International Journal of Information Technology and Knowledge Management
January-June 2012, Volume 5, No. 1, pp. 5-8

COMPARATIVELY ANALYSIS OF AGILE SOFTWARE DEVELOPMENT

Ajaydeep[1], Lekha[2], Kavita Dawra[3], Vipul Sethi[4], and Manisha[5]

ABSTRACT: For Developing software there are two main methodologies: the traditional sequential or "the waterfall method" and the iterative or "The Agile Method". In this paper we emphasis on Agile Manifesto and its principles. It provides the general understanding of Agile Software Development and can act as a foundation for choosing methodology for the software projects as we have compared different methods and listed their Applicability and Limitations.

Index Terms: Agile, Agile manifesto, Principle of Agile manifesto, Usability, Waterfall Method.ss

1. INTRODUCTION

The agile methods analyzed and reported in this paper are those consistent with the Manifesto for Agile Software Development. There are two famous process models in this area, which are the traditional Waterfall Process Model and the Agile Software Development. [1] The Agile software development techniques are gradually being accepted as viable alternative to traditional software development methodologies. It leads to better quality software in a shorter period of time. [2]

2. WATERFALL MODEL

In Royce's original waterfall model, the following phases are followed inorder: Analysis, Requirements Specification, Design, Implementation, Testing and Integration, Operation and Maintenance. The waterfall model proceeds from one phase to the next in a sequential manner. When requirements are completed, one proceeds to design. The software in question is designed and a blueprint is drawn for implementers (coders) to follow-this design should be a plan for implementing the requirements given. [3] When the design is complete, an implementation of that design is made by coders. Towards the later stages of this implementation phase, separate software components produced are combined to introduce new functionality and reduced risk through the removal of errors. [4][5]

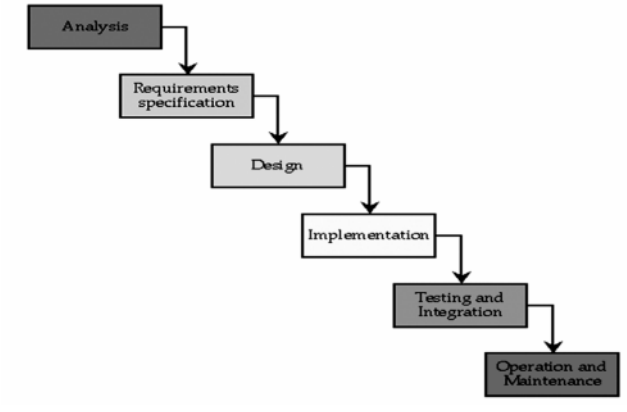

Figure 1: Waterfall Model

3. AGILE MODEL

Adaptable software creation, also known as agile software development. Agile development is a style of software development that emphasizes customer satisfaction through continuous delivery of functional software. The concept of the Waterfall Process Model is that the requirement analysis has to be done in the beginning phase, whereas, the Agile Software Development emphasizes that the requirement is changeable throughout the process. [5] Thus, it seems that using the Agile Software Development is becoming a trend for software development companies in order to improve the software process. Agile is a group of software engineering methodologies. [6]

(a) Principles of Agile Manifesto

The values described above are realized in the principles of Agile Manifesto. The principles are the following:

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

[1] E-mail: er.ajaydeep@gmail.com,
[2] E-mail: bhambhu@gmail.com,
[3] E-mail: kavitadawra285@gmail.com,
[4] E-mial: sethi.vipul@yahoo.com,
[5] E-mial: er.manishajain@yahoo.in

- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- 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.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

(b) The Agile Project Lifecycle

As mentioned, the Agile Development Framework is an iterative, incremental, and collaborative methodology for software development project. The Figure shows that the Agile Software Development Lifecycle (ASDL) starts from an initial plan and ends with deployment. [7] [8] Each iteration consists of planning, requirements, analysis and design, implementation, deployment, testing, and evaluation.

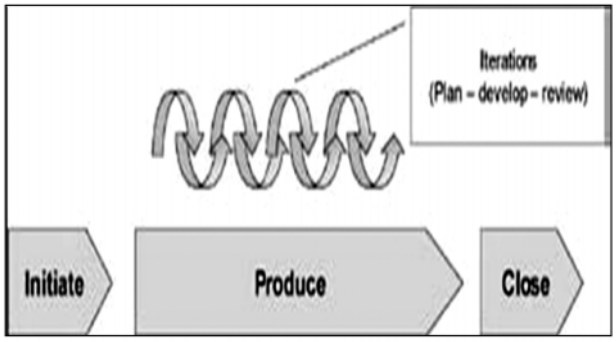

Figure 2: Agile Project Life Cycles

The Agile Software Development was developed to solve the Waterfall's weakness. Differing from the Waterfall Process Model, the Agile Software Development emphasizes its flexibility. For example, requirements are allowed to change in an agile project. A face to face communication also takes an important place in the Agile Software Development since it requires a quick feedback. [9]

(c) General Understanding of the Agile Software Development

Generally, agility is defined by ability which is able to be flexible and adaptable to change. The idea of the Agile Development Framework is to create a pain-free working environment for those small, co-located, self-organized teams in order to assist companies to take full advantage of the customer value of the delivered software product. In the Agile Development Framework, user requirements are written from users' perspective, elaborating on what users want to do with a feature of the software. The concept of the Agile Software Development can be summarized as below: [10]

Eliminate Waste: The Agile Development Framework advocates a "barely sufficient" approach to plan, process, and control software development process. "Barely sufficient", in other words, is to find the simplest things to meet needs of requestor instead of wasting unnecessary resources.

Sustainable Pace: The Agile Development Framework requires a daily meeting

Intense Collaboration: Unlike the traditional approach, which relies on documents, the Agile Development Framework requires a daily face to face communication with customers and coworkers to understand and fulfill their requirements.

Frequent Delivery: Frequent delivery offers incremental business value to customers. Thus, frequent delivery is one important way to seek feedback on the quality of the application.

Continuous Feedback: The Agile Development Framework was invented to achieve customer's value.

Include Change: Differing from the traditional approach, change of requirements is considered in the Agile Development Framework since remaining adaptable is a key to building a trusting relationship with customers.

Empowerment: The Agile Development Framework also pays attention to the team working atmosphere.

Iterative and Incremental development: Plans, requirements, design, implementation, deployment and testing are developed incrementally through iterations.

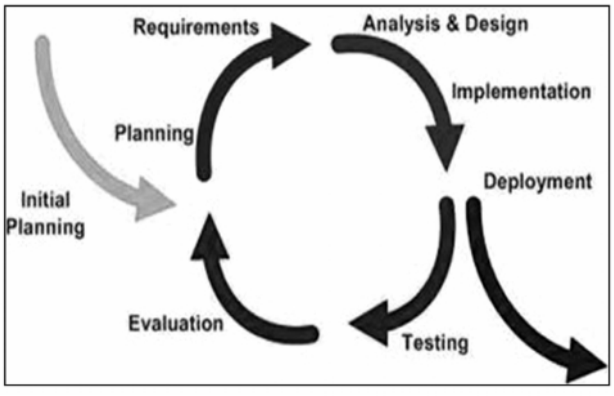

Figure 3: Iteration in the Agile Development Framework

(d) Limitations of Agile Processes

The assumptions listed above do not hold for all software development environments in general, or for all "agile" processes in particular. In this part I describe some of the situations in which agile processes may generally not be applicable. It is possible that some agile processes fit these assumptions better; Such extensions can involve incorporating principles and practices often associated with more predictive development practices into agile processes. [11]

(a) Limited support for distributed development environments.

(b) Limited support for subcontracting.

(c) Limited support for building reusable artifacts.

(d) Limited support for development involving large teams.

(e) Limited support for developing large, complex software.

(e) Traditional vs. Agile Software Development Methods

Throughout the literature we have consulted, a clear distinction between traditional and agile software development methods is made. My analysis found that a typical traditional method tells the developers pass through several phases in a prescriptive manner, phases conceived out of a plan based on the requirements specifications of the customer. Security in and stability of the project is therefore emphasized and extensive documentation is encouraged. An agile method on the other hand puts virtually all of the above in second place to complete flexibility and customer collaboration. Developers are agile as to being responsive to changing needs of the customers. Prototypes and beta versions are delivered constantly to the end users who in turn report back with changes that have to be made and wishes for additional functionality. The well-being of the development team also has a strong focus in agile software development.

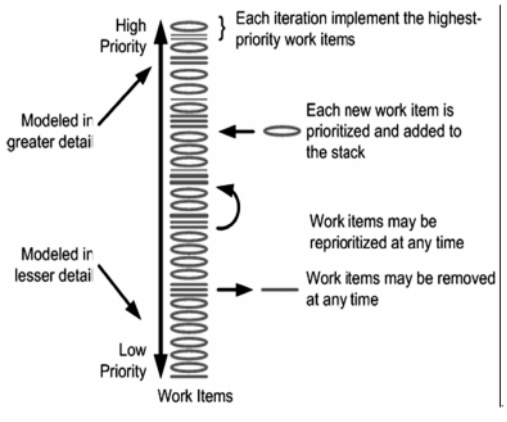

Figure 4: Process of Agile Requirements Change Management

To illustrate the somewhat varying degree of agility in all discussed software development methods, an "Axis of Agility" is created where the methods relative placing are shown on a one-dimensional scale with the non-existent extremes of traditional and agile values at each end of the barometer. This Axis of Agility depicts subjective feelings for how each of the methods stands in relation to each other with regard to these values.

Finally, in order to summarize the characteristics of the traditional and the agile software development methods, we once again turn to the agile values which truly captures the essence of what agile software development methods, relative traditional software development methods, are all about.

5. CONCLUSIONS

This paper aims for the identification of factors that help to decide for Agile or traditional project management methodologies in different types of projects. The following are the characteristics of the project which would benefit significantly if done with agile methodologies:

(a) Poorly defined requirements

(b) High level of complexity

(c) Risk of failure is high

(d) High development platform flexible.

Also several other factors were identified:

(a) High risk tolerance of the project champion

(b) High programmer experience with chosen development platform

(c) High end-user ability to adopt to change

(d) High budget elasticity

In comparison with Agile it was found that projects with high resource count and sub-contract development benefit more from traditional methodologies.

Future Research

The paper shows that agile software development methods requires a lot of skill and effort from those who use them. This may lead to that teams consisting of mainly inexperienced developers cannot fully reap the benefits of these methods thus leaving them dissatisfied with their choice of software development method. There may be reason for dissatisfaction among users of certain methods is that why a method may fail to satisfy a specific organization and succeed in another.

Another interesting aspect to be studied is, no or low customer participation would mean that Agile would not work at all, while low complexity would still mean that Agile could be used with this project.

REFERENCES

[1] Rashina Hoda, James Noble, Stuart Marshall: "The Impact of Inadequate Customer Collaboration on Self-Organizing Agile Teams", 2010.

[2] Maarit Laanti, Outi Salo, Pekka Abrahamsson: "Agile Methods Rapidly Replacing Traditional Methods at Nokia: A Survey of Opinions on Agile Transformation", 2010.

[3] Subhas Chandra Misra, Vinod Kumar, Uma Kumar: "Identifying Some Important Success Factors in Adopting Agile Software Development Practices", 2009.

[4] Agile. (2009), "Cambridge Advanced learner's Dictionary Online". Retrieved 09/12/2009.

[5] Helen Sharp, Hugh Robinson, Marian Petre: "The Role of Physical Artefacts in Agile Software Development: Two Complementary Perspectives", 2009.

[6] Simona Motogna, I. Lazar, B. Pˆarv, I. Czibula: "An Agile MDA Approach for Service-Oriented Components", 2009.

[7] Ambler, S. (2008a), "Acceleration: An Agile Productivity Measure" Retrieved 08/12/2009.

[8] Ambler, S. (2008b), "Results from Scott Ambler's February 2008 Agile Adoption Survey".

[9] Frank K.Y. Chan, James Y.L. Thong, "Acceptance of Agile Methodologies: A Critical Review and Conceptual Framework", 2008.

[10] Woi Hin, Kee: "Future Implementation and Integration of Agile Methods in Software Development and Testing", 2006

[11] Beck, K., and Fowler, M. Planning Extreme Programming, Boston: Addison Wesley, 2001. Bossi, P., and Cirillo, F. "Repo Margining System: Applying XP in the Financial Industry", in Proceedings of the 2nd International Conference on eXtreme Processing and Agile Processing Software Engineering (XP 2001), Villasimius, Italy, May 2001.

[12] Cockburn, A. & Highsmith, J. 2001, "Agile Software Development": The People Factor. Computer, 34, No. 11, pp. 131.133.