Scenario Based Requirement Engineering (SBRE) in eXtreme Programming (XP) through Agile Modelling (AM)

Sundar Kunwar

Department of Computer Science and Engineering, Nepal Engineering College, Pokhara University Changunarayan, Bhaktapur, Nepal
sunfrens2002@gmail.com

Sundar Kunwar is working as an Assistant Professor in the Department of Computer Science and Engineering at Nepal Engineering College, where he has been since 2007. He has received BE in computer engineering from Tribhuvan University in 2006 and Master in Software Development from University of Tampere, Finland in 2013. His research interest is in improving the agile software development methodologies through agile modeling. He has made numerous contributions to robotics projects. He has successfully coordinated robotics competition organized by IOE held on 2067 BS on theme "Working together for better Nepal" and had stood first position among all the engineering colleges of Nepal.

Abstract

Software development methodologies have been enhancing significantly all the time and have drawn the attention of software professionals from past few years. The software development methodologies are expanding rigorously with wide range of diverse fields and have become more complex demanding the agility to rapidly changing needs of the customers. As a result, agile software development methodologies have evolved and gaining popularity day by day. Agile software development methodologies have come up with modifications demanded in traditional software development process to make them faster, more flexible, light weighted and productive. Extreme Programming (XP) is one of the well-known agile software development methodologies and is driven by a set of values including simplicity, communication, feedback and courage. Planning game, very short release stories and test first coding are some interesting extreme practices of XP; however it is criticized for some deficiencies like light weight requirement, onsite customer, pair programming and so on. The study has only focused on only one of the most criticized extreme practice-lightweight requirement and has followed agile modelling approach to make improvement on requirement engineering process in XP. Requirements are the user stories that consist of a few sentences (1-3 sentences) written on an index card which describes the functionality given by single onsite customer in XP. There is lack of analysis of stakeholders and their roles in requirement process. Therefore, it is very difficult to know the specific requirements of the specific stakeholder in XP. This means that the requirement engineering process is incomplete in XP. This may result high chances of providing deficient requirements. Agile Modelling (AM) is the chaordic practice based methodology for effective modelling. The interesting part of AM is that it does not tell how to model, but tells about how to be effective as modellers. A Scenario Based Requirement Engineering (SBRE) is proposed using AM as alternative to light weighted requirement which one of the most criticized extreme practice of XP. SBRE is the implemented description of techniques that helps to understand the task related activities and also facilitates communication among stakeholders and experts. Stakeholders are analysed for understanding a system by identifying the stakeholders in the system, and assessing their respective requirements and influence on the system. This will help to get better requirements for system development, but still needs to be validated.

*[**Keywords**—Agile, eXtreme Programming (XP), Agile Modelling (AM), pair programming, onsite customer, Scenario Based Requirement Engineering (SBRE)]*

I. Introduction

Software development methodologies are the frameworks that are used for structuring, planning and controlling the processes involved in software development. Waterfall model, V shaped model and Rational Unified Process (RUP) are the most popular traditional software development methodologies. These methodologies involve sequential series of steps that need to be planned and documented in detail before implementation and are very rigid to change that will be occurring in future. As a result, plan driven heavyweight traditional methodologies have transformed into agile software development methodologies to address the rapidly changing requirements of the market. Agile software development methodologies are developed with the ability to respond quickly with changing requirement [1]. Therefore, it is not simply the size of the process or the speed of delivery; it is about the flexibility of the process or methods [2]. Agile methodologies include modification in software development process to make them faster, more flexible, light weighted and productive. In the late 1990s, several software development methodologies drew the attention of the public and each method has a combination of old ideas, new ideas and transmuted old ideas [3]. The common among all these methodologies was that they all emphasized personal interaction over process, direct communication, short and frequent release, iterative process, self-organization and code crafting among others. Extreme Programming (XP), Scrum, Feature Driven Development (FDD), Crystal Methodologies Family (CMF) and Adaptive Software Development (ASD) are the most popular agile methodologies in use today. XP and Scrum are the most commonly used agile software development methodologies in software industry. One of the well-known methods of Agile is extreme programming (XP in short) and is driven by a set of values including simplicity, communication, feedback and courage and is characterized by a short development life cycle, incremental planning, continuous feedback and reliance on communication and evolutionary design and is characterized by some extreme practices such as planning game, small releases, metaphor, simple design, test driven development (TDD), refactoring, Pair Programming (PP), collective ownership, continuous integration, 40-hour week, onsite customer and coding standards [4]. XP was developed by Beck and Jeffries with interesting composition to improve the quality of software development process as well as to respond the changing need of customer [3]. XP is known to be a lightweight agile software development methodology with some extreme practices which are lightweight in nature but very difficult and sometimes unrealistic to implement and there are only a few analytical studies on XP. Most of the literature and books have been drafted by the inventors of the Agile Manifesto and are concerned with the promotion and commercialization of the agile methods and the services they provide. Figure 1 shows the general overviews of XP. Release planning is done with the help of system metaphor obtained architectural spikes and the requirement specifications obtained from user story. Release plan helps to carry out the iteration which in turn produces a piece of software. Small releases are released after the acceptance test approved by customer.

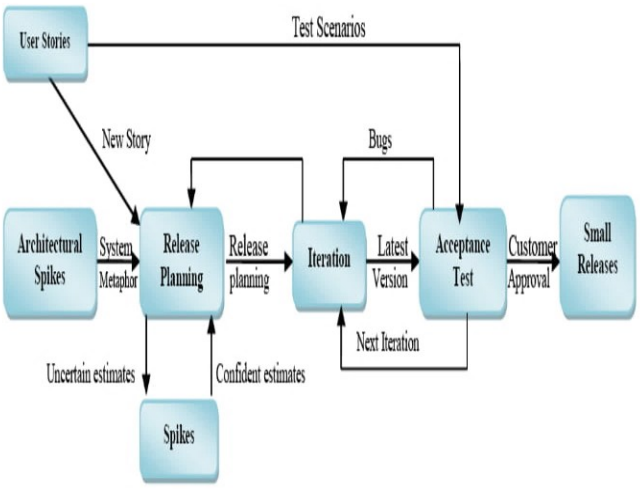

Fig 1. eXtreme Programming (XP) Release Cycle [5]

Extreme Programming is used as research framework to examine the chances of deficient requirements in XP. An interpretive approach is followed to conduct the literature review and this approach is concerned with the hermeneutic cycle derived from document and literary analysis. The hermeneutic cycle is used for the modelling of extreme programming regarding the most criticized practice-user story of XP. User story is criticised as lightweight

requirement with high chances of acquiring deficient requirement that propagates throughout the software development process. Interpretive approach was used to propose Scenario Based Requirement Engineering (SBRE) practice using agile modelling to address the chances of deficient requirements to make it realistic and practical. SBRE describes the implemented techniques that aid to understand and communicate the task related activities among stakeholders and experts.

II. Research Methodology

The aim of this study is to gain a thorough understanding of requirement process in extreme programming and to build the agile models of extreme practices addressing the pitfalls in requirement process that best suits for XP. The work is more concerned with the applicability of requirement engineering in XP and is considered XP requirement process as an initial research framework for explaining and evaluating various aspects of it. An interpretive approach was followed to conduct a literature review. A research can be interpretive if it builds on the assumptions that humans learn about the reality from the meanings they assign to social phenomena such as language, consciousness, shared experiences, publications, tools, and other artefacts [5]. The most fundamental principle of the interactive research approach is a hermeneutic cycle derived from documents and literary analysis. The different components of the hermeneutic cycle are illustrated in Figure 2. The first component of the hermeneutic cycle is concerned with the pre-understanding of researchers on the subject matter and the second component is concerned with the absorption of more knowledge from different sources to widen knowledge to expand the researcher's interpretation potential. The third component is concerned with theory building on the basis of an interpretation of knowledge, explanation attempts and missing knowledge. The last component is concerned with documenting the new theories and knowledge acquired through interpretive research approach [3]. The same approach of the hermeneutic cycle is used for agile modelling. The common purpose of modelling is to provide a basis for deeper understanding with experiments, predicting the behaviour of the system and saving the cost of actual case controlled experiments.

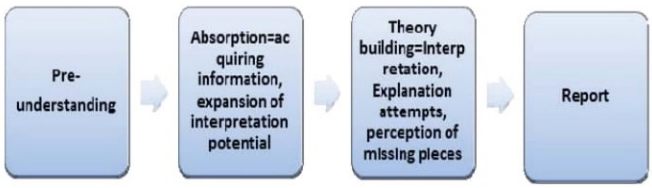

Fig 2. Hermeneutic Cycle [3]

III. Agile Modelling (AM)

Agile Modeling (AM) is the chaordic, practice based methodology for effective modeling and documenting software based systems [6]. It does not tell about how to build the model, but it tells about how to be effective as modelers. In other word it is not prescriptive process. It is a chaordic because it blends the chaos of simple modeling practices and blends it with the order inherent of software modeling artifacts.

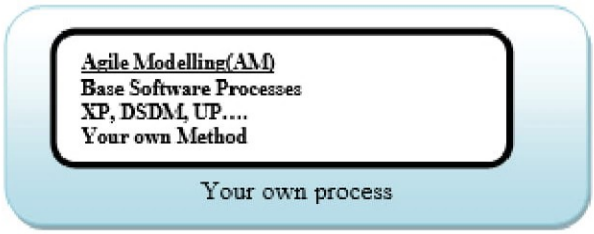

Fig. 3 Agile Modelling and Base Software Process [6]

Figure 3 shows the base software processes such as XP, Scrum, UP or your own personal process which can be tailored with AM. The best part of the AM is that it is possible to pick the best features from different existing software process and can be modelled it using AM to make your own process according to your need. AM is independent of other processes such as XP or UP, but it plays a significant role in enhancing those processes. Any person who follows the agile methodology applying the AM practices with its principle and values are agile modellers. An agile developer is who follows the agile approach to software development. Therefore, agile modellers are agile developers but not all the agile developers are agile modellers.

IV. Requirement Eegineering in XP

XP is a lightweight agile methodology with four core values: simplicity, communication, feedback and courage [4]. Although XP has many interesting practices such as planning game, very short releases and test first coding among others, it is not free of critics. One of the most criticized practices in XP is light weighted requirement. Requirement engineering is the process of specifying requirements by studying stakeholder needs and the process of analyzing and refining those specifications systematically [7]. Specifications are the concise and clear statements that serve as a requirement that the software should satisfy. Requirement engineering must include four activities: elicitation, modeling, validation, and verification to produce clear and faultless requirements. Unclear and deficient requirement is one of the biggest causes of software failure. The process to acquire requirements in XP is different from the traditional methodologies. In XP, requirements are the user stories that consist of a few sentences (1-3 sentences) written on an index card which describes the functionalities of the customers' values. It serves as the starting point for developers and customers generate more precise detail [10]. And then the developer decomposes the user story on a card into manageable chunks of tasks recording each task and its status on the card. As there is no analysis of stakeholders and their roles in requirement process, it is very difficult to know the specific requirements of the specific stakeholder.

Information about the requirements of the whole system by a single onsite customer may lead unclear and deficient requirements because single customer does not know all the requirements of the concerned stakeholders. A stakeholder is defined as any group or individual who can affect or is affected by the achievement of the organization's objectives. One of the best solutions to avoid unclear and deficient requirement is to collect use scenarios and perform stakeholder analysis. A use scenario is the implemented description of techniques that helps to understand the task related activities and also facilitates communication among stakeholders and experts. Stakeholder analysis is an approach for understanding a system by identifying the stakeholders in the system, and assessing their respective interests in, or influence on the system.

Another big problem in XP requirement is that the customer wishes high expectations exaggerating the computer capacity and proposes the more functionalities request and hope that the developers deliver the product in very short time. This usually happens if the customer is unknown about the new technology and available platforms for development. Another major problem in XP is paying less attention towards the changing requirements which leads to project stagnation, modification on finished work and even abandon the finished work [8].

Modifications to the XP requirements process are reported by many researches and studies. There are various solutions suggested by different studies. But, most of the suggestions are based on the comparative studies. Scenario Based Requirement Engineering (SBRE) practice is proposed in this study.

V. Addressing Light Weighted Requirement in XP through Agile Modelling

Why Agile Modeling approach was used in modeling XP? The answer is very simple; Agile Modeling is a part of XP. It uses many Agile Modeling techniques such as User story, Component Responsibility Collaborator (CRC) cards, models and sketches. There are mainly two primary purposes for using modeling approach. First is to understand and make others understand what is being built and what are the processes involved in it. Second is to analyze the requirement and present detail design of the system. This work is concerned with both of the primary purposes of using modeling approaches. Agile Modeling is used for clarifying the necessity and analyzing them in term of agile models. Agile Modeling approach is used for requirement modeling to make the process realistic and practical in real XP project.

Requirements play significant role to make any project successful. However, unclear and deficient requirements in software development often lead to disappointment with an unreliable product which may even results dangerous accidents. Therefore, with unclear and deficient requirements create more problems than they solve. One of the major determining factors to make the software development organization successful is how well they understand and manage their requirements. Requirement engineering is the process of developing requirements through an iterative cooperative process of analyzing the problems,

documenting the resulting observations in a variety of representation formats and checking the accuracy of the understanding gained [9].

To get clear and adequate requirements, collection of use scenarios is proposed in this study. Use scenarios can be defined as the implemented description of techniques that helps to understand the task related activities and also facilitates communication among stakeholders and experts. The effectiveness of using scenarios in several subjects can work as the capability of simulating thinking. In simple words, scenarios are the representation of the real world and can be generalized for requirement analysis to produce the required models which are familiar to requirement engineers or software engineers. Figure 4 shows how requirement specifications are related to real world scenarios and how real world scenarios can be used for designing rational models and concept prototypes which helps to ex-tract real requirements from the real world. The best ways of obtaining requirement specifications from usage scenarios are inspection and observation which helps in brainstorming to get the real requirement of the project.

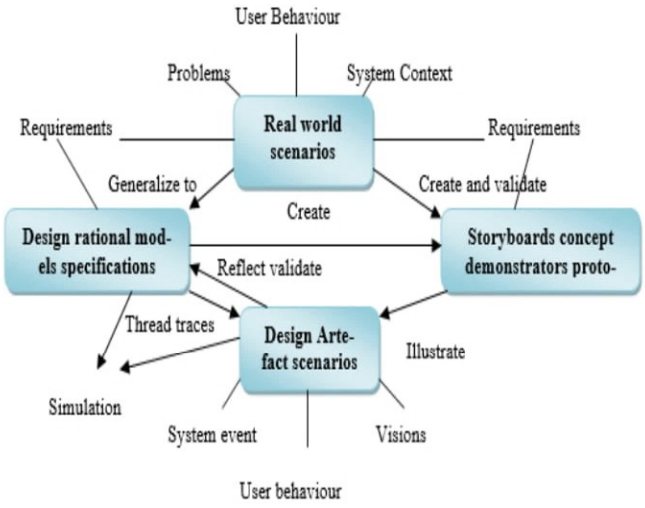

Fig. 4 Roles of scenarios and their relationship with requirements [10]

Real World Scenario is the real world of interest that is inspected or observed. Real worlds are always concerned with problems, behaviors and system context. Close inspection and observation helps in brainstorming to derive the real requirement specifications of the system. The real world scenarios can be generalized to rational models with generalized specifications derived from real world scenarios. Storyboard-Concept demonstrator prototype is a story or example of real world events or grounded theory abstracted from real world experience. Designed Artifact Scenarios is the final designed artifact scenarios derived from the real world scenarios. It is the use case collected from the real world scenario and can be represented in a variety of formats. It can be sequences of use case diagrams or list of use case requirement specifications. There are two methods in scenario based requirement engineering, and they are ScenIC method and SCRAM method [11]. Scenic Method was proposed by Colin Potts in 1999 and it consists of goals, objective, task, actor and obstacle [12]. Scenarios are made up of episode and actions. Man or machine can be actor and goals can one of the following-achieving states, maintaining states or avoiding states. Obstacles show the successful completion of tasks. In this method, every cycle involves in criticism and inspection of the scenarios that helps to further refine the requirement specifications. General guidelines are provided to format scenario narratives and to identify goals, actions and obstacles. Goals are achieved in episodes and episodes are evaluated with goals achieved. Goals are achieved with the help of system tasks which are carried out by actors. Dependencies are examined among goals, actors, tasks and resources to make sure that all the requirements of the system are met [11]. SCRAM stands for Scenario Based Requirement Analysis and this method does not explicitly provide modelling and specification. It works in parallel with software engineering methodology chosen by the practitioner. It is used for requirement elicitation with reasoning about the problem extracted from scenario about use context [10]. It is usually done after preliminary design.

After requirements are collected from scenario based requirement engineering process, the next step is to identify the stakeholders and perform analysis. There are many approaches to identify the stakeholders. From the viewpoint of software and requirement engineering, following are the most appropriate stakeholders staked with the software end product and software development processes Fig 5.

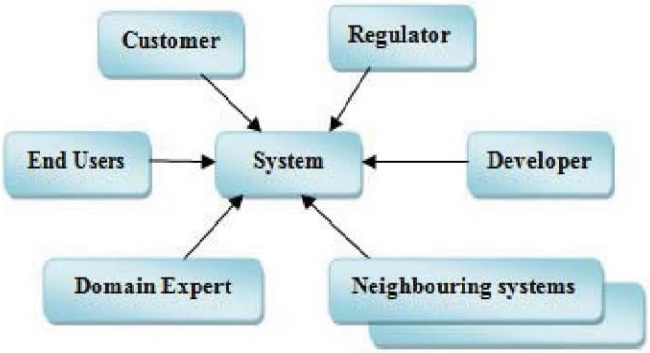

Fig. 5 Different types of stakeholders

Stakeholder analysis is a technique of understanding a system by identifying the stakeholders staked to the system and assessing their relationships, interests and expectation from the system or project. Following are the general steps of stakeholder analysis Fig. 6 [13].

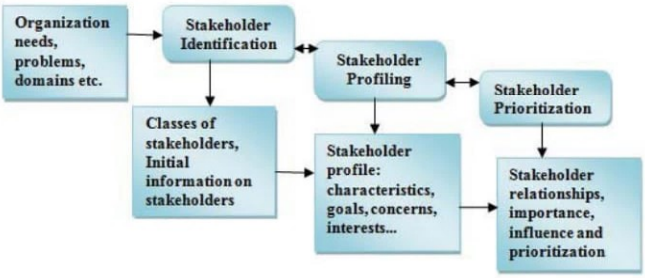

Fig. 6 Stakeholder analysis process [13]

All the above discussed changes are modelled in the release cycle of XP and are shown in Fig. 7. The requirement changes are carried out in three stages- collect user scenarios, stakeholder identification and analysis; and then only detail user e-story is prepared.

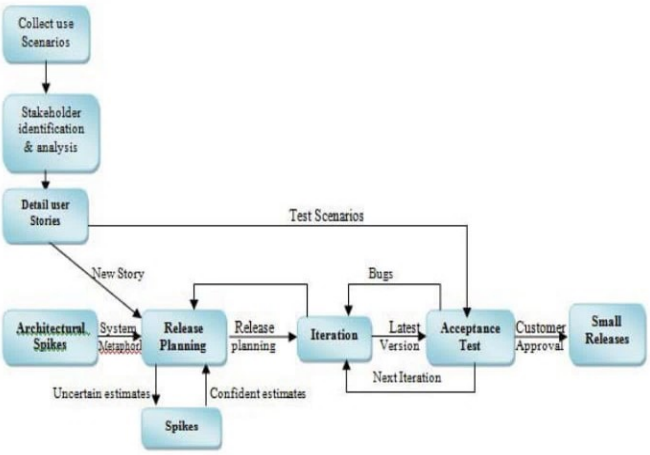

Fig. 7 Requirement model in release cycle

VI. Conclusions

Agile software development methodologies have been evolving continuously with the demanded improvements needed in traditional software development methodologies. Agile methodologies are characterized by personal interaction over process, direct communication, short and frequent release, iterative and incremental process, self-organization, code crafting and many more. Extreme Programming (XP) is one of the well-known agile software development methodologies with sets of values including simplicity, communication, feedback and courage. One of the most criticised lightweight processes introduced in XP is lightweight requirement (user story) which is taken as one of the extreme practice in XP. The extreme practices and composition variation has made the software development process more complex. Lightweight requirement is taken into consideration for the study and agile modelling for lightweight requirement was performed to overcome the pitfalls found during the study. The study concluded with requirements in XP can be improved with collecting use cases from scenario based requirement engineering practice followed with stakeholder analysis. Agile model needs to be validated which can be further studied in the future. The study concentrates on the most criticized practices-lightweight requirement of XP. In the future, further study about other extreme practices can be carried out to refine the practices and make them simple, practicable and applicable.

References

[1] Kuda Nageswara Rao, G Kavita Naidu and Praneeth Chakka, "A Study of the Agile Software Development Methods, Applicability and Implications in Industry", *International Journal of Software Engineering and Its Applications*, vol. 5, no. 2, p. 35 , 2011.

[2] P Kruchten, "Agility and Architecture: Can They Coexist?", *IEEE Software*, vol. 27, no. 2, pp. 16–22, 2010.

[3] J Kalermo and J Rissanen, "Agile Software Development in Theory and Practice", University of Jyväskylä, Finland, 2002.

[4] K Beck, "Embracing Change with Extreme Programming", *IEEE Computer*, vol. 32, no. 10, pp. 70-77, 1999

[5] E David, *Research Methods for Political Science: Quantitative and Qualitative Approaches*, 2nd ed. Reading, *ME Sharpe*, 2004, [E-book] Available: http://books.google.com/books?id=8PJYznDXQIcC&pgis=1.

[6] S Ambler, "Agile Modeling; Effective Practices for Extreme Programming and the Unified Process". *Internet:* http://www.scribd.com/doc/37142147/Agile-Modeling-Effective-Practices-forExtreme-Programming-and-the-Unified-Process#outer_page_99, 2013

[7] C Jones, "Software Estimation Rules of Thumb", *Proceedings of the 1998 IFPUG Conference,* vol. 5, no. 3, pp. 1–11, 1998.

[8] M Fowler, *Planning Extreme Programming.* New York, Addison Wesley, 2002, pp. 51-90.

[9] S Misra and V Kumar, "Goal-Oriented or Scenario-Based Requirements Engineering Technique - What Should a Practitioner Select?", *Canadian Conference on Electrical and Computer Engineering*, vol. 12, no. 3 pp. 2288–2292, 2005.

[10] Bas de Bar, "Using Stakeholder Analysis in Software Project Management", July, 2006. [Online]. Available : http://www.theicpm.com/blog/item/ 186-using-stakeholder-analysis-in-software-project-management. [Accessed: May 20, 2018]

[11] M Williams, J Packlick, R Bellubbi and S Coburn, "How We Made Onsite Customer Work - An Extreme Success Story", *IEEE Computer Society*, vol. 8, pp. 334–338, Feb. 2007.

[12] C Potts, ScenIC: a strategy for inquiry-driven requirements determination, *Proceedings IEEE International Symposium on Requirements Engineering (Cat. No.PR00188)*,1999, 58–65.

[13] A Sutcliffe, "Scenario-Based Requirements Engineering", *Proceedings of the 11th IEEE International Conference on Requirements Engineering*, vol. 12, no. 10, pp. 320–329, 2003.

* * *