ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

A STUDY ON THE DIFFERENCES BETWEEN AGILE AND TRADITIONAL WATERFALL SOFTWARE DEVELOPMENT METHODS"

Rashmi Pawar

Research Scholar, Computer Application , Bharati Vidyapeeth University, Pune, India,

Abstract: This paper will focus on traditional SDLC method like Waterfall model and will brief on comparative study of same with Agile.

Traditional SDLC models like Waterfall model is sequential development methodology where output of one development phase becomes input for other. To brief cons, like risk of failure are high and process itself is not flexible for change.

Agile software development methodology is widely used software development process which overcomes the drawbacks of traditional software development methods.It provides means for rapid development and great deal of flexibility to adopt new changes during development process.

The primary data collection method was interviews of the industryexpertise. The secondary source of data is reference books and Internet articles. This paper will help to understand basics of Agile methodology.

Keywords-Agile methods,Agile methodology, Software development, Software Development Life *Cycle(SDLC)*

Introduction

Software development is an organized process that thrives to deliver products in faster, better and cheaper ways. There have been many studies and suggestion in improving the development process. Recently, this interest has paved way to a new software development method called Agile Software Development.Agile methods strive to deliver small sets of software features to customers as quickly as possible in short iterations.

As part of this paper scope most commonly used methods will examined from the angle of their applicability, strengths and weaknesses and their adoption in industry. This will lead us to find benefits, limitations and difficulties in agile software development. The rest of paper is as follows: section 2 contains the previous related works on agile methods, section 3 contains the features of Agile methodology, and section 4 contains the Agile development methods,Section 5 contains the analysis of the study, section 6 contains the validity threats and limitations to

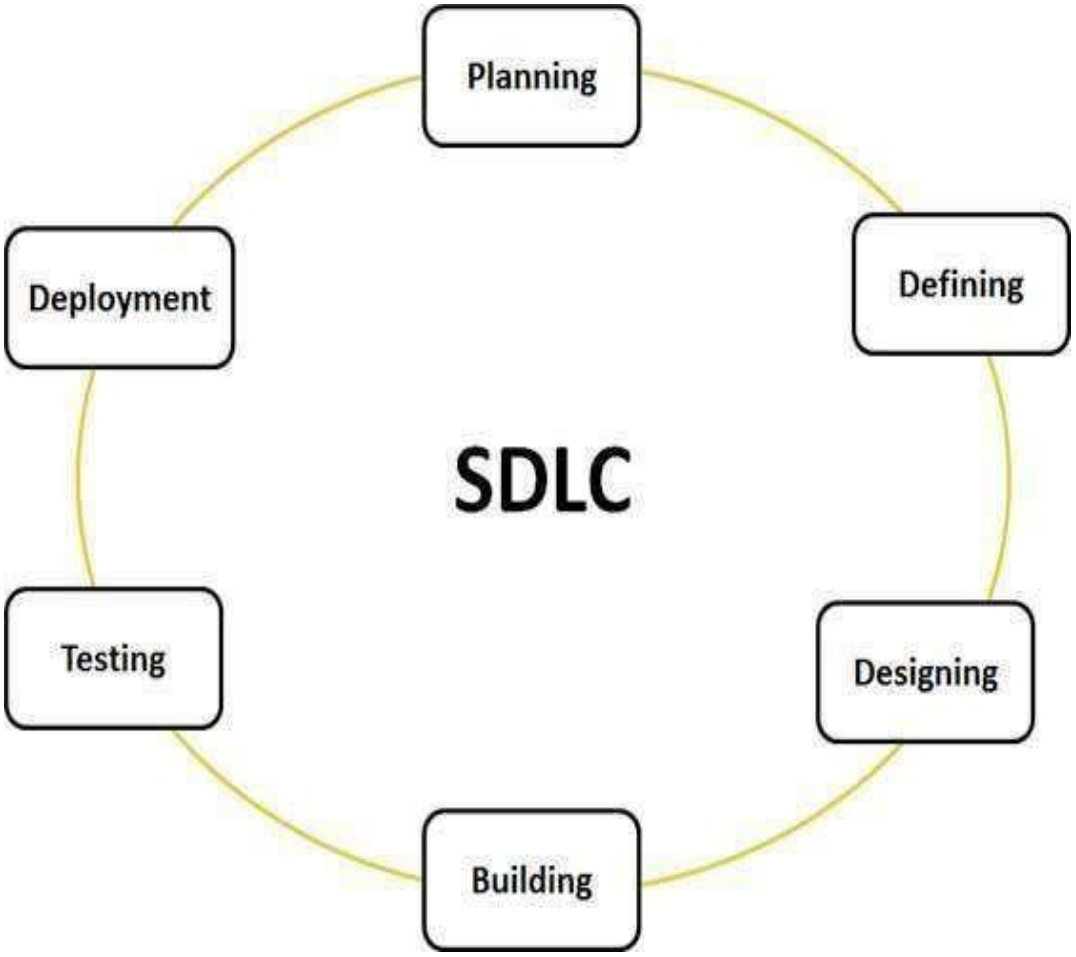
Background and Related Work

1.1 Software Development Life Cycle(SDLC)

SDLC [3] is the process consisting of a series of planned activities to develop or alter the software products.SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal



A typical Software Development life cycle consists of the following stages:

Following are the most important and popular SDLC models followed in the industry:

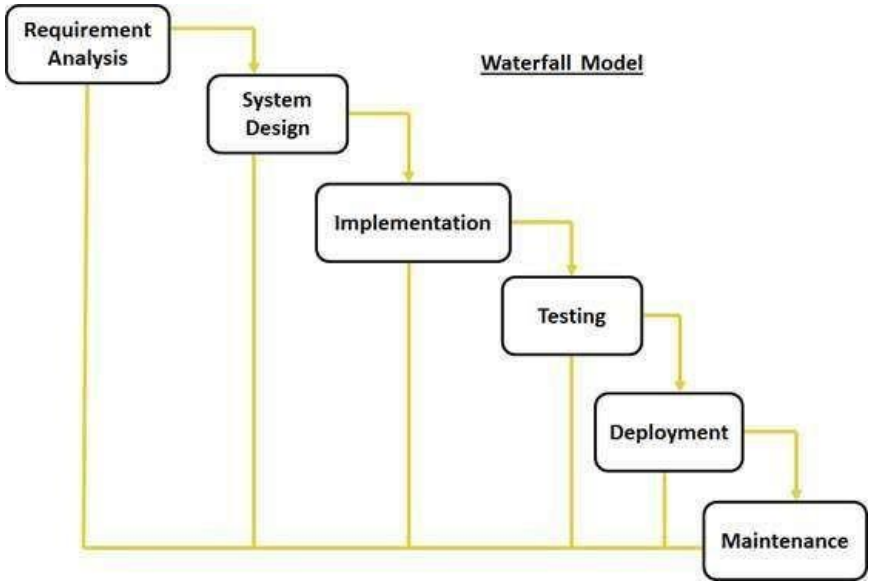
1. Waterfall Model
2. Iterative Model
3. Spiral Model
4. V-Model
5. Big Bang Model

2.1.1 Waterfall Model

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear- sequential life cycle model. It is very simple to understand and use.

Testing is carried out once the code has been fully developed. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal



2.1.1.1 Pros

1. Requirement is clear before development starts.
2. Each phase is completed in specified period of time after that it moves to next phase.
3. As its linear model, it's easy to implement.
4. The amount of resources required to implement this model are minimal.
5. Each phase proper documentation is followed for the quality of the development.

2.1.1.2 Cons

1. The problems with one phase are never solved completely during that phase and in fact many problems regarding a particular phase arise after the phase is signed off, this result in badly structured system.
2. If client want the requirement to be changed , it will not implemented in the current development process

I. Agile Methodology

3.1.History of Agile

In 1970, Dr. Winston Royce presented a paper entitled "Managing the Development of Large Software Systems," which criticized sequential development. He asserted that software should not be developed like an automobile on an assembly line, in which each piece is added in sequential phases. In such sequential phases, every phase of the project must be completed before the next phase can begin. Dr. Royce recommended against the phase based approach in which developers first gather all of a project's requirements, then complete all of its architecture and design, then write all of the code, and so on. Royce specifically objected to this approach due to the lack of communication between the specialized groups that complete each phase of work.

In The 2013 State of Agile Development survey [6], conducted by Version One says 83% of the responders

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

are

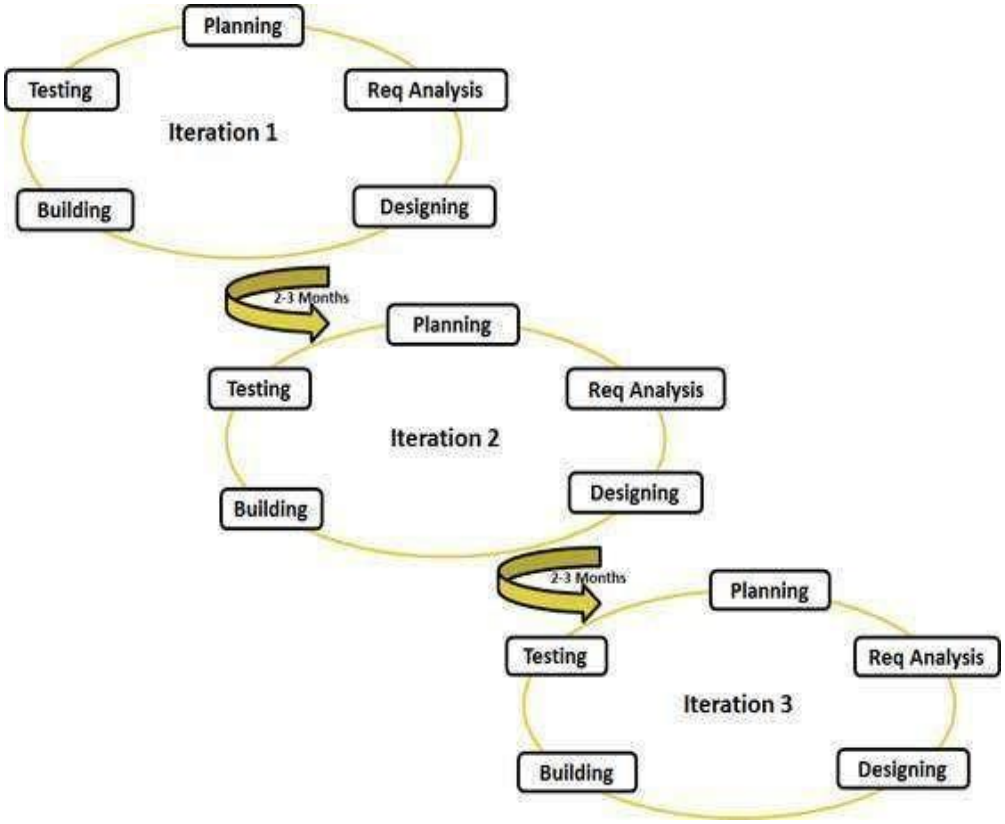
using agile software development, however Agile may not be the silver bullet or the destination for the software development it has his own challenges which may force to rethink to adopt agile as a software development methodology.

In a recent survey conducted by Version one 57% stated that their teams are distributed [13], and the figure is growing as industry considers distributed development as a cost effective model.

3.2What is Agile methodology

The term agile stands for 'moving quickly' .Agile methodology is a lightweight methodology for software development. [1]Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In agile development, rather than a single large process model that implemented in conventional SDLC, the development life cycle is divided into smaller parts, called “increments” or “iterations”, in which each of these increments touches on each of the conventional phases of development. Iterative approach is taken and working software build is delivered after each iteration. Each build is incremental in terms of features; the final build holds all the features required by the customer[6].

Here is a graphical illustration of the Agile Model:



ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

3.3Principles of Agile methodology

There are twelve principles behind the Agile methodology as below,

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

Agile development methods

There are various methods that are collectively known as agile methodology or Agile manifesto**.**

4.1 DSDM (Dynamic Software Development Method)

Dynamic System Development Methodology is an agile framework for software projects, it was used to fine tune the traditional approaches. The most recent version of DSDM is called DSDM Atern. The name Atern is a shortening of Arctic Tern - a collaborative bird[citation needed] that can travel vast distances and epitomizes many facets of the method which are natural ways of working e.g. prioritization and collaboration. DSDM addresses the most common failures of information systems projects, including exceeding budgets, missing deadlines, and lack of user involvement and top-management commitment.

4.1.1 . The main features of the DSDM method are as follows:

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

1. User involvement
2. Iterative and incremental development
3. Increased delivery frequency
4. Integrated tests at each phase

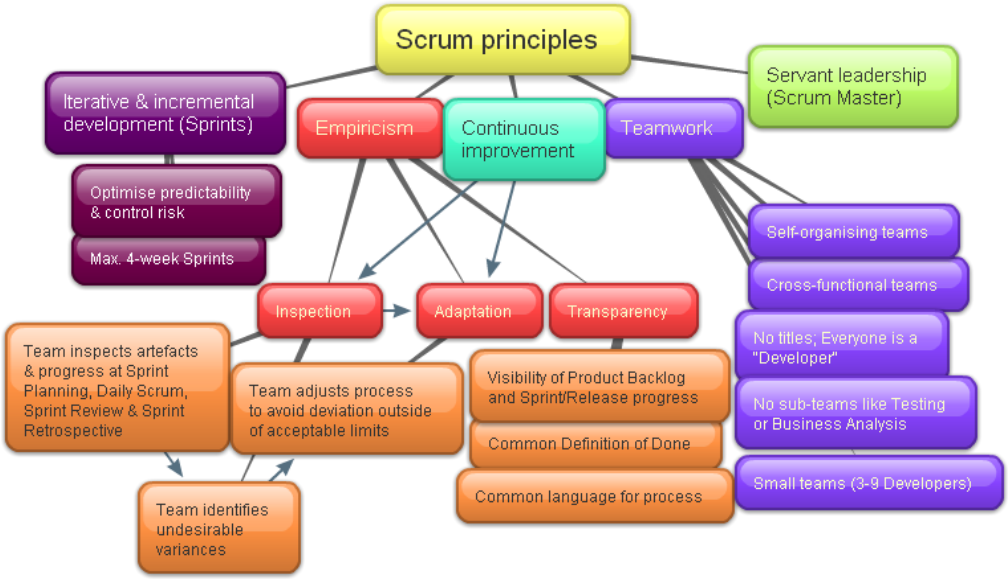
The acceptance of delivered products depends directly on fulfilling requirements

4.2 Scrum

Scrum is most popular agile framework in the world, Scrum uses iterative and incremental development model. Scrum concentrates particularly on how to manage tasks within a team-based development environment. Scrum provides the simple framework of basic tenets to solve problems and deliver good results - more valuable software faster. Scrum beautifully defined in below diagram ,that shows that scrum is based on following principles,

4.2.1 Scrum Principles

1. Iterative and incremental development
2. Continuous improvement
3. Empiricism (A theory that stats that knowledge comes only from experience)
4. Servant leadership (Scum Master)



4.3 XP

According to Wiki definition: "Extreme Programming (XP) is a software development methodology

which is intended to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles, which is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted."

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

Extreme Programming is a type of agile software development, it advocates frequent "releases" in short development cycles, which is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted. The methodology takes its name from the idea that the beneficial elements of traditional software engineering practices are taken to "extreme" levels. Extreme Programming is a software- development discipline that organizes people to produce higher-quality software more productively. XP addresses the analysis, development and test phases with novel approaches that make a substantial difference to the quality of the end product.

4.4 TDD

Test-driven development (TDD) is a software development process that relies on the repetition of a

very short development cycle: first the developer writes an (initially failing) automated test case that defines a desired improvement or new function, then produces the least amount of code to pass that test, and finally refractors the new code to acceptable standards.

4.5 Lean

Lean is a production practice that considers the expenditure of resources for any goal other than the

creation of value for the end customer to be wasteful, and thus a target for elimination. Working from the perspective of the customer who consumes a product or service, "value" is defined as any action or process that a customer would be willing to pay for. Lean is centered on preserving value with less work.

Lean comes from Lean Manufacturing and is a set of principles for achieving quality, speed, and customer alignment.

4.5.1 There are seven Principles of Lean Software Development:

1. To eliminate Waste
2. To build quality product
3. Create Knowledge
4. To deliver software fast
5. Respect People
6. Optimize the Whole
7. Dissimilar Commitment

4.6 Kanban

Kanban is a system to control the logistical chain from a production point of view, and is not an inventory control system. Kanban was developed by Taiichi Ohno, at Toyota, to find a system to improve and keep up a high level of production. Kanban is one method through which JIT is achieved. Kanban became an effective tool in support of running a production system as a whole, and it proved to be an excellent way for promoting improvement.

4.7 Plan

In Plan Driven Development a project is successful if it goes according to plan, so in

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

software

development it depends on the requirements stability, on having clear and fixed requirements. As you probably know, that is a luxury most software projects don't have.

In plan-driven methodologies, it is less costly to change requirements during the design stage and it is more expensive to adapt to changes when construction has already started. So, a lot of energy is put into the planning phase. But software development is different. There is no guarantee that a good design will make construction predictable.

"*Walking on water and developing software from a specification are easy if both are frozen.*" - *Edward V. Berard*

Comparison between Waterfall model and Agile Methodology

Traditional development methods for example Waterfall model is a heavyweight and Agile models are based on flexibility and welcoming changes any time in software development so Agile models are lightweight models. Despite the success of traditional models it has a lot of drawbacks, like linearity, inflexibility in changing requirements, and high formal processes irrespective of the size of the project. Kent Beck took these drawbacks into account and introduced Extreme Programming, the first agile methodology produced. Agile methods deal with unstable and volatile requirements by using a number of techniques, focusing on collaboration between developers and customers and support early productdelivery. A summary of the difference of agile and heavyweight methodologies is shown in the table below.

The agile and heavyweight methodologies both have their strengths and weaknesses. People usually follow either one of these methodologies or follow their own customized methodology. There are major factors affecting methodology decision and selecting which is suitable for various conditions. These factors can be categorized into project size, people and risk. .

5.1 Project Size

The key elements of project size are project budget, duration and project team organization. The larger the team or more budget you need, the bigger the project is. Thus compiling more requirements, requiring more people and more coordination. Heavyweight methodologies support this by providing plans, documentation and processes for better communication and coordination across large groups. Agile claims that for a given problem size, "fewer people are needed if a lighter methodology is used, and more people are needed if a heavier methodology is used,"

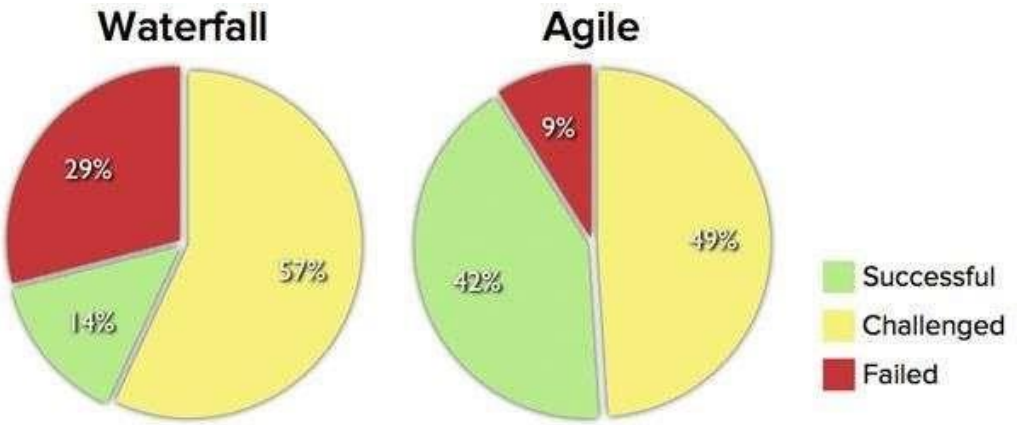
5.2 People Factor

Agile methods mostly based on human factors, "Individuals and interactions…" and "Customer collaboration...". Even NASA has concluded that technology and training are not the big factors, "The most effective practice is leveraging human potential". Having skill and experienced people in a team is a key factor for agile methodologies. Encouraging domain experts to be part of the team gives developers rapid feedback on the implications to the user of their design choices. Customer adaptability is another great factor, the customer gets the power to check the progress and change the direction of the software development during each iteration. Gaining this level of commitment from the customer makes agile methodology a more attractive process than heavyweight.

5.3 Risk Factors

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

The most important risk factors in the development of a software process are project criticality and responding to change. Agile methods are used in applications that can be built quickly and do not require extensive quality assurance. Critical, reliable, and safe systems are more suited to a heavyweight methodology. If a project is critical, all requirements must be well defined before the development of the software. Changes can be resolved using an agile method. Practices defined in agile methods allow for better handling the changes, such as constant feedback from customer and short iterative development. So that risk factor is minimized by using Agile methodology .But in waterfall model risk factor is high because whole software development is based on initial requirements and step by step development.



Source: The CHAOS Manifesto, The Standish Group, 2012.

Above figure shows the success,challengesand failure of software by using traditional Waterfall model and Agile models.

Improvement

Agile software development methods were developed to provide more customer satisfaction, to shorten the development life cycle, to reduce bug rates, and to accommodate changing business requirement during the develop.

Conclusion

Agile approaches are increases the flexibility, agility and to be more adjusted to the environment where software development projects are present and working today. The idea behind Agile methodology would likely to break large projects into smaller projects which would become more flexible. Also its proven that agile provides means for software development with minimum risk where project requirements are well defined, development team is self dependent. It also ensures that every team member understands, evaluates and actively participate in development which ensures no single point of dependency and failure. It also facilitates

ISSN No: 2350-1278
Peer Reviewed & Refereed Journal
National Research Journal

more flexibility to adopt changes in requirement and control development life cycle through micro breakdown of requirements and planning of same.

This idea is brought up from the primary data - interviewees, respondents of the survey as well as recommended by the agile methodologists.

Understanding the differences among various methods improves the decision-making regarding selection of the most suitable methodology in a suitable way.

REFERENCES

1. K. Beck, Extreme Programming explained: Embrace change. Reading, Mass., Addison-Wesley, Nov16, 2004
2. L. A. Williams, "The XP Programmer: The Few-Minutes Programmer", IEEE Software, pp. 16-20, May/June 2003
3. K.Mar and K.Schwaber, "Experiences of using Scrum with XP", http://www.controlchaos.com/XPKane.htm, Accessed on 2/2/2005
4. Robert C. Martin "Agile Software Development: Principles,Patterns and Practices"
5. Alistair Cockburn "Agile Software Development: The Cooperative Game(2nd edition)
6. Jim Highsmith "Agile Project Management:creating Innovative product" (2nd Edition)
7. http://www.agilemanifesto.org
8. http://www.agilesoftwaredevelopment.com
9. Thesis, Jonna Kalermo and Jenni rissanen, "Agile software development in theory and practice"
10. Cho, Juyun. Issues and Challenges of agile software development with SCRUM. Issues in Information System.
11. W. Cunningham, "Agile Manifesto." http://www.agilemanifesto.org/, Accessed on 10/7/2004
12. http://ipcsit.com/vol37/030-ICINT2012-I2069.pdf