Improvement in Agile Development Technique Using Requirement Engineering

Engr. Muhammad Fahad Khan, Engr. Waqas Ali and Engr. Zia Ur Rehman
Software Engineering, University of Engineering and Technology Taxilla, Pakistan

***Abstract*— World has turned into a global village. Fast growing economies has turned the way how world works. Old methods of software development are losing their grounds against agile development. Agile is paving its way to become the first choice of every person involved in software development. Agile development techniques are another name for delivering software product on time and within budget. Like all other developing techniques agile must have mechanisms to collect and manage requirements. This paper presents how different requirement engineering principles can be incorporated in agile development. Paper will cover different agile methods and requirement engineering techniques. There are lot of integration and differentiation in both techniques appears in this paper.**

***Keywords*— Requirement Engineering, Agile, Prototype and Techniques**

I. INTRODUCTION

Agile software development methods are now the first choice of software developers [1]. It is in high demand in academia and industry due to its benefits in developing software quickly, meeting customer needs, and keeping pace with the quick variation in requirements [2]. Agile development is solely for client contentment by the means of swift and incessant readiness of effective system's segments came into being by an interactive mechanism with the design point that uses minimum requirements [3]. They have left behind all other methods which were once used to deliver quality software. Agile software development methods stand on principles of iterative and incremental development, in which requirements come up through joint effort of self-organizing, cross-functional teams [1]. All this stuff helps to make product which is ready to accept change. In agile we prefer people over processes, limited documentation, lot of customer involvement.

Agile development techniques are quickly replacing every other method used in a lot of industrial projects. Its flexible feature provides the means to address many common problems faced in the development of software systems [4]. Many organizations have started to use agile development processes to speed up the development cycle and to improve the ability of the company in reacting to changing requirements [5].

In contrast to agile, requirement engineering is structured approach involving specific steps in course of developing a software product. Steps involve identify, analyse, document and validate. There are bundles of documentation which covers each and every step in course of development. Requirements engineering can benefits us in terms of reduce work, increase efficiency and efficacy, more reliable estimates, lower development time, reuse of successful projects components, predictable results and ease of moving from one project to another. Integrating agile development techniques with requirement engineering is a challenging task.

In this paper we will try to identify some requirements engineering techniques which can be applied to agile development and how can agile development be benefited from these techniques to develop a more acceptable product for a customer.

II. REQUIREMENT ENGINEERING

Requirement Engineering means requirements of product to be developed are clear, managed and tested methodically. The purpose of RE is to ensure that a product development team builds a system that satisfies customer and user needs [1]. Requirement engineering focuses on "what" not "how". Requirement engineering in the start is cost effective; this activity in the start saves money and time in future. As the time passes in software development, mistakes discovered are more expensive to correct [6].

High-level activities in the requirements engineering are requirements elicitation, requirements management, requirements analysis and negotiation, requirements specification, requirements validation, and documentation.

Following section briefly describes all these techniques:

A. *Requirement Elicitation*

Another name of requirements gathering, involves collection of requirements of software product from different stake holders i.e., users, managers and customers. Requirements elicitation is a collection of interviews, questionnaires, user observation, workshops, brain storming, use cases, role playing and prototyping [7]. Each one is described below:

1) Interviews

Conversations in which questions are asked by the interviewer to requirements from the interviewee. There is huge probability of extracting mistakes and inconsistencies from requirements. Interviews can be of

a) Closed Interview: Predefined set of questions to be asked by requirement engineer.

b) Open Interview: Open conversation between requirement engineer and company representative.

Interviews result in very huge information from different stake holders. This information is some time hard to analyse due to difference in stake holder's views and inconsistencies

2) *Questionnaires*

A questionnaire is a research instrument consisting of a series of questions and other prompts for the purpose of gathering information from respondents [8]. They are of much importance because they are cheap and don't require much effort. Actually it is one time effort which involves setting of general questions relating to software product.

3) *Prototyping*

It is the technique of constructing a partial implementation of a system so that customers, users, or developers can learn more about a problem or a solution to that problem [7]. Prototypes may be Throw Away which is used to determine feasibility of system. Another type of prototype is evolutionary prototype which focuses on delivering working product to customer [7].

4) *Use Cases*

It has noticed that successful software products are the outcome of carefully planned requirements, which results from effective communication and coordination between different stake holders. Uses cases in software development maps different scenarios about the usage of software product.

5) *Brainstorming*

It is very creative activity which mostly results in solutions of problem. In brainstorming we generate ideas then there is discussion on these ideas which may or may not work [7]. All these results in better understanding of problem.

6) *Observation and Social Analysis*

This is an interactive kind of technique in which requirement engineer notices user working on manual system and take notes. It may be direct involving requirement engineer presence or some indirect mean involving video conferencing.

B. Requirements Management

It is name of process of managing changes to the requirements for a system. Requirements cannot be managed effectively without requirements traceability. It is concern with change management and version control.Requirement management is a corner stone for Project success [9]. With proper requirements management and definition, organizations can reduce project overruns by as much as 20 percent by limiting the number of inaccurate, incomplete, and omitted requirements [9].

C. Requirements Validation

Requirement validation ensures that whether the view of different stake holders do not contradict each other. This is mostly carried out by reviews and testing. Requirement validation extracts problem in requirement document which must be corrected for successful execution of software development life cycle.

D. Requirements Analysis and Negotiation

Very important activity in elicitation done by gaining of an insight of the relationships among various requirements and to achieve a successful result through applying this [10]. The techniques which are mainly used for analysis are:

1) *Requirement Prioritization*

When having many alternative situations, decision making becomes much more complex. To avoid this complexity is to prioritize things. Same is the case with software requirements. Customer demands functionality which is least risky, least costly [11]. There is need for us to develop the functionality which is most desired, least risky and least costly for the customer. This process is mostly applied in the case of tight deadlines. Quality is determined from the ability to satisfy customer from functionality pint of view [11].

2) *Modelling*

Models act as a bridge between transformations from analysis to design phase. Models can be of static and dynamic type. Static models are use cases, class diagrams, object models, package models, etc. Dynamic models are interaction diagram, activity diagrams etc [12].

It is name of process of managing changes to the requirements for a system. Requirements cannot be managed effectively without requirements traceability. It is concern with change management and version control.Requirement management is a corner stone for Project success [9]. With proper requirements management and definition, organizations can reduce project overruns by as much as 20 percent by limiting the number of inaccurate, incomplete, and omitted requirements [9].

E. Requirements Validation

Requirement validation ensures that whether the view of different stake holders do not contradict each other. This is mostly carried out by reviews and testing. Requirement validation extracts problem in requirement document which must be corrected for successful execution of software development life cycle.

F. Requirements Analysis and Negotiation

Very important activity in elicitation done by gaining of an insight of the relationships among various requirements and to achieve a successful result through applying this [10]. The techniques which are mainly used for analysis are:

1) *Requirement Prioritization*

When having many alternative situations, decision making becomes much more complex. To avoid this complexity is to prioritize things. Same is the case with software requirements. Customer demands functionality which is least risky, least costly [11]. There is need for us to develop the functionality which is most desired, least risky and least costly for the customer. This process is mostly applied in the case of tight deadlines. Quality is determined from the ability to satisfy customer from functionality pint of view [11].

2) *Modelling*

Models act as a bridge between transformations from analysis to design phase. Models can be of static and dynamic type. Static models are use cases, class diagrams, object models, package models, etc. Dynamic models are interaction diagram, activity diagrams etc [12].

III. AGILE DEVELOPMENT TECHNIQUES

Agile is an evolution in software development techniques. As the name implies it is characterized by Swiftness, nimbleness, and ease of movement. Agile Modelling (AM) is a practice-based methodology for effective modelling and documentation of software-based systems [13]. Agile is carried out by following practice like Daily Stand Up Meetings, Continuous Integration, Burn Down Tracking, Code Refactoring, retrospectives [14]. Agile is most suitable to accommodate change in requirements. Team's members in Agile are all rounders and have ability to work on tight schedule. In agile there is quick response to customers which results in early realization of future software product.

Following section briefly describes some important agile methods:

A. Scrum

It is project management approach carried out by incremental and iterative techniques. The name "Scrum" initiated by the scrum in rugby which is a way to restart the game after an accidental violation or change. It works on the principle of team working together to ensure quality in changing requirements [15]. Artefacts supported by scrum are Product Backlog, Sprint Backlog, Product Burn down Chart by Day, Product Burn down Chart by Sprint, Sprint Burn down Chart, Sprint Task Board View, Impediments. Scrum is mainly supported by three fundamental roles, which are Product Owner, Scrum Master, and team member.

B. eXtreme Programming

New but contentious approach towards software development. In this technique requirements take the form of user stories. Then the analysis of each story is carried out to estimate for the cost and duration of specific functionality. Then next story is listened for next build. Then each build is divided into tasks. Pair programming is employed in extreme Programming in which single computer is shared by two developers working on task. There is continuous presence of client with developers in large room to make development effective and to the point. The compulsion in extreme programming is that no team is allowed to work on their task more than two weeks [6]. It is very effective in scenarios where overall scope is limited.

C. Dynamic System Development Method

Kind of rapid application development methods which are organized and has primary focus on delivering product efficiently. It is best suited in situation where requirement time is fixed. It is UK's de-facto standard for RAD. Dynamic System Development Method is useful because its results of development are direct and visible. It offers the ability to divert ongoing project direction. This method indicates early about the feasibility of system to be developed [16]. Basic functionality is delivered quickly, with more functionality being delivered at regular intervals [16]. It has less intrusion from bureaucracy and breaks down the communication barrier between interested parties. User is continuously involved in development, thus developer gets feedback on the usability and appropriateness of the product. DSDM revolves around "test as you go" [17].

D. The Crystal Family

As name indicates that these are the mix of techniques to be applied on each project because every project is different has its own needs. In this technique every member can be used in change environments. It is light weight and adoptable approach to software development. The members of family are Crystal Clear, Crystal Yellow, Crystal Orange all has effect on a project due to several factors like tem size, project priorities etc [18].

E. Adaptive Software Development

This technique focuses large complex projects. It has short iterations. It incorporates the feature of just in time development in which functionalities are added when there is need of them. Also there is constant incrementation, iteration in development with constant prototyping [19]. In ASD there is review at the end of every iteration. This is not a defined model instead it should be tailored to specific domain. Main phases of ASD are the product development cycle, the iteration cycle, the daily development cycle. Is ASD main responsibility on developer for developing the product are involved for as much of the process as possible, since their decisions and steps make the product successful.

IV. REQUIREMENT ENGINEERING METHODS FOR AGILE METHODS

Typical approaches of requirement engineering employs only subset of development team on requirement gathering while in agile every member of team is involve in this activity. These approaches works well in small scale project but lose their significance in large and complex projects.

Now it is time to see how to combine two techniques:

A. Customer Interaction

Customer is a supreme stakeholder who is involve in software development process. Inconsistencies about requirements between customers and development are very complex topic [20]. To remove these inconsistencies agile method prefers on a representative of customer which is well aware of needs of customers, specially representative must have known everything about the domain in which product is going to be implemented, must have some authority to take important decisions at time. Agile method must ensure every time availability of customer to ensure smooth ongoing of project.

Agile methods may incorporate fast review to remove inconsistencies among requirements.

B. *Prioritization of Requirements*

Time taken by each functionality to complete and time to deliver it to customer is very important from project point of view. This importance makes division of complex requirements to simple tasks to complete in terms of time and business priorities from customer point of view. The risk with each functionality is identified. Prioritization is carried out through all development process and there is frequent prioritization.

C. *Non- Functional Requirements*

Kind of requirements which specify attributes and quality of future product. External constraints, restrictions on product and development process are targets of non functional requirements. They are of most importance in selection of tools of system to be developed. Examples are performance, reliability, security, usability etc.

Non functional requirements has two type of constraints:

i) Internal Constraint

Implemented by rule, constraints which specifies a limit not to cross during software construction.

ii) External Constraint

Implemented by restriction, constraint which specifies a limit not to cross during software execution.

In agile methods internal quality can be assured by: i) Pair programming in which two teammates work together, and ii) Peer reviews

D. *Documentation*

The major element in requirement engineering which separates it from agile methods. Complete and effective documentation in agile methods seem inapplicable. It is all up to the developer to make documentation. Less documentation causes problem when new developer comes, due to tight schedule in agile, conversation with other team members will result in slow pace in agile. That's why concerns and questions of new comer can be easily addressed with documentation. There is no chance of loosing of information in case of absence of any member.

Agile team must be accompanied by one technical writer from the start to ensure smooth documentation.

E. *Evolving Requirements*

It is implicit in agile that in starting phase it is almost impossible to extract all requirements. Other fact which is also implicit in that requirements change with the passage of time following customer's mind to in relation to changing environment or any technicalities. Agile is well aware of these facts always supports change even late in development. Cost of change is variable depending upon the phase. Evolution can be managed by consistent conversation with customer and prioritization, avoiding coupling and promoting cohesion.

F. *Prototyping*

In requirement engineering two type of prototyping are used. Throw away and Evolutionary prototype. Throw away is generally temporary used to elicit difficult functionalities, determining the feasibility of system, uncover missing requirements, while evolutionary prototypes is for delivering workable system to customers. First version implements the requirements which are best understood and which are important from end user functionalities. As agile is also composed of much iteration, evolutionary nature of evolutionary prototype can be used in agile. Overall process repeats until the prototype system satisfies all needs and has thus become desired system.

V. CONCLUSION

Agile methodologies are new and advanced topic. Now the developers have no choice left but to adopt agile as their development technique. Customer interaction in development process is main concern in both approaches. Each has its own pros and cons. Agile is best in small projects to manage requirements. It has overlapping in many phases to distinguish one from each other. In contrast to requirement engineering, which is governed by many phase. Agile has least documentation and heavily relied on developer and requirement engineering has documentation of each and every step.

REFERENCES

[1] Frauke Paetsch, Dr. Armin Eberlein, Dr. Frank Maurer "Requirements Engineering and Agile Software Development" in Proceedings of the Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03).

[2] Xuesong (Sonya) Zhang, C. Zhu and Bradley Dorn, "Accelerating Software Development through Agile Practices - A Case Study of a Small-scale, Time-intensive Web Development Project at a College-level IT Competition," Journal of Information Technology Education: Volume 11, 2012 Innovations in Practice.

[3] Bernd Waldmann, "There's never enough time Doing requirements under resource constraints, and what Requirements Engineering can learn from agile development" in 2011 IEEE 19th International Requirements Engineering Conference.

[4] Peter Maher, "Weaving Agile Software Development Techniques into a Traditional Computer Science Curriculum," in 2009 Sixth International Conference on Information Technology: New Generations.

[5] Juha Savolainen, Juha Kuusela and Asko Vilavaara, "Transition to Agile Development Rediscovery of Important Requirements Engineering Practices," in Proc. 2010 18th IEEE International Requirements Engineering Conference.

[6] Kent BeckExtreme Programming explained, Addison-Wesley, 1999.

[7] Gerald Kotonya and Ian Sommerville: Requirements Engineering, John Wiley & Sons, 1997.

[8] (2010) The WikiPedia website. [Online]. Available: http://en.wikipedia.org/wiki/Questionnaire

[9] The IBM website. [Online]. Available: http:// www-142.ibm.com/software/products/us/en/category/SW740

[10] Software Requirements: Objects, Functions, and States by A. Davis, PH, 1993

[11] Patrik Berander & Anneliese Andrews, Engineering and Managing Software Requirements, Aybuke Aurum & Claes Wohlin (Eds.), Springer 2005

[12] Grady Booch, James Rumbaugh, Ivar Jacobson, Unified Modeling Language User Guide. Publisher: Addison Wesley First Edition October 20, 1998 ISBN: 0-201-57168-4.

[13] The Free Dictionary Website. [Online]. Available: http://www.thefreedictionary.com/agile

[14] (2009) The AmbySoft Website.[Online]. Available: http://www.ambysoft.com/surveys/practices2009.html

[15] Ken Schwaber, Mike Beedle: Agile Software Development with Scrum, Prentice Hall, 2001.

[16] Marc Clifton, J. Dunlap. (2009) [Online]. Available: http://www.codeproject.com/Articles/5097/What-Is-DSDM

[17] Jennifer Stapleton: DSDM - Dynamic System Development Method, Addison-Wesley, 1995.

[18] Alistair Cockburn: Agile Software Development, Addison-Wesley, 2002.

[19] James A. Highsmith III: Adaptive Software Development, Dorset House Publishing, 1996.

[20] Bailey P, Ashworth N, Wallace N (2002) Challenges for stakeholders in adopting XP. In: Proceedings of 3rd International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP2002), Alghero, Italy, 26-29 May