MANIFESTATION OF AGILE METHODS FOR PROMPT SOFTWARE DEVELOPMENT: A REVIEW

Munish Mehta*

Naveeta Adlakha**

ABSTRACT

Nowadays agility has become the most powerful feature of software development. In the era of rapidly changing requirements of the user, the developers require the new methodologies with higher degree of flexibility. Agile Methodologies were thus introduced to meet the new requirements of the user or the software development companies. Agile methodologies enable companies to deliver flexible, scalable and adaptive software in less time. It helps in delivering the quickly create working software, in frequent iterations, building the highest priority features first. This paper presents a review of three agile approaches including SCRUM, Extreme Programming and FDD, describes the differences between them and recommends when to use them

Keywords: *Agile Methodology, Scrum , Extreme Programming, FDD.*

*Assistant Professor, Maharishi Markandeshwar University, Mullana, Ambala.

**Associate Professor, Maharishi Markandeshwar University, Mullana, Ambala.

1.0 INTRODUCTION

Nowadays, In the software companies the current issue is the rapid software development. As the user requirements are changing rapidly, so the software is. In the starting stages of the software development, the users requirements were stable, so it was in practice to collect all the requirements in the initial phases of software development life cycle and development proceeds without major changes. But, as now software development has become more dynamic due to complex software projects, so it has become difficult to work with the static requirements. Some other issues associated with it are as follows [1]:

- Embryonic requirements

  Its become difficult to collect all the requirements in the initial sages of development, as well as the user requirements may change time to time due to embryonic business needs.
- Dependency on resources

  Since different teams are responsible for different phases of development, there are more dependencies on their respective resources.
- Heavy Documentation takes lot of time

  Documentation takes a lot of time. It's very difficult to maintain the documentation for every iteration of the project.
- Lack of Customer involvement

  As the customer is involved only in the starting , in traditional developments methods , so it leads to higher chances of project failure. These methods usually do not allocate any effort for customer involvement.
- On Time delivery within budget

  In the traditional methods low budgets projects are set with the tight deadlines, while at the same time, requiring high ROI, and all of this is because of competition in the markets.
- Miscommunications

  Once the requirements are misunderstood, it results into dissatisfaction of the customer. The lack of involvement of the customer during the development stages, gives the products which are not according to the users requirements.

With the existence of such problems, the Traditional development methodologies cannot satisfy the objectives of software development companies. So the new development methodologies i.e. Agile Methodology came into existence. Now the agile development is the

most common practice in the software companies. By using agile methodologies, the companies can deliver adaptive software in less time with high ROI. Agile methodologies help in fast software development in small iterations. Development with agile methodologies starts with small modules that focus only on the few requirements of the user. As the user approves a module the project moves to the next iteration. These methodologies enable companies to deliver scalable, flexible and adaptive software. In the next section of the paper we have discussed some of the agile methodologies with comparison.

2.0 AGILE METHODOLOGIES

In the literature there are various agile methodologies are available. This paper presents a review of three agile approaches including SCRUM, Extreme Programming and FDD, describes the differences between them and recommends when to use them.

2.1 SCRUM

SCRUM has been used with the objective of simplifying project control through simple processes, easy to update documentation and higher team iteration over exhaustive documentation [5]. SCRUM shares the basic concepts and practices with the other agile methodologies, but it comprises of sprints. Scrum is an agile software development process designed to add energy, focus, clarity, and transparency to project teams developing software systems. It is based on a 30-day iteration called a "Sprint." Technically Sprints can be either two or four weeks, but the default sprint time is four weeks. Scrum not define the way how software is developed, what documents are to be produced, how requirements should be gathered rather it is a guide how an agile implementation team should be managed. Scrum is not an independent stand alone development methodology; rather it is a wrapper for existing engineering processes. It manages and controls development work incrementally, improves communications, removes the problems to development with all in environment with rapidly changing requirements.

Advantages of SCRUM

1. It fully involves the customer for the software approval at every stage of development.
2. It delivers a quality Product in a scheduled time due to short sprints and constant feedback from the customer.
3. Through the daily meetings, issues are identified well in advance and hence they can be resolved in time.
4. It helps the companies in saving time and money.
5. In scrum daily meetings make it possible to measure individual productivity.

Disadvantages of SCRUM

1. It is good for small, fast moving projects as it works well only with small team. Its not easy to make large projects with this methodology.
2. When the scrum master trust the team, this methodology works well. If he put too much strict control over the team members, it can be frustrating for them and become the reason for failure of product.
3. Experienced staff is necessary for the success of this methodology. If the team consists of beginners, the project can not be completed in time.
4. Tasks should be well defined. If the task is not well defined, time and estimating project cost will not be accurate. As a result task can be spread over several sprints.
5. It is not best suited for products where the focus is on usability. It fails to address usability needs of the user, because product owners keep their focus mainly on business issues and forget about usability.

2.2 Extreme Programming

Extreme Programming can be said as a package of several practices and ideas, most of which are not new. But this combination is definitely new. Extreme Programming is a methodology that consists of small teams working for one target. XP provides a list of simple principles and values that guide the software development process throughout SDLC. XP results the product what the customer wants exactly because of its ability to accept changes at anytime during the development.

Rules of Extreme Programming

1. **Target current problem only**

   XP only write code for the current task. It try to use the actual task time for developing code of a different task, even if the task are very related.

2. **Modularity**

   Try to get rid of the duplication of code, make complex code simpler.

3. **Integrate often**

   Continuous integration often avoids diverging or fragmented development efforts, where developers are not communicating with each other about what can be re-used, or what could be shared.

Advantages of Extreme Programming

1. It is suitable for small projects only.
2. It works for those projects in which requirements are not clear at the time of beginning.

3. It gives good results with small teams. Teams with 2 to12 members are best for XP approach.
4. If there is a strong co-operation between developers and customers are available, then XP is very good. If this is missing then XP approach does not produce good results.

2.3 Feature Driven development

The Feature Driven Development method was originally conceived by Peter Coad and his colleagues as a practical process model for object oriented software engineering. Stephen palmer and John Felsing have extended and enhanced Coad's work, describing an adaptive, agile process that can be applied to moderately sized and larger software projects. It has some common characteristics with XP such as Short iterations, Customer representative on the team & using the notion of features which are comparable to the XP. Prioritization are encouraged in FDD and it is ensured that most valuable features are delivered. FDD feature short iteration cycles i.e.Two weeks. This methodology recommends individual code ownership and clearly defined roles.

3.0 PROBLEMS IN ADOPTING AGILE

The most common problems faced by the development companies while try adopt agile are as follows:

- Fear of change
- Specialized skills
- Outdated skills
- Documentation heavy mind set
- Do-it-all-at-once attitude
- Serial Thinking
- Closed Mindedness
- Office Politics

4.0 COMPARISON BETWEEN EXISTING AGILE METHODOLOGIES

1. SCRUM and FDD provides agile management mechanisms whereas the extreme Programming provides engineering practices. Scrum teams typically work in iterations called sprints that are from two weeks to one month long. XP teams typically work in iterations that are one or two weeks long. FDD forms the small volatile teams of 3-5 developers concentrating on few features to be developed.
2. Changes into sprints are not allowed by Scrum teams. XP teams are much more amenable to change within their iterations. As long as the work hasn't started by team

on a particular feature, a new feature of equivalent size can be swapped into the XP team's iteration in exchange for the un-started feature. FDD offers better predictability if requirements for the project are more stable.

3. In SCRUM, validation of the software is completed at the end of each Sprint during the Sprint review, not at each step within the Sprint while XP requires that the software be validated at all times to the extent that tests are written prior to the actual software. FDD has methods to track project's progress, which is more appealing in corporate environment.
4. Extreme Programming teams work in a strict priority order. Features to be developed are prioritized by the customer and the team is required to work on them in that order. By contrast, the Scrum product owner prioritizes the product backlog but the team determines the sequence in which they will develop the backlog items. However, at some point one of the high priority items may not be a good fit for the sprint being planned—maybe a key person who should work on it will be swamped by work on higher priority items. Or maybe it makes sense to work on a slightly lower priority item.
5. FDD has design stage and design review meeting. During the design stage the developers declare main classes, methods and properties required to implement the given feature. XP does not have the formal design stage. The design may be performed at the beginning of the iteration and later on as-need basis.
6. XP promotes re-factoring. Re-factoring is a process of changing code to make it better, without breaking the existing functionality. The resulting code has higher quality. Unlike XP, FDD strongly discourages re-factoring. The main argument against re-factoring here is that it takes time and does not bring any value to the customer. The quality of code is addressed during code review meetings.

5.0 CONCLUSION

In this paper, we discussed some of the agile methods and after comparing all the agile methodologies, we can conclude that the various methodologies have their advantages and disadvantages. We can not say that any one of them is best, rather all of these can give good results depending upon the situation on which they are implementing. XP seem to be better suited for volatile projects, where user requirements are obscure or change often. XP deals with such projects better since it deliberately avoids any activities that are not immediately required for current implementation stage. FDD is better prepared to work with highly

experienced team. FDD has methods to track project's progress, which is more appealing in corporate environment.

6.0 REFERENCES

1. Agile Alliance. Manifesto for Agile Software Development. [Online] Retrieved 16th March 2009.Available at: http://www.agilemanifesto.org .
2. C.Balan et.al. , "The Need To Adopt Agile Methodology In The Development Of Cyber Forensics Tools,IEEE ,2010
3. David Fox et. al., "Agile Methods and User-Centered Design: How these two methodologies Development Methodology", Journal of software, Vol. 3, No. 4, April 2008.
4. Glen Litton Van der Vyver et. al., "Agile methodologies and the emergence of the agile organization: A software development approach waiting for its time", 7th Pacific Asia Conference on Information Systems, 10-13 July 2003.
5. M. Cristal, D. Wildt and R. Prikladnicki, Usage of SCRUM Practices within a Global Company. Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on, 2008, 222-226.
6. Peter E. Maher and Janet L. Kourik et. al., "Exploratory Study of Agile Methods in the Vietnamese Software Industry", Fifth International Multi-conference on Computing in the Global Information Technology, 2010.
7. Veerapaneni Esther Jyothi et. al., "Effective implementation of agile practices", International Journal of Advanced Computer Science and Applications, Vol. 2, No.3, March 2011.