Sindh Univ. Res. Jour. (Sci. Ser.) Vol.43 (2) 133-138 (2011)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

An Inside View into the Role of Customer in Agile Methods

Z. HUSSAIN, M. A. SOOMRO, J. A. MAHAR*, A. KEERIO** AND Y. A. MALKANI**

Quaid-e-Awam University of Engineering, Science and Technology, Pakistan

Corresponding author: Z. Hussain, email: zhussain@quest.edu.pk Cell. 92-3412802830

Received 28rth June 2011and Revised 16th August 2011)

Abstract. The successful software product demands various factors to play their part. One of the important factors is the successful role played by the stakeholders of the software domain. Extreme programming (XP) is one of the famous agile software development methods which advocates for the role of onsite customer who actively participates in team and remains onsite with a team physically. This paper presents the survey study of the role of onsite customer in XP and critically analyzes how this role is being carried-out practically in agile methods, particularly in XP. The data was collected from XP series conference from XP 2005 till XP 2010 as well as agile series conference from agile 2005 till 2010. The result shows that the role of the customer is being played both as onsite as well as offsite with team. However, few teams have no customer role at all. The communication channels used mostly between the customer and the team are: face-to-face, email, video conference, phone call, wiki, instant messaging, requirement document, online conference, presentation, dictionary, and story cards.

Keywords: Agile Software Development, Extreme Programming, Role of Customer.

1. INTRODUCTION

Software development is an art. The successful software product demands various factors to play their part. One of the important factors is the successful role played by the stakeholders of the software domain. Extreme programming (XP) is the famous agile software development methodology which advocates for the role of onsite customer who actively participates in team and remains onsite with a team physically. Agile software development is a key term mostly used for incremental software development methodologies. Agile software development includes many methods, each agile method particularly XP methodology requires quick feedback from customer for the functionality which is recently integrated by the team of programmers (Andrzeevski, 2007).

Research Questions:

In this paper we review the literature of agile methods particularly XP, a well-known software development method, for studying the role of customer. We have conducted a mini systematic literature review, which is a secondary study that evaluates and finds out the results from already conducted research regarding to a particular research question or interested phenomena (Kitchenham and Charters, 2007). The main research question is "How the role of the customer is being carried-out in agile software development, particularly in XP". This research question is further divided into following three sub-questions:

R Q #1a: Whether the customer is co-located in the team or not?

R Q # 1b: Which approach has been suggested in related research in order to overcome the problems of offsite customer?

R Q # 1c: Which communication channels have been used between the customer and the team members?

For this purpose, the data was collected from XP and agile series conferences from 2005 to 2010 whose research papers are available in Springer (springerlink.com) IEEEX plore (ieeexplore.ieee.org) digital libraries, respectively. Both XP and agile series conferences were selected as the main source of data gathering as both conferences are the leading conferences on agile methods, are held every year with agile professionals gathered from around the world. The keyword for the search criteria, word "customer".

This paper is organized into following sections. Section 2 briefly presents the background of agile and extreme programming. Section 3 presents the role of customer in XP. Section 4 presents the results and finally section 5 concludes the paper.

2. MATERIAL AND METHODS

2.1 Agile methods

Agile software development is a key term mostly used for incremental software development

ftware development includes

* Faculty of Engineering, Science and Technology, Hamdard University, Karachi

** University of Sindh, Jamshoro, Pakistan

many methodologies. Each agile method particularly XP methodology requires quick feedback from customer for the functionality which is recently integrated by team of programmers Andrzeevski, (2007). All agile methods set the values, principles and practices according to the rules set in agile manifesto Beck, (2001). The agile values as defined in agile manifesto are:

"Individuals and interactions over processes and tools

- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan That is, while there is a value in the items on the right, we value the items on the left more. Beck, (2001)
- The principles are:
- "Over highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile process harness change for the customer's competitive advantage.
- Delivery working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developer must working 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 conversion.
- 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.
- Continuous intention to technical excellence and good design chances agility.
- Simplicity --the art of the maximizing the amount of work not done –is essential.
- The best architectures, requirements, and design emerge from self-organizing teams.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly. Beck, (2001)

Being light weight agile methods are the best response against the heavy weight software development methodologies. A few years ago many software development companies working on traditional software, do not deliver software incrementally to the customer, and do not perform unit test and there was no concept of continuous integration. Agile software development solves such problems efficiently. All agile methods always try to release the first version of the software with minimum required features, prioritized by the customer within a limited time frame called release cycles, which is normally from one month to three months. After releasing the first version of the program a team of developers start work on the user stories of second release. Each release cycle is divided into iteration cycles which are normally one to two weeks long. All tasks performed by programmers in each iteration are tested before integrating with main development track. In agile methods the tasks are broken down into small increments with minimal planning. A task done by the programmer in each iteration should be passed through the design, coding and testing phase before integrating to the main program and this process is continuously done up to the last integration of user story (user required features). Each task is helpful to adapt changes quickly and also minimizes the failure rate of software. Agile methods also welcome change. One of the major causes of software project failure is to take more time to develop system Sohaib, (2010) Micheal *et. al.,* (2001)

Team collaboration is very important in agile methods; normally a team consists of 5 to 10 persons, all in single room, are responsible to perform tasks or functionality as an individual or with other developers according to the requirements of the onsite customer who is usually appointed by stakeholders. A customer shares the ideas and tasks are written on the index card usually known as a user story. A user story is a narrative that the customer describes some kind of problems or requirements; a user story has the priority and estimation. A priority is always set by a customer and the estimation is set by programmers, the priority of the user story can be changed, the programmers take those tasks first having the highest priority and most valuable stories are picked by the programmer to work for the next few weeks. A team of developers may be in single room or in different geographical locations, they communicate with each other through email, mobile phone, instant messages, documents or sometime, if needed, they communicate through video conferencing.

Some of the agile methods commonly share the same practices and values, where as in the implementation point of view many methods have different terminologies and techniques. The famous agile methods are XP, Scrum, Crystal, Dynamic Systems Development Method (DSDM), Feature Driven Development (FDD) and Lean.

2.2 Extreme Programming (XP)

XP (Extreme Programming) is most widely used methodology of these days. During the last few years XP becomes more popular in software

development community due to its simplicity, iterative and incremental approach, and for various engineering practices. XP always provides disciplined approach to develop high quality software quickly and continuously. It promotes high customer involvement, rapid feedback loop, continuous testing, and close team work to deliver working software at frequent intervals Micheal *et. al.,* (2001). The term XP was coined by Kent Beck in late 90's. The XP is relatively new and most widely used agile methodology of these days. Mostly, the companies selecting XP as a software development is due to its simplicity, light weight, iterative and incremental approach. Beck, (2000).

XP Values and Practices: The Kent Beck published 4 values and 12 primary practices in his first book Beck, (2000).and in its 2nd edition of the same book he explained 5 values and few more corollary practices as well as few principles. Values are Beck, (2000) Stapel, *et al.,* (2008).

Communication: Face to face communication is preferred among software developers and stake holders, they also communicate through other channels and some sort of documents. Through this value they share the ideas and get the knowledge from their experience.

Simplicity: XP always focuses on the simplicity, a simplest code and design may integrate with software easily and XP also tries to strictly follow KISS (keep it simple stupid) and YAGNI (you aren't gonna need it) principles.

Feedback: A positive feedback from customer and also from other stakeholders including end-users help developers to work on customer described user stories promptly.

Courage: A developers should have courage and feel comfortable with refactoring their codes.

Respect: The team of developers work in a single room they are all equally respectable, XP always advocates self respect and the respect for all other team members.

The few XP practices are Beck, (2000).

Pair programming: Two programmers work on one task, one is responsible to write a code while the 2nd programmer reviews the code.

Planning game: A team of developers with the customer carries-out detailed planning about releases and iterations, they arrange meetings weekly or once per iterations.

Test driven development: unit tests are conducted by the programmers that test the functionality of the code. In XP methodology, a test will be written first before writing the actual code.

Whole team: a team consists of 8 to 10 members, customer who is always be with the developers for questions and answers and is supposed to be the part of whole team.

Continuous integration: The developers are working on the user stories; they include the required minimum features in the program. After the successful test, the latest version of the software is integrated to main software.

Design improvement: XP always emphasizes to include those required minimum features which are needed today with as simple as possible design and continuously advocates for improving the design.

Small releases: As the XP is incremental approach; software is delivered to the customer in small frequent releases. Through the small releases customer gains the confidence in the progress of the software.

Coding standard: a code which is written by developers should follow the set of rules, format and appearance throughout the designing of the software.

Collective code ownership: All the developers are the owner of the code, and every programmer is allowed to change any part of the main program. XP does not advocate owning code individually.

Simple design: A simple code and simple design is always be appreciated by XP methodology. Developers may adapt a simplest approach during the design of software.

Metaphor: Proper names are assigned to classes and objects to understand the functionality of classes and methods.

Sustainable pace: XP does not allow any developer to work more than 8 hrs and 40 hrs/week. Overtime may be done by the developers not more than once in a week; a tired programmer may attempt many mistakes in code.

Few corollary practices are:

Real customer involvement: customer who is available with the developers throughout the designing of the software, is fully involved with the developers to develop, discuss and clarify the user stories.

Incremental development: A required minimum functionality is integrated with main program on incremental basis. A huge, complex program may not be delivered to customer.

Team continuity: A small effective developers' team is kept together up to the completion of project.

This methodology maintains software quality and minimizes the failure rates of projects due to frequent releases. The first version of the software product is released in three to four weeks with minimum functionality as prioritized by the customer.

One of the XP practice 40 hours week, recommends that a good programmer may not work more than 8 hours a day and 5 days a week as developers sit together in same room they increase

their knowledge rapidly by sharing their experience. The code or piece of program is written by the programmer in single or in a pair with other programmer. In pair programming, two programmers work on the same task, and the tasks are completed within estimated time. One programmer may be more experienced to the other; due to this pair programming the chances of errors are minimized. One programmer is busy in writing the code and other programmer is responsible to analyze the code and check the errors. Huge and lengthy code is discouraged and a complexity in a program is also avoided. XP encourages to include a required features with minimum number of lines of code and insists its rule of simplicity. The unit test of the code is important when a code is integrated to the main code. A unit test is conducted by a programmer before the integration. Technically, programmer writes the test first once the test is successful then the code is written by the programmer. Initially it is so difficult for a new programmer to write test first but with experience a programmer is used to write the test first before writing the actual code. After the integration of each code with the main program, a new system containing latest integrated feature is ready for the production. The acceptance test is also important in the XP methodology. A final version of a program is released to the customer. After the delivery of software to the customer, an acceptance test is conducted by the end users and in some times by the customer. All the features required by end users are critically analyzed; if any problem or bug is found during the test of program, end users may report. Sometimes a GUI based test may be conducted to check the features of program graphically (usability test). In case an acceptance test fails and program does not satisfy the requirement of the business organization, such programs are sent back to the programmers for fixing the errors in the coming iterations.

3. The Role of the Customer in XP

In any traditional software development there is a great involvement of end user Salihoglu, (2008) during the designing and running of software. An end user is a person who is responsible to use and run software. XP is one of the most widely used software development methodology of agile software development which advocates full time onsite customer, who can be an end user, for improving the quality of software Mohammadi *et al.,* (2008). Onsite customer is well known practice of XP, this practice successfully improve the efficiency of software. Full time customer should participate as a team member with the other developers within a single room or in a separate office within a same building. A customer is appointed by stakeholders and he should be committed, cooperative, knowledgeable and expert of his domain and have a direct interest in software development. A good well-behaved onsite customer is responsible to make good decisions about the software which fulfills the business requirement now and also in near future. An onsite customer is a member of team, is responsible to share the basic requirements of the business value with the programmers, A customer is authorized to change the requirements time to time. With the help of onsite customer practice a number of errors related to business requirement can be reduced, with this practice 60% rework was reduced in William *et al.,* (2007). Onsite customer is a non-developer team member, has a implicit and explicit responsibilities Martin, *et al.,* (2004) and has to drive the software. A customer has a close contact with software developers, stakeholders, clients, users and fund providing agencies. He is responsible about planning of iteration and releases the software, he is the integral part of the team and remains onsite on each release, (s)he is an authorized person who is responsible for all business decisions, (s)he write user stories usually on index card and prioritizes such stories, (s)he has a full rights to change the priority or requirement of user story at any time. Customer and team of programmers discuss on these user stories time to time . Programmers always oblige the customer and take critical and highest priority user story first, to work on minimum required feature. After completing the tasks, customer conducts the test and verifies that these tests run correctly Deursen, (2001) Mikko *et al.,* (2009). Customer also maintains the contact with the organization and informs them about the progress of software and justifies the time taken by software team Deursen, (2001) Mikko *et al.,* (2009).

4. RESULTS AND DISCUSSION

All the papers of agile and XP series conferences from 2005 to 2010 were downloaded from IEEEXplore and Springer digital libraries, respectively. Total 620 papers were downloaded. The search criteria used the word “customer” in the mentioned digital libraries. The word “customer” was found in 443 papers. Fig (a) shows the year wise relationship between XP and agile series conferences having word customer.

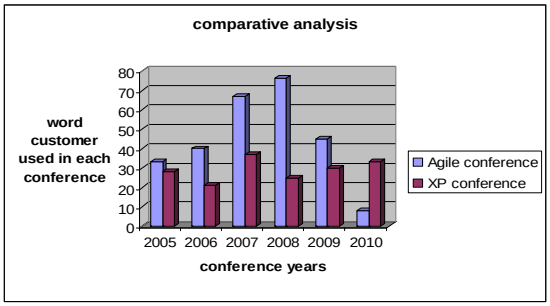

Fig (a):comparative analysis between agile and XP conference Fig (b) and fig(c) show total number of papers

having word customer from agile and XP series conference respectively.

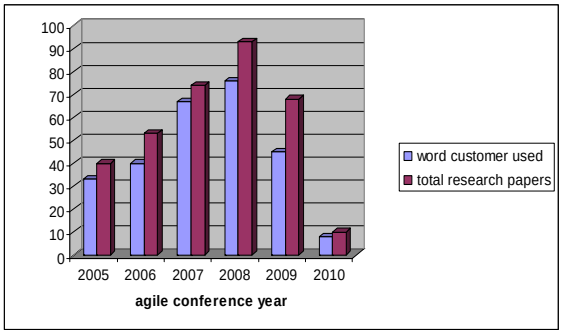

Fig (b) agile conference series from 2005 to 2010

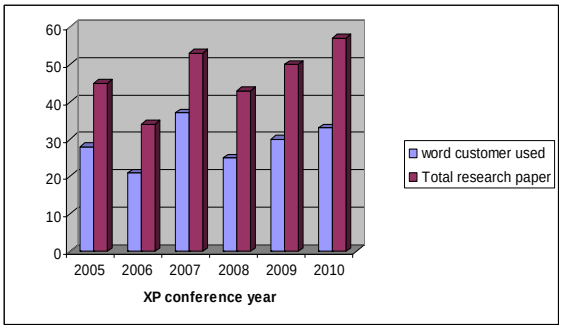

Fig (c): XP conference series from 2005 to 2010

Among these 443 papers the search was further limited to the title and abstract of the papers containing the word customer. Out of 443, only 105 papers contained the word customer in their titles and abstracts. These 105 papers were finally selected and read word by word to find the answers of above mentioned research questions. Among these 105 papers the terms "product owner" and "product manager" were also included/ considered as the equivalent of the customer role.

R Q#1 (a):Whether the customer is co-located in the team or not?

According to the results, in 27 papers, the customer is co-located with the team members, which is over 25% of the finally selected papers.

R Q#1 (b): Which approach has been suggested in related research in order to overcome the problems of offsite customer?

In 78 projects/papers the customer is offsite or there is no customer at all. Korkala et al. found out that "an increased reliance on less informative communication channels results in higher defect rates" Mikko (2006). Less informative communication channels may lead to inaccurate, incomplete or misunderstood requirements. They insist on face-to-face communication and the feedback between the customer and the team members. If the customer is offsite, remotely located, then videoconferencing is the next best communication channel after face-to-face, which "produces better decisions, a wider range of options, greater analysis, and it creates the psychological distance needed to engage in a more open exchange of options" Mikko (2006). Telephone is of good usage followed by email which is effective for clearly-understood issues. However the communication channels should be decided at the initial stage of a project, in collaboration with all stakeholders including offsite customer Mikko (2006). These findings are also supported by Cockburn (2002) and Mikko, *et al.*, (2009) where also use wiki. Broza (2005) stresses the need to use effectively the information radiators such as release plan, sticky notes, charts, to-do list besides their online versions. He also recommends establishing an online learning repository as well as a common vocabulary for the understanding of the process and product terms among team members and other stakeholders Broza (2005).

R Q #1(c): Which communication channels have been used between the customer and the team members?

Mostly the communication channels have been used are: face-to-face, video conference, phone call, email, wiki Deursen, (2001) meeting, Instant messaging, requirement document Olly and Leip, (2007), online conference, Twiki Chubov and Droujkov (2007), presentation Broza (2005) translators (a person who is a member of a team is responsible to facilitate communication between team and customer through their languages), dictionary (an online editable document used by customer to define business terms, their meaning and their context) and story cards Honda, *et al.*, (2010). Acceptance tests were also used as a communication tool between the customer and other stakeholders Nyman *et al.*, (2010).

5. CONCLUSION

This paper presented the results of a mini systematic literature review with some implications. This review is not a full-fledged literature review rather a mini review as we had only searched two digital libraries albeit very important and most relevant to agile methods; 105 papers in total were read. Another point is that we mostly focused on the word customer who covers the XP agile method but a future work would enhance this research with focus on "product owner" as used in Scrum (another important agile

method) and other terms used for the role of customer in other agile methods.

A customer with one voice for the business requirements, and always present with the team all the time for requirement elicitation and clarification is indeed a positive factor for building a successful software product. The results show that agile methods, particularly XP emphasizes on a dedicated customer being part of a team. However, as the results revealed that mostly the role of the customer is being performed both as onsite as well as offsite with the teams with the exception of few teams having no customer role at all. With the onsite customer present, a team has the advantage of face to face communication with her/him for quick clarification for any requirement feature to be implemented quickly. If the customer is offsite then the teams use email, video conference, phone call, wiki, instant messaging, shared online requirement document, online conference, presentation, online dictionary, and shared online story cards for the communication purpose to develop a better and a successful software product.

REFERENCES:

Andrzeevski S., (2007) Experience Report 'offshore XP for PDA development, in proceeding of agile development conference, IEEE, 376-381.

Beck K. (2000) Extreme programming Explained, Addison-wisley.

Beck K. (2001) agile manifesto, agilemanifesto organisation, USA.

Broza, G. (2005) Early Community Building: A Critical Success Factor for XP Projects, in proceeding of agile development conference, IEEE 2005, 167-172.

Cockburn, A. (2002) Agile Software Development, Addison-Wesely, Indianapolis.

Chubov, I. and D. Droujkov (2007) User Stories and Acceptance Test as Negotiation Tools in Offshore Software Development, in lecture notes in computer science (LNCS), Springer, vol (**4536**): 167-168.

Deursen, A.V. (2001) Customer Involvement in Extreme programming, XP2001 workshop report, XP 2001 Workshop on Customer Involvement, ACM software engineering, 70-73.

Gotel, O., and D. Leip, (2007) Agile Software Development Meets Corporate Deployment Procedures: Stretching the Agile Envelope, in lecture notes in computer science (LNCS), Springer, vol. (**4536**): 24-27.

Honda, R., J. Noble, and S. Marshall, (2010) What Language Does Agile Speak, in lecture notes on business information processing (LNBIP), Springer, XP2010, vol. (**48**): 387-388.

Kitchenham B. and S. Charters, (2007) Guidelines for performing Systematic Literature Reviews in Software Engineering, Keele University and Durham University Joint Report EBSE, 2007.

Micheal K., J. Prashant C. Angelo and L. David (2001) Distributed eXtreme programming, second international conference on extreme programming and agile processes in software engineering, 66-71.

Mohammadi S., B. Nikkhahan and S. Sohrabi (2008) An analytical survey of "onsite customer" Practice in Extreme Programming, in proceeding of international symposium on computer sci. and its application, 1-6.

Martin A., R. Biddle, and J. Noble (2004) The XP customer role in practice: Three studies, in proceeding of agile development conference, IEEE 2004, 42-54.

Mikko K., M. Pikkarainen, and K. Conboy, (2009) Distributed Agile Development: A Case Study of Customer Communication Challenges, in lecture notes on business information processing (LNBIP), Springer, XP2009, 161-167.

Mikko K. (2006) A case study on the impact of customer communication on defects in agile software development, in Agile 2006 conference, IEEE.

Nyman R. (2010) Automated acceptance testing of high capacity network gateway", in XP 2010, LNBIP vol. (**48**): 307-314.

Sohaib O. and K. Khan (2010) Integrating usability engineering and Agile Software Development, In proceeding of international conference on Computer Design and Application, IEEE 2010, vol. (**2**): 32-38.

Stapel, K., D. Lubke, and E. Knauss (2008) Best Practices in extreme programming course design, In proceeding of international conference on software engineering, IEEE, 769-776 .

Salihoglu Y. (2008) User Involvement in software project development:A Review of Models, Software development research paper.

William M., J. Packlick R. Bellubbi and S. Cobun (2007) How we made onsite customer work-An Extreme success story, In proceeding of agile development conference, IEEE 2007, 334 -338.