Volume 4, No. 2, Jan-Feb 2013
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

Introducing Agile into a Non Agile Project: Analysis of Agile Methodology with its Issues and Challenges

Lipika Bose*
HCL Technologies Noida, India.
lipika.bose@gmail.com

Prof. Sanjeev Thakur
Amity University, Noida, India.
sthakur@ascs.amity.edu

Abstract: This paper will mainly focus on the most suitable software development methodology in A Small Medium Business. It will present a glimpse of traditional and the most upcoming methodology named AGILE. The issues and challenges for practicising Agile will be discussed. It will also provide management guidelines to help organizations avoid and overcome obstacles in adopting the Scrum method as a future software development method. At the end the research will illustrate a hybrid model which showed how agile methods can be adopted and utilized to effectively support the development of mission-critical, small and medium project.

Keywords: Traditional Model, Agile Methodology, Hybrid Model

I. INTRODUCTION

Traditional methodologies are plan driven in which work begins with the elicitation and documentation of a complete set of requirements, followed by architectural and high level design development and inspection. Due to these heavy aspects, this methodology became to be known as heavyweight. Some practitioners found this process centric view to software development frustrating and pose difficulties when change rates are still relatively low. The agile methods claim to place more emphasis on people, interaction, working software, customer collaboration, and change, rather than on processes, tools, contracts and plans Agile methodologies[1] are gaining popularity in industry although they compromise a mix of accepted and controversial software engineering practices. The software industry would most likely find that specific project characteristic such as objective, scope, requirements, resources, architecture and size will determine which methodology suits them best. Either agile or heavyweight or maybe a hybrid of the two.

In the past few years, anecdotal evidence and success stories from practicing professionals suggests that agile methods are effective and suitable for many situations and environments. The negative aspects of the agile methods mentioned earlier imply that there are issues, problems, and challenges faced in developing high-quality software products using these methods. Identifying the issues, problems, and challenges of the agile methods should be more beneficial to organizations considering them than merely showing their positive benefits.

Therefore, it is worthwhile to conduct a research to identify the issues and challenges of agile methods and propose a new hybrid model suitable for Small Medium Business projects[2].

For this research, one research site XYZ was chosen for an in-depth case study. XYZ organization has produced several high-quality software products through traditional and agile software development methodology with. An exploratory research process using observations, surveys, documentation, and interviews was conducted at the organization. The contribution of this research is fivefold:

(1) It identifies critical issues of traditional model. (2) It highlights the issues of Agile (3) It provides management guidelines to help many organizations avoid and overcome obstacles when adopting the agile method as future software development method, and (5) it suggests a new framework for further research on the application of traditional and agile methods.

II. REVIEW OF LITERATURE CLASSICAL & AGILE

A. Traditional Software Development Methods:

The analysis phase, about 15% of the SDLC [2], analyzes the current system, its problems, and then identifies ways to design the new system through requirements gathering. The design phase, 35% of the SDLC, decides how the system will operate in terms of hardware, software, and network infrastructure. The implementation phase occupies about 30% of SDLC and is the phase where actual coding occurs. The maintenance phase occupies the remaining 5% of SDLC and focuses on going-live, training, installation, support plan, documentation, and debugging. Figure 1 and Table 1 below show a typical waterfall lifecycle and deliverables, respectively.

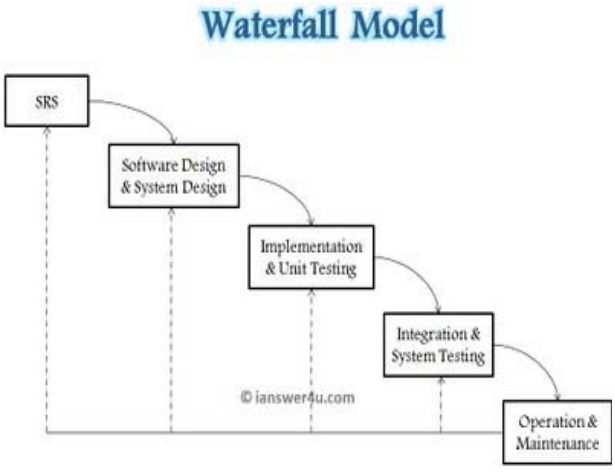

Figure 1. Waterfall model lifecycle

Table 1. Waterfall Model Deliverables

Phases	Deliverables
Planning Phase	Planning Specifications
Analysis Phase	Analysis Specifications
Design Phase	Design Specifications
Implementation Phase	Completed Product

B. Advantages and Disadvantages of Traditional Model:

a. Advantages:

a) Waterfall model is simple to implement and also the amount of resources required for it are minimal.
b) In this model, output is generated after each stage (as seen before), therefore it has high visibility. The client and project manager gets a feel that there is considerable progress.
c) Project management[3], both at internal level and client's level, is easy again because of visible outputs after each phase. Deadlines can be set for the completion of each phase and evaluation can be done from time to time, tocheck if project is going as per milestones.
d) This methodology is significantly better than the haphazard approach to develop software.
e) This methodology is preferred in projects where quality is more important as compared to schedule or cost.
f) The advantage of waterfall development is that it allows for departmentalization and managerial control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process.
g) Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order, without any overlapping or iterative steps.
h) Fits well into a contractual setting where each phase is considered a milestone.

b. Disadvantages:

a) Real projects rarely follow the sequential flow and iterations in this model are handled indirectly. These changes can cause confusion as the project proceeds.
b) It is often difficult to get customer requirements explicitly. Thus specifications can't be freeze. If that case arises baseline approach is followed, wherein output of one phase is carried forward to next phase. For example,even if SRS is not well defined and requirements can't be freeze, still design starts. Now if any changes are made in SRS then formal procedure is followed to put those changes in baseline document.
c) In this model we freeze software and hardware. But as technology changes at a rapid pace, such freezing is not advisable especially in long-term projects.
d) This method is especially bad in case client is not IT-literate as getting specifications from such a person is tough.
e) Even a small change in any previous stage can cause big problem for subsequent phases as all phases are dependent on each-other.
f) Often, the client is not very clear of what he exactly wants from the software. Any changes that he mentions in between, may cause a lot of confusion.
g) Until the final stage of the development cycle is complete, a working model of the software does not lie in the hands of the client. Thus, he is hardly in a position to inform the developers, if what has been designed is exactly what he had asked for.

C. Agile Software Development Methods:

As a remedy for the traditional software development methods[2] shortcomings, agile software development methods (ASDMs) were developed.

Table 2: Principles Behind The Agile Manifesto

S.No	PRINCIPLES
1	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4	Business people and developers must work together daily through the project.
5	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7	Working software is the primary measure of progress.
8	Agile processes[4] promote sustainable development. The sponsors, developers,and users should be able to maintain a constant pace indefinitely.
9	Continuous attention to technical excellence and good design enhances agility.
10	Simplicity—the art of maximizing the amount of work not done—is essential.
11	The best architectures, requirements and designs emerge from self-organizing teams.
12	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

D. Advantages and Disadvantages Of Agile:

a. Advantages:

a) Customer satisfaction by rapid, continuous delivery of useful software.
b) People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
c) Working software is delivered frequently (weeks rather than months).
d) Face-to-face conversation is the best form of communication.
e) Close daily cooperation between business people and developers.
f) Continuous attention to technical excellence and good design.
g) Regular adaptation to changing circumstances.
h) Even late changes in requirements are welcomed.
i) The team does not have to invest time and effort and finally find that by the time they delivered the product, the requirement of the customer has changed.

j) The end result is the high quality software in least possible time duration and satisfied customer.
k) The documentation is crisp and to the point to save time.
l) The company is given a competitive advantage since it is continuously changing its approach to satisfy its customers.
m) Innovative designs based on the customers' demands are provided. This in turn gives a wider variety for the customer to choose from.
n) Even tough the production could change rapidly; mass production could still be reached while flexibility is still possible.

b. Disadvantages:

a) Active user involvement and close collaboration are required throughout the development cycle..
b) Agile requirements[5] are barely sufficient. This eliminates wasted effort on deliverables that don't last (i.e. aren't part of the finished product), which saves time and therefore money
c) Testing is integrated throughout the lifecycle. This helps to ensure quality throughout the project without the need for a lengthy and unpredictable test phase at the end of the project. However it does imply that testers are needed throughout the project and this effectively increases the cost of resources on the project. However there is an additional cost to the project to adopt continuous testing throughout.
d) Less importance for designing and documentation.
e) Project will be mess if client is unclear about requirements.
f) In case of some software deliverables[2], especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
g) Necessity of experienced and senior resources: Since Agile method is more about less planning and more decision making, it is absolutely necessary to have experienced and senior resources in the team or experienced resources mentoring new resources.
h) Time consuming and wastage of resources because of constant change of requirements: If the customers are not satisfied by the partial software developed by certain iteration and they change their requirements then that incremented part is of no use. So it is the total wastage of time, effort and resources required to develop that increment.
i) More helpful for management than developer: The agile methodology helps management to take decisions about the software development, set goals for developers and fix the deadline for them. But it is very difficult for the baseline developers to cope up with the ever changing environment and every time changing the design, code based on just in time requirements.

III. RESEARCH METHODOLOGY AND PROCEDURES

A. Glimpse of XYZ Firm:

The XYZ firm has been providing ecommerce application primarily to clients over three years. It is a Small and Medium Business enterprise. The major client is Symantec. The firm has been using the traditional software development method and has successfully completed several projects. Then as per the clients demand the made a transition to Agile to increase customer satisfaction by rapid, continuous delivery of useful software. They mainly practiced Scrum Methodology in their projects. Agile also overcome some of the major drawbacks of Traditional Model. The duration of a project at XYZ firm is now 3-6 months. But again many challenges and issues were faced and they concluded that "Pure agile"[12] is not appropriate for all projects. The issues and challenges practicing SCRUM[6] are listed in the next section.

In an attempt to gather data, two types of data were collected from the firm totriangulate findings. First, observations of software development process were conducted through onsite visits and field notes were taken during the observation to make the strange familiar. Finally, a formal face-to-face interview was conducted with project managers and lead software engineers. This triangulation in the process of data collection provides more useful information and different perspectives on the issues, allows for cross-checking, and yields stronger substantiation of constructs.Data collection focused on issues and challenges of Scrum. It sought information on:

The roles of the product owner, team members, and Scrum master; the daily Scrum meeting, the daily Scrum of Scrums meeting, the Sprint review, and the Sprint planning meeting[13]; the product backlog, the Sprint backlog, and burn down charts; the flow of Scrum process; user involvement; training; documentation; communication; individual and team experience with Scrum; applicability of Scrum; bug tracking system; project estimation and planning; and the working environment. The following open questions were used as survey questions:

a. What have worked and have not worked for you on a project since you began using Scrum?
b. What are the most unique and interesting aspects of Scrum?
c. What have you learned from Scrum?
d. What would you do next time with Scrum?
e. What advices do you have for others involved in Scrum?
f. Do you have any comments or opinions on any of the following Scrum process?
   a) Daily Scrum meeting
   b) Daily Scrum of Scrums
   c) Sprint planning meeting
   d) Sprint review meeting[7]
   e) Product backlog[7]
   f) Sprint backlog[7]
   g) Scrum master

B. Data analysis and research results:

Table: 3

S. No	Common Categories	Concepts	Data
1	Human Resource Management Factor	Training	• Developers need more formal step-by-step training. • Some of developers do not see the benefits of Scrum.
		Collaboration	Scrum provides good collaboration mechanism between developers. Collaboration between

			developers and product managers should be improved Scrum helps developers to be aware of other developers' tasks.
		Multiple Responsibilities	One person is responsible for many tasks in different field • Product manager is a bottleneck in the development process • Developers have a communication problem with a product owner
2	Structured Development Processing[8] Factor	Scrum Framework	Scrum ceremonies force developers to be on the same page • Daily Scrum meeting, Sprint planning meeting, and Sprint review meetings are sometimes inefficient • Setting up the meeting time is difficult
		Formal Code Review	Company employs informal code review
		Documentation	Less documentation • More comments on the code • Hard to complete the system without having any documents • Equally shared skills and knowledge among team members
		Use Cases	Developers realize the importance of creating use cases • Use cases help developers understand the system they are going to build • Users do not know what user cases are
3	Environmental Factor	Customer Involvement	Not involved in the decision making process • Biggest roadblock in the development process • Customers do not know what they really want • Customers give unclear system requirements
		Social Loafing	Individual hard work is not recognized Accurate measurement of individual performance is needed
4	Information Systems and Technology Factor	Communication	Daily scrum meetings improve communication within a team • Lack of communication between teams • Work is being duplicated • Lack of communication with customers
		Bug Tracking System	□ JIRA is a issue tracking product was used for bug tracking, issue tracking, and project management. □ Developers want customers to report their bugs in JIRA.

C. Issues and Challenges of Scrum:

a. Human Resource Management[8]: On one occasion at XYZ, a situation developed between developers and a particular project manager. Due to insufficient human resources, one person had multiple responsibilities as product manager, product owner, and accounting manager. The first problem was that developers did not try to talk with the individual about issues that came up outside of the Scrum meeting because they knew he was too busy and overloaded with too many other tasks. The second problem was that developers had to wait to get any questions answered. The third problem was the developers had the notion that the overloaded manager was a bottleneck for the rest of team. A fourth problem was that developers had communication issues with him because he was rarely available to people.

b. Training: One of the biggest problems XYZ faced was in new employee training.
Due to the complexity embedded in the system, new employees need to spend a lot of time becoming familiar with the system. XYZ deemed training new employees separately in each Scrum team a big waste of time.

c. Scrum Framework[9]: Some developers did not like inefficient Sprint planning and review meetings. They felt that keeping daily Scrum meetings to 15 minutes was difficult because people gab a little too long, there is an excessive amount of material that needed discussed, and taking care of 2 or 3 projects at once. Another problem was setting up the meeting time. It was difficult to get all developers together at one time without interrupting their work because of the flexible work schedule.

d. Documentation: The lack of detailed design, as indicated at ABC, caused many problems in complex projects. One main area affected considerably was testing, because QA personnel depend heavily on documentation to find problems. Developers tried to place more comments and explanations for any tricky logic in the code, along with explanations for any changes that they made, to compensate for the lack of documentation.

e. Formal Code Review[9]: Developers at XYZ did not have a formal code review, but they had an occasional informal code review. Not having a formal code review invoked some issues. First, developers did not pay extra attention to their code, because they believed no one would look at it again. If they believed that at some point somebody would go back and look at their code, they would have more accountability. Second, developers lost opportunities to improve the quality of their code and enhance their coding skills through feedback from other developers.

f. Use Cases: Three issues were identified related to creating use cases. First, some developers were not well prepared to write use cases because they were unfamiliar with a system. Second, clients did not have a clear and precise idea what they really wanted to have in their system. Third, clients did not know what use cases are or how to use them.

IV. A HYBRID MODEL: A COMBINATION OF AGILE AND WATERFALL

A. Hybrid Model[11]:

Both agile methods and traditional methods have strengths and weaknesses as shown in the literature review

section. It would be very beneficial if we can come up with a new method that accommodates the strengths while suppressing the weaknesses of both methods.

a. ***Preliminary Aggregation And Formulation:*** We get the use case stories from the customers. Analyse the use case stories and prioritize the requirement as per the business requirement. Depending upon the prioritization we along with the business decide upon the scope of the each iteration. Once the requirements of the first iteration are freeze a SRS is prepared specifying the mock-ups required and the requirements in clear. Since we consider less number of features in each iteration document preparation and finalization takes less time as compared to waterfall model. Estimation of the efforts, time and cost required is all estimated in this phase.The business, functional and the non functional requirements are all formulated and finalized in this stage. Since we are taking few requirements at a time depending upon their priority this phase is less time consuming. Agile requirements[10] are barely sufficient. However this can mean less information available to new starters in the team about features and how they should work. It can also create potential misunderstandings if the teamwork and communication aren't at their best, and difficulties for team. This phase will overcome the drawback of agile.

b. ***Picture Plan:*** The primary objective of the picture plan phase is to create a design that satisfies the agreed application requirements. The Design stage describes how the proposed solution is to be developed. The solution design is specific to the system's technical environment and the tools to be used in constructing the system. The results of this stage will be inputs to the Execute and Implement stages. Detailed requirements are transformed into detailed specifications for the system to guide the work of the Execute stage. For complex projects there may be an iterative relationship between the Design and Execute stages. Infrastructure architecture decisions are made to address how the system will meet the defined functional, physical, interface, information protection and data requirements.

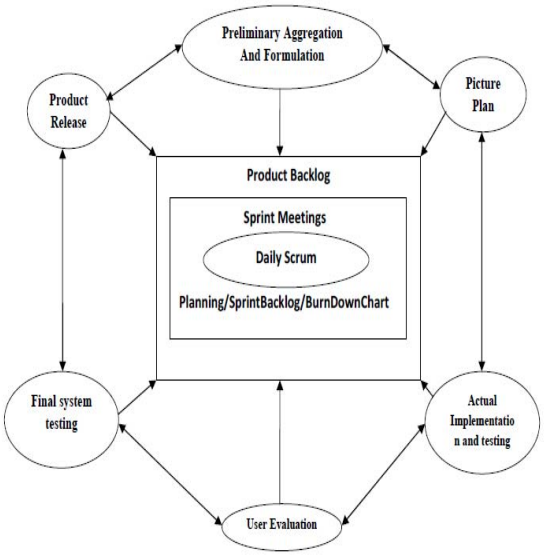

Figure 2. A new Hybrid Model.

c. ***Tracking Of The Iteration:*** Underlying the iteration-based development process is the continuous activity of tracking status and adjusting course. Even within the course of a short iteration, scope must be a managed, and deviations from plan will occur. The tracking and adjusting activity is focused on getting an objective, real-time picture of where the software development effort is and whether the team is likely to "land" (complete on time) the iteration in process. Tracking progress of the current iteration requires having visibility into the status of the stories, defects, and other tasks that are being worked on during the iteration. In particular, it's important to be able to understand how quickly the team is moving through the scheduled work and how accurate its estimates were. The progress toward the release can be understood by considering the status of the stories, defects, and other tasks across the iterations in the release. Iterative and incremental processes[11] tend to favour a schedule driven approach, so at the release level, it is most important to understand which chunks of planned work are done and how fast the team is producing work. This information allows the team to deliver the most valuable software on the committed release date by making decisions about what work to do next and what work to defer.

d. ***Actual Implementation:*** This phase includes the actual coding. In this phase developers are involved. The main focus in this phase goes to coding of the software. All the system components and features, including user interfaces, business logics, data access functions, and help functions, are implemented according to the specifications designed in the previous phase. This phase should produce a releasable working system so that the system can be deployed during the next phase. This phase can include several iterations that continue the design and implementation of the system.

e. ***Testing:*** Testing is the process of evaluating a system or application, to check whether the application meets all requirements of the client and to detect the errors. Testing will be carried in the same way as it is conducted in Agile. Agile Testers are part of the cross-functional team and talk directly to developers and have their say in all phases of Software Development Life Cycle. With Agile Testing you can influence developers to think abouttestability in their code and you can help understand and refine new requirements. Agile teams are collocated, so the flow of information is uninterrupted. When participating in Agile Testing, keep your feedback loop short. If you find a bug, talk about it with developers. If you need some clarification about the requirements Agile Testing, talk to the Product Owner. Be active on Planning Meetings and Retrospective Meetings. Your influenceon quality assurance in agile development methodologies is needed to improve the product. With Agile Testing you actually have that power. There is no time or need for writing lengthy Test Plans because you only need a few pages stating the vision, technologies, and your approach. Early testing means testing in phases of gathering requirements and design. And as I mentioned above, this really happens in Agile Testing. Agile Testing starts working on stories and evolves into Test Driven Development and ends as automated acceptance

testing on Continuous Integration for every new change committed to source code repository.

f. ***User Evaluation****:* Once the prototype passes the system test, representatives from the client evaluate the prototype against the requirements and the use cases. They also access if they want more functionality to be added to the product as the initial requirement gathering stage may not have collected the entire user requirements or were not clear at that time.

g. ***Are More Changes Required:*** This is not a stage but a decision point in the lifecycle where client decides whether they need more changes to the products or not based on the Evaluation doe in the last stage. Prototype may go into one or more iterations if client wants more changes to the products or it can be handled in the next sprint. If the client is happy with the product and are ready to accept this as final product then the product may go to the final round of system test.

h. ***Final System Test:*** Once the customer approves the prototype as the final product a final system integration test might be required to check the integration of the product with client environment and other application.

i. ***Product Release:*** This phase involves training of both user and support teams and dispatch and endorsement to production. Once final test is done product is ready to be released and deployed at the client location. Figure 12 shows the diagrammatic representation of the Hybrid Model. It to a certain extent overcomes the shortcomings of the waterfall and the agile model. The result of implementing hybrid, the limitations and assumptions are listed in the below sections.

B. Result On Implementing Hybrid Model:

Following will be the benefits we will earn from the Hybrid agile:

a. It provides us a required detailed documentation. This overcomes both Agile and waterfall limitation.
b. The Client can change the requirement at any time in the project. Depending upon the estimation it can be incorporated in the same or in the upcoming sprint. This overcomes the waterfall limitation.
c. It allows much reflection and revision. It overcomes both Waterfall and Agile limitation.
d. More clear idea about the final outcome of the project. This feature overcomes both Agile and Waterfall limitation.
e. Testing is included in the two steps which will help to verify the final outcome of the project to clients. It overcomes both Agile and Waterfall limitation.
f. Face to face communication between the team, daily standup meetings help to understand more about the project. This overcomes Waterfall limitation.

C. Limitations of this Model:

There are several limitations of this study.

a. First, the most important limitation of this research is that the unit of analysis of this research was narrowed down to the Scrum software development process as utilized by the XYZ firm. Hence, the finding of this research may not be directly generalizable to the larger population.
b. Secondly, in some cases due to client interaction in every phase iterations might get long resulting in high cost and more time.
c. Thirdly people having expertise in agile as well as sound knowledge of traditional can easily implement and manage software development using this model.
d. Fourthly, interviewees were selected from various roles, levels of experience, and positions, including developers, lead software engineers, Scrum masters, project managers, and executive officers. Selection bias may have affected the process of selection.
e. Finally, because the quality of data collection and the research results is highly dependent on the skills of the researcher and on the rigor of data analysis, the quality of the research might be influenced by skills and experience of the researcher.

V. CONCLUSION

This research started with the description of the early stages of Software development. It provided a glimpse of Traditional and Agile. It then highlighted the Critical steps for smooth transition from Classical to Agile. This research identified the four main categories of critical issues and challenges that may affect the quality of the application of agile methods and illustrated a theoretical model which showed how agile methods can be adopted and utilized to effectively support the development of mission-critical, small- scale projects. It also provided management guidelines to help organizations avoid and overcome obstacles in adopting the Scrum method as a future software development method. The lessons about Scrum obtained through the case studies will be valuable assets to many Scrum practitioners, and the suggested new framework for further research on the application of traditional and agile methods [14]will provides a basis for further research for those who want to further explore hybrid software development[11] methods

VI. REFERENCES

[1]. AgileLogic. (2006). Agile logic. Retrieved March 20, 2009, from http://www.agilelogic.com

[2]. Advanced Development Methods, Inc. (2009). Scrum, Retrieved February 2, 2009, from http://www.controlchaos.com.

[3]. Balijepally, V. (2005). Collaborative software development in agile methodologies –Perspectives from small group research. Proceedings of the Eleventh Americas Conference on Information Systems, August 11-14, Omaha, NE.

[4]. Batra, D., Sin, T., & Tseng, S. (2006). Modified agile practices for outsourced software projects. Proceedings of the Twelfth Americas Conference on Information Systems, Acapulco, Mexico, August 4-6, 2006, 3872-3880.

[5]. Beck, K., Beedle, M., Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). Principles behind the agile manifesto. Retrieved January 19, 2009, from http://www.agilemanifesto.org/principles.html

[6]. Boehm, B. (2002, January). Get ready for agile methods with care. Computer, 35(1), 64-69.

[7]. Boehm, B., & Philip, P. (1988). Understanding and controlling software costs. IEEE Transactions on Software Engineering, 14(10), 1462-1477.

[8]. Boehm, B., & Turner, R. (2003, June). Using risk to balance agile and plan driven methods. Computer, 36(6), 57-66.

[9]. Boehm, B. & Turner, R. Management challenges to implement agile processes in traditional development organizations. IEEE Software. 22(5), 30-40. 2005.

[10]. Cockburn, A. (2007). Agile software development: The cooperative game. Upper Saddle River, NJ: Addison-Wesley.

[11]. Cho, J. (2009). A hybrid software development method for large-scale projects: rational unified process with Scrum. Issues in Information Systems, X(2), 340- 348.

[12]. Lawrence, R. Avanade Inc., Seattle, WA Yslas, B. Three-way cultural change:introducing agile within two non-agile companies and a non-agile methodology. Agile Conference, 2006. Conference Publications. 5pp.-262.2006.

[13]. Livermore, J.A. Factors that impact implementing an agile software development methodology. Southeast Con, 2007. Proceedings. IEEE. Conference Publications .82 86.2007.

[14]. Schatz, B. & Abdelshafi, I. Primavera gets agile: A successful transition to agile development. IEEE Software. 22(3). 2005.