Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 10, Issue, 01, pp.64764-64770, January, 2018

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

LEVERAGING OF CAPABILITY OF AGILE SOFTWARE FOR OPTIMIZATION NEEDS

***Mrityunjay Kumar and Shachee Mishra**

Department of Computer Science and Engineering,
Rajarshi Rananjay Sinh Institute of Management and Technology, Amethi, UP, India

ARTICLE INFO

Article History:
Received 21st October, 2017
Received in revised form
09th November, 2017
Accepted 28th December, 2017
Published online 31st January, 2018

Key words:

Agile methodology,
Requirement,
Importance of agile software,
Software quality.

ABSTRACT

Agile software design and development methodologies have been gaining heavy-handed attention in the field of the software engineering in the research community and additional to being highly adopted by the software development industry. Basically, Innovative industry needs an agile concept to rapidly respond and changing conditions until encourages customer satisfaction by early incremental delivery of software. A great number of notable methods have been proposed and various surveys have been presented by many researchers. People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.

Copyright © 2018, Mrityunjay Kumar and Shachee Mishra. *This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.*

Citation: Mrityunjay Kumar and Shachee Mishra. 2018. "Leveraging of Capability of Agile Software for Optimization Needs.", *International Journal of Current Research*, 10, (01), 64764-64770.

INTRODUCTION

Agile software development approach allows fast changing organizational business needs using traditional methods. Agile is an iterative and adaptive process that identifies and addresses risk, promotes higher quality results, implements cost control and keeps pace with technological changes. The quality assurance and testing activities add significant cost to the project which asks for the rational management and allocation of testing. The term agile means very active. As a cure to many of the problems that have afflicted software project work, in recent years, agile software development has been put forward (Gauri yogesh kapure, 2008) In addition, emphasizing productivity might encourage your team to take shortcuts and to be less rigorous in their work, which could actually harm productivity (James Shares, 2008) Agile method is a term that embraces a number of techniques that share common principles. These principles are articulated in what is called the agile Manifesto. The principles emerged from an analysis that older methods were simply too big and unwieldy that there was a need to use more lightweight approaches to development. These new methods explicitly recognize that software development is primarily about individual skill and communications between people (between developers and with the clients).

****Corresponding author:*** **Mrityunjay Kumar,**
Department of Computer Science and Engineering, Rajarshi Rananjay Sinh Institute of Management and Technology, Amethi, UP, India.

Agile team recognizes that software is developed by individuals working in team and that the skills of these people, their ability to collaborate is at the success of the project.[1] A major concern with such lightweight requirements "engineering" is that non-functional requirements, such as security, are often neglected since system functionality is the focus (Ramesh, 2010). One of the best-known of the methods is named. Extreme programming (XP), but others are DSDM, SCRUM, Crystal and FDD. (Douglas Bell, 2000). This paper represents various type of approach to determine its capabilities basically, how can we use its methods for optimization needs. Higher quality is achieved with integrated, cross functional teams that are able to identify and address issues earlier, and faster, with more cost-effective solutions.

An Agile Overview

Agile is a software development methodology to develop the software incrementally using phases of 1 to 4 weeks so that the development process is classified with the changing business needs. Instead of a single-pass development of 6 to 18 months where all the requirements and risks are predicted upfront, Agile adopts a process of frequent feedback where a workable product is delivered after 1 to 4 week iteration. Basically, Agile breaks down larger projects into small, manageable chunks called iterations. At the end of the each iteration something of value is produced.

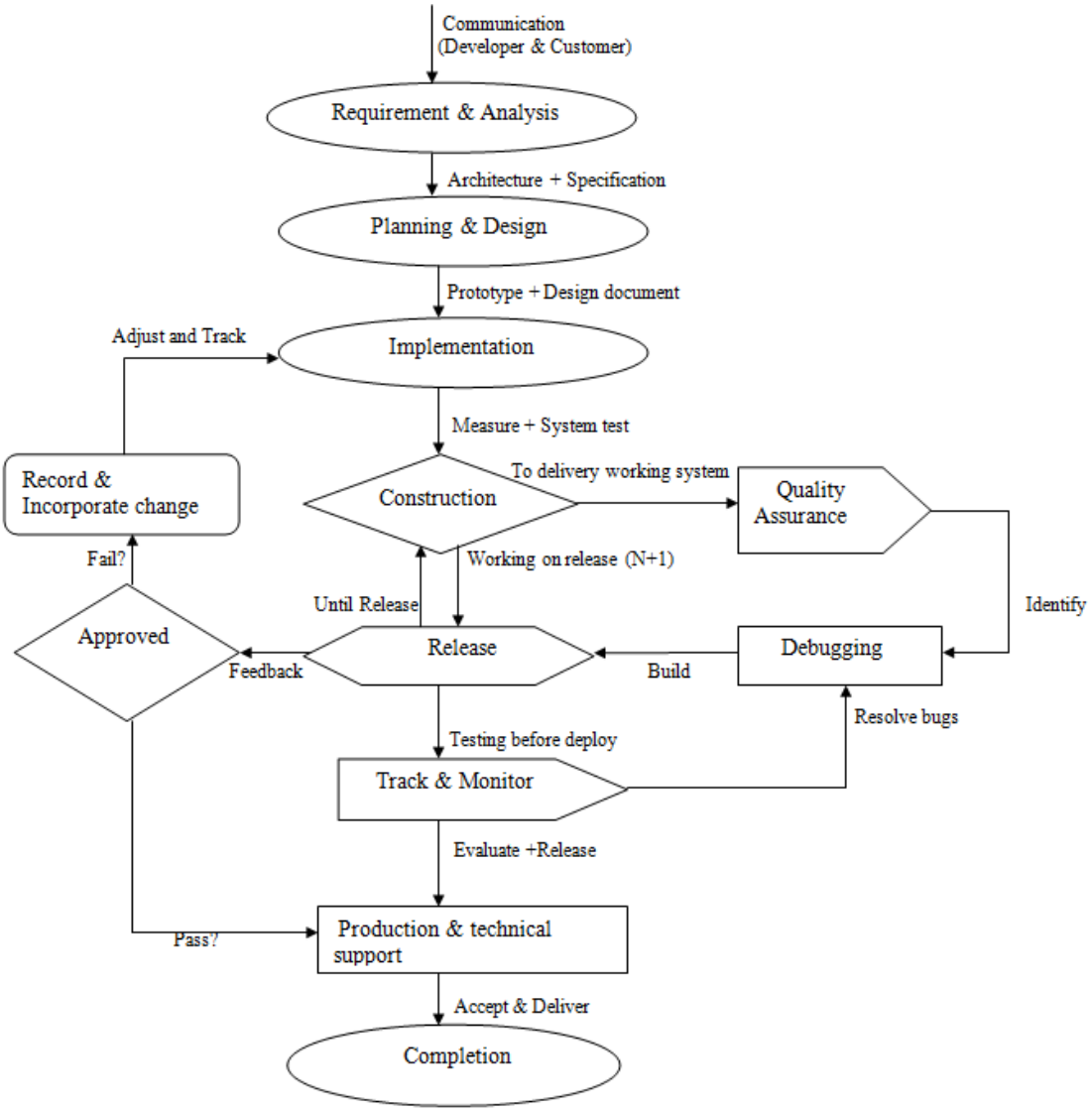

Fig. 1. Lifecycle of agile software development

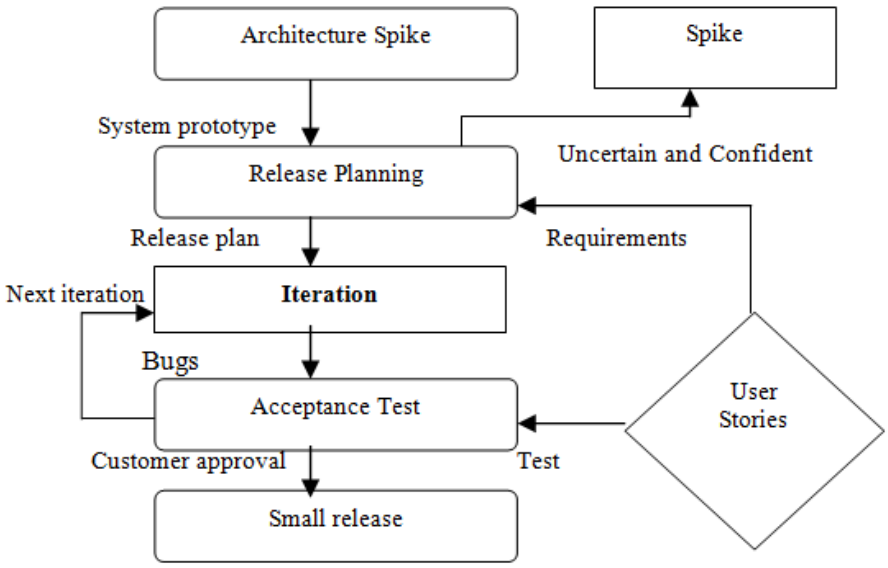

Fig. 2. Process of extreme programming

The product produced during the each iteration put into the any industry or any software company to gain feedback from users or stakeholders. Agile software engineering combined a philosophy and set of development guidelines. The approach of agile software customer satisfaction and development of the delivery of software, small highly motivated project teams, informal methods; minimal software engineering work products; and overall development simplicity. The development guidelines stress delivery over analysis, design, active and continuous communication between development and customers (Roger, 2010). This might mean a computer aided software engineering (CASE) tool but it also includes non-computer tools. Here are examples. Sketches can be made on paper, using color as appropriate for all sorts of diagrams, informal and more formal. A scanner or digital camera can record the results for computer storage (Douglas Bell, 2000) The modern business environment that spawns computer based systems and software product is fast-paced and ever changing. Agile software engineering represents a responsible alternative to conventional software engineering for certain classes of software and certain types of software project. It has been demonstrated to deliver successful system quickly (Roger, 2010) A major concern with such lightweight requirements "engineering" is that non-functional Requirements, such as security, are often neglected since system functionality is the focus (Ramesh, 2010). The capability that will differentiate the most is the hardest to bring about:

- What are your experiences?
- Has anyone seen agile impact on program office?
- How has agile worked well in program office?
- Have you faced evolving towards an agile way of thinking?

MATERIALS AND METHODS

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 (Agile Alliance, 2009). 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 (Erickson *et al.,* 2005) After the process of agile methodologies, software development approaches evolved into Object-Oriented (OO) methodologies, which include: They include not only the objects and object oriented approach, but also the best practices from structured extreme programming of agile methodologies defined as following:

- Evolving requirements: Most of the customers do not have a clear vision about the specifications of their requirements at the early stages.
- Customer involvement: lack of customer involvement leads to higher chances of project failure.
- Deadlines and budgets: 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.

Agile methodology basically used for

- Helps to explain how your team works.
- Helps us understand responsibilities and priorities.
- Helps measure progress and show progress.

On this methodology, will help to overcome the problems mentioned earlier, by always may if customers, wants to change, 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

Extreme Programming

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. [(9)] 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. To many, XP is a set of 12 interdependent software development practices. Used together, these practices have had much success, initially with small teams, working on projects with high degrees of change. 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. In XP, every contributor to the project is a member of the "Whole Team," a single business/development/testing team that handles all aspects of the development. Central to the team is the "Customer," one or more business representatives who sit with the team and work with them daily. XP also emphasize teamwork; experiences from all stakeholders are employed to meet the specific goals, and within the given constraints (Extreme Programming, 2009). Next iteration XP concentrates on producing executable code and automated test drivers. This focus on source code makes.

Agile Modeling

Agile Modeling (AM) was established in 2002 by Scott Ambler. Agile modeling 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 Home Page, 2009). The values of agile modeling, 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 (Agile Modeling Home Page, 2004). Basically, agile methodology creates as intermediate between rigid drastic and lightweight methodologies, by designate that developers communicate architectures through applying its practices to the modeling process (Erickson *et al.,* 2005). 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 (Agile Modeling Home Page, 2009). In agile modeling, consist to the communication between customers and development of the projects in the reputed companies. Basically, agile modeling provide a way for creating meaningful software which is required by customers and it's also always welcome to change if present, there any type of error and additional requirement by the customers.

SCRUM

Scrum is agile methodology, SCRUM undertake the concept of empirical process control for the management of complex and changing software projects. Scrum holds that straight forward defined processes cannot be used to effectively manage complex and dynamic software projects. Scrum is a management and control process used for developing and sustaining complex products in order to build software that meets business needs, incrementally and empirically. It is considered a widely used agile method, first described in the year 1996 (Schwaber, 1996). Objective of SCRUM to simplifying project control through simple processes, easy to update documentation and higher team iteration over exhaustive documentation (Cristal, 2008). The mechanics of SCRUM expects a team to build a product backlog, a place where one can see all requirements pending for a project, sized based on complexity, days or some other unit of measure the teams decide. Inside a product backlog you have a simple sentence for each requirement, something that will be used by the team to start discussions and details of what is needed to be implemented by the team. The SCRUM process consists through the project team and SCRUM defines simple roles as following: (Carmel, 2005).

- The product owner: responsible to be the voice of business inside a SCRUM project. May be also seen as the business partner role.
- SCRUM Team member: In SCRUM team member defined by its developers, testers and other roles you can find.
- SCRUM Owner: SCRUM owner responsible for to keep the team focused on the practices and values that are needed to be applied inside the project and also responsible to help the team.

The authors proposed a framework to utilize the user experience design in SDLC in organizations that use SCRUM in association with Capability Maturity Model Integration practices. The authors utilized the user experience design dimensions recommended by the Human Factors Institute, which involve training of professionals, creating and managing metrics to evaluate the usability, and establishing a successful cases database for training purposes. Many studies focus on using the scrum agile method in software companies, such as that study using of SCRUM in small business.

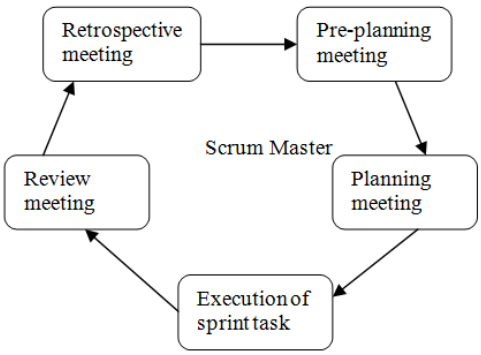

Fig. 3. Process of SCRUM

Crystal methodologies family

The methods are color-coded to signify the risk to human life. For example, projects that may involve risk to human life will use Crystal Sapphire while projects that do not have such risks will use Crystal Clear. Crystal focuses on six primary aspects: people, interaction, community, communication, skills, and talents. Crystal clear is designed for the small projects having six team members. Crystal clear needs project team to be located at same place due communication structure (Boehm, 2007).

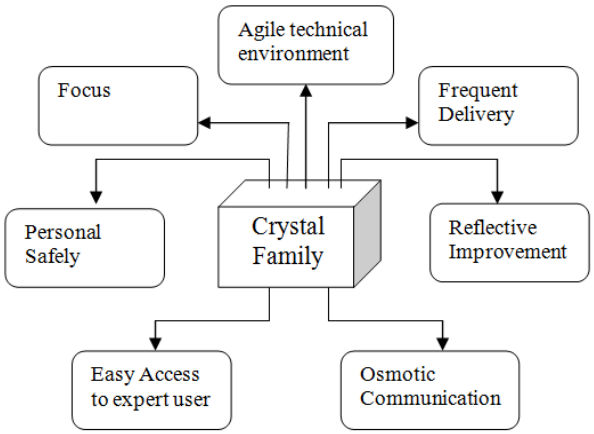

Fig. 4. Quality of Crystal family

Feature- Driven Development

Feature-Driven Development (FDD) was introduced by Jeff De Luca in 1997. Jeff De Luca and Peter presented the FDD model by combining the concept of feature with software development process in their book Java Modeling in Color with UML (Abrahamsson, 2002) FDD is highly adoptive development model that greatly stress on designing and modeling aspects of project (Mnkandla, 2007). Feature-Driven Development presents the role of software like that; key roles, supporting roles and additional roles (Abrahamsson *et al.,* 2002). This section explain key roles in FDD which includes, project manager, development manager, chief architect, chief programmer, class owner, domain expert and feature team. FDD teams specially focus on quality throughout the development phases (Abrahamsson *et al.,* 2002). This is an iterative phase in which multiple features teams actually implement the design classes and modules. After coding, code inspection, unit testing and integration testing is performed.

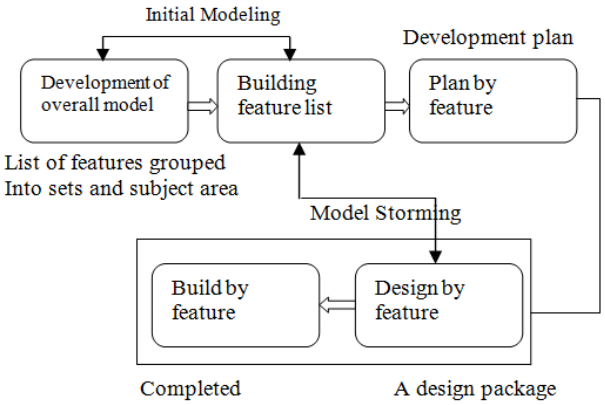

Fig. 5. Process of FDD

Classes are built in the sequence which these are defined in plan by feature phase. After completing successfully, developed features are included in main build and another with new features set is started again. Uniquely sees very small blocks of client-valued functionality, called features, organizing them into business-related groupings Focuses on producing working results every two weeks and facilitating inspections; managers know what to plan and how to establish meaningful milestones (Palmer, 2002).

Adaptive Software Development

In ASD (Adaptive software development) is a move towards for the adaptive software and adaptive resources and leaving the deterministic practices in the context of complex systems and environments. Adaptive Software Development focuses on cooperation and learning build complex systems. It is also include from the RAD (Rapid Application Development). The phases are named in a way to emphasize the role of change in the process. "Speculation" is used instead of "Planning", as a "plan" is generally seen as something where uncertainty is a weakness, and from which deviations indicate failure. Similarly, "Collaborate" highlights the importance of teamwork as the means of developing high-change systems. "Learn" stresses the need to acknowledge and react to mistakes, and the fact that requirements may well change during development (Salo *et al.,* 2002). "Adaptive Software Development is lifecycle as the involvement and evaluate of each model, with the phase names speculate, collaborate and cooperation the unpredictable coverage area of increasingly complex systems."

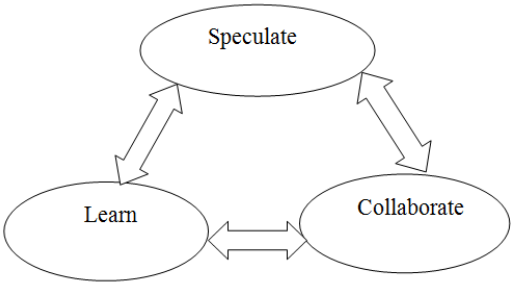

Fig. 6. Process of Adaptive software development

ASD is explicitly component-oriented rather than task-oriented. In practice, this means that the focus is more on results and their quality rather than the tasks or the process used for producing the result. The way of adaptive model this point of view is through adaptive development cycles that contain the Collaborate phase, where several components may be under concurrent execution and development. Planning the cycles is a part of the iterative process, as the definitions of the components are continuously refined to reflect any new information, and to cater for changes in component requirements. There are define the lifecycle of adaptive software and discuss to the place of Speculate, Learn and collaborate. Adaptive development goes further than its evolutionary heritage in two key ways. First, it explicitly replaces determinism with emergence. Second, it goes beyond a change in Life Cycle to a deeper change in management style.

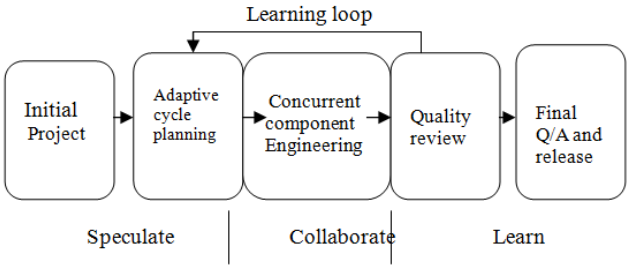

Fig. 7. Process of ASD Lifecycle

Agile Manifesto

In February 2001, a group of developers of any type of project interested in advancing lightweight development methodologies got together to discuss about their views and to find common ground, and agile was born. The developers who created agile understood the importance and advantage of creating a model in which each the phases in the development cycle "learned" from the previous phases. The result was a methodology that was more flexible, efficient, and team-oriented than any other of the previous models. Thus, we are providing better ways of developing software by using this type of approach and doing it. Through this work we have performs some type points of 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.

Agile Manifesto to have 12 core principles for guidance as:

- **Customer Satisfaction -** Highest priority is given to satisfy the requirements of customers through early and continuous delivery of valuable software.
- **Welcome Change -** Changes are inevitable during software development. Ever changing requirements should be welcome, even late in the development phase.
- **Deliver Software** –Deliver working software frequently, ranging from a few weeks to a few months and considering shorter time-scale.
- **Cooperation between peoples -** Business people and developers must work together during the entire life of a project.
- **Motivation -** Projects should be built around motivated individuals. Provide a way to support individual team members and believe them.

- **Face-to-face Conversation -** Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.
- **Measure progress on working software -** Working software is the key and it should be the primary measure of progress.
- **Maintain Pace of Constant -** Agile processes aim towards sustainable development. In the industry, the developers and the users should be able to maintain a constant pace with the project.
- **Monitoring-** Pay regular attention to technical excellence and good design to enhance agility.
- **Simplicity-** Keep things simple and use simple terms to measure the work that is not completed.
- **Self-organized Teams-** An agile team should be self-organized and should not depend heavily on other teams because the best architectures, requirements, and designs emerge from self-organized teams.
- **Regularly, Work review-** Review the work as a regularly and done at regular intervals so that the team can oppose on how to become more effective and adjust its behavior accordingly.

Importance of Agile Software

Agile software address perfectly defined for customer's needs. During the each phases of life cycle, user invited as the encouraged, it providing visibility and transparency and other showing the actual progress or value of projects. Agile method is all about iterative planning, making it very easy to adapt when some requirements change. There are defined the importance to testing program when working in an agile methodology defined as following:

- **High product quality:** Incorporating continuous integration and daily testing into the development process, Defining and elaborating requirements just in time so that the knowledge of the product features is as relevant as possible.
- **Higher customer satisfaction:** Delivering products to market quicker and more often with every release. The clients get early access to the product during the phases of life cycle.
- **Reduce risk:** Adaptation to the client's needs and preferences through the development process. Agile commonly uses user stories with business-focused acceptance criteria to define product features.
- **Better Communication:** Because the tester and developers are working on the same team, they can more easily communicate with each other.
- **Early and predictable delivery:** By using time-boxed, fixed schedule Sprints of 1-4 weeks, new features are delivered quickly and frequently, with a high level of predictability.
- **Focuses on goal:** In this, the client to determine the priority of features, the team understands what's most important to the client's work of business, and how to deliver the features that provide the most business value.

Conclusion

The benefits of agile development are both numerous and critically important. To encourage long-term value and reduce the threats of project success, it pays role as a flexible. Many teams and organizations have realized benefits, advantages and improvement by adopting agile methodologies. An independent working module is built after the completion of phases. Each type of phases should not consume more than two weeks to complete a code. Agile methodologies provide a way to invite the developers to get involved in testing, rather than a separate quality assurance team. This type of methodology used for advanced business requirements became more demanding. However, challenges continue to come because of unique requirements and transforming business landscapes, especially difficult in large enterprises. Thus, the problem of measure agile delivery process continues to be difficult to solve. It includes a set of software development approaches, they have some type of variations, but still they can share the same basic concepts. Therefore, the objective of all industries, or the main objective of all companies to achieve that repeatable, predictable process that will improve productivity and quality continues on.

REFERENCES

Abrahamsson, P., Salo, O., Ronkainen, J. and Warsta, J. 2002. "Agile software development methods: Review and analysis.

Agile Alliance. Manifesto for Agile Software Development. (Online) Retrieved 16th March 2009.

Agile Modeling Home Page. Effictive Practices for Modeling and Documentation. (Online) Retrieved 17th March 2009.

Beck, K. and Andres, C. 2004. Extreme Programming Explained: Embrace Change (2nd Edition), AddisonWesley, Boston.

Boehm, B. 2007. "A survey of agile development methodologies." Laurie Williams.

Carmel, E., Tija, P. 2005. "Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce", UK: Cambridge.

Cristal, M., Wildt, D. and Prikladnicki, R. 2008. Usage of SCRUM Practices within a Global Company. Global Software Engineering, 2008.

Douglas Bell, Software engineering, Addison-Wesley, 2000.

Erickson, J., Lyytinen, K. and Siau, K. 2005. Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research. *In Journal of Database Management,* 16(4).

Erickson, J., Lyytinen, K. and Siau, K. 2005. Agile Modeling, Agile Software Development, and Extreme Programming: The State of Research. In Journal of Database Management, 16(4).

Extreme Programming. What is Extreme Programming? (Online) Retrieved 18th March 2009.

James Shares and Shane Warden, 2008. ''The art of Agile Development.

Lisi Romano, B. and Delgado Da Silva, A. 2015. Project Management Using the Scrum Agile Method: A Case Study within a Small Enterprise. 2015.

Mnkandla, E. and Dwolatzky, B. 2007. "Agile Software Methods: State-of-the-Art." Agile Software Development Quality Assurance, 1

Mrs. Gauri yogesh kapure, 'Software Engineering', 2008.

Palmer, S and Felsing, J. 2002. A Practical Guide to "Feature-Driven Development.

Ramesh, B., Cao, L., Baskerville, R. 2010. Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal* 20(5) (November).

Ramesh, B., Cao, L., Baskerville, R. 2010. Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal* 20(5) (November).

Roger S. Pressman 2010. Software Engineering practitioner approach.

Salo, P., Ronkainen, O., J. and Warsta, J. 2002. "Agile software development methods: Review and analysis.

Schwaber, K. 1996. Controlled Chaos: Living on the Edge.

****