Assimilation of Four Layered Approach to NFR in Agile Requirement Engineering

Nupur Chugh
Assistant Professor
Galgotias University, India

Aditya Dev Mishra
Assistant Professor
Galgotias University, India

ABSTRACT
In agile Requirement Engineering Techniques handling of non-functional requirements is ill defined. Customers or users talking about what they want the system to do normally do not think about resources, and other quality attributes. [1] This paper provides a solution to requirements related issues in agile environment and proposes an approach which integrates four layered approach to NFR with Agile Requirement Engineering that is helpful to improve requirement analysis process which would help in the prioritization of user stories. It also describes in detail the factors that could be considered for prioritization of user stories.

Keywords: Nonfunctional requirement, agile, requirement engineering, user stories

1. INTRODUCTION
Requirement Engineering acts as a foundation for any software and is one of the most important tasks. The process of eliciting, analyzing, specifying, validating and maintaining requirement is known as Requirement Engineering. [8]
Right requirements produce a number of benefits such as preventing errors, improving quality, and reducing risk throughout software development projects. [12]

Essentially a system's utility is determined by both its functional requirements and its nonfunctional requirements, such as reliability, maintainability, portability, and security. Both functional and non-functional requirements must be taken into consideration in the development of a quality software system. [5] Functional requirements tell about system's functionality, whereas nonfunctional requirements are quality attributes. Although the requirements engineering community has classified requirements as either functional or non-functional, existing agile requirement engineering lacks the proper treatment of quality characteristics.

2. AGILE REQUIREMENT ENGINEERING TECHNIQUES
Customer Involvement: Agile methods often assume an "ideal" customer representative: the representative can answer all developer questions correctly, and is empowered to make binding decisions and makes the right decisions. The different elicitation techniques aim to get as much knowledge as possible from all stakeholders and resolve inconsistencies.

Interviews: Interviews are the most common techniques to gather requirements. Interviews provide direct and 'unfiltered' access to the needed knowledge. It helps in establishing relationship between developer and customers.

Prioritization: is found in all the agile approaches. The highest priority feature is implemented in first increment that delivers the most business value.

Modeling: In RE, models on different levels of abstraction are used. Models are used for communication to make customer understand the system in a better way. The models are mostly throw-away models.

Documentation: In agile software development, creating complete and consistent requirements documents is seen as infeasible or at least. The scope of documentation in agile is limited.

Validation: Review meetings and acceptance tests are used for validation in Agile RE. Review meetings show that the project is in target. Different kinds of review meetings are used to present the new software.

Management: Agile methods provide a good base for requirements management. Index cards or feature backlog /list is used to write requirements.

The issues in Agile Requirement Engineering are:

- The requirements are evolved during time that leads to missing requirements interface.
- Accepted technique for non-functional requirements elicitation and management is not provided by agile methods.
- User stories are prioritized without considering interdependencies.
- Lack of focus on non-functional requirements. [1]

3. NON-FUNCTIONAL REQUIREMENTS
In the area of software requirements, the term non-functional requirements [9] have been used to refer to concerns not related to the functionality of the software. However, different authors characterize this difference in informal and unequal definitions. For example, a series of such definitions is summarized in:

a) "Describe the non-behavioral aspects of a system, capturing the properties and constraints under which a system must operate. "

b) "The required overall attributes of the system, including portability, reliability, efficiency, human engineering, testability, understandability, and modifiability."

The distinction between functionality and other qualities makes clear to the software engineers that requirements are meant to deal with quality and not only just single requirement which provides benefit in the field of requirement engineering. There are various classification schemes present for non-functional requirements. FURPS is a model for classifying non-functional requirements. FURPS emphasize on various quality attributes or non-functional requirements. [5]

Functionality: Feature set, capabilities

Usability: Human Factors, Documentation

Reliability: Accuracy, mean time to failure, frequency of failure.

Performance: Speed, Efficiency, Throughput, Resource consumption

Supportability: Testability, maintainability, portability, extensibility.

The classification schemes are inconsistent with each other. The software practitioner can chose any of the classification scheme but he/she must know the non-functional requirement terms such as performance, usability such that it can be communicated with the user as well as with the software developer so that end product will be produced as expected.

Non-functional requirements play an important role during requirement engineering and must be considered for producing quality software.

3.1Four Layered Approach to NFR

Four Layered Approach includes some rules and non-functional requirement. Four layered Approach to NFR is used to identify the goals, sub goals and finally non-functional requirements. The main objective of this approach is to find out non-functional requirements that are considered for software success. The non-functional requirements identification process can be divided into the four steps as follows and shown in activity diagram. [2]

Step 1: Identify the key stakeholders of the system.

Step 2: Generate the goals from Stakeholders based on developers' Knowledge and experience.

Step 3: Decompose the goal into sub goals.

Step 4: Identify non-functional requirements for each sub goal.

The following rules are used in the above process.

The rules are: [2]

<Who> are the stakeholders?

<What> are the services (goals)

<What > are the sub goals of each service?

<How> the sub goals are achieved under constraints.

The layered approach has following advantages: [2]

- **Cost Effective**: The layered approach is the most economical way of developing and implementing any system.
- **Rapid Application Development**: The layered Approach helps in identifying requirements in less amount of time which leads fast application development.
- **Scalability**: the layered approach scales better.
- **Task Segmentation**: Large components are broken into manageable subcomponents that are easy to develop.
- **Enhanced Understanding**: Layering helps in easy testing of sub component.

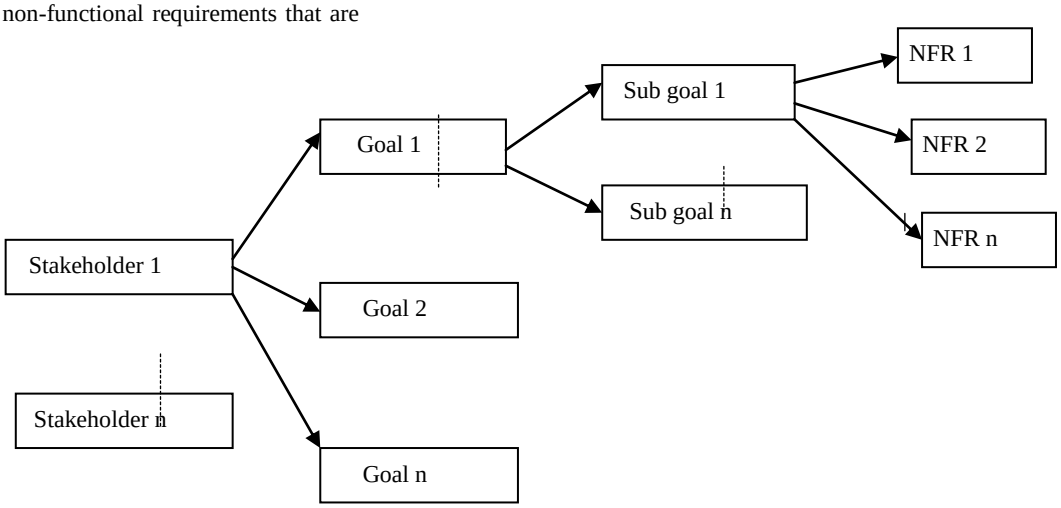

Fig 1: General Architecture for Four layered Approach to Non Functional Requirements. [2]

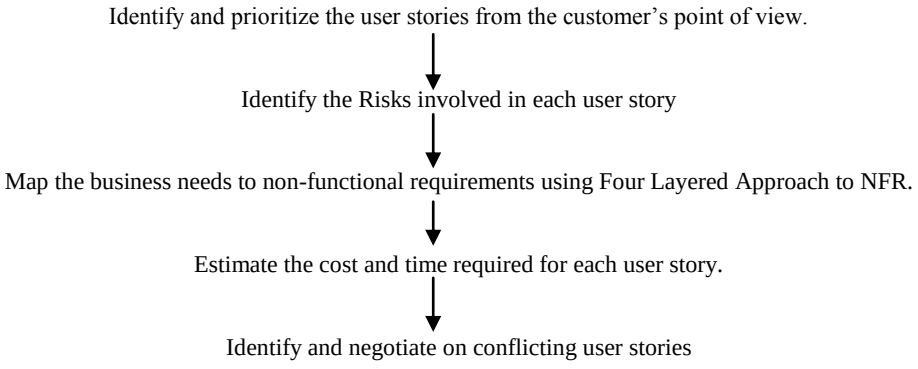

Fig 2: User Stories Prioritization Process

4. USER STORIES PRIORITIZATION

Five key principles of Extreme Programming (XP) are communication, simplicity, feedback, courage, and quality work. The first framework activity of Extreme Programming is "Planning", in which the customer writes the user stories.

The user stories define the functionality desired for the software. Different stakeholders have different needs. As end user rarely has a picture of a clear system to write the user stories, it leads to problems like requirements conflicts, missing requirements, etc. The user stories also lack focus on non-functional requirements. Two third of the projects fail because of ambiguous and incomplete user requirements and poor quality of the requirements. [11] For small to medium organizations, appropriate requirements prioritization and selection can increase the possibility of project success. Requirement Prioritization is the process of making a choice among multiple options. It is considered as important activity in requirements engineering, as it helps the developers to properly analyze requirements, in order to rank them according to their business value.

The following steps have been suggested for prioritization of user stories that lay focus on non-functional requirements. The non-functional requirements may be handled in requirement specification and may help in development of quality software.

STEP 1: Identify and prioritize the user stories from the customer's point of view: The user stories are identified and prioritized on the basis of business value. The agile teams distinguish between *"must have" and "nice to have"* requirements. This is achieved by frequent communication with customers.

STEP 2: Identify the Risks involved in each user story: The risk exposure may be calculated for each user story. Risk exposure is calculated on the basis of probability of a risk and its impact.

STEP 3: Map the business needs to non-functional requirements using Four Layered Approach to NFR: The business needs identified by customers may be mapped into non-functional requirements using four layered approach. The goals given by different stakeholders are analyzed and partitioned into sub goals that are further analyzed for identification of non-functional requirements.

STEP 4: Estimate the cost and time required for each user story: The cost and time required for each user story is estimated. The estimation may be based on historical data.

STEP 5: Identify the conflicting user stories: Identify the user stories from developer's view conflicting with the one from customer's view. The user stories that are conflicting have to be negotiated by communicating with the customers.

5. CONCLUSION

The success of software depends on whether the customer is satisfied and it's all requirements have been met by the software. The paper has proposed an approach to improve the requirement analysis by considering nonfunctional requirements that plays an important role in software success. The Four layered approach to NFR have been integrated with Agile Requirement engineering which increases the probability of acceptance of software and helps in rapid development of software.

6. REFERENCES

[1] Waleed Helmy, Amr Kamel and Osman Hegazy, "Requirements Engineering Methodology in Agile Environment", International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012

[2] A. Ananda Rao and M.Gopichand, "Four Layered Approach to Non-Functional Requirements Analysis", International Journal of Computer Science Issues, Vol. 8, Issue 6, No 2, November 2011

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

[4] Saravana. K.M, G. N. Basavaraj, Rajkumar, Dr. A. Kovalan, " Case Study On Agile User Stories Prioritization Using Imaginative Standard", International Journal of Engineering Research and Applications (IJERA), Vol. 2, Issue 5, September- October 2012

[5] Lawrence Chung1 and Julio Cesar Sampaio do Prado Leite, "On Non-Functional Requirements in Software Engineering", Springer-Verlag Berlin Heidelberg 2009

[6] Malik Qasaimeh, Alain Abran, "Extending Extreme Programming User Stories to Meet ISO 9001 Formality Requirements", Journal of Software Engineering and Applications, 2011

[7] Evita Coelho, Anirban Basu, " Effort Estimation in Agile Software Development using Story Points" , International Journal of Applied Information Systems (IJAIS), Volume 3– No.7, August 2012

[8] Dr. Sohail Asghar & Mahrukh Umar, "Requirement Engineering Challenges in Development of Software Applications and Selection of Customer-off-the-Shelf (COTS) Components", International Journal of Software Engineering (IJSE), Volume (1): Issue (2)

[9] Naresh Kumar Nagwani, and Pradeep Singh, "An Agile Methodology Based Model for Change-Oriented Software Engineering", International Journal of Recent Trends in Engineering, Vol.1,No.1,2009, pp.128-132

[10] Chetankumar Patel, and Muthu Ramachandran,"Story Card Based Agile Software Development", International Journal of Hybrid Information Technology,Vol.2, No.2, 2009,pp.125-140

[11] Veerapaneni Esther Jyothi, and K. Nageswara Rao "Effective Implementation of Agile Practices", International Journal of Advanced Computer Science and Applications, Vol. 2,No. 3, 2011, pp.41-48

[12] International Journal of Internet Computing (IJIC), ISSN No: 2231 – 6965, Volume-1, Issue-2, 2011 34An Insight into the Importance of Requirements Engineering.

[13] A. Eberlein, F.Maurer, F.Paetsch, "Requirements Engieering and Agile Software Development", Proceedings of the Twelfth International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003

[14] S. Ambler, "Agile Requirements Modeling", 2012 available at: http://www.agilemodeling.com/essays/agile Requirements.htm

[15] R.S Harirs and M. Cohn, "Incorporating Learning and expected cost of Change in Prioritizing Features on Agile Projects," Proc. XP 2006, pp.175-180