" [4]. Hulme tells us that "MIS projects and associated procurements take place in an environment characterized by the following: Lack of management continuity and an incentive system that encourages overly optimistic estimates of the benefits that can be attained from doing the project" [5]. Leicht explains that high user expectations can actually be the cause of project failure [6], while Hoffman tells that projects fail because of poor alignment between IT departments and business users [7]. But Hodgson puts it simply by saying, "projects fail – that's the fact of life. Too many fail because the average project is like an iceberg – 9/10ths of it lay

hidden from view" [8]. Every scholar has their own reasons for why most projects are unsuccessful.

The goal of this research is to find out whether human factors or technical factors contribute more to the outcome of a project. The next section will outline different human and technical factors that must be taken into account when starting and working on an IS project, especially one that is being completed during a time of change in methodologies.

2.2 The Role of Human and Technical *Factors in Software Projects*

In the table below, there is a list of human and technical aspects that may be attributed to the outcome of an IS project. These factors are in no particular order. Each of these factors is put in a category, human or technical, to help us better understand what leads to the success or failure of an IS project. The factors have been added to this list after extensive research and personal experience working on different IS related projects.

Table 1: Human and Technical Factors [9]

Human	Technical
Communication	Troubleshooting/testing phase
User participation	Use of a model
Support from top management	Chosen methodology
Responsiveness to client	Correct tool for the job
Understanding clients goals	Data migration
Feedback capabilities	Customization of commercial software
Self-organization/collaboration	Project plan
Decision making ability	Adapting system late in development
Mutual trust and respect	

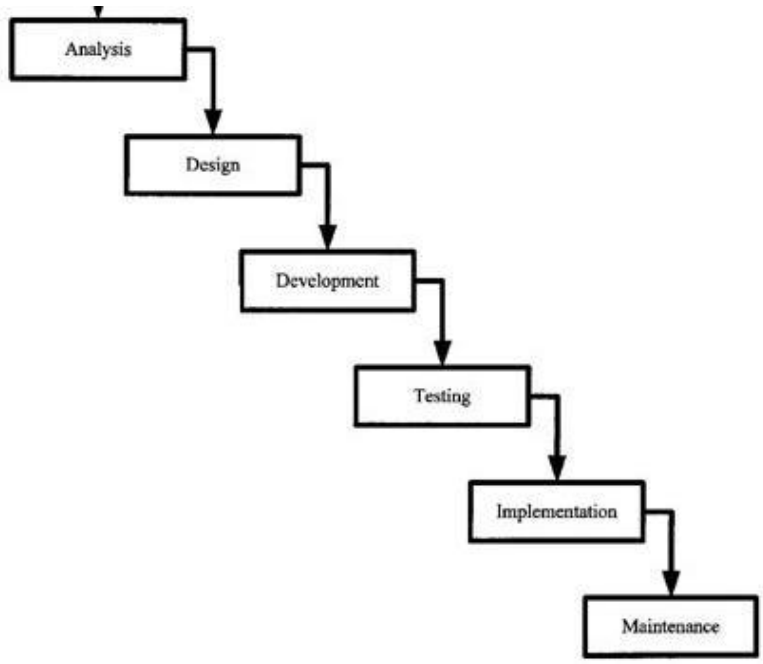
This table exhibits aspects that are important when working on and IS project. All of these factors are must be good in order for a project to run smoothly. Without them, there will be problems during the development, which could lead to failure. Following this information, this research is going to look into different software development methodologies, Waterfall and Agile. The goal is to create an understanding of the history, similarities and differences between these methodologies. This will allow us to understand why transitioning from one methodology to another is difficult and how this transition impacts the factors above and in turn, the outcome of IS projects.

3.0 The Role of Human and Technology Factors in the Transition Between Software Development Methodologies

3.1 Waterfall Methodology

There are many types of traditional software development methodologies. Some examples are the V-model and Waterfall illustrated in Figure 1. These methodologies are based on a series of steps like defining requirements, solution building, testing and deployment [10]. This analysis will focus on the Waterfall methodology specifically; it is the oldest of the SDLC models and the most well known [11]. There are four phases that are essential to traditional software development methods. The first phase is the create requirements, the second phase is the planning phase, the third phase is the development phase and the last phase is the testing phase. This is not to say that there are not intermediate steps involved with the creation of a piece of software using this type of methodology, but these are the four main categories that other steps fall under.

Figure 1 Waterfall Methodology



Source: Balaji, Sundararajan (2012) [9]

The first phase in traditional software development methodologies is the set up of requirements. These requirements are crucial to developing software in this methodology. Requirements must be clear before starting the next phase and in many cases, the changing of requirements will not be considered [11]. The second phase is a planning phase. In this step, the design and architectural infrastructure is created using models. This allows for potential issues to surface. If this step were to be missed, it could lead to more problems during the development and testing phases because these issues are not as easy to fix farther into the development. The third phase is the development phase. This phase is where the code is created until the goals for the project are reached. Normally, development is broken up into different teams that code different aspects of the system and testing overlaps with this to ensure any issues that remain are corrected. The last phase in the development lifecycle is user testing. This is when the customer becomes a part of the testing and gives feedback. After this, the project can be fully delivered

with to the satisfied customer. Figure 1 shows a flow of a project being created using Waterfall methodology. It can be seen that there are more than four steps. The general steps that are accomplished in a Waterfall project are requirements gathering, analysis, design, development, testing, implementation and maintenance. It is important to note that all of these steps fall in to the phases that were previously mentioned.

Waterfall is characterized by "separate and distinct phases of specification and development" [12]. In Waterfall, each step must be fully completed before the next step can begin. At the end of each step it is reviewed to ensure compliance with the requirements specified in the first step [13]. In other words, it is a sequential model [11]. The model works best with the requirements are clearly laid out and well understood by all parties involved [13]. This model has origins in the manufacturing and construction industries. Both industries are highly structured and making changes are extremely costly [12]. At the time when the Waterfall methodology was created, there were no formal software development methodologies in existence. This model was adapted for this purpose. The Waterfall model is credited to Winston W. Royce in 1970. Royce originally presented it as an example of a flawed, non-working model. This is currently the way that the model is described now, criticizing a widely used software development practice [12].

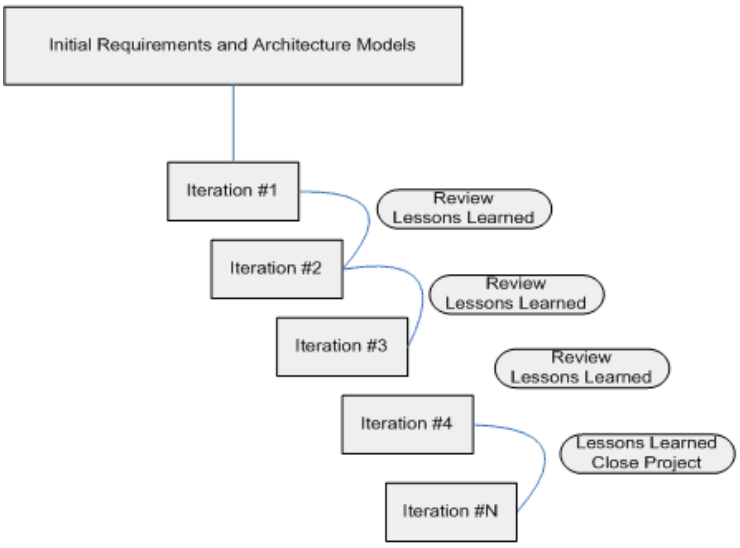
3.2 Agile Methodology

In recent years, iterative models of the SDLC have emerged. They are generally referred to as Agile models. They are many different methods of Agile but they all share common visions and values as described in the Manifesto for Agile Software Development or simply the Agile Manifesto. The manifesto notes twelve principles that are to be followed. Simply put, they are:

- Satisfy the customer

- Welcome change
- Deliver working software often
- Business people and developers work together
- Build around motivated individuals
- Face to face conversation
- Working software is the primary measure of progress
- Process promote sustainable development
- Attention to technical excellence
- Self-organizing teams
- Reflection at regular intervals [14]

Figure 2: The Agile Development Methodology



Source: Balaji, Sundararajan (2012) [11]

Figure 2 graphically depicts the agile development methodology. It is noteworthy that there are many different models of Agile, similar to traditional methodologies. Some examples are Adaptive Software Development (ASD), Feature Driven Development (FDD), Crystal Clear,

Dynamic Software Development Method (DSDM), Rapid Application Development (RAD), Scrum, Lean, Extreme Programming (XP), and Rational Unify Process (RUP) [13].

Agile models are, as the name suggests, designed for effective response to change, with the main goal to yield rapid and periodic delivery of the software [18]. They are centered on the idea of "incremental and iterative development" [10]. This means that the phases are repeated over and over again until the customer is satisfied with the final product. Agile methodologies utilize multiple, smaller processes called "increments" or "iterations" and each of these iterations have aspects of all of the original phases of development.

3.3 A Comparison of Waterfall and Agile Methodologies

There are favorable and unfavorable aspects for both Waterfall and Agile methodologies. They can both complete the task that they are assigned to but each methodology has its strengths and weaknesses. A project that was not successfully completed using one of the methodologies may have been better suited for the other.

Waterfall methodology is known for having clear requirements, being easy to implement, use and manage. But there is also a high documentation and an inability to change or update the project after the requirements have been defined [11]. Waterfall also has high risk and uncertainty. It is not well suited for complex or object oriented projects and its better for short-term projects. Agile methodology is known for its ability to adapt and for requiring the team to communicate face to face on a regular basis. But agile is difficult to complete when new developers; they must have good technical skills [11].

Table 2 summarizes the differences between traditional development methodologies, like Waterfall and iterative methodologies, like Agile.

Table 2: A Comparison of Traditional and Agile Methodologies [10], [13]

Aspect	Traditional development	Agile development
Fundamental hypothesis	Systems are fully specifiable, predictable and are developed through extended and detailed planning	High quality adaptive software is developed by small teams that use the principle of continuous improvement of design and testing based on fast feedback and change
Management style	Command and control	Leadership and collaboration
Knowledge management	Explicit	Tacit
Communication	Formal	Informal
Development model	Life cycle model (waterfall, spiral or modified models)	Evolutionary-delivery model
Organizational structure	Mechanic (bureaucratic, high formalization), targeting large organization	Organic (flexible and participative, encourages social cooperation), targeting small and medium organizations
Quality control	Difficult planning and strict control. Difficult and late testing	Permanent control or requirements, design and solutions. Permanent testing
User requirements	Detailed and defined before coding/implementation	Interactive input
Cost of restart	High	Low
Development direction	Fixed	Easily changeable
Testing	After coding is completed	Every iteration
Client involvement	Low	High
Additional abilities required from developers	Nothing in particular	Interpersonal abilities and basic knowledge of the business
Appropriate scale of the project	Large scale	Low and medium scale
Developers	Oriented on plan, with adequate abilities, access to external knowledge	Agile, with advanced knowledge, co-located and cooperative
Clients	With access to knowledge, cooperative, representative and	Dedicated, knowledgeable, cooperative,

	empowered	representative and empowered
Requirements	Very stable, known in advance	Emergent, with rapid changes
Architecture	Design for current and predictable requirements	Design for current requirements
Remodeling	Expensive	Not expensive
Size	Large teams and projects	Small teams and projects
Primary objectives	High safety	Quick value

The table shows a multitude of aspects where the different methodology types differ. This helps explain which types of projects are better suited for either Waterfall methodology or Agile methodology.

In this table, there are aspects that can be categorized as either human or technical. The majority of the aspects are human. But there are some key aspects that would fall under a technical umbrella. The difference between human and technical aspects is that the human aspects are decisions and ideas that are controlled by the humans who are working on a project. The technical aspects are those that just come with the implementation plan. Humans decide upon these technical aspects but they are not the biggest factor. The technical factors listed in the table above are development model, development direction, user requirements, architecture and remodeling. These aspects are directly linked to technical skills and requirements that go in to an information system. All other aspects listed would be considered human aspects. It is important to note that there are a large number of human aspects compared to the amount of aspects that are regarded as technical.

3.4 Challenges of Methodology Transitions

A few articles have been written on some of the challenges that companies face when transitioning from Waterfall to Agile. For example, a company works on multiple projects at a time. But when companies think abut transitioning between Agile and Waterfall it seems daunting to manage the change of multiple projects at one time [14]. Another example that was mentioned was the issue with assigning tasks to team members and managing their time. A solution for this during a transition is the change the sprint time from two weeks to one, which allows the team members to think in a way that is more natural, and to also measure the amount of work in hours so that the team can see how much time everyone is spending on a project [14]. When making this transition, the lanes of communication and amount of communication seem to be a problem for team members. Completing a project under Waterfall methodology requires a lot of written communication and does not require as much verbal communication and collaboration, so this is a big change during the transition to Agile [15]. Another article sited an issue to be that people do not know what Agile looks like [16]. This is a problem because if people cannot fully understand what they are working towards they will not be able to reach the goal. And the last issue that came up, in some form or another, in multiple articles, is that people are generally resistant to change. People's unwillingness to change and learn new methods for completing projects is the biggest obstacle that companies face when transitioning from Waterfall to Agile.

From each of these cases it is apparent that both human and technical factors are involved when transitioning between methodologies. To gain a richer understanding of issues faced by organizations when making a transition from the Waterfall methodology to the Agile

methodology a case study was conducted. The following section summarizes the findings from the case studies.

4.0 A Case Study

4.1 The Company

Utilizing and understanding the difference between Waterfall and Agile methodologies is important but it is also essential to look into how companies transition from one methodology to another. Waterfall is the well known and traditionally used but there has been a transition to agile in larger corporations. For the purpose of privacy, this research will refer to the employee being interviewed as John Smith, and the company he works for as Company X. Company X is a Fortune 100 company with well over 40,000 employees and branches worldwide. John Smith has been working for the company for 7 years and has been in his current management role in this department for about 2 years.

4.2 The Challenge

Company X has started to implement a new set of practices and tools that align with their business principles and creed to change and better the way their employees work. Some other these newly implemented practices and tools are smaller, group meeting every morning to debrief on the teams work, creating boards to visualize and keep track of the team's goals and incentivizing team members to continuously improve their workflows and to create ideas that will benefit their team and the company. This change is impacting the entire company but this research will be focused on how it will effect the IT departments and teams specifically.

The goal of implementing these tools and practices is to empower employees and build up the company's capabilities. The roll out of this new system will take years to complete. It will

touch all aspects of the business, not only IT, but most of IT has been starting to go through the change already. A typical deployment spans one year with seven separate phases. The phases are planning, preparation, diagnostic, design, pilot, deployment and continuous improvement.

Over the course of 2 months, I conducted a series of interviews with John Smith, who gave some of his insight to the company's transition. All the interviews notes were compiled then synthesized to permit the identification and categorization of each of the human and technical factors being encountered during the transition from one methodology to another. The following sections present the highlights of the interviews.

4.3 A Synthesis of the Interviews

John Smith explained how upper management is learning about the new system, and gave insights as what he sees to be the most challenging and rewarding aspects of the experience so far. John Smith noted a lot of challenges that are coming to light with this new experience but the biggest challenge is time. He says "it's difficult to grapple with the fact that we're starting and initiative to change the way we work, and its going to take four or five years to get it out to everybody in IT". Because the deployment is only happening with a couple of teams at a time to help ensure success, there will be years before everyone will be working on the same system. This could be beneficial because each team will get the time they need to make sure the system is working right for them but it can also be problematic because over the span of these five years there could be a lot of changes Company X will want to make and if all the teams are in different locations of the roll-out, making the changes will be more difficult.

The second challenge for Smith is the understanding of the benefits of the new system. He explains that the outcomes of the system are both tangible and intangible. It is easy to see the

tangible outcomes like numbers, facts and other data that will be collected but the intangible changes like engaging, involving and inspire people, will be harder to see immediately.

Despite some challenges, the most rewarding that the John Smith sees in this experience is that it will lead to challenging the teams to find and solve new problems, working together in their small groups to find answers and allow time for management to have more time to analyze what they need to focus on.

This change and transition to a new system has a lot of similar goals and working to move towards an Agile methodology. The smaller teams, many meetings and deployment that focuses on continuous improvement are selling points for Agile. Even though Agile is a software development methodology, we can see here that Company X is working to make their enterprise more agile and this new system implementation is the first step in the process. Looking deeper into the comments made by John Smith, he notes mostly human factors that are influencing success and being influenced by this change. There is a technical or system aspect as well, but Smith is more focused on the employees and people who will be working with this new system for the years to come.

4.4 Transitional Technical and Human Challenges

Throughout this research, multiple technical and human challenges have been identified as problems when transitioning from Waterfall to Agile. Some examples are how upper management is learning about a new system, the long roll out period, the far geographic distance of the teams being affects and more. These findings coincide with the findings of other scholars' research. Based on the readings and personal findings, it seems as though human challenges are a deciding factor when it comes to transitioning. As previously noted, people are resistant to change. This is the largest roadblock a company can face when transitioning. There can be

adequate training and time given but if the team members do not adopt the new idea there will be no changes made. It will be very important to motivate the employees and ensure that senior management fully understands and supports the new system. This will allow for a trickle down effect to occur. Based on the readings, these issues that Company X is currently facing and the issues that will surface later down the road are normal for all companies that are making this transition. No company or person is exempt from these problems, but there are ways to mitigate their negative impacts. There are many tools that have been documented by professionals who have been part of a transition like this. The best recommendations seem to be make note of the differences, read the Agile Manifesto, observe in order to learn and go slow [15]. These recommendations will allow the team members to fully understand the goals that they are working toward and not rush them into something until they are ready.

5.0 Conclusion

5.1 Discussion and Implications

Understanding the difference between human and technical factors in the outcome of a project is difficult. The multitude of factors that are always changing depending on the project make it hard to pinpoint on factor or even type of factor that has the most bearing on that outcome. The literature notes human factors are the main cause of projects failing because needs are never fully understood and poor project management. But there are other scholars who have said that technical aspects can be what bring a project down; like poor testing and no model being created before building a new system. These factors become multiplied when a company is going through a transition. Transitioning between methodologies can lead to even more issues because the workflow and phases that a project goes through are changing, which affects the

people and the technology. Based on the literature alone, it seems evident that most scholars would agree that some human factor is the key to a successful IS project.

The case study and interview that I conducted gives insight into a company that is currently going through a transition. They were known to have used Waterfall in the past but are now moving to Agile and making their business more agile as well, with the new system they are implementing. John Smith gave his thoughts on the changes and noted all human factors as being key components of the transition and of success for the project. He said that he thinks it is a good thing that people are trying to implement some of these agile techniques in their home lives. This shows that people are on board with this new system.

Based on the literature review and the study conducted, I find that the multiple different human factors have the biggest influence on the outcome of an IS project. Technology and technical factors do play a large role in the success or failure of a project but not as large as human factors. This was difficult to conclude because many of the technical factors have human components. For example, not having a methodology chosen is a technical factor. This is because methodologies are used in the creation of something technical but the choice to not have a methodology chosen is human error. This shows that each factor is not black or white; there are some gray areas.

5.2 Limitations and Suggestions for Future Research

This research has it limitations. As previously noted, there are many places where decisions had to be made but the answers were not clear one way or the other. This may have influenced the results because if certain factors were noted as technical and not human or vice versa, then the results may have differed. Another limitation of this research is that it is difficult to prove causation. Based on the information in this research, human factors seem to be related to

the outcome of a project more than the technical factors but there is no empirical evidence proving that.

Due to a lack of time and resources this research is not fully complete and there is room for this research to be expanded upon in the future. Because of the limitations of this study, there is room to look deeper in this research. Doing multiple case studies and comparing those insights on human and technical factors in a changing environment could be a part of future research. In the future, with more concrete data, it may be possible to determine which factor or combinations of factors specifically influence the success or failure of a project the most and why that is.

Although this research is completely finished there are still lessons to be learned about changing methodologies and what factors influence IS project outcomes. Based on this research, we have learned about the different types of software development methodologies, keys to success when completing IS projects and actions that lead to failure. But the key takeaway from this research is that understanding human factors is important to focus on when transitioning between methodologies and completed IS projects. By focusing on these factors, there can be a higher chance of that project succeeding.

Reference List

[1] Hoffer. H., George. J., & Vlacich. J (2006). Modern Systems Analysis and Design. *Pearson Prentice Hall*

[2] *Introduction to Software Engineering/Process/Methodology*. (2015). Retrieved from https://en.wikibooks.org/wiki/Introduction_to_Software_Engineering/Process/Methodology.

[3] Parasuraman, N. (n.d.). Understanding and Managing Agile Transitions.

[4] Field, T. (1997). When Bad Things Happen to Good Projects. CIO Magazine, 11(2), 54-59.

[5] Hulme, M.R. (1997). Procurement Reform and MIS Project Success. Journal of Supply Chain Management, 33 (1), 2.

[6] Leicht, M. (1999). Managing User Expectations. Retrieved from http://www.umsl.edu/~sauterv/analysis/user_expectations.html

[7] Hoffman, T. (2003). Corporate Execs Try New Ways to Align IT with Business Units. Retrieved from http://www.computerworld.com/printthis/2003/0,4814,86466,00.html

[8] Hodgson, I. (2002). Keeping Your Head Above Water, Retreieved from http://www/conspectus.com/2002/november/article19.asp

[9] Dorsey, P. (2005). Top 10 Reasons Why Systems Projects Fail. Dulcian, Inc.

[10] Leau, Y. B., Loo, W. K., Tham, W. Y., & Tan, S. F. (2012). Software Development Life Cycle Agile vs Traditional Approaches. International Conference on Information and Network Technology, 37, 162-167.

[11] Balaji, S., & Sundararajan Murugaiyan, M. (2012). Wateerfall Vs V-Model Vs Agile: A Comparative Study on SDLC. International Journal of Information Technology and Business Managment, 2(1), 26-30.

[12] Ragunath, P., Velmourougan, S., Davachelvan, P., Kayalvizhi, S., & Ravimohan, R. (2010). Evolving A New Model (SDLC Model-2010) For Software Development Life Cycle (SDLC). International Journal of Computer Science and Network Security, 10(1), 112-119.

[13] Stoica, M., Mircea, M., & Ghilic-Micu, B. (2013). Software Development: Agile vs. Traditional. Informatica Economica, 17(4), 64-76.

[14] Johnston, J. (2014, July 9). How to Start the Transition from a Waterfall to an Agile Process [Web log post]. Retrieved from http://www.mightybytes.com/blog/transition-waterfall-to-agile/

[15] Francis, K. (2016, May 16). From Waterfall to Agile: A Guide for Project Managers [Web log post]. Retrieved from http://inviqa.com/blog/from-waterfall-to-agile-project-managers

[16] Lester, R. (2013). Transitioning to Agile: Seven Strategies for Business Executives. Scrum Alliance, 2-8.

[17] Principles behind the Agile Manifesto. (2001). Retrieved from http://agilemanifesto.org/principles.html

[18] Pressman, R. S. (2009). Software Engineering: A Practitioner's Approach (7th ed.). Boston, MA: McGraw Hill.