A Concept Note on Impact of Agile Software Project Management Methods in Midsized IT Product Development Companies.

***Dr. Mayanka Sharma, **Mr. Iqbal Kotwal**
** Associate professor at the D Y Patil School of Management Pune, INDIA.*
*** Persistent Systems Ltd. in Pune, INDIA.*

Abstract

Agile methods are very helpful in cost effective development of software, it will greatly benefit the organizations in IT field. Empirical data comparing the effectiveness and limitations of agile and non-agile approaches would greatly enhance our understanding of the true benefits and limitations of agile processes. In this study we will be analyzing traditional development methods its disadvantages and agile software development methods with its advantages' This research study will be helpful to IT companies for proper implementation of agile software methods and for maximization of profits while minimizing the cost and delivery time.

Key words: Project management, Agile software, Methods of software project development

I - INTRODUCTION

In the 1970s and 1980s, the software industry grew very quickly, as computer companies quickly recognized the relatively low cost of software production compared to hardware production and circuitry. To manage new development efforts, companies applied the established project management methods, but project schedules slipped during test runs, especially when confusion occurred in the gray zone between the user specifications and the delivered software. (Robiolo 2014). To be able to avoid these problems, software project management methods have focused on matching user requirements to delivered products.

In the 1990s, new processes and methodologies that deal with software development projects appeared. As they evolved, and because of their particular characteristics, these development methodologies fell into two broad categories: traditional and Agile methods. (Larson,2001).

On the one hand, traditional methods involve those in which the systems are fully specified, they are predictable and they are built according to meticulous and extensive planning. Besides, these projects are run by a clearly defined head that controls the activities, based on explicit knowledge. The organization where this happens is usually large, bureaucratic, with a high degree of formalization, which makes communication formal too. In addition, the software life cycle of their products may be described as waterfall or spiral, and the testing is most surely performed at the end of the cycle. (Beck, etal,2001).

On the contrary, Agile methods are based on the premise that high quality software -adaptable to different conditions is developed by small groups, using a design which gets continuous improvement. Besides, constant testing provides a rapid feedback, thus making the early introduction of improvements possible. The management structure of the organization in which Agile methods are used tends to be informally defined; it is based on natural leadership and collaboration. Regardless of the size of the organization, work is divided into small groups, where communication within the teams is informal, the internal organization is flexible and their members are participatory. The life cycle of their products is often evolutionary, which includes requirements management and continuous testing. (Burton,2010).

II - REVIEW OF LITERATURE

The exact definition of agile is elusive, and it is important to note that the term encompasses several different methodologies, including Feature Driven Development (FDD) (Coad, deLuca, & Lefebvre, 1999), Dynamic Systems Development Method (DSDM) (Stapleton, 1997), Crystal (Cockburn, 2005), Agile modeling (Ambler, 2002), Scrum (Schwaber & Beedle, 2002),

DOI : 10.21904/weken/2016/v1/i3/103459

Adaptive Software Development (Highsmith, 1999), and Extreme Programming (XP) (Auer & Miller, 2001; Beck, 2000, 2004; Jeffries, 2001).

Agile methodologies provide a structure for highly collaborative software development. Developed in the 1990's, the adaptive methodologies were formulated by and for developers in reaction to perceived deficiencies in conventional 'top down' or 'plan driven' methods. Commonly associated with 'lean' engineering (e.g. Poppendieck & Poppendieck, 2003), agile software development closely follows the flow of business value, with a focus on activities that directly contribute to the project end goal of quality software. The agile manifesto (Beck et al., 2001), a guiding force for agile practitioners, was created by 17 influential figures in the nascent software development movement

III - AGILE SOFTWARE DEVELOPMENT

The Agile software development method is not new by any means, yet its principles and methods of execution are very different from the way software projects have historically been run for the DoD. The Agile Manifesto was developed by a group of industry experts in 2001 to formally consolidate and agree to a set of some principles for lean software development. The Agile Manifesto revolves around four main principles as follows. (Agile alliance,2001)

a. Individual and interactions over processes and tools

b. Working software over comprehensive documentation

c. Customer collaboration over contract negotiation

d. Respond to change over following a plan

The principles of Agile development aim to break down the software development cycle into multiple smaller cycles with each cycle ending in a complete working product. As per research done by Burton D. (2010) the Agile method allows for changes in requirements and direction of the project and argues that lining up to the customer's most current needs in each cycle actually helps save cost while gaining better customer satisfaction.

At the early years of software development, most of the users' requirements were fairly stable, and development followed the plans without major changes. However, as software development involved more critical and dynamic industrial projects, new difficulties emerged according to the growth of companies.

These difficulties as pointed out by D. Turk et.al. (2002) are,

- Evolving requirements: customer requirements are changing due to evolving business needs or legislative issues. Most of the customers do not have a clear vision about the specifications of their requirements at the early stages. Some customers realize what their true requirements are only when they use an application that does not really meet their needs. Another source of change comes from experiences gained during the development.
- Customer involvement: lack of customer involvement leads to higher chances of project failure. Many companies usually do not allocate any effort for customer involvement.
- Deadlines and budgets: often, customers do not accept failure. On the other hand, companies usually offer low budgets, tight deadlines, while at the same time, requiring high demands, and all of this is because of competition in the markets.
- Miscommunications: one cause of the misunderstanding of requirements is the miscommunication between developers and customers. For example, each party uses its own jargon, and this leads to misunderstanding of customer's needs.

IV - AGILE METHODS

With the existence of such problems, the OO software development methodologies cannot satisfy the objectives of software development companies. New development methodologies have to be applied in order to overcome these problems.

A number of IT professionals started to work individually on new approaches to develop software. The results of their researches were a set of new development methodologies that have many common features.

DOI : 10.21904/weken/2016/v1/i3/103459

When they met in 2001 in conference in Utah, they created the so called: Agile Manifesto. These approaches were developed based on the same rule that the best way to verify a system is to deliver working versions to the customer, then update it according to their notes. Agile authors built their methodologies on four principles. First, the main objective is to develop software that satisfies the customers, through continuous delivering of working software, and getting feedback from customers about it. The second principle is accepting changes in requirements at any development stage, so that customers would feel more comfortable with the development process. The third principle is the cooperation between the developers and the customers (business people) on a daily basis throughout the project development. The last principle is developing on a test-driven basis; that is to write test prior to writing code. A test suite is run on the application after any code change. (The SCRUM Alliance ,2010)

Agility in short means to strip away as much of the heaviness, commonly associated with traditional software development methodologies, as possible, in order to promote quick response to changing environments, changes in user requirements, accelerate project deadlines, and the like as defined by J Erickson (2005). Agile methodologies prefer software development over documentation. Their philosophy is to deliver many working versions of the software in short iterations, then update the software according to customers' feedback. Applying this philosophy will help to overcome the problems mentioned earlier, by welcoming changes, satisfying user requirements, faster development, and at the end, users will have just the system they need.

Agile methodologies include:

- Extreme Programming
- Agile Modeling
- SCRUM
- Crystal methodologies family
- Feature-Driven Development
- Adaptive Software Development

V - EXTREME PROGRAMMING (XP)

Extreme Programming was introduced by Kent Beck in 2000. Being an emerging agile methodology, XP offers a number of practices, values and principles which are advised to be adopted in order to run a software development project, K Beck, etl.al. (2005). XP is a package of several practices and ideas, most of which are not new. The combination and packaging of all of these is, however, new. Extreme Programming was in fact targeted especially at small co-located teams developing non-critical products. It has been suggested that the early adopters of agile methods have been small high-tech product companies C. Schwaber (2005). Currently, however, it has already been proven at many companies of all different sizes and industries worldwide .

XP provides a list of simple, specific, and seemingly naïve principles and values that guide the software development process throughout the main four phases of software development: planning, coding, designing, and testing (Figure 1). The main purpose is to deliver what the customer needs, at the time it is needed. In addition to this, one of the main reasons of its success is its ability to accept changes at anytime during the development. XP also emphasises teamwork; experiences from all stakeholders are employed to meet the specific goals, and within the given constraints.

Figure No- The flow in a project using XP



DOI : 10.21904/weken/2016/v1/i3/103459

There are some programming principles that are encouraged by XP. One of them is that, simplicity and flexibility will reduce maintenance costs of the software in the future. Another programming practice is the intensive and robust testing mechanism, which will reduce the number of bugs reported later by the customer after delivering the final version of the application. Embracing changes is also encouraged by XP, since customers are most likely to notice new opportunities for improving the system to meet their goals, while still in the development process. Moreover, XP encourages creating high quality code.

VI - AGILE MODELING (AM)

Modeling is an important step in software development. It enables software developers to think about complex issues before addressing them in programming. Agile Modeling (AM) was established by Scott Ambler in 2002. It is a collection of values, principles, and practices for modeling software that can be applied on a software development project in an effective and light-weight manner . Agile Modeling was built to be adapted to, and used with existing methodologies, as XP and RUP, aiming to allow a developer to build a software system that truly meets the customer's needs.

The values of AM, which are considered to be an extension to the values of XP include: communication, simplicity, feedback, courage, and humility. Humility means to admit that you may not know everything; others may know things that you do not know, and thus, they may provide useful contribution to the project .

Again, the principles of AM are quite similar to those of XP, such as assuming simplicity, embracing changes, incremental change of the system, and rapid feedback. In addition to these principles, AM principles include the knowledge of the purpose for modeling; having multiple effective models; the content is more important than the representation; keeping open and honest communication between parties involved in the development process; and finally, to focus on the quality of the work .

The practices of AM have some commonalities with those of XP, too. An agile modeler needs to follow these practices to create a successful model for the system. AM practices highlight on active stakeholder participation; focus on group work to create the suitable models; apply the appropriate artifact as UML diagrams; verify the correctness of the model, implement it and show the resulting interface to the user; model in small increments; create several models in parallel; apply modeling standards; and other practices .

ASSUMPTIONS OF AGILE SYSTEM

Assumption 1: Customers are co-located with the development team and are readily available when needed by developers. Furthermore, the reliance on face-to-face communication requires that developers be located in close proximity to each other.

Assumption 2: Documentation and software models do not play central roles in software development.

Assumption 3: Software requirements and the environment in which software is developed evolve as the software is being developed.

Assumption 4: Development processes that are dynamically adapted to changing project and product characteristics are more likely to produce high-quality products.

Assumption 5: Developers have the experience needed to define and adapt their processes appropriately. In other words, an organization can form teams consisting of bright, highly-experienced problem solvers capable of effectively evolving their processes while they are being executed.

Assumption 6: Project visibility can be achieved primarily through delivery of increments and a few metrics.

Assumption 7: Rigorous evaluation of software artifacts (products and processes) can be restricted to frequent informal reviews and code testing.

Assumption 8: Reusability and generality should not be goals of application-specific software development.

Assumption 9: Cost of change does not dramatically increase over time.

Assumption 10: Software can be developed in increments.

DOI : 10.21904/weken/2016/v1/i3/103459

Assumption 11: There is no need to design for change because any change can be effectively handled by refactoring the code

VII - IMPORTANCE OF THE STUDY

Project Management methodology can dramatically improve the efficiency of any project whether large or small. However, it is important to understand not all projects will benefit from a particular project management methodology. Hence, before applying any project management methodology due diligence needs to be done in choosing a particular methodology for a particular project. The study will focus on Agile project management methodology and attempt to establish a co-relation between Agile methodology and the types of projects it will be suited for. This will help create guidelines for project managers and other stakeholders, based on which they can choose Agile methodology for their projects.

In addition, the study will also assess the impact Agile methodologies have on the efficiency of projects. The increase in efficiency will in turn increase the bottom line for the companies that execute the projects.

Agile methodologies provide some practices that facilitate communication between the developer and the customer, and undergo develop-deliver-feedback cycles, to have more specific view of the requirements, and be ready for any change at any time. The main aim of agile methodologies is to deliver what is needed when it is needed.

Agile methodologies include a set of software development approaches. They have some variations, but still they share the same basic concepts. The main agile methodologies that are being used include XP, Agile Modeling, and SCRUM. XP is the coding of what the customer specifies, and the testing of that code. Agile Modeling defines a collection of values, principles, and practices which describe how to streamline modeling and documentation efforts. SCRM, on the other hand, supports management role in software development.

VIII - NEED FOR AGILE SOFTWARE PROJECT MANAGEMENT METHODOLOGY

As the industry has matured, analysis of software project management failures has shown that the following are the most common causes:

1. Insufficient end-user involvement

2.Poor communication among customers, developers, users and project managers

3.Unrealistic or unarticulated project goals

4.Inaccurate estimates of needed resources

5.Badly defined or incomplete system requirements and specifications

6.Poor reporting of the project's status

7.Poorly managed risks

8.Use of immature technology

9.Inability to handle the project's complexity

10.Sloppy development practices

11.Stakeholder politics (e.g. absence of executive support, or politics between the customer and end-users)

12.Commercial pressures

The first five items in the list above show the difficulties articulating the needs of the client in such a way that proper resources can deliver the proper project goals. Specific software project management tools are useful and often necessary, but the true art in software project management is applying the correct method and then using tools to support the method. Without a method, tools are worthless. Since the 1960s, several proprietary software project management methods have been developed by software manufacturers for their own use, while computer consulting firms have also developed similar methods for their clients. Today software project management methods are still evolving, but the current trend leads away from the waterfall model to a more cyclic project delivery model that imitates a software development process.

Agile software project management methodology is an approach that helps the teams to respond to unpredictability in execution of the project by

DOI : 10.21904/weken/2016/v1/i3/103459

engaging with all stakeholders throughout the execution of the project. It follows an incremental and iterative approach to software development, where the customer has early and regular access to the product under development so that any corrective actions to rectify deviation from requirement or enhancement for improvements can be done earlier in the lifecycle of software development.(Turk,2001)

IX - CONCLUSION

Agile methodologies are not best suited for all projects. When communication between the developer and the customer is difficult, or when the development team includes mainly beginners, agile methodologies will not give the best results. These methodologies exhibit optimum results when there is a strong communication between the developer and the customer, and the development team compromises skilled team members. When there is a big chance for misunderstanding the exact customer's requirements, or when the deadlines and budgets are tight, then Agile methodologies are among the best software development approaches to apply.

REFERENCES

1. (n.d.). Retrieved from http://www.agiledevelopmentconference.com/2003/files/AlanAgileSoftwareJun03.ppt

2.Abrahamsson, P. S. (2002). Agile Software Development methods:Review and Analysis . Espoo, Finland: VT Publications

3.Agile Alliance. Manifesto for Agile Software Development . (2009, March 16). Retrieved from http://www.agilemanifesto.org

4.Agile Modeling Home Page. Effictive Practices for Modeling and Documentation. (n.d.). Retrieved from www.agilemodeling.com .

5.Agile_software_development. (n.d.). Retrieved fromhttps://en.wikipedia.org/wiki/Agile_software_development

6.Ambler, S. ,. (2002,2003). Agile Modeling, Agile database techniques:Effective strategies for the agile software developer. . New York: Indianapolis, IN: Wiley Publishing Inc.

7.Auer, K. ,. (n.d.). XP Applied(The XP Series). Reading, MA: Addison Wesley.

8.Beck, K. B. (2001). Manifesto for Agile Software Development. Retrieved

from http://agilemanifesto.org/.

9.Beck, K. C. (2000,2003,2004). Extreme Programming explained.Reading, MA:Addison Wesley,Test driven development:By example. Reading:Addison-

10.Wesley,Extreme programming explained: Embrace change(2nd Ed.). Addison-Wesley Professional .

11.Burton, D. H. (2010). Considerations for Using Agile in DoD Acquisition. Retrieved from http://www.sei.cmu.edu/reports/10tn002.pdf

12.C Schwaber, R. F. (2005). Corporate IT leads the second wave of agile adoption. . Forrester Research, Inc .

13.Explore the Top 4 Project Management Methodologies. (n.d.). Retrieved from http://www.devx.com/enterprise/explore-the-top-4-project-management-methodologies.html

14.Fowler, M. (n.d.). New methodology. Retrieved fromhttp://www.martinfowler.com/articles/newMethodology.html

15.Frank Maurer, S. M. On the Productivity of Agile Software Practices: An Industrial Case Study, University of Calgary,. Canada: Department of Computer Science, Calgary, Alberta, Canada, T2N 1N4, {Maurer, smartel}@cpsc.ucalgary.ca.

16.Gabriela Robiolo, D. G. (2014). Do Agile Methods Increase Productivity and Quality? . American Journal of Software Engineering and Applications.

Vol. 3, No. 1 doi: 10.11648/j.ajsea.20140301.11 , pp. 1-11.

17.GUSTAFSSON, J. Model of Agile Software Measurement: A Case Study.

DOI : 10.21904/weken/2016/v1/i3/103459

18.J. Erickson, K. L. (2005). Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research. . Journal of Database Management, 16(4) , 88-100.

19.Jensen, R. W. (n.d.). Improving Software Development Productivity, Effective Leadership and Quantitative Methods in Software Management.

20.Jim, H. (2002). Agile Software Development ecosystems.Boston,. MA Pearson Education .

21.K. Beck, C. A. (2004). Extreme Programming Explained: Embrace Change (2nd Edition), Addison-Wesley, Boston.

22.Larson, E. P. (2001). Preparing the U.S. Army for Homeland Security: Concepts, Issues, and Options, . RAND Corporation,Issue 1251 .

23.MacCormack, A. (n.d.). Agile software development: Evidence from the field.

24.Manifesto for Agile software development. (n.d.). Retrieved from http://agilealliance.com

25.Pekka Abrahamsson, O. S. (n.d.). Agile Software Development methods-Review and analysis .

26.Scrum Is an Innovative Approach to Getting Work Done.

The SCRUM Alliance. (2010). Retrieved from http://www.scrumalliance.org/pages/what_is_scrum.

27.Turk, D. F. (2002). Agile Software Processes: Principles, Assumptions and Limitations. . Technical Report. Colorado State University.

28.Turk, R. F. (2002). Limitations of agile software processes. . The Third International Conference on Extreme Programming and Flexible Processes in Software Engineering,.

29.Types of Agile Methodologies,. (n.d.). Retrieved from https://www.versionone.com/agile-101/agile-methodologies

About Author



Dr. Mayanka Sharma

She has done her PhD in Management. She is associate professor at the D Y Patil School of Management Pune. This is affiliated to Savitribai Phule Pune University. She is an approved Guide at Savitribai Pune University. She has published many research articles in reputed journals and ISSN proceedings at National and International Level. She has expertise in conducting training program.

(Email : drmayankasharma@gmail.com)

About Co-Author



Mr. Iqbal Kotwal

He is a computer engineer, now working with Persistent Systems Ltd. in Pune from last 16 years. His areas of interest are Software Project Development and Management. He is working on end-to-end software project management across multiple customer projects that are being executed using traditional and agile software project management methodologies.

(Eamil: iqbal.kotwal@gmail.com)

DOI : 10.21904/weken/2016/v1/i3/103459