Agile Software Process Model: A Comparative View

P. Joshi
PG Student
Thapar University
Patiala, Punjab, India

A. Agarwal
Assistant Professor
Thapar University
Patiala, Punjab, India

S. Goel
Assistant Professor
Thapar University
Patiala, Punjab, India

ABSTRACT
Agile software development methods have been developed and evolved since early 1990's. These methods are widely used due to its short development lifecycle and through an interactive and iterative software development process. This paper explains a brief description of agile methodologies and difference between the traditional development methods and agile developments.

General Terms
Software Life-cycle Management

Keywords
Agile software lifecycle , agile software development, agile method comparison

1. INTRODUCTION
Traditional software development methods are referred as heavyweight development method because they makes the heavy use of documentation and design upfront. Although they are very effective as they planned every steps but the problem arises whenever an unpredictable risk/error occurs or when customer demands sudden changes.

To reduce these problems various agile methods such as Extreme programming(XP), Dynamic System development methodology(DSDM), Feature Driven Development(FDD), Scrum, Adaptive System development(ASD) were developed to handle the continuously changing requirements. The agile method makes lower use of the documents. Agile methods enables team to more rapidly response to change [11]. As changes are costly to accommodate later in the project [9], the ability to respond rapidly to change reduces project risks and their costs [10].

The software process is becoming a major concern in most software development organizations as one of ways to assure the software quality while developing software system, with the software process [12], there are still questions that which agile model is to be used, what is the primary difference between them and how their framework activities are different from each other. So the aim of this paper is to present a comparatively analysis of some of widely used agile methodologies.

Remainder section of this paper presents a basic concept of agile methodologies and their basic characteristic. Then it is described that how they are different from the traditional method and finally a comparatively analysis of some of agile method is done and results are represent in tabular format.

2. WHAT IS AGILE SOFTWARE DEVELOPMENT
Agile software engineering combines a philosophy and a set of development guidelines [3]. The philosophy encourages the early delivery of software product, delivery in small increment and less focus on documentations. And development guidelines focus on analysis, design of software products. Agile software development focuses on keeping code simple, testing often and delivering functional bits of the application as soon as they're ready. Agile methodologies get increasing attention of public in late 1990s. Each had a different combination of old ideas, new ideas, and transmuted old ideas. But they all emphasized close collaboration between the programmer team and business experts; face-to-face communication (as more efficient than written documentation); frequent delivery of new deployable business value; tight, self-organizing teams; and ways to craft the code and the team such that the inevitable requirements churn was not a crisis[2].

Agile manifesto was given on 17th February 2001, which is a formal statement which describes the agile methodologies. It stated the "We are uncovering better ways of developing software by doing it and helping others do it."[4]

2.1 Agile development methodologies
Agile methods are a subset of iterative and evolutionary methods [5] and are based on iterative enhancement and opportunistic development processes . In all iterative products, each iteration is a self-contained, mini-project with activities that span requirements analysis, design, implementation, and test.

2.1.1 Extreme Programming
Extreme Programming is one of several popular Agile Processes. It has already been proven to be very successful at many companies of all different sizes and industries worldwide. advocates frequent "releases" in short development cycles (timeboxing), which is intended to improve productivity and introduce checkpoints where new customer requirements can be adopted. Phases of XP includes planning, designing, coding, testing and then release the software increment. In first phase, initially, customer requirements are collected. These requirements are referred user stories. These user-story are analyzed by project manager, team member and the value is given to every user story. Then these user stories are ranked or prioritized according to their value. The acceptance criteria is set and interaction plan Is defined. Only the project manager has right to change the priorities of user stories. Now first phase is over here and second phase is started.

In designing phase, CRC are generated and spike solution of the problem is given. CRC cards stands for class responsibility collaborator cards, which is used to identify and organize the object oriented classes revalent to the current software increment. If difficult design problem encountered as a part of design then Extreme Programming recommends immediate creation of operational prototype of that portion known as spike solution. And if it is the second phase of iteration, then refactoring is done which is process of changing a software in such a way that it does not alter the external behavior of the system yet improves the internal structure. In third phase of Extreme Programming, which is coding, stories are developed and the team should not move to code, but rather then develop a series of unit test which will be used to exercise each of user stories in current release[3]. Once the unit tests have been created the developer is better able to focus on what must be implemented to pass the unit test. Key concept is to use pair programming in which two programmer work together on a one computer work station as it provides the mechanism for real time problem solving and real tile quality checking. And the last phase is testing. Extreme Programming encourages the regression testing whenever code Is modified.

2.1.2 Dynamic system development methodology

DSDM is an agile software development approach that provides a framework for building and maintaining system which meets the time constraints through use of incremental prototyping[3]. DSDM is an iterative development approach like other agile development process. When we are using DSDM then identification of crosscutting requirements is necessary throughout the DSDM development lifecycle.
The DSDM consortium is a group of member companies. The consortium has defined the life cycle activities of DSDM. The following is the brief discussion of DSDM lifecycle activities. Process start with feasibility study. Followed by business study, functional model iteration, design and build iteration and finally implementation phase. This is an iterative model i.e. if customer is not satisfied with current increment then process starts again. During the feasibility study, high level of separation of concern can take place. In this phase, high level functional or business requirement are defined and then asses that whether the application is viable candidate for DSDM process. During the business study phase of the DSDM, non-functional requirements are identified such as maintainability and other quality attribute. Next phase is functional model iteration.

The prime focus of functional model iteration is on prototyping to elicit requirements[7]. In this phase a set of prototype is produces that demonstrates the functionality for the customer. During design and build iteration, the focus is on ensuring that the prototypes are sufficiently well engineered for operational use and the functional prototypes are refined to meet the non-functional requirements . this phase ensures that each prototype has been engineered in a manner that will enable to provide operational business value to the end user. Now the last phase comes which is implementation phase. In this phase the software increments are produced or developed on the basis of prototype and then It places the latest software increment into the operational environment. It should be noted that changes may be requested by customer after the completion of increment. In this case DSDM work will continue by returning to the functional model iteration activity.

2.1.3 Feature driven development

FDD was first introduced to the world in 1999 via the book Java Modeling In Color with UML, a combination of the software process followed by Jeff DeLuca's company and Peter Coad's concept of features. In the context of FDD, feature is a client valued function that can be implemented in two weeks or less[8].

Features are expressed in form <action> <result> <object>[8] . For example, "Calculate the total of a sale". Features are as important in FDD as user stories are in Extreme Programming. Features provide a better view of system as user can describe them more easily and can define how they are related to each other. Since feature in a deliverable, team deliver feature in every two weeks and project planning, scheduling and tracking is done with the help of feature rather than software engineering tasks. It's development framework includes five collaborative activities, first two activities establishes overall shape and last three activities are iterated for each feature. Process starts by developing an overall model, which provide high level scope of the system and its context. Next activity is build feature list, in this activity the knowledge that was collected in first phase/activity is use to identify the feature list. For this, domain area is functionally decomposed into subject area and subject area contains business activity and business activity forms feature. Thus the feature are client valued function represented in form of <action><result><object>.

When the feature list is build, next step is to develop a plan by feature. In this step, each feature is organized as classes and assigned to chief programmer. In design by feature phase, chief programmer selects the set of feature that is to be delivered in two weeks and make a detailed sequence diagram together with the other class owner and refines the overall model. And finally the design inspection is held. After a successful design build by feature step take place. This is the main implementation step take place. The class owner or chief programmer develop the actual code for the feature and after the unit test and code inspection the completed feature is added to main build. If the deadline pressure is high then it is difficult to determine if feature are properly scheduled[3]. To mark the progress of each feature, FDD defines several milestones that must be completed sequentially for each feature. These are domain walkthrough, design, design inspection, code, code inspection, promote to build.

2.1.4 SCRUM

scrum is an agile method for completing the complex project. Scrum is an activity used in rugby match, as the whole process is performed by one cross-functional team across multiple overlapping phases, where the team "tries to go the distance as a unit, passing the ball back and forth"[9]. The scrum model was initially developed by Jeff Sutherland and his team during early 1990s. Scrum originally was formalized for software development projects, but works well for any complex, innovative scope of work. Today there are records of Scrum used to produce financial products, Internet products, and medical products. Scrum emphasizes the use of a set of software process patterns that have been proven effective for project with tight timelines, changing requirements, and business criticality[3].

In scrum framework Lifecycle activities includes creation of backlogs, followed by sprints, scrum meeting and demos to customer. Initially a customer or product owner creates the backlogs/ requirements and priorities' the backlogs. Items can be added to backlog at any time that added the value to the customer. Project manager can access the backlogs and can update priority at any time. When the backlog is completed then sprint phase is started. In this phase, work units, that are required to achieve a requirement defined in backlog is are defined i.e. sprint consists of work unit that defines that how to implement specified backlog.

When the sprints are defined then sprint teams have some time to implement these sprints usually two to four weeks. Scrum master is a key personnel in the project which keeps the team focused on their goal. Scrum meetings are daily short-meeting of scrum teams with their scrum master. During the scrum meeting several question are asked by the scrum master to their scrum teams such as what they have did since last meeting, what they will do by the next meeting. This scrum meeting helps to uncover the potential problem.

When an increment of the software is developed or any backlog is created/implemented than it is demonstrated to the customer. This phase is known as DEMOs. In this phase customer also evaluates the functionality of developed increment. If the customer is satisfied with the functionality of current software increment than scrum team begin to work on another backlog otherwise changes required by customer are logged and cycle starts again.

2.2 Common properties of agile methods

Agile is a different way of managing projects and development teams. Four agile methods are considered in this paper contains some of the common properties and behavior. A small group of people got together in 2001 to discuss their feelings that the traditional approach to managing software development projects was failing far too often, and there had to be a better way. They came up with the agile manifesto, which describes 4 important values that are as relevant today as they were then. It says, "we 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" [10].

Ever since then, the use of methods that support these values has become increasingly popular. From the detailed study of four agile method a brief discussion of common properties in all agile method is given as:-

- Agile methods applies Pareto principle rule that is also known as 80/20 rule.
- Focus on frequent delivery of the product.
- Testing is integrated throughout the lifecycle.
- Complete each feature before moving to next.
- Changes in the requirements are welcomed even late in the development cycle.

3. DIFFERENCE BETWEEN AGILE METHODS AND TRADITIONAL METHODS

Traditional Software Development Methods (TSDMs) including waterfall and other models, utilize extensive planning, codified process, rigorous reuse, heavy documentation and big design up front [1]. Due to these characteristics, TSDMs are often called heavyweight development methods. Following table represents the differences between TSMD and Agile methods.

Table 1 Difference between AM and TSDM

Attribute	Agile Methods	TSDM
Customer Participation	Customer is part of development team	Not actively participates
Team Empowered to make decision	Yes	No
Increment Size	Small	Depends on project
Primary Focus	Rapid delivery of software product to customer	High Assurance
Testing	Integrated throughout the lifecycle	Separate phase is used for testing
Communication between team member	Communication is high	Communication is low
Quality Process	Integrated in lifecycle	Separate and well defined Quality process
Lifecycle phases	Iterative and incremental	Depends on model used
Requirement	Changes throughout the lifecycle of product	Are fixed
Documentation	Not too much important	Very important
Team size	Small teams	Depends on project
Refactoring	Inexpensive	Expensive

4. COMPARISON OF AGILE METHODS

in this section, four agile method discussed above are compared with each other based on various factors.

Comparison is listed below in tabe-2 based on detailed study of four agile methods:-

Table 2 Comparatively analysis of agile methods

Attribute	Extreme Programming	Dynamic System Development Methodology	Feature Driven Development	Scrum
Proposed by	Kent Beck in 1999	DSDM Consortium in 1994	Jeff De Luca in 1997	Sutherland and Schwaber in 1995
Philosophy	Programmer Centered	objective of "jointly developing and promoting an independent RAD framework" by combining best practice experiences	Use feature list to manage functional requirements and development tasks	whole process is performed by one cross-functional team across multiple overlapping phases, where the team "tries to go the distance as a unit, passing the ball back and forth"[9]
Focus on	User Stories and their value and acceptance criteria	Time constraints and prototypes	Feature which is a client valued function	Backlogs and sprints
Prototype	Recommend use of operational prototype	Builds a set of prototype in functional model iteration	Does not recommends creation of prototype	Recommends creation of operational software increment instead of prototype
Addresses risk	Early in lifecycle	Early in lifecycle	In middle of lifecycle	Early in lifecycle
Supporting tools	Ex:- xm.tracker-1.0.6	DSDMAtern	FDD Tools 2.0, Snap-on Tools	Scrumwise, Eylean
Iterative process	Yes	Yes	Yes	Yes
Management activity completed in phase	Planning	Business Study Phase	Plan by feature	Depends on scrum-master
Artifacts	Working software , code, unit test cases, user manual and installation guide	Working software , documents to explain model and enable maintenance, user manual and installation guide	Working product, document consisting of feature, user manual and installation guide	Working software, user manual and installation guide
Team Member	Chief Programmer, programmer, tracker, consultant, customer	Executive sponsor, Ambassador user, project manager, Technical coordinator, Team leader, Solution Developer	Project Manager, chief architect, Development Manager, chief Programmer, Class owner, Domain Expert	Product owner, Scrum master, Scrum teams
Phases	Planning, Design, Coding, Test	Feasibility study, Business Study, Functional model iteration, Design and build iteration. Implementation	Develop an overall model, Build a feature list, Plan by feature, Design by feature Develop by feature	Product Backlog, Sprints, Scrum Meeting, Demos

5. CONCLUSION

Agile development method were developed to provide more customer satisfaction, frequent delivery of software product, to reduce the heavy need of documentation and most important is to welcome the changing in the requirement even late in development cycle. Agile address the key assumption that it is difficult to predict how customer requirements will change as project proceed. This paper presents some of the key differences between traditional development methods and agile method. Then some basic characteristic are addressed of widely used agile methods. And finally presents a detailed comparison between these methods. Agile methods provides another method to development organization to develop a software product with continuously changing requirements.

6. REFERENCES

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

[2] "Agile Alliance" http://www.agilealliance.org/the-lliance/what-is-agile/

[3] Pressman. (2010,6/e). Software Engineering, A practitioner's approach 64-69

[4] "Manifesto for Agile software development" http://www.agilemanifesto.org/

[5] C. Larman, *Agile and Iterative Development: A Manager's Guide*. Boston: Addison Wesley, 2004.

[6] Abufardeh S., Magel k. "Software Internationalization: Crosscutting Concerns across the Development Lifecycle". International Conference on New Trends in Information and Service Science, 2009

[7] Coad P., E. Lefebvre, J. Deluca, Java Modeling in colors , 1999

[8] Takeuchi, Hirotaka; Nonaka, Ikujiro. "The New Product Development Game" *Harvard Business Review*. Retrieved June 9, 2010.

[9] Paetsch, F., A. Eberlein, and F. Maurer, "Requirements Engineering and Agile Software Development," Proceedings of the 12th IEEE international Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, June 2003, pp. 308 – 313.

[10] K. Beck. *eXtreme Programming Explained*. Addison-Wesley, 2000.

[11] Coram M., Bohner C. "The Impact of Agile Methods on Software Project Management" Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05) 0-7695-2308-0/05 $20.00 © 2005

[12] Altarawneh H., El Shiekh A. "A Theoretical Agile Process Framework for Web Applications Development in Small Software Firms" Sixth International Conference on Software Engineering Research, Management and Applications.