IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 3, March 2015.
www.ijiset.com
ISSN 2348 – 7968

Agile Software Development Methodologies and Its Quality Assurance

Aslin Jenila.P.S
Assistant Professor, Hindustan University, Chennai

Abstract:

Agility, with regard to software development, can be expressed as the flexible, ready to change and quick-responsive nature of software development process. In agile development, software industry moved development from process oriented to people oriented. Some of the agile methods are Extreme programming, Scrum, Crystal, Feature Driven Development, etc. Quality Assurance activity to be considered while moving towards the agile development. This paper furnishes the Agile Software development methodologies and generalizes the importance of the Quality while using those methodologies for software development.

Keywords: Agile Software Development, Quality Assurance, Agile Testing.

1. INTRODUCTION:

Agile Software Development approach is short iterative, incremental and people-centric. Agile software development is actually a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.

Agile software development encourages people collaboration through the project. As compared to conventional way of software development, it responds to change efficiently as it is incremental and iterative.

"Software Quality Assurance (SQA) is defined as a planned and systematic approach to the evaluation of the quality and adherence to software product standards, processes, and procedures". Agile shifted the responsibility of testing to developers by adopting the techniques like Pair Programming and Test Driven Programming.

This paper is organized as follows: Section II describes what the agile development is, how the agile differ from the traditional. Section III explains the different

methodologies of agile development and its processes. Section IV briefs the Software Quality Assurance on Agile methodologies and its agile testing.

2. AGILE SOFTWARE DEVELOPMENT:

Agile is defined as Values, principles and practices that foster team communication and feedback to regularly deliver customer value through working software." Agile software development processes are built on the foundation of iterative development. Agile processes use feedback rather than planning as their primary control mechanism.

Characteristics of agile are Iterative, Modularity, Time Boxing, Parsimony, Incremental, Adaptive, Convergent, Collaborative, and People Oriented.

Fig. 2.1 Single Iteration of Agile Development Process

The traditional way to develop software methodologies follow the generic engineering paradigm of requirements, design, build, and maintain. These methodologies are also called waterfall–based taking from the classical software development paradigm. They are also known by many other names like plan–driven, documentation driven, heavyweight methodologies, and big design upfront.

Due to constant changes in the technology and business environments, it is a challenge for TSDMs is the

ISSN 2348 – 7968

inability to respond to change that often determines the success or failure of a software product.

Where the waterfall approach is based in predictability and processes, an Agile approach focuses on adaptability and response time to changing requirements. Another important advantage of Agile over the waterfall model is the recursiveness of the work pattern. This means that we can make modifications to the completed stage in Agile while it is not allowed under waterfall model.

Agile Vs Waterfall Model

Waterfall model is the primitive model type and is the basic of all elements. Agile is the successor of the waterfall model.

Difference between Agile and Waterfall Model

1. The main advantage is the backward scalability in Agile. Under waterfall approach we cannot change the decisions and implementations that we had made under the previous stages. If we want to make changes under waterfall we have to build the entire project from the scratch once again.
2. The flexibility to error check under any part of the development stage makes Agile more bug free and less erroneous as compared to Waterfall which can only test bugs at the end of the development module.
3. Since Agile provides flexibility to make changes as per customer requirements it is more inclined towards better client satisfaction. This is a real set back for the Waterfall model which doesn't allow any modifications once the module has been completed.
4. Under Agile development modular partitioning of the software can be effectively carried out as compared to its counterpart. Though both of them allows option for segregation the later lacks the modifications in the implementation stage. The rules are set down before the commencement of the project hence it hinders further break down of the logical module. Whereas Agile can be of great help under such situations and can allow simultaneous development of different modules at the same time as per time bound requirement.

3. AGILE DEVELOPMENT AND METHODS

Agile methods try to avoid this weakness of "waterfall" by doing iterative development. Each iteration is meant to be short (1-3 weeks) and includes all of the above steps. This guarantees that design errors are discovered at the early stages of development.

Some of the Agile Methods are Extreme programming, Scrum, Dynamic System Development, Adaptive software Development Method, Crystal, Feature Driven Development etc.

3.1 EXTREME PROGRAMMING

Extreme programming promises productivity improvement and concentrates on the development rather than the managerial aspects of the software projects.

The technical premise of XP is that the cost of software change over time does not have to increase exponentially over time, but rather can be made to increase much more slowly, eventually reaching an asymptote. This leads to dramatically different behavior, where decisions are deferred until as late as possible, and implementation focuses on doing the simplest thing that could possibly work.

Some practices that are needed to be follow in the development process.

- The Planning Game
- Small Releases
- Metaphor
- Simple design
- Testing
- Refactoring
- Pair
- Collective ownership
- Continuous integration
- 40-hour week
- On-site

ISSN 2348 – 7968

• Coding standards

Test Driven Development, refactoring, system metaphor, and pair programming play the main role in achieving the QA (Quality Assurance). These main practices are harmonize with each other, test driven development verify that written code is bug free. Refactoring always make sure the simplicity of code to avoid the complexity in the developing system. System metaphor provide the basic understanding of system architecture, it reduces the possibility of system failure if development work carried out according to the architecture. Pair programming is most popular practice of XP in which two programmers share their ideas and identify mistakes collectively that helps to develop the system bug free. So we may say these practices play their role to develop the better quality product with minimum risk of errors.

3.2 SCRUM

Scrum methodology includes both managerial and development processes. Scrum provides project management with frame work that includes development tasks like requirement gathering, design and programming are take place. It does not provide any specific method to be applied; it guides the management how their team should function to maintain the flexibility of the system, in applying the environmental changes. The main practice in scrum is daily 15 minutes meeting to coordinate and integrate the development issues

Scrum practices:

Product backlog: The team writes all currently identified tasks, in a list called the Backlog
Sprints –Sprints are 30-days in long. Developers are assigned with number of task to execute a sprint.
Sprint planning meeting
Sprint Backlog – A list of features are assigned to a particular Sprint.
Daily Scrum – the scrum team conducts frequent meetings. These daily meetings are more or less 15 minutes long, to know about the process.

This methodology uses the iterative and incremental approach towards software development.

There are three main roles in it, namely, the 'Scrum Master', 'Product Owner', and the 'Team'. The Scrum Master is the person who maintains the entire project. His role is similar to that of a project manager. The Product Owner is the person who represents the stakeholders and the business. Last but not the least is the team. Normally, the team is cross functional. It is this team that takes care of the actual analysis, design, implementation, testing, etc., of the project. Generally, the team size is limited to 10 people.

Agile Scrum Process Flow:

In this process, the entire software development life cycle is divided into small parts known as sprints, which have a defined period of time for development. Iterative processes are included within them. The project is defined completely by taking help of the cross functional teams. The project is then divided into small parts. Once the parts have been made, the team decides who is going to be responsible for which part of the sprint. Accordingly, in the stipulated time period, the work is carried out.

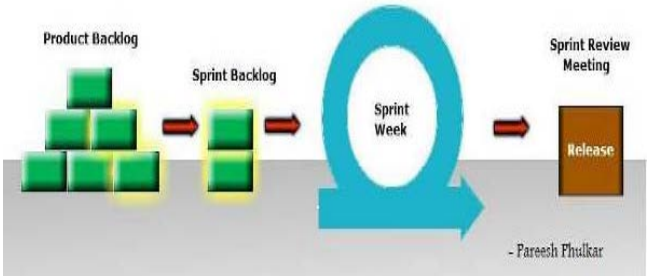

Fig. 3.1 Agile-scrum-process

After the first iteration, it is the job of the Product Owner to define the backlog. The tasks in the backlog are prioritized by the owner. In some cases, the team might be asked to prioritize the items in the backlog. The plan is then reviewed. The Scrum Master, Product Owner, and the team are a part of the sprint planning meeting.

The size of each item is determined by the team after the review meeting. This helps in deciding the items which can be completed in the current sprint. Often, the time used for this decision is not more than 4 hours. This is

ISSN 2348 – 7968

followed by the work of clearing the backlog. The highest priority items are completed first and the items with lower priority are taken into consideration later.

Depending on the policy decided, there might be a daily or a weekly review meeting. This meeting helps in deciding the tasks accomplished as well as the next items to be worked upon. For the items which have not been completed, the problems are discussed and solutions are worked upon. Modifications are done to the sprint backlog either everyday or weekly. This helps create the burn-down chart where the progress made as opposed to the time remaining for the sprint to get over, are charted out. At the end of the sprint, the progress is displayed.
Finally, the entire team gathers together to review the sprint. This meeting is aimed at finding out what went wrong in the computer programming of the preceding sprint. At the same time, the change of action that needs to be carried out is also decided. After this starts the next sprint.

In order to achieve quality, scrum practices daily sprint meetings, continuous integration and acceptance testing are supposed to be followed.

3.3 FEATURE DRIVEN DEVELOPMENT:

The key advantage of this method is to design the domain of the software to be produced before development. The method starts with collecting the requirements from the users and building up the overall model of the project. Next step is to make a list of features which are the client-valued functions. Next step is to make a plan for developing the features. Last step is modeling iteration in which first UML modeling is done for each feature.

Feature Driven Development (FDD) is an iterative software development methodology intended for use by large teams working on a project using object-oriented technology.

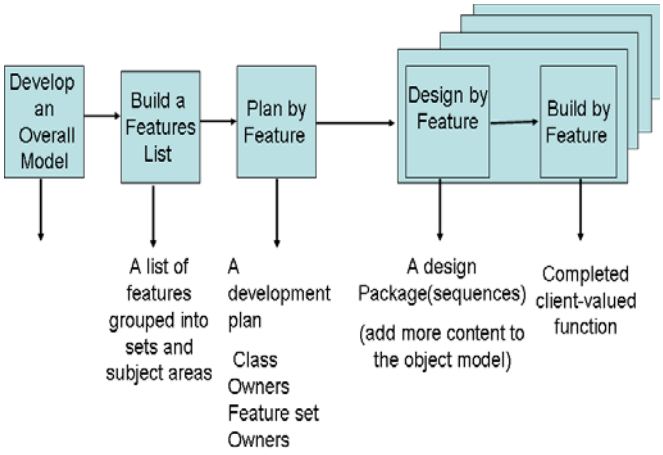

Fig. 3.2 The five processes of FDD with their outputs

FDD is useful because it shows that teams can spend a short amount of time at the beginning of the project to establish a clear understanding of the domain in which they are working and use that understanding to formulate a rough plan without getting stuck in analysis and design paralysis.

FDD starts with the creation of a domain object model in collaboration with Domain Experts. Using information from the modeling activity and from any other requirement activities that have taken place, the developers go on to create a features list. Then a rough plan is drawn up and responsibilities are assigned. Small groups of features feature that lasts no longer than two weeks for each group and is often much shorter are taken up. FDD consists of 5 processes.

3.4 CRYSTAL:

Crystal methodologies are categorized according to the project size that they address.
Every Crystal methodology:

- Enforces a development process framework.
- Requires that a set of certain general process elements be used.
- Requires that certain work products be produced.

Crystal Methods

ISSN 2348 – 7968

There are mainly three crystal methodologies constructed these are: Crystal Clear, Crystal Orange and Crystal Orange web. All of these methodologies provide tools and standard roles to be implemented in software development process.

Crystal Methods is primarily focused on communication, with special focus on interaction, community, people, and skills.

As noted, Crystal Methods is a collection of methodologies based on two fundamental assumptions: (1) teams can streamline their process as they work and become a more integrated, optimized team and (2) projects are unique and dynamic and require methods that are specifically designed for each effort.

Another theme that has run through Cockburn's work from his first articles to his current Crystal Methods development work is the concept of product development as a game, a cooperative, interactive activity that should be designed in a way.

Crystal practices are mandatory to follow but can be replaced with any other practice because Crystal do not limit to adopt any practice to maintain the development process in the crystal practices automated testing, direct user involvement and side by side programming work are mainly focused by development team to achieve and maintain the quality of product.

4. SOFTWARE QUALITY ASSURANCE IN AGILE:

"Software Quality Assurance (SQA) is defined as a planned and systematic approach to the evaluation of the quality of and adherence to software product standards, processes, and procedures".

Quality assurance activities, in software development, are the backbone of whole project. These activities are not only responsible of product quality, but also for process quality. In conventional software development QA is a separate group of QA experts. As the trends of software development moved towards agile development, QA activities also got changed.

Ultimate purpose of quality assurance is to attain better quality in software product. Different approaches and several quality models are followed in this discipline. SQA activities are practiced during project and these activities include process control, documentation, audits and verification and validation.

SQA is not only responsible for a particular project but also maintain the processes and culture of organization. QA activities have gained the importance of back bone in an organization. These activities are responsible for development process, quality of product and these activities keep the project on track.

Agile shifted the responsibility of testing to developers by adopting the techniques like Pair Programming and Test Driven Programming.

Pair programming and Test-Driven Development (TDD) are some of the key practices of agile development to achieve quality in software products. Using these approaches, agile development makes testing an integral part of project.

In agile, developers are supposed to write tests and test their code or each other's while doing Pair Programming. Acceptance testing is the responsibility of customer who is participating in project. Agilists claim that customer involvement in project and in testing leads to develop the software for higher conformance to requirements.

Test-First Development (TFD) approach where developers are to write all test cases and tests before starting actual programming.

Scott W. Ambler expresses TDD as:
TDD = Refactoring + Test First Development

Pair Programming

Pair Programming (PP) means, "All production code is written by two people at one screen/keyboard/mouse." Purpose of adopting PP is to monitor and learn from each other continuously, while writing the code. "Pair programming can improve design quality and reduce defects".

5. CONCLUSION:

Agile methodology improves the quality of the software products both in organizations running large and distributed projects, and the approach recommended

ISSN 2348 – 7968

overcoming these challenges to achieve a successful agile deployment. In future Agile development methodologies can be used in cloud computing which Provides an Unlimited Number of Testing and Staging Servers, turns Agile Development into a truly Parallel Activity, Encourages Innovation and Experimentation, Enhances Continuous Integration and Delivery and eases Code Branching and Merging. The major challenge of Agile methodology is its scalability (Size matters).

REFERENCES:

1. Malik Imran Ullah, Waqar Ali Zaidi " Quality Assurance Activities in Agile"
2. Nari Kannan, "How Cloud Can Enhance Agile Software Development", 2012.
3. Selby. R.W, Enabling reuse based software development of large scale systems, 2005.
4. Sandhya Kakkar, Implementation Aspects of Software Development projects, 2006.
5. Aoyama.M. "Agile software process and its experience",1998
6. Chowdhry,A.F; Huda.M.N, "Comparison between Adaptive Software Development and feature driven development", 2011.