HAAGA-HELIA
University of Applied Sciences

Assessing the applicability of Traditional and Agile methodology to enhance success of software development projects.

Bushra Mostafa

HAAGA-HELIA
University of Applied Sciences

Abstract

15 April 2015

Author(s) Bushra Mostafa	
Degree programme Business Information Technology	
Report/thesis title Assessing the applicability of Traditional and Agile methodology to enhance success of software development projects	**Number of pages and appendix pages** 50+4

Over the past few decades, various types of software development methods have evolved and implemented in the software industries. Each method has its own strengths and weaknesses as distinguished from another. Agile software development (ASD) has emerged as a major evolutionary step in the software development process. There is a general classification in the software development process. Process-wise software development is considered to follow either a heavyweight method or a lightweight method. Agile software development (ASD) methods facilitate the product development process and provide high quality software products.

For this research, among different software development methods, Waterfall and Scrum methodologies will be highlighted mostly because of their popularity and principles. Despite of critics arguing about Waterfall method, it was adopted by companies based on the project characteristics. Scrum has been adopted by a vast number of software industries in recent years. Scrum is classified as a lightweight method which is iterative, features incremental development and provides a strong customer collaboration and high quality products within a defined timeframe. Business requirements change so frequently and the scrum methodology principles can adopt the changes quickly and provide the product with lower rates of bugs and a shorter development cycle.

The software component covers a substantial area of any business operations. Many software development methodologies have evolved because of the competition. It’s a tough decision to switch from one method to another. Before that, many factors have to be measured inside the organization. This thesis provides some guidelines to overcome some common challenges of the agile methods.

This study investigates how the proper selection and adoption enhance the success of the software development projects. This study also finds out which factors influence the method selection procedure for a project. The opinion of the majority of experts interviewed in the study indicates that agile software development methodologies are in more demand than traditional ones because of their strong focus on customer collaboration and response to changes.

Keywords:

Agile software development (ASD), Scrum methodology, agile methods, Software industry, Iterative and incremental development.

Table of contents

List of Figures

List of Tables

ABBREVIATIONS

Heavyweight methodology	Heavyweight methodologies are considered to be the traditional way of developing software.
Lightweight methodology	Software development method based on few rules and practices.
Agile software development	Software developing framework. In agile methodology, software is develop based on iterative approach.
Waterfall methodology	Sequential software development process including series of stages.
Spiral model	Consider as heavyweight methodology
FDD	Feature Driven Development. Agile methods.
IID	Iterative and incremental development.
Iteration	In agile, software is developed in small cycles.
Scrum	Agile method
Daily meeting	Scrum meeting. Daily stand-up meeting which is inform about the project status.
Product Backlog	Functionality, features and technology which should follow and cover during the project.
Product owner	Product owner represent customer requirement. Product owner is responsible for the product requirement and manage the schedule.
XP	Extreme Programming
Sprint	One time box iteration used in the scrum method.
Scrum Master	Helps product owner and development teams to manage the Scrum rules in the project.
Spring backlog	Detailed document about the requirement and procedure about the upcoming sprint.

1. Introduction

Deployment of software methodologies in software industry is very common habit. Currently, many software industry implement agile methodology in the project to finalize the project faster with less resources. Running a software development projects very complicated and it needs different kind of management skills. Knowledge about software methodologies are very important for the IT worker.

Based on the project characteristics, organization choose their development methodology to manage and execute project. Agile methodology support the development of small-scale, large-scale and other type of projects. But the challenges arise when the principals does not followed correctly. Its common problem to identify the best methodology to follow for the software project because of the business requirements. The applicability of the software development methods are varies depend of the project nature. This thesis will tell about the possible issues and challenges of the methodology and illustrates how software development methods can be adopted and followed efficiently to support and increase success for the project.

Several projects were done previously based on the software development methodologies. But the focus area was different. Most of the project were done to evaluate specific agile methods inside the company. Some projects focused on the error in the agile methodologies and one project I found which focus on the security issues in the agile methodology. Some earlier projects are given in Appendix 2.

This thesis focus on the applicability on the software development methodology. The growing popularity of agile methods need sources and research on the agile methodologies. So might be this thesis will be helpful for the student and organizations.

2. Research question, objectives and Scope

Software Industry environment is highly distributive in nature. In order to gain the competitive advantages, software industry adapt the business changes so quickly. Traditional methods are so straightforward and slow to adapt the constant changes of business requirement. The shortcoming of the traditional method create the demand of the new solution which has evolved as a remedy of the shortcomings of the traditional

Methods. Agile software development (ASD) methods are designed to deliver the product frequently through fast development cycle. It is common practice to take advantages from different agile methods and according to the project requirements, the method is selected and followed. This research review the agile methods principals, usage, challenges, strengths and weaknesses. The aim of this research to assess different agile methods based on the principals and functionality. The objectives are listed below:

Research question:

1. How the combination of proper software development methodologies selection and adoption enhance the success of the software development project?

2. Which factors influence the selection of development methodology to manage and execute software project?

Objectives:

- This paper will provide some guidelines related to management which help to avoid known obstacles when adopting Scrum.
- This paper tell briefly about agile project management (APM)
- Present the facts when organization switch of from traditional to agile methodologies.

3. Research methodology

This research paper will collect data by taking interviews. Three interviews will be taken for data analysis. According to the nature of research topic, qualitative methodology will be followed.

The purpose of choosing the interview tool for this research because it suits best to this topic. The questions for the interview will be set to find out what software development method the company use, why and how they follow and manage. Also I want to find out which are the most common problems and difficulties when the organization transit form traditional to agile methods.

The interview will be held among three different organizations to get three different views. Among of three, two of them are experts in Scrum methodology and other interviewee had an experience in traditional methodology. So it will fulfill the purpose of this research. The interview will be semi- structured. Semi- structured interview is the most common type of interview used in qualitative social research. At the semi structured interview, specific information is compare and contrasted with information which gained from other interviews. (Dawson, C., 28). The interview will have certain questions to be answered and along with there will be freedom to discuss beyond the questions. Based on the information from the interview, data analysis will be done systematically. There are two criteria for this interview to be successful. First is to review the agile methodology and the second is the challenges which organization faces when they are switch from traditional methodologies to agile approach.

4. Heavyweight development approach:

Heavyweight methodology practiced a long time in software industry to develop and deliver the products. The characteristics were so strict and predictive to manage the process. Sometimes those characteristics slow down the development process because of too much documentation. The common characteristics of the heavyweight methodologies are listed and described below (Awad 2005, 10):

- Predictive approach
- Comprehensive documents
- Orientation

Predictive approach

Heavyweight methodologies focus on requirement analysis for a long time and spend huge time on the design phase. The design phase draw the full process which include the system requirements and how to fulfill them. All of the planning complete before it goes to next groups who are responsible to develop the system. During the planning phase, several issues are predicted such as project delivery time, budget and task distribution. This approach is not good for the small project and new technology because a lot of unplanned issues can be arise at the development phase.

Comprehensive documents

One of the common characteristics of heavyweight methodologies is huge documentation. According to the predictive approach nature, team spend more time on planning and design phase before the system building. There are the possibility that perhaps all of the features are not needed to construct the system but the team have to do documentation for all features which slow down the project process (Williams 17 May 2013).

Orientation

To deliver each of the task, developer has to use specified tools which sometimes not serve the best purpose. At the developing phase, it's create complication to follow the drawing as it is every time.

According to Fowler," The goal of engineering methods is to define a process that will work well whoever happens to be using it" (Fowler 2010, 3). The objectives of the process are to support the developer to complete their task. Heavyweight methodologies planned the process beforehand which sometimes hinders the team progress. There is always conflict between manager and developer because of their purpose. Manager or planning team focus on mainly schedule and budget. Because of that, they choose some traditional processes for the completion of the project which might not suit to the project.

5. Traditional software development methods (TSDM):

There are two main category in software development process. TSDM are considered as heavyweight methodology. TSDM follow the traditional approach to development software. At the beginning phase, TSDM methods works fine and get the popularity but later, the deficiencies were find out by the software developer and stakeholders. During the process time, the communication between stakeholders and developers were not strong and maintained so the delivery product quite often does not face the stakeholder requirement.

Traditional methods were very disciplined and sequential. There were no testing phase before the product was ready. So a little fault had a strong and big impact because each phase is depend on the previous step and nothing was check between the stages. In traditional methodology, there was too much documentations were needed and requirement analysis was done before the projects started.

Among different kinds of TSDM, two most significant traditional methods will be discussed in this thesis. Those are: Waterfall and Spiral model.

5.1 The Waterfall development method

The waterfall model is mostly used in software development for large software project both in public and private sector. It was created in 1970 by Winston Royce and this model addresses all of the phases in software development life cycle phases (Dooley 2011, 9). This plan-driven model designed with seven stages and it requires detailed documentation at each stages. There are many drawbacks in waterfall methods but the main three deficiencies in waterfall methods hindered its acceptance and make it very difficult to implement. The first one is, gathering requirement before architectural design. It is very difficult to design if the requirements are fixed and not changeable. The second deficiencies is there is no testing phase before the project is completed. The third one is each phase must be complete before the next phase begins. The below figure shows that there is no way to back to previous step. So that it call waterfall.

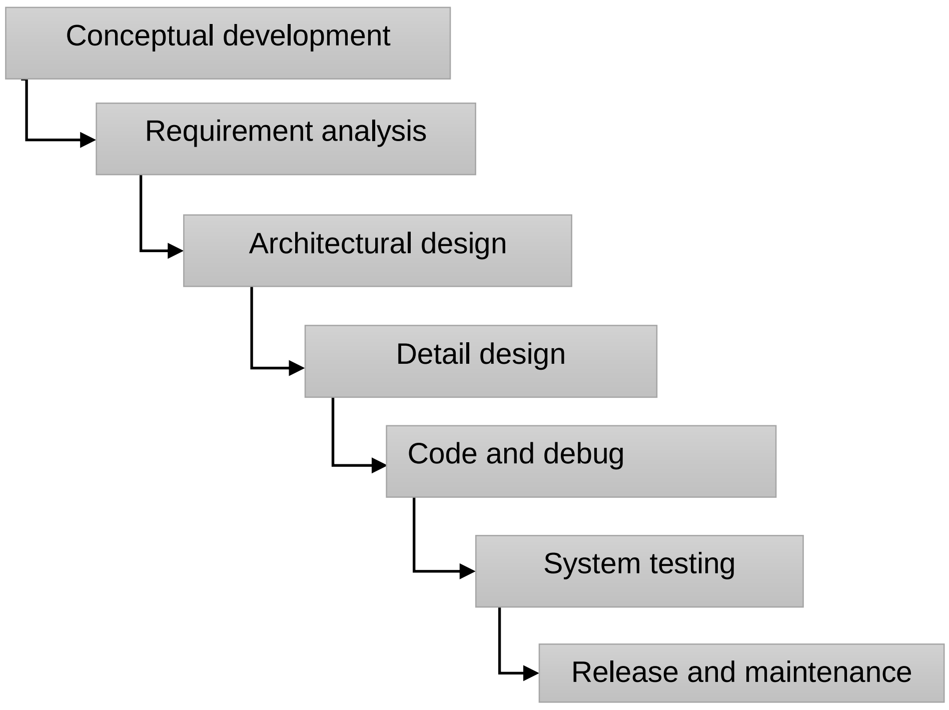

Figure 1: The waterfall process model

5.1.1 Deliverables:

Each phase of the waterfall model has specific deliverables which shows in the below table (Cho 2010, 20).

Phases	**Deliverables**
Planning phase	Planning specification
Analysis phase	Analysis specification
Design phase	Design specification
Implementation phase	Final product

Table 1: The waterfall made deliverables

5.1.2 The strengths, weaknesses and applicability of Waterfall method:

Strengths:

- This methodology deliver a very complete and well understood requirement specification.
- The waterfall method is a disciplined and systemic approach
- Works well when quality is more important rather than schedule and budget.
- Identifies deliverables and milestones (Munassar & Govardhan 2010, 3)

Weaknesses:

- It doesn't have feedback opportunity between stages.
- This model doesn't have good communication with customer or product owner.
- This method doesn't support change management, risk management methodology (Farrell, 2007, 12).
- Lots of documentation required. Not suitable for small scale project.

Applicability

- Based on the characteristics of waterfall methods, it doesn't suit the project which require rapid development.

- The waterfall method is best for projects where requirements will not change and are able to be defined and frozen at the initial stage of the project (Farrell, 2007, 21)
- Research and development oriented project is not suitable to implement waterfall methods because it has possibility to change the requirements.
- It suits best for the long term government project where the requirements are fixed and not needed testing between the phases.
- At the current time, large software projects follow the combination of waterfall and agile methods. At the initial phase, project team emphasis on modified waterfall methodology and during the developing phase, agile methods are adopted.

5.2 Spiral method:

Spiral model is a risk-driven approach which adopt elements from other methods such as waterfall or evolutionary prototyping methods. This model was first described by Barry Boehm in 1986 (Wikipedia, 2015). Risk-driven approach means where the risks are focused in each step. The risk-driven approach has three main steps. These are: 1) Identify and prioterize the risks. 2) Select and apply the set of architecture techniques and 3) Evaluate risk reduction (Fairbanks 2010, 1).

Increment and iteration are the main concept of spiral method. Those two works as the foundation or basement of spiral method. Iterative development requires a sequence of design-code-test cycles which is followed in several iterations. Frequent iterations are needed to fulfill the project requirements. Through the iterations, project proceed incrementally. Each cycle is complete when the requirements are fully completed and then next cycle will start (Farrell 2007, 14).

According to Boehm, spiral model evolved as a refinements of waterfall method (Boehm 1988, 7). In waterfall method, there were no testing phase before the project is complete but in spiral model was proposed to develop different prototype in various stages until the project is completed. One of the most significant feature of this model is: each cycle is completed by a review where all concerned parties are involved and by mutual commitment the planning and design for the next phase are decided (Farrell 2007, 9).There are three round in spiral model and each round consists of different deliverables such as objectives, constraints, alternatives, risk, risk resolution, risk resolution result, plan for next phase, commitment. According to Boehm, Round 0 is called, feasibility study, round 1 is called as concept of operation and round 2 called as Top-Level Requirements Specification (Farrell 2007, 12). Each cycle of spiral process involves:

- **Planning phase**: Analyzing objectives, identifying alternative approaches, and establishing constraints for the next process cycle.
- **Risk analysis phase**: Planning the next cycle by evaluating alternative approaches, identifying the risk factors of each approach, and selecting an approach.
- **Engineering phase**: Implementing the selected software and at the end of the phase the development tested.
- **Evaluation phase:** Evaluating the outcome and deciding what to do next. At this phase stakeholders evaluate outcome of the project and participate in planning for the next spiral (Fairley 2009, 86).

There are so many features include in spiral method but risk-driven documents, particularly specification and plans are the most important features of this model (Farrell 2007, 16).

5.2.1 Risks scale and decision points

This model is based on risk driven approach so any shortfalls in evidence are considered as a risks and it should be discusses in risk mitigation plan. The risk and risk mitigation plans are considered by the stakeholders and developers where both parties are addressed different risks based on risk type. There are four type of scaling to address risks and based on these, the decisions are taken (Boehm, Lane, Koolmanojwong & Turner 2014, 20).Those four scales are listed below:

- The risks are accepted and well covered
- The risks are too high but addressable
- The risks are negligible
- The risks are too high and un-addressable.

Decision Points:
Based on the risk, the decisions are taken in risk mitigation plan to proceed the project. The decision points are listed below (Boehm & al.):

- If the risk are accepted and well covered then the next spiral will be started.

- If the risks are too high but addressable then the project remain in current spiral until the risks are well covered.
- If the risks are negligible then the project move to the next spiral
- If the risks are too high and not possible to address then the project is terminated or re-scoped.

5.2.2 Strengths, weaknesses and applicability of Spiral method

Strengths:

- During the development cycle, this method has the ability to change the requirements.
- It is risk-driven model. Risk analysis in each spiral help to get success in project (Agarwal, Tayal & Gupta 2010).
- Spiral model support prototyping so that it helps to manage control, risks as well as costs.
- End- users are able to see the product in development cycle (Sabharwal 2008, 18).
- It provides framework to generating iterative development models (Fairley 2009, 88).

Weaknesses:

- This method accept changes and requirements in several stages of the development cycle. Sometimes it can be difficult to manage changes. Project manager should expert to accept and reject changes coming into the project (Sabharwal 2008, 19).
- This model is costly to implement because it requires to build prototype in each cycle or spiral.
- This model is a risk-driven model, so if the risk analysis goes wrong, the entire project might be not successful or need to re-works.

Applicability:

- According to the spiral model nature, this model is suitable for the large-scale projects with medium to high risks.
- For the small- scale project, the spiral model is not work well because it need risk analysis in each spiral which might be go over the budget.

- Spiral method work best where the requirements are not well defined such as new technology (Sabharwal 2008, 19).

6. Lightweight Development Approach

Software industry has to choose software development methodology based on the project requirements. During past few decades, lightweight methodologies got rapid interest because of its characteristics. Lightweight methodology, the name itself implies that it’s the opposite of heavyweight methodology. Though lightweight methodology is not the official name, they are referred by this term because it’s easy to categorize the opposite’s heavyweight methodologies. There are three main significant nature of lightweight methodologies are listed and described below (Fowler 2000):

- Light methods are adaptive rather than predictive
- Less-document oriented
- People oriented

Light methods are adaptive rather than predictive:

Lightweight methodologies tend to split the whole project into several pieces. Planning and design phase has been done for each phase separately. In heavyweight methodology, planning for the whole project has been done before the construction so no changes are not welcome later. But lightweight methodologies accept change in later phase of SDLC cycle.

Less-document oriented

In lightweight methodology, documentations are needed for a given task which are usually source codes.

People oriented

Rather than process oriented, lightweight methodologies emphasize on people. People are the key factor in any industry. Due to adaptive nature, people get focus in lightweight methodologies.

7. Agile Software Development Methodology (ASDM)

Agile software development method consider as lightweight methods because of the similarity in their principals. Usually a method means way of working. Agile methods are considered as processes which support agile's own principals. According to James Shore and Shane Warden, "agile development is a philosophy, it's a way of thinking" (Shore & Warden 2008, 9). Agile methods has its own value and principals which are needed to put into agile's individual element 'practices'. Practice element continue during the
Whole project time and agile methods has its own way to combine practice elements.

In 2001, the creators of different agile software methods and others who were implement agile methodologies in their sector, decide to form an agile manifesto where they summarize and promote their belief. Original agile manifesto is attached in appendix 1(Cunningham 2001).

This manifesto has advocates the importance of individual and team motivation as well as customer satisfaction. Customer collaboration get high priority and need close relationships with project members. The software process must flexible to accept change and documentation get less priority which help to focus on the develop software.

7.1.1 Principles behind the Agile Manifesto:

The 12 principles of the Agile Software development made by the Agile Manifesto (agilemanifesto 2001). They act as a foundation agile philosophy and serve as a common framework for agile methods.

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 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 behavior accordingly.

Table 2: Agile Manifesto Principals

The following section provide strengths, weaknesses and applicability of agile software development methods.

7.1.2 Strengths, weaknesses and applicability of ASDM

The 12 key principals are the strengths of ASDM. The most important strengths are listed below:

Strengths

- The main strength of the agile methodology is the flexibility to accept changes. In opposite to traditional software development methods, agile methods do their planning for each iteration and based on the findings and success, next iteration is begin.
- Customer collaboration is another main strengths of the ASDM. After each iteration, customer or stakeholder can evaluate the development and give feedback. It is recommended that, a full time customer can take place in agile development team to avoid any misunderstanding of the requirements and resolve any emergencies to discuss with the customer. (Mohammad, Alwada'n & Ababneh 2013, 458).

- Each iteration, testing and integration has been done which help to view and measure the progress in accordance with assure that the current phase is fully working and fulfill the requirements.
- Agile methods work in small team which help to communicate with the team member and speed up the development process as well as accelerate time to market.
- Less documentation help to focus on the development process which save time and reduce costs.

Weaknesses

- It difficult to implement ASDM in large projects and large organization. In large project, team size is big and project continues for a long time. If the team size is big, it's difficult to co-ordinate team. According to VersionOne, Inc. 2014 survey, 10% agile projects failed because of the broad organization or communication problem. (VersionOne, Inc. 2014, 5)
- Some organization require certification and due to lack of documentation, it sometimes difficult to worth it.
- Sometimes it difficult for the product owner to attend the meeting with the developer at a regular interval.
- In some company, agile core values has conflict or don't match with company's own culture or philosophy. According to VersionOne, Inc. 2014 survey, 13% respondent tell about this issue (VersionOne, Inc. 2014, 5).

Applicability:

- ASDM work best in small scale project as well as small team size.
- It suits when project requirements are not static. For example, new technology or new software where customer can change their requirements at any period of development lifecycle.
- ASDM work best where produce working software frequently (Cockburn 2007, 305).

8. Well known Agile software methods:

According to VersionOne, Inc. 2014 survey, the most popular and adopted agile methods are Scrum and Scrum variants (73%) (VersionOne, Inc. 2014, 5)

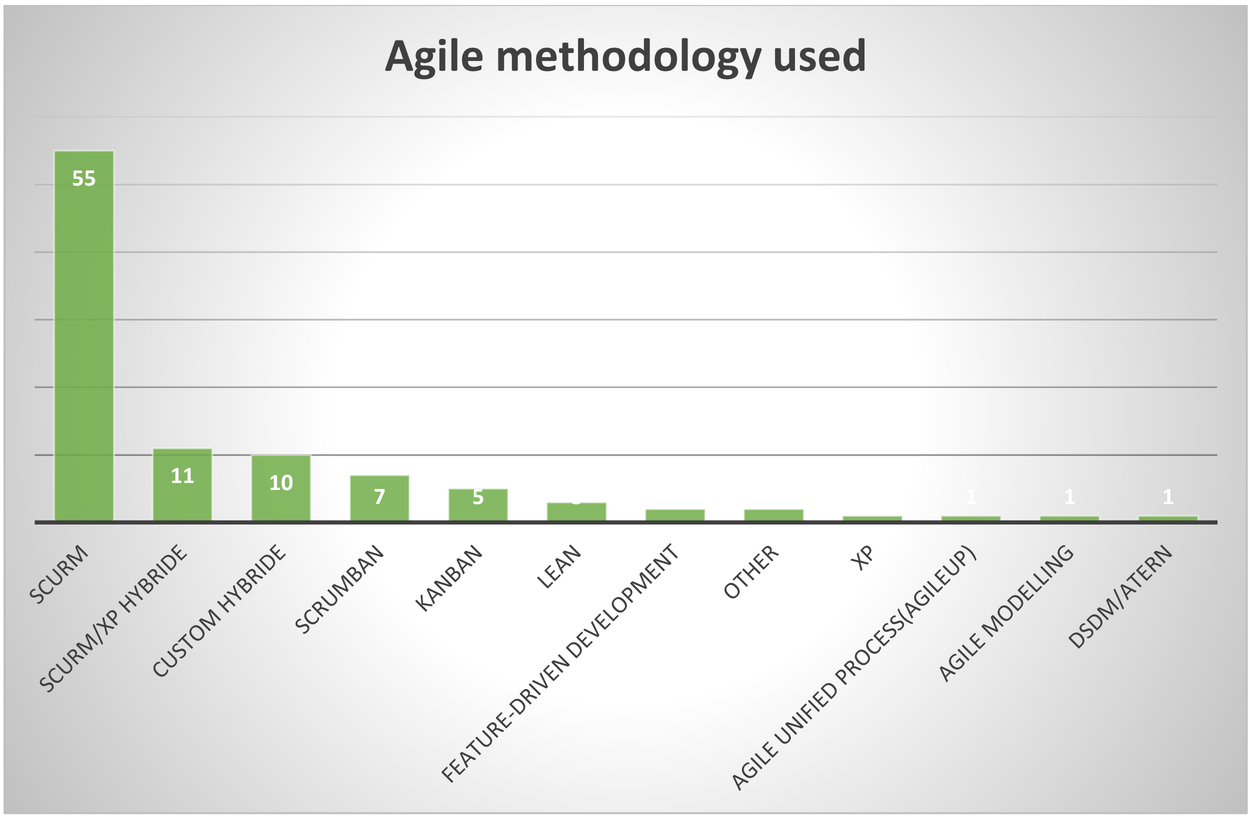

Figure 2: Agile Methodology Used (VersionOne's 2014 Agile Methodology Survey)

Survey of most widely adopted agile methods. 8th Annual State of Agile Development Survey 2014, Courtesy of VersionOne, Inc.

Based on their survey, this report focus on two well adopted agile methods: extreme Programming (XP) and Scrum.

8.1 Extreme Programming (XP):

Extreme Programming was created by Kent Beck and Ward Cunningham in 1995. It is an agile methodology to develop software in a flexible and efficient way. This methodology is based on five values and 15 principals. XP's five underlying values are described briefly in below:

8.1.1 Values:

- **Communication:** Each team member should be aware of what other doing. To fix bugs in the system and redesign the features, the collaboration is needed inside team member. To do so, collective knowledge is needed to spread around the team. The communication is constantly going on during the whole project lifecycle (Dooley 2011, 17).

- **Simplicity**: Simplicity is one significant value of this methodology where developers emphasize on the current requirements and write the code most simple way to meet the customer needs. Refactoring concept is used in the XP which means that the XP developers can redesign codes to make it simpler without changing the behavior (Dooley 2011, 17).

- **Feedback**: In XP methodology, each iteration contains constant check and feedback loops which help to assure the product quality. Customers also involve in implementation feedback by writing functional tests which focus on the development.

- **Courage:** XP developers' should be courage to find out the best solution for the project. It can be happen that, existing solution doesn't fit to the project and needed to re-code again. Developers check the schedule daily to track any fault and if anything doesn't suits to the project, programmers need courage to redesign the project. The main focus to create the solution which easier to maintain in the long time (Stellman & Greene 2015, 210).

- **Respect:** According to Leffingwell, XP is a team-based practice where each team member shows the expect to each other and also to the projects related things. All of the members and decision are important in the project. The shared value reflect cooperation, communication and feedback of the processes (Leffingwell 2007, 34).

8.1.2 Principals:

XP has 15 basic principles which fully based on the values. These principals can guide the development team and help to apply the XP practices on the real-life project. Understanding the XP principals help to embrace changes and planning. XP principals are described below:

- **Rapid feedback:** Each iteration has feedback loop which help to focus on the requirements and if needed, change the design and put it again in the system to fulfill the requirements. Automated tests are important to run the unit test to integrate the changes.

- **Assume simplicity:** According to XP values, simplicity has to maintain for the design, coding and planning. Developers focus on the today's task and solve the problems as simple way.

- **Incremental changes:** After doing any changes, it needed to integrate into the system which help to find integration and interface errors. It also engage customer to examine the system.

- **Embracing changes:** One of the core agile methodologies principal is to accept change any phase of the software development process. XP also follow this principals to accept changes any stage of the development process. XP project success depend on the discipline for the accommodate changes.

- **Quality work:** All of the software development process focus on less defect in the coding. To do so, XP programming aimed at test-driven development (or TDD) and Pair programming (Stellman & Greene 2015, 178).In pair programming, two developer do the coding together which inject less bugs and doing constant brainstorming. Test-driven development make sure that each individual code is working. It helps to avoid some common problem which arise in development process frequently. TDD allow developers to change the code easily.

- **Teach learning:** Unlike the traditional methodologies, XP focuses on innovative way to do coding, testing and managing development process.

- **Small initial investment:** XP use less budget and fewer resources at the initial phases of the project which help to focus on lean design and code.

- **Play to win:** XP developers' focus on today's task rather than focus a lot of thing. This behavior help to do work more relax way where developers don't have to think about the project schedule and deadline. Every day the work should be done perfectly and successfully which lead the project to success.

- **Concrete experiment:** In XP methodologies, each requirement has to test which give direction for the developers whether they are doing right planning or wrong. Through the dirty proof-of-concept, XP developers' draw the outline their decision.

- **Open, honest communication:** For the good design and coding, constructive criticisms are welcome to do. It help to improve the coding quality as well as it shows the courage of the team member.

- **Work with people's instinct, not against them:** It is human nature to compete with each other in team. XP methodology recommend that team member don't do anything which go against other work.

- **Accepted responsibility:** In XP project, the whole team is responsible to deliver the project. XP team has the coach to direct them with the process rather than having manager to assign work. There is another role which is project manager, help team to do the administrative work. The development team decide the task and assign.

- **Local adaptation:** People and circumstances changes frequently and people are the key part of software. Local adaptation is required to gain success for the project. The team will select which process they are need to adapt for the development process.

- **Travel light:** Travel light means the team and process artifacts should be simple which help to follow easily. Whenever the team need to change direction, those artifacts help to do that.

- **Honest measurement:** XP methodology recommend to measure only those level which are needed and valuable for the project. Accuracy and precision has to focus for the measurement.

8.1.3 Key practices of XP

To implement XP value and principals in the project, XP practices needed to mention. Each method must define practices which provide guidelines how to apply XP. XP has 13 practices, among them 10 practices can be divided into four categories to understand well (Stellman & Greene 2015, 210). They are described below:

- **Category 1 - Programming Practices:**

Programming practices focus on test-first programming and pair programming practices which directly related with the programmers to write the better code.

1. Test-first programming:
Test-first programming means developers write test before starting coding. Obviously, this test will fail because there are no code but this failing test help to find problem and prevent defects in the development. These automated test are called by unit test.

2. Pair-programming:
Pair programming is a unique practice of XP. Beck stated that, “If someone refuse to pair, they can choose to look for work outside of the team” (Leffingwell 2007, 38). In pair programming, two developer work together to do coding, at the same time pair review also done which induce the quality of the code.

- **Category 2- Integration Practices:**

Integration practices category include two primary practices of XP. Those are:

3. Ten-Minute Build:
An XP team should be able to build the whole system automatically where all the tests run within 10 minutes and generating report contain which tests are passed and which are fail. The time should be under 10 minutes, otherwise team can be unwilling to do this practice. Through ten-minute build practice, team member can able to know which tests are functioned and which tests are need to re-work.

4. Continuous Integration:

Martin Fowler define continuous integration as:

“A fully automated and reproducible build, including testing, that runs many times a day. This allows each developer to integrate daily, thus reducing integration problems.” (Leffingwell 2007, 169).

Continuous integration has three steps: source code integration, automated build management and automated build verification testing. By following those steps, projects contains fewer bugs and project always being monitored for the fault.

- **Category 3 – Planning practices:**

As an agile methodology, XP based on the iterative and incremental development (IID) to manage their projects. Weekly cycle, quarterly cycle, stories and slack practices are under the planning practices category which are described briefly in below:

5. Weekly cycle:
XP use weekly iteration concept like Scrum. Team decide tasks in weekly basis and do the tasks in daily basis. At the end of the week, iteration must be completed to reach the next iteration. No weeks can go without progress.

6. Stories
Stories are the functionality of XP where it use instead of calling by requirements. Stories can be change during the implementation and prioterize on the need base. Stories are choose for the weekly cycle and divide into task to assign to developers.

7. Quarterly cycle:
Major Deliverables such as- themes, epics, release etc. should be planned in quarterly (Leffingwell 2007, 37). Quarterly cycle help to see the big picture of the project and connect them with the real time business problem.

8. Slack:
In XP, slack means low priority stories that can be ignored or skipped if the iteration has high priority stories has to fulfill.

- **Category 4 – Team practices:**

Final category is team practices which focus on the team behavior. XP values based on the team and there are two practices emphasize on team.

9. Sit together:
The name implies itself the meaning of sit together. Stated above, Pair programming is the unique practice of XP and to do so, it needed to sit together. Though there are contradiction about sitting together because developers need private and quiet environment to work efficiently but at the same time, it quite beneficial for the coding through constant discussion.

10. Informative workspace:

Communication is the first value of XP. Each team member need to know about the current state of the project whether developer or project manager. To know about the highest requirements, planning for the current and next iteration, workspace should be informative. Workspace can be informative by different way like task board.

- **Rest of the practices:**

11. Whole team:

In XP, project should always be functional state. To do this, multiples skills are needed to do that. Just only programmer or project manager or coaches can’t deliver the project. Whole team can work together to manage and deliver the project.

12. Energized work:

In workspace, team member just focus on the project related work to be effective. XP methodology recommend to do 8 hours work per day to be productive. Respect is the value of XP which requires that team member should be dedicate to their work.

13. Incremental design:

XP team design the project little by little. In opposites of Waterfall, in XP methodology, teams invest time for the design in daily basis and the design increase incrementally. It helps team to redesign and refactor the code if any complexity discover in design. For incremental design nature, the project don’t need to focus the whole design, just focus on the complex or problem areas.

8.1.4 Strengths, weaknesses and applicability of Extreme Programming:

Extreme programming has lots of strengths. Some of them are described below:

Strengths:

- This methodology embrace changes any stages of the development period. It is flexible to adapt changes in the requirements.
- In XP methodology, customers are strongly involved in the process which increase customer satisfaction.
- Through the constant feedback loop, integration and refactoring of the code increase product quality.
- Developers can focus on the programming sector rather than doing administrative works.
- Developers focus on daily task which put less pressure on them and deliver project on time.

- At the end of the each iteration, small release of tested software lead that software is in functional state.

Weaknesses:

- Sometimes pair programming leads to less productivity. Two heads working on the same thing. The good side is two pair of eyes are looking same thing and thinking about the same thing. But at the same time, one programmer sit idle which unproductive.
- XP methodology is team oriented and team members has to maintain communication among the team members which sometime time consuming.
- Though constant feedback loop increase the product quality but along with it gives chance to customer change their requirements.

Applicability:

- XP work well for small project rather than for the big project because XP methodology is design centric. Lack of documentation causes a huge problem for the large project.
- The programming part of XP are mostly done by the pair programming which also not suits for the large project.
- Like other agile methodology, XP suits for the new technology where requirements can change in any phase.

8.2 Scrum Software Development Methodology

8.2.1 Basic facts of Scrum:

Scrum methodology based on the iterative and incremental practices which use to manage complex software development process. The term Scrum comes from an article published by Hirotaka Takeuchi and ikujiro Nonaka in the Harvard Business Review in 1986. The title was “The New New Product Development Game” where Takeuchi and Nonaka compared to the Scrum formation in rugby (Pham & Pham 2012, 5). In the early 1990s, Ken Schwaber used Scrum at his company. In1993, Scrum processes was first applied in software development teams in Easel Corporation where Jeff Sutherland and other developer built the first object-oriented design and analysis (OOAD) tool that incorporated round-trip engineering in the initial Scrum-based project (Sutherland 2004, 1) Jeff and Ken working together and created a new software development methodology based on their learning in 1995.

Scrum is based on an empirical process control model rather than the traditional defined process control model (Schwaber & Beedle 2002, 89). When activities are so complicated and complex that they can't be defined in advance and aren't repeatable, they require the empirical process model (Schwaber & Beedle 2002, 100).

According to Schwaber, when the define process control can't work well because of the complexity of intermediate activities, empirical process approach employed in the development systems. (Leffingwell 2007, 46). To achieve better concept of the scrum process, empirical process control is needed to understand well. At below, empirical process control briefly described:

8.2.2 Empirical Process Control:

Leffingwell stated empirical process control as fundamental tenant of Scrum (Leffingwell 2007, 45). By nature, software development is a very complex process and not easy to predict. A single fault can destroy the whole project. There are so many risky factor going around the development process such as technology, customer requirements can be changes, productivity among the team members vary, delivery urgency and so on. There are two major approaches for the software development process. First one is define approach which including planned and predictive control. The second one is empirical approach which including measured and adaptive control. Empirical approaches are needed when the process can't be determined by the defined approaches. Empirical approaches has three control: Visibility, Inspection and Adaption. They are briefly discussed at below: (Leffingwell 2007, 46).

Visibility: Visibility control requires to see the intermediate steps of the process. In complicated software development process, every step is important to reach the final steps. "Scrum provides direct visibility into the progress of the project" (Schwaber & Beedle 2002, 69). The outcome of each process should be visible and clear to everybody involved in the project.

Inspection: Inspection control allow to evaluate and measure the immediate result. Based on the result, next iteration or step starts which direct to a successful end. Each process need to inspect from various aspect to identify any detect in the process.

Adaptation: Adaptation allow to accept changes in any stages of development cycle. This is one of the most important control of the scrum to accept constant changes.

8.2.3 Scrum principles and practices:

Scrum principals are based on the principal of new product development process defined by Takeuchi and Nonaka (1986). Those principals are applied in Scrum and key practices of scrum are related to those principals. The principals are new product development process are listed below (Leffingwell 2007, 44):

- Built-in-instability
- Self-organizing project teams
- Overlapping development process
- Multi-learing
- Subtle control
- organizational transfer of learning

Based on the above principals, scrum key practices evolved. Scrum key practices are described below (Leffingwell 2007, 44):

- Cross-functional and collocated teams of eight or fewer team members develop software in sprints.
- Sprints are iterations of a fixed 30-day duration. Each sprint delivers incremental, tested functionality of value to the user.
- Work within a sprint is fixed. Once the scope of a sprint is committed no additional functionality can be added except by the development team
- The Scrum Master mentors and manages the self-organizing and self-managing teams that are responsible for delivery of successful outcomes of each sprints.
- All work to be done is carried as product Backlog, which includes requirements to be delivered, defect workload, as well as infrastructure and design activities.

- The product Backlog is developed, managed and prioritized by the Product Owner, who is an integral member of the team and who has the primary responsibility of interfacing with the external customers.
- A daily 15-minute stand-up meeting, or "daily Scrum" is a primary communication method.
- Scrum focuses heavily on time-boxing. Sprints, stand-up meetings, release review meetings, and the like are all completed prescribed times.
- Scrum allows requirements, architecture, and design to emerge over the course of the project.

8.2.4 Scrum teams, events and process:

The Scrum Team:

According to Scrum principal and practices, Scrum team must be self-organizing and cross functional. Self-organizing team choose the best way to proceed project. Cross-functional team manage the option to accomplish project. There are three main roles in Scrum project. Product Owner, Development team and Scrum Master. Each role has distinguish responsibility to lead project. The responsibility of each role are described below (Sutherland & Schwaber 2013, 5-7):

The Product Owner:

- Product Owner is the main person to manage and express clearly about product backlog.
- To achieve the best goal for the project, product owner prioterize the items of product backlog.
- Evaluate development teams work performance to optimize value
- Ensuring the product backlog is clear to development team and keep track on the work of development team

The Development Team:

- Development team is self-organizing. Developer decide how they turn the product backlog into the incremental and functional deliveries.
- Development team should be cross functional. As a cross functional characteristics, all of the skills are required to create the product increment.

- There are no title for the development team member. Each member of development team is titled as developer.
- There are no sub-teams exists under the development team except testing and business analyst team. This is one of the core characteristics of development team.
- The whole team is accountable for the performance. Individually team member can have special skills and knowledge but Scrum focuses on the whole team.

The Scrum Master:

“The Scrum Master is responsible for the success of Scrum” (Schwaber & Beedle 2002, 31).Scrum Master is responsible for three distinguish part of the project. Scrum Master has to give service for the development team, product owner and the organization. The responsibilities are described below:

Role	**Service for Product Owner**	**Service for Development Team**	**Service for Organization**
Scrum Master	Finding the best technique to manage the product backlog.	Provide guidance to the development team in self-organization and cross-functionality.	Help to adopt and implement Scrum inside the organization.
	Ensuring product owner arrange product back-Log to gain maximum success.	Helping development team to follow scrum practices to provide high-value products.	Doing changes which increases productivity Of the scrum team.
	Empirical environment Is applied in product planning.	Coaching the development team to follow and adapt Scrum principles and practices.	To increase the effectiveness of the Scrum application, work with other Scrum masters.
	Ensuring agility is understand and practiced.	Help to remove obstacle in the way of development team.	

	Facilitated Scrum events based on the project requirements.	Ensuring development team follow and implement Scrum during the whole project.	

Figure 3: Responsibilities of Scrum Master

Scrum Events:

Scrum is a very organized methodology and Scrum events play an important role for the Scrum project. All of the events are designed to achieve the best goal. Each events has a time box and when an event start, it can’t be cross it time limit. Scrum events are necessary to inspect and adopt the progress critically. Scrum events are consists of the following events:

- The Sprint
- Sprint Planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective

The Sprint:
Sprint is considered the heart or most important events of Scrum. It is a time-box which duration is one month or less. Sprint deliver fully functional product increment at the end of the sprint and based on the success of sprint, next sprint starts. Each sprint has specific planning to deliver the product. During the sprint, no changes can’t be change and quality goal can’t be reduced. Scope and functionality can be discussed with the product owner ((Sutherland & Schwaber 2013, 7).

Sprint Planning:

During the Sprint Planning meeting, the whole team decide what they deliver and how the work to deliver. The planning can’t be change during the sprint. Scrum Master ensure that all of the team member understand the requirements well. Product owner present the product backlog to the developer but the ultimate decision depends on the developer. They select and assign work for the upcoming events. At the end of the sprint planning meeting, the sprint goal and planning should be exists.

Daily Scrum:

Daily Scrum is a regular event perform every day during 15 minutes. Three important matters are discussed in daily scrum. They are: What I did yesterday, what I will do today and is there any known obstacle to meet the goal. Communication is so important in software development because of its complexity. Daily scrum allow the team member to communicate effectively.

Sprint Review:

During Sprint Review meeting, all of the parties can inspect the development of sprint together. Product Owner explain what done and not done in product backlog, development team presents their product increment and answer to product owner.

Sprint Retrospective:

Sprint Retrospective event allow the development team to inspect the progress and find out what went well and what item need improvements. So it likes a team judgment to perform well and remove the same obstacle for the upcoming sprints.

Events	Duration	Attendance
Sprint Planning	8 hours	Entire Scrum team
Daily Scrum	15 minutes/day	Development team
Sprint Review	4 hours	The Scrum team and stakeholders
Sprint Retrospective	3 hours	Developers and Scrum Master

Figure 4: Sprint Events

The process of Scrum:

In real life, the Scrum process is not as simple as the below figure. But this figure shows the fundamental process of Scrum which vary depend on the project requirements (Leffingwell 2007, 47)

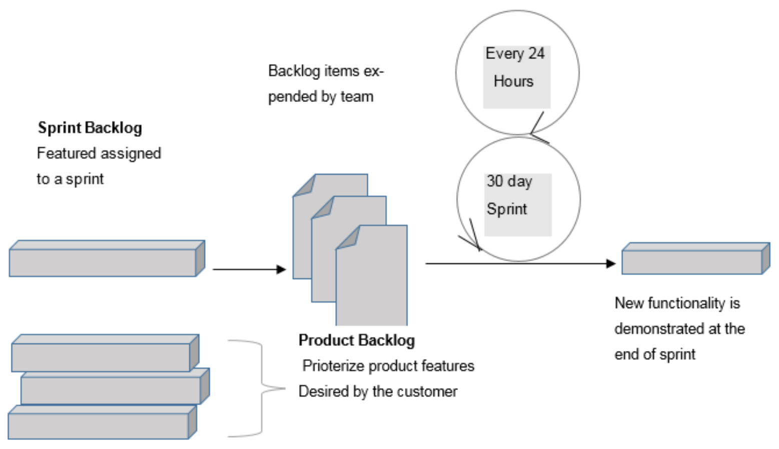
Backlog items ex-
pended by team
Every 24 Hours
Sprint Backlog
Featured assigned to a sprint
30 day Sprint
Product Backlog
Prioterize product features
Desired by the customer
New functionality is demonstrated at the end of sprint

Figure 5: Scrum Process Model Overview

This figure illustrates the basic process of the Scrum which based on the empirical model. There are two inspections cycle going on the every Scrum process. First one is daily basis and the second one is 30-day basis. Product backlog provides by the Product Owner which contains desired functionalities of the customer and then backlogs are assigned to the sprint. Next team start to work on it. Through the daily and monthly basis, the inspection model continuing in Scrum. At the end of the sprint, team should deliver a tested and fully functional product increment to the customer.

8.2.5 Strengths, weaknesses and applicability of Scrum

Strengths:

- Scrum focus very well on management and organizational aspect of enterprise (Leffingwell 2007, 48).
- Constant inspection and feedback loops allow the Scrum team to address and solve the issues.
- Product owner is a team member of the Scrum project which remove miscommunication with customer. Development team can always discuss with product owner if something need to be change.
- Less documentation and management work help developers to focus on their work.
- As an agile methodology, Scrum also accept changes any phase of software development and short sprint help the development team to accomplish it.
- Development team decide the workload and assign by their own discussion which relief them from bossy management.

- Due to constant inspections, the bug rates are fewer in final product.
- Customer get fully functional product increment after every sprint so that they can able to see their product partly functioning.

Weaknesses:

- In Scrum, documentation get low priority which sometimes leads situation towards bad.
- According to Scrum methodology, the Scrum team must deliver a fully functional product increment after every sprint by which developers might be feel stress.
- Development team decide the workload for each sprint but no structured working processes are involved.
- Development team focus mostly on the sprint and low visibility over project outside sprints obviously a weakness of Scrum.
- Management get low priority in Scrum. Most of the tasks are titled towards the developers and stakeholder value is not clearly shown in Scrum.

Applicability:

Scrum applied in numerous projects over the world and it has been well adopted by thousands company. But like other methodology, Scrum principles and practices doesn't suits every projects. The applicability depends on some factors. They are described below:

- Scrum is not work perfectly for the large and distributed teams. According to Scrum guides, the perfect Scrum team size is fewer than 8 peoples. Though Scrum adopted company can manage large-scale project by dividing teams but the small-scale projects suits better than large-scale projects.
- Scrum team is self-organizing and cross functional. If the company doesn't focus on the team, it might not good to apply Scrum.
- There are some projects where continuous integration and testing is not feasible to develop their products. Scrum doesn't work well for them also (Leffingwell 2007, 48).
- Scrum accept changes in any stages of the development cycle which is difficult for the large-scale and mission critical projects.

- Scrum work well where requirements can changes frequently like new technology or web development. The product owner continuously keep eye on the changes and manage the product backlog constantly. These kind of project are most suitable for the Scrum.
- The development cycle is short in Scrum and the team deliver product increment after each iteration so that in small-scale project, Scrum works well.
- Scrum can combine any other software development methodologies based on the project nature.

9. Agile Project Management (APM)

Agile Project Management (APM) is a value-driven approach where project managers has to deliver high quality work to their customer. The APM practices are not new but the implementation of the practices in project is a new form. APM help to draw conceptual project framework with useful guidelines to manage projects. APM defines a strategic capability to identify, create and respond to changes and lead the project through uncertainty and challenge.

To understand APM, it needs to focus the business value of APM, they are listed below (Highsmith 2004, 6):

- Continuous innovation- focus on current customer requirements
- Product adaptability- focus on future customer requirements
- Reduced delivery schedules- streamlining processes and focus on skilled people.
- People and process adaptability- respond and adapt business changes fast.
- Reliable results- Deliver innovative results within cost and schedule constraints.

APM framework:

APM framework consists of five correlated and supporting practices. These are listed below (Highsmith 2004, 18):

- Envision- Define project vision and scope, decide how the team work.
- Speculate- focus on the product specifications and plan how to support to the vision.

- Explore-Preliminary features are tested and implemented, focus to reduce uncertainty and risk.
- Adapt- Review the delivered result and adapt if needed.
- Close- conclude the project and pass the key learnings.

The characteristics of APM

The characteristics of APM are varies depend on the project nature and size. But there are some common characteristics which most of the APM has. They are listed below:

- Teams are self-organizing and self-directed.
- Requirements can be change during the project lifecycle.
- Testing and customer feedback happen constantly.

The successful implementation of APM depends on many variants. APM is strongly depended its values, principles and practices. In current uncertain and turbulent world, company's success depend on the creation of and respond quickly to changes. Changes are acts as a disruptor for competitors. APM support to develop new products where requirements are changing frequently during the project development cycle, reducing product development time and creating new sales channel.

10. Data collection

According to the thesis topic nature, interview method was select as data collection method which performed by conducting 3 interviews where the interviewee has prior knowledge and work experience on different software development methodology. First interviewee focus on traditional methodologies whereas second interviewee focus on agile software methodologies and third interviewee mostly answer based on knowledge and personal preference. At below the expertise of the interviewee listed:

- 1st interviewee: Agile software methodology Expert (Former Scrum Master)
- 2nd interviewee: Traditional software methodology expert (Have a very good knowledge on Scrum).
- 3rd interviewee: A good knowledge on both Agile and traditional software development methodologies.

The questions of the interview are attached in Appendix 3.

Interview structure was semi-structured where interviewee mostly answer narratively and for some questions they got pre-defined options but most questions they answered descriptively. Interview was recorded under their consent. After taking 3 different interviews, no interview was conducted because a certain level of information had been achieved and not much new insights are needed.

11. Data analysis

All information, collected by interviews will be discussed in this section. Interview questions are categorized in 10 different categories. Those are analyses in this section based on the interviewee's answer.

Category 1: Software capability quality standard

The purpose of asking software capability quality standard related question to know how this standard practice in the project and which software development methodologies use this standard a lot. First interviewee answered that, his company currently use Capability Maturity Model Integration (CMMI) version 1.3 where the level is now 4 but a couple of years ago, it was 2. CMMI has five level to measure the project. He stated that, CMMI is very important not only for the process improvement but also it provide guidelines for the stable and mature process. For project planning, they use CMMI mostly 100% for XP and Scrum projects and their level going around between 3 to 4. According to second and third interviewees' statement, their company use different maturity standard for different sector but mainly they are managed by other department not the software development team. Based on the interviewees' answer, whether it is agile or traditional, software quality standard was very useful to measure the maturity and quality of the software.

Category 2: Company's nature while adopting new methodology

Company play different role when they adopting new methodology. Couple of company's nature listed below:

- Market leader (adopt and implement new technology in their total market)
- Market follower (follow those company who are successful after implementing new technology)
- Conservative (Those company implement new technology when it lasts for long time)
- Static (not willing to adopt new technology)

First interviewee said that his company by nature one of the technology leader but during last ten years, company was strictly follow agile methodology for all projects, so according to this, it behaves static. Second interviewee told that his company stand between market follower and conservative position, mostly near to conservative. According to third interviewee's answer, company's position as a market leader when he was working there. Now the discussion for this question start from here. This question try to find out the scenario that when company used traditional methodology for manage their project, what was their nature in market and what is the nature when they are using agile methodologies. According to interviewees' statement, it was clearly shown that the company who use agile methodologies during last 10 years, they are static because it helps them to manage their development procedure. Second and third interviewee had work experience on traditional methodologies and they choose market follower and market leader option which shown that, while they use traditional methodologies, they are looking for new technology which can support the development.

Category 3: Knowledge, preference, likes and dislikes of traditional software development methodology

All respondents has knowledge about the traditional or heavyweight software development methodology but only second interviewee had the work experience on this. Rest of the respondents answer that questions by knowledge from their institute and personal preference. For the knowledge, second interviewee's statement will get more focus because he has the work experience from 2002. He told that, his company use pretty much waterfall and there were separate phase for analysis, design and implementation phase. So the project mostly use waterfall. All respondents opinion about the good characteristics of traditional methodologies was- following plan strictly but at the same time the dislike of traditional methodologies, they answered all is- not respond to changes. In most of the traditional methodologies, one was common that requirements are designed once and development start after that and no requirements were not change and accepted after requirement analysis phase. That was a strong point to dislike traditional methodologies. Second interviewee had work experience with waterfall and spiral methodologies and he preferred spiral is more manageable than waterfall because spiral concept comes later than waterfall and most of the weak sides of waterfall are absent in spiral methodology. According to first interviewee, some of the spiral concept are still use in his company but it doesn't call as spiral. Another weak point is heavy documentation. Same information are written in several places. So when it need to change, it has to be change in every places separately which requires a lot of time.

Category 4: Knowledge, preference, likes and dislikes of agile software development methodology

All of the respondent has knowledge about agile methodologies which is above than average and all have the work experience on the Scrum software development methodology. According to the first interviewee's statement-"Scrum is the best way to allocate limited resources"". He also told that investment wise it good to develop software project by following agile methodologies especially Scrum. His company's supporting unit use Kanban to manage their work. According to second interviewee Scrum is the best methodology for the software development which has a strong customer collaboration. Third interviewee talked about Scrum and FDD methodologies. Currently he is doing a project where his team uses FDD methodology to manage their work. All respondent like customer collaboration characteristic for the agile methodologies. According to second interviewee, when his team was use traditional methodologies, mostly waterfall, they were always late in requirement phase, design phase and implementation phase. After that, most of the time, the customer was not satisfied. So customer collaboration is the strong nature of agile methodologies. The weak point of agile methodologies is low documentation which on first and second interviewee talked about. Because of low documentation most of the time reworks are needed though first interviewee told that heavy documentation and Scrum is the two opposite's side of a same coin but people orientation can replace documentation. Third interviewee choose lack of process structure as the second weak point of agile methods. As a summary of this section, Scrum is very popular to manage software development project in the current world and the customer collaboration is the strongest point of agile methodology where customer can change their requirements and can able to see the small release before the final product release.

Category 5: Behind factors of choosing specific software development methodology

Based on the interviewees' reply, the summary for this category is there are several aspects play behind when a company choose a methodology for project. Now a day's most of the product owner decide or discuss to follow specific methodology for the project. Also development team has opinion about that. Other factors are: requirements types, features, development time and cost also play role to select development methodology. Project size, development team size, communication between customer and developers also influencing factors when choose development methodology. According to first interviewee's statement, experienced company and team can choose the development methodology by an easy effort where new company has to give strong effort because if the development methodology not match with the project, it can’t reach to the solution properly.

Category 6: Effect on software development while choosing agile methodologies

According to first interviewee, agile has different variants and it is flexible and allow to do work incrementally where requirements are getting step by step in a small chucks and customer can see their product before the final release. So agile of course work best than rational methodologies. Second interviewee told that in traditional methodology, there were no phase for the evaluation so the quality decrease but by following agile methodologies, not every stages increase in quality but most of the stages eventually increase the quality because of the faster iteration by which developer and customer can evaluate and improve the product. Third interviewee told that, by following agile methodologies of course increase the quality of the software development because agile come after the traditional methodologies where the weak points of traditional methodologies of course not present in agile. All of the respondents told agile methodologies work best in the current world for software development but may be parts of the project can follow some aspects of traditional methodology so by this way it can be mix methodology but the portion of agile should be triple out of the total. Second interviewee told that the database implementation can follow the traditional methodologies still now. It will be good if project can use the strong part of the traditional methodologies and rest of the parts can follow specie agile methodology.

Category 7: Factors which increase or decrease bug rate, development time and cost

Based on the first interviewee’s experience, agile methodologies goal to release software product without bugs or fewer bugs but if there are always a release rush, minor details or low priority features can’t possible to focus. According to his opinion, one week release is not helpful for the developer because they don’t get enough time to think but if the iteration time 2 week to one month, then developer can focus on the minor details also though developer take task in every sprint by their ability but it good to release product by minimum two weeks. His team do weekly scrum and after two weekly scrum, they have one release. By focusing small details, it decreases the bug rate. The development time and cost also decrease by following agile methodologies because time to time the customer get fully functional product increment after each iteration. According to second interviewee, process model is advance in agile which decrease development time and cost. He also said that iterative and incremental methodology also decrease the bug rate where developer test first and put more feature to it. Third interviewee told that agile support iterative and incremental way to do work which help to decrease bug rate along with development time and cost. Based on the interviewees' answer, agile methodology help to decrease bug rate because step by step developer get the requirements and by strong collaboration with customer, developer understand the requirements of customer very well which help to release the final product on time and keep the cost figure unchanged.

Category 8: Common issues while transit from traditional to agile and key challenges while implementing agile methodologies

All of the respondent answer is similar for the common transition problem. They told it change the culture of the company. With the traditional methodology, management habitually order and request to release products by some sudden decision or they have the chance to order to the development team but in agile, management have to wait until the existing sprint over and before the next release they can request about some issues. In Scrum, once the release over, requirements comes from the product backlog for the next phase. So the culture of the company change specially the management changes while transit from traditional to agile methodologies. According to second interviewee, along with the culture of the company, the attitude of the people and in some cases, tools need to be change because might be the tool support the traditional methodologies but when transit to agile, it needed to change tools to do work with agile methodologies for example travel agencies. Third interviewee told about the above points with another matters, sometimes interior office design need to change while transit to agile. Because in XP or Scrum, they needed pair programming and product owner need place in office. So the arrangement of the office also change. Based on their answer, when a company transit to agile methodologies, several issues can be occur but people can adjust with new system or nature by time and explore the new way to manage work.
The key challenges to implement agile methodologies depends on many variants. The interviewee told about different challenges but among them one was common. Agile mythologies does not need skilled people but it really difficult to expert on this. According to second interviewee, attitude must be bigger problem than skills. Third interviewee talk about lack of top management support for the agile methodologies. Management don't play very active role in agile but at the same time they are happy because company earn money due to short development time, cost and fewer bugs by following agile methodologies. Based on the information from interviewees', most of the company welcome agile though it need some time to adopt new way of working.

Category 9: More on Scrum

In this category, there are 3 sub-categories which will be discuss in this section. They are listed below:

- How effective the daily Scrum meeting for the developer
- Scrum only for small-scale projects
- To what extent, agile methodologies are used in the company

Sub-category 1: Effectiveness of daily Scrum meeting for the developer

According to the first two interviewees', daily Scrum meeting has a strong effect for the development team where team can discuss about the daily progress. It helps to deliver the sprints without stress. It is an effective way to manage the progress and according to the Scrum guide, 2 minutes per people in daily Scrum meeting. According to first interviewee, it is effective not only for the developer but also for the management because developer is responsible and accountable what they are doing and for management, they can know about what development team doing and how they are progressing. So daily Scrum meeting increase the visibility of the work during the sprints. Third interviewee's opinion about the daily Scrum meeting is sometimes it seems so routine work to participate daily Scrum meeting to inform about the progress where in the current world, everyone use so smart technical devices such as skype, email, Facebook for communication. Based on their opinion, both aspects are correct. Developers are very smart people but at the same time they are not willing to talk much or usually don't prefer routine work. From this aspects, third interviewee is right. But at the same time, communication is so important for the developers to inform what are they doing so it can avoid re-works. From this point of view, the aspects of other interviewees' are right.

Sub-category 2: Scrum work well only for small-scale projects

First two interviewees' opinion about that, it's not true that Scrum work well only for small-scale projects. According to second interviewee, there is one rule that not more than 9 developers should work in same team but for big project, there can be several Scrum team where the whole work are divided which called Scrum of Scrum. The first interviewee told that Scrum is a way of managing software development whether it is large-scale or small-scale projects. For large-scale projects, release can be every one-two months and there are several team work on it to develop the product. Third interviewee has the different opinion. He told that Scrum work well for the ongoing projects where the projects life span less than 4 months because sprints are designed to do in one month and for the long project, if the customer change requirements in several times, it will difficult to manage the development work. Every variable in Scrum is design for a short period and for the long project it really difficult to manage the product and sprint backlog. Based on their opinion, there are two different opinion found. There are rumor going on Scrum not for the mission critical and government project because the project life span is usually more than a year and it is difficult to find product owner and some political issues can be exists but for the small-scale projects, Scrum principals, values, artifacts match with the project duration. Actually so many variants have to measure when it is a large-scale project. Any kind of project is possible to manage by Scrum but it needs careful planning and expert development team to manage the development.

Sub-category 3: To what extent, agile methodologies are used in the company

This question answered only by first interviewee because he is currently working in the company. His company use 100% agile for the software development but still some aspects of spiral methodology use partly for developing software. Second and last interviewee had the work experience mostly on traditional methodologies.

Category 10: Which factors play important role for the success of software projects

Based on the interviewees' opinion, the summary for this category are: communication, customer collaboration, proper requirement analysis, willingness to learn and skilled person. Those elements were talked about by the interviewee. First interviewee mostly focus on the communication. He told that there are so much oral communication happen in agile team. So customer and development has to speak without ambiguity. Both parties need the clear understanding. Second interviewee focus on skilled people, attitude and willingness to learn matters. He told that skilled people is an asset for any company who can analyses development stages beforehand and give useful suggestion to gain competitive advantages. Third interviewee focus on customer collaboration and proper requirement analysis. He told that without proper requirement analysis and good design, it is not possible to satisfy the customer fully.

12. Discussion

12.1 Limitation of traditional and software development methodologies

Both traditional and agile software development methodology has its own strengths and weaknesses but in the discussion section, only the limitation will be discuss because limitations or shortcomings need focus to make correction or improvement. The limitations depend on different variable and they are not same in traditional and agile software development methodologies.

Traditional software development methodology:

One of the biggest limitations of traditional software development methodologies is unwillingness to respond changes. Once the analysis and design phases are complete, traditional methodologies doesn’t welcome to get new changes. For this approach, sometime customer not satisfied on the final output. Heavy documentation is another common shortcomings. According to interviewees’ opinion, this is the second big shortcomings of traditional methodologies where the team have to spend too much time for the documentation and perhaps same information are documented in several places. For each phase of the project, traditional methodology requires documentation for which developers can lose their motivation and focus to concentrate on the project. Up-front design is comprehensive which is also consider as limitation of traditional methodologies where team has to spend time for the analysis and design phase before the coding part is start. Because of this nature, team has to spend time to think about different potential factors which can be arise after the coding part. The design and analyze phase are complete at the beginning of the project and after that any kind of changes are not easy to do. So team has to spend time to predict issues which can be happen in future. Command-control culture is another limitation. Because of the predictive approach, the management know what the final output are so they behave autocratic to the team and sometimes give sudden command to complete part of the project because of the pressure from the upper management. This direct to the conflict between development team and management.

Agile software development methodology:

The first limitation is difficult to manage large-scale projects. According to agile methodologies principals, they work best on small-scale projects. Like Scrum, team size should not be more that 9 people. In XP, pair programming is mandatory according to their principals. Projects who implement agile methodology to manage their work, have to deliver the product in short period which also one reason that agile suited best in small-scale project.

In large-scale projects, the team size is big, sometimes based on the project size, 200 people can be under one team which really not suitable with agile methodologies. Agile methodologies respond to changes any stages of software development lifecycle which also a reason being difficult to manage large-scale project. The next limitation is lack of documentation.

Though the heavy documentation is a limitation of traditional methodologies but in agile, less documentation is a limitation. Due to lack of documentation, sometimes developers need to re-work for similar projects. Required skilled people also count as a limitation of agile. In agile methodology, projects lifecycle is usually short and being creative to innovate deliver products fast. To manage customer requirements, agile team work efficiently and provide functioning increment in each iteration.

12.2 The combination of proper selection and adoption enhance the success of the software development project

The success of software project is mostly depend on the development methodology. The proper selection and adoption of the methodology guide the project to success. To gain competitive advance in this current world, it’s really important how the project is managed and how swiftly organization deliver the products. It’s important for the organization to perform research on the previous projects to identify which methodology work best and which was failed so that the behind factors can be determined. The first step to select the appropriate methodology for the specific project and the next step is to adopt the methodology in the organization. For the success both steps are need to combine to lead the project to success. Most of the IT projects fails due to poor management where a lots of issues involved. Some of them are: Scope are not well defined, poor project planning, requirements are not properly analyzed, lack of communication with customer etc. The selection of proper software development methodology help management to avoid these issues. The adoption of methodology also a crucial part where lots of challenges are involved such as, the organization culture, the team skill, communication, ability to adopt new technology and methodology etc. The way of doing work is depend on the methodology. The proper adoption is not easy for the team so it needs expert management to observe team performance. There are some important issues involves while organization adopting and implementing methodology in the project. Some of them are: team need dedication to learn the methodologies principals, values and practices which needs extra time and effort, information should be available inside the organization, small pilot projects need to organize to see how the methodology works in reality.

In the current world, changes happen so frequently and in software development organization, customer need innovative and new product to win in competition. Besides that, project lifecycle can't be long. So software development organization need to select and adopt appropriate methodology to deliver product which meet customer satisfaction.

12.3 Influencing factors for the selection of development methodology to manage and execute software project

Selecting software development methodology isn't an easy task for the organization. Lots of factors are taking in consideration while selecting development methodology. Some of the basic and common influencing factors are: organization structures, project nature, product features, development time, cost, development team size, communication between customer and developer etc. Among them two factors play vital role. Those are: Organizational structure and project nature. Others are considers as secondary or sub factors. In current world, agile methodologies are very popular in software industry. But not every agile methodologies are getting the same popularity. According to VersionOne's.com, Scrum is the most widely use agile methodology because of his superfast adaptability character. Scrum methodology can be compatible to different kind of organization and it can deliver products in short development time with fewer bugs rate. So when company select the methodology not only the main factors are considered but also other sub-factors come into the ground. To avoid project failure, company examine and determine carefully the methodology. Last but not least, the company culture plays also very important role to select methodology. If the company is resist in nature to welcome changes, company stick to the previously used methodology where the influencing factors are not well examined.

To survive in current business world, every resources need to carefully use. To manage and execute project company need to focus all of the factors whether they are primary or secondary while choosing appropriate methodology. The ultimate project success depend on how the project is managing and deliver to the customers where appropriate methodology shows the right way to reach to the success.

13. Recommendation

13.1 Guidelines for management support while adopting Scrum

This section provides guidelines for those organization who are interested to implement Scrum or already using in the organization. Some common issues arise while adopting Scrum in the company. Guidelines are listed below:

1. Manager must aware about the knowledge and skills levels of the team members about Scrum.
2. Need a skill Scrum master to observe team performance and guiding the team when necessary. According to Scrum principals, development team is self-organizing and assign work by themselves. Scrum master should be monitor progress of the team.
3. Allocate budget for the training session by which team members can able to up-to-date.
4. Customer collaboration is very important characteristics in Scrum methodology. To implement Scrum in organization, company must arrange meeting with the customers and welcome them to take participate in decision making process.
5. Organization must change the working environment while adopting or utilizing Scrum. In Scrum, daily Scrum meeting is the crucial events to discuss with each other about the project so company need to focus on the interior design to help team member to communicate easily.
6. Product owner is one of the team member of the project. So product owner has to work in the same place where developer doing their work. So developer can discuss and clarify requirements whenever they need. Organization must create collaborative workspace for the Scrum team.
7. Scrum strongly depend on its principals, values and practices. Scrum master should ensure that each of the events, artifacts are well followed in the team.
8. Scrum team provides fully functional increment after each iteration. Organization must provide efficient bugs tracking tools and project manage tools for the team.
9. Daily burndown chart should be maintain to track the team progress.
10. Company must be prepare for the changing culture which includes way of working and attitude of the people.

13.2 Avoid common issues for the transition from traditional to agile

Transit from traditional to agile method requires lots of changes inside the organization. It's not only about the way of managing project but also it's the attitude, office culture and principals. There are some common issues arose before which can be avoid by taking some useful steps. It of course depend on the company's nature, location and other things. But some common steps should be done or taken before the transition. They are listed below:

- Teams must be organized according to the agile methodology and each member role should be clarify. In traditional, team size usually large and there are several groups for separate actions like verification team, requirement analysis team, design team, testing team etc. In agile, a team do all of the work by themselves. So team need to be organized.
- The schedule of the project should be fixed before project has been started and the iteration length need to be decided beforehand so that team member can assign the task based on the schedule.
- Weekly demo is needed if team decide two weeks iteration for the project. So that team can test their project inside company weekly and get finding so that they can make the correction before the iteration.
- Agile training is needed for the management and the developer. Each parties should be know their responsibilities to manage the project works.
- Communication is always a crucial point for every methodologies. Organization must need to provide open platform, required tools to manage effective communication inside the team. Company can have their own communication page which only active inside the company between stakeholders, management and developer team to share information.

14. Conclusion

A good number of research about software development methodology was done before but every research has a unique focus. This paper focus on the traditional and agile software development methodologies to find out how the proper selection and adoption can enhance the project success, how the organization avoid issues while transits from traditional to agile and how organization can implement Scrum. Scrum was focused more than others methodologies because of its popularity and demand in the current world. Selecting an appropriate methodologies is a very critical decision for software organization because budget, time and other factors are related with this. This paper can give insight for both organization and students to know about factors which play behind the decision of select software development methodology. The guidelines can help the start-up company who are new to the software grounds and it also can give useful information about the transition from traditional to agile methodologies.

In the current world, to win in the competition, every company need to response to the changes fastly and be strict in development time and budget. Agile methodologies comes from the shortcomings of traditional methodologies. So it will be beneficial for the organization to practice agile mythologies and take the strong part from the traditional so it can be benefitted from both kinds of development methodology.

15. References

Awad, M, A. A Comparison Between Agile and Traditional Software Development Methodologies. URL: http://www.unf.edu/~broggio/cen6940/ComparisonAgileTraditional.pdf. Accessed 27.01.15

Williams, C. 17 May 2013. Agile Manifesto – Working Software Over Comprehensive Documentation. URL: http://blogs.sourceallies.com/2013/05/agile-manifesto-working-software-over-comprehensive-documentation/. Accessed: 07.02.15

Fowler, M. The New Methodology. URL: http://dcc.uab.es/sites/default/files/21290/teoria/Fow2005-TheNewMethodology.pdf. Accessed: 07.02.15

Dooley, J. 2011. Software Development and Professional Practice. Apress Publication. ISBN-13 (electronic): 978-1-4302-3802-7. URL: http://it-ebooks.info/book/587/
Accessed: 07.02.15

Cho, J, J. 2010. AN EXPLORATORY STUDY ON ISSUES AND CHALLENGES OF AGILE SOFTWARE DEVELOPMENT WITH SCRUM. UTAH STATE UNIVERSITY Logan, Utah. URL: http://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1595&context=etd
Accessed: 09.02.15

Farrell, A. 2007. Selecting a Software Development Methodology based on Organizational Characteristics. ATHABASCA UNIVERSITY. URL:
http://dtpr.lib.athabascau.ca/action/download.php?filename=scis07/open/AdrienneFarrell_Final.pdf. Accessed: 09.02.15

Munassar, N, M, A & Govardhan, A. September 2010. A Comparison Between Five Models Of Software Engineering. IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010 ISSN (Online): 1694-0814. URL:
http://www.ijcsi.org/papers/7-5-94-101.pdf. Accessed: 07.02.15

Wikipedia 2015, Spiral model. URL: http://en.wikipedia.org/wiki/Spiral_model
Accessed: 28.1.15

Fairbanks, G. 2010. Just Enough Architecture. The Risk-Driven Model. URL:
http://www.crosstalkonline.org/storage/issue-archives/2010/201011/201011-Fairbanks.pdf

Accessed: 2.2.15

Boehm, B, W. 1988. A Spiral Model of Software Development and Enhancement. TRW Defence Systems Group. URL: http://csse.usc.edu/csse/TECHRPTS/1988/usccse88-500/usccse88-500.pdf. Accessed: 01.02.15

Boehm, B. Lane, J, A. Koolmanojwong, S & Turner, R. 2014. The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software (Google eBook). Addison-Wesley Professional. Pearson Education, Inc. URL:

https://books.google.fi/books?id=bi-tAwAAQBAJ&pg=PA104&lpg=PA104&dq=deliverables+of+spiral+model&source=bl&ots=T-OuGtNJD7&sig=xbpUlChH-0e7sl616Li-JYQ3vRdA&hl=en&sa=X&ei=bPzOVIn-NYKfygPJ9YKYBQ&ved=0CDMQ6AEwAzgK#v=onepage&q=spiral&f=false
Accessed: 2.2.15

Fairley, R, E. 2009. MANAGING AND LEANDING SOFTWARE PROJECTS. A JOHN WILEY & SONS, INC., PUBLICATION. IEEE Computer Society. ISBN: 978-0-470-29455-0. URL: http://it-ebooks.info/book/2754/. Accessed: 03.02.15

Agarwal, B, B. Tayal, S, P & Gupta, M. 2010. SOFTWARE ENGINEERING & TESTING. Jones and Bartlett Publisher. ISBN: 978-1-934015-55-1. URL:

Sabharwal, S. 2008. Software Engineering. New Age International Publisher, India. ISBN: 978-81-224-2377-8.

Fowler, M. 2000. The New Methodology (Original). URL:
http://www.martinfowler.com/articles/newMethodologyOriginal.html#TheMethodologies
Accessed: 08.02.15

Shore, J & Warden, S. 2008. The Art of Agile Development. Published by O'Reilly Media, Inc. ISBN-10: 0-596-52767-5

Cunningham, W. 2001. Manifesto for Agile Software Development. URL:
http://www.agilemanifesto.org/. Accessed: 08.02.15

Manifesto for Agile Software Development 2001. URL: http://www.agilemanifesto.org/

Accessed: 09.02.15

Mohammad, A, H. Alwada'n, T & Ababneh, J, M, A. 3 March 2013. Vol.5 No.3, ISSN: 0975-5462. URL: http://www.ijest.info/docs/IJEST13-05-03-045.pdf. Accessed: 09.02.15

Cockburn, A. 2007. Agile software development. The Cooperative Game. 2nd edition. ISBN: 0321482751. Pearson Education, Inc.

8th Annual State of Agile Survey 2014. VersionOne, Inc. URL: http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf. Accessed: 10.02.15

Dooley, J. 2011. Software Development and Professional Practice. Apress Publication. ISBN-13 (electronic): 978-1-4302-3802-7. URL: http://it-ebooks.info/book/587/ . Accessed: 12.02.15

Stellman, A & Greene, J. 2015. Learning Agile. Understanding Scrum, Lean, XP and Kanban. Published by O' Reilly Media, Inc. ISBN: 978-1-449-33192-4.

Dooley, J. 2011. Software Development and Professional Practice. Apress Publication. ISBN-13 (electronic): 978-1-4302-3802-7. URL: http://it-ebooks.info/book/587/ . Accessed: 12.02.15

Stellman, A & Greene, J. 2015. Learning Agile. Understanding Scrum, Lean, XP and Kanban. Published by O' Reilly Media, Inc.ISBN: 978-1-449-33192-4.

Pham, A & Pham, P, V 2012. Scrum in Action. Agile Software Project Management and Development. Course Technology, a part of Cengage Learning. ISBN-13:978-1-4354-5913-7

Schwaber K. & Beedle M. 2002. Agile Software Development with Scrum. Pearson Education International. Prentice-Hall, Inc. Upper Saddle River. ISBN: 0-13-207489-3

Leffingwell, D 2007. Scalling Software Agility. Best practices for Large Enterprises. Pearson Education, Inc. ISBN: 0-321-45819-2

Sutherland, J. 2004. AGILE DEVELOPMENT: LESSONS LEARNED FROM THE FIRST SCRUM. URL: https://www.scrumalliance.org/resources/35

Leffingwell, D 2007. Scaling Software Agility. Best practices for Large Enterprises. Pearson Education, Inc. ISBN: 0-321-45819-2

Sutherland J & Schwaber, K. 2013. The Scrum Guide™. The Definitive Guide to Scrum: The Rules of the Game. URL: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf#zoom=100. Accessed: 02.03.15

Highsmith, J. 2004. Agile Project Management. Pearson Education, Inc. ISBN: 0-321-21977-5

Appendices

Appendix 1 Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck	James Grenning	Robert C. Martin
Mike Beedle	Jim Highsmith	Steve Mellor
Arie van Bennekum	Andrew Hunt	Ken Schwaber
Alistair Cockburn	Ron Jeffries	Jeff Sutherland
Ward Cunningham	Jon Kern	Dave Thomas
Martin Fowler	Brian Marick	

Appendix 2 Earlier Projects

Project name	**Author**	**Focus area**	**University name**
Selecting a Software Development Methodology based on Organizational Characteristics	Adrienne Farrell	Organization characteristics	Athabasca University, 2007
An Exploratory Study on Issues and Challenges of Agile Software Develop ment with Scrum	Juyun Joey Cho	Scrum	Utah State University, 2010
New agile process errors in software development	Petri Koivisto	Agile process errors	Haaga-Helia University of Applied Sciences, 2010
Evaluating Agile methods and their implementations	Minna Väänänen	Agile methodologies	Tampere University of Applied Sciences, 2008

Appendix 3 Interview Question

1. Do you use any Software Capability Quality Standard?
2. What is your company's nature when adopting new methodology?
3. How much do you know about the heavyweight software development methodologies?
4. How much do you know about the agile software development methodologies?
5. Which agile methodology do you prefer mostly for the software development based on the project size?
6. Which traditional methodology do you prefer mostly for the software development based on the project size?
7. Which characteristics of agile methodologies you like for software development?
8. Which characteristics of traditional methodologies influence you for software development?
9. Which agile methodology characteristics, you dislike for software development?
10. Which traditional methodology characteristics, you dislike for software development?
11. Based on which factors, company decide software development methodology?
12. What's your opinion about choosing agile methodologies rather than traditional methodologies have any effect in different stages of software development?
13. What are the key challenges while agile methodology is implementing in the company?
14. In your opinion which methodology is the most suitable or work best for different kinds of software development?
15. To what extent, your company choose agile technique for different kind of software development?
16. What are the factors which increase or decrease the bug rate, development time and cost?
17. What's your opinion about Scrum work well only for small-scale projects?
18. What are the common issues when company transit from traditional to agile methodologies?
19. Do you belief, daily scrum meeting is effective for developers?

20. In your opinion, which factors play important role for the success of software projects?

Appendix 4 **Interviewee**

1st interviewee:	Working at Basware Oy in Espoo since 2010.
2nd interviewee	Senior lecturer of Haaga-Helia University of Applied Sciences
3rd interviewee	Previous work experience in Israel and Finland, currently involve in studying.