INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X

(Volume2, Issue5)

Available online at: www.Ijariit.com

Review on Agile Method with Text Mining

Parveen Kaur

M.Tech scholars Department of Computer Science and Engineering,
GGS College, Kharar, Punjab, India
Parveenkaur17@gmail.com

Abstract— working software measures the progress. Basically, agile method involves interleaving the specification, implementation, design and testing. Series of versions are developed with the involvement of and evaluation by the stake holders in each version. Agile methods aim at reducing the software process overheads (like documentation) and concentrate more on code rather than the design. Customer involvement, incremental delivery, freedom of developers to evolve new working methods, change management, and last but not the least simplicity is the basic essence of Agile development. Agile methodologies are well suited for small as well as medium sized projects.

Keywords— agile, software engineering, adaptive.

I. INTRODUCTION

Agile software development is defined as rapid development and delivery of the software in the requirement changing environment which is well suited for present day business scenario. Working software measures the progress. Basically, agile method involves interleaving the specification, implementation, design and testing. Series of versions are developed with the involvement of and evaluation by the stake holders in each version. Agile methods aim at reducing the software process overheads (like documentation) and concentrate more on code rather than the design. Customer involvement, incremental delivery, freedom of developers to evolve new working methods, change management, and last but not the least simplicity is the basic essence of agile development. Agile methodologies are well suited for small as well as medium sized projects. However, uniform customer involvement throughout the project, appointing appropriate team to adapt to Agile methodology, ranking of changes to be accommodated in software, maintaining simplicity, difficulty in scaling Agile procedures to larger projects and deciding upon the contract terms account for the major disadvantages involved in the Agile development.

1.1 Agile Requirement Engineering

Requirement Engineering is the most important step in any software development. Success or failure of correct and complete requirement engineering becomes the most important part in guiding the project to its final success or a complete failure. Requirement Engineering process is composed of five major activities namely, requirement elicitation, analysis and negotiation, documentation, validation, and management.

1.2 Text Processing

General text pre-processing [37] is nothing but preparing the available text for computer analysis. In involves steps that prepare the text for computer understandable representation and extracts the necessary useful matter from the text. There are mainly three steps involved in text pre-processing, namely:

- **Tokenization:** This involves the process of converting a stream of characters into tokens (generally word tokens). Delimiters like spaces; punctuations etc. are used for separating one word token from another.
- **Stop-word Removal:** In this step, words that carry negligible or little meaning for the sentence like 'is', 'of', 'and', 'the', etc. are removed.

- **Stemming:** It is the process of reducing words in the word stem to its root form like ‘writes’, ‘writing’, ‘wrote’, ‘written’ these all correspond to a single root “write”.

1.3 Clustering

Clustering is nothing but partition of data into sets of similar items. Each of the sets is called a cluster. Document clustering aims at increasing cohesion in a single cluster and minimizing coupling between two or more clusters i.e., trying to reduce the intra-cluster distance and increase inter-cluster distance. Clustering is considered to be a part of unsupervised learning. Three most popular clustering methods are described below:

- **K-Means:** It is the simplest flat and hard clustering algorithm. This algorithm’s objective function tries to minimize average squared distance of items from the cluster centers (which is mean of items in a cluster).
- **Expectation Minimization:** It is a flat model-based clustering technique which assumes data to be generated by a model and tends to recover that original model from data.
- **Hierarchical Clustering:** As described by [39], hierarchical technique creates a nested structure of partitions with an all-inclusive single cluster at root and singleton clusters of singular points at bottom.
- **Agglomerative:** It is a bottom up approach. Points are considered as individual clusters at the starting. At each step, most similar clusters are merged according the cluster similarity/distance definition.
- **Divisive:** It is a top down approach. Process starts with root cluster and is split until singleton clusters of individual points are formed.

1.4 Machine Learning

Machine learning is basically a subfield of artificial intelligence which lends computers the learning ability without explicitly programming them. This involves development of those computer programs which have the ability of teaching themselves for growing and changing when they are exposed to fresh data. Following are the two main learning styles [40]:

- **Supervised Learning:** Input data (training data) has recognized labels or output datasets. Training process prepares a model which makes predictions and the model is corrected for the wrong predictions. This training process is continued until a desired accuracy level is achieved on training data.
- **Unsupervised Learning:** Input data does not have recognized labels or output datasets. Structures available in input data are deduced to create a model which may be for extracting general rules or for organizing data by similarity.

II. LITERATURE REVIEW

John Mylopoulos et al [1], to propose a comprehensive process oriented qualitative framework that integrates non-functional requirements into the process of software development. To illustrate the application of proposed methodology by taking examples of accuracy requirements in design phase and performance requirements in implementation phase for information systems.

Armin Eberlein and J. C. S. P. Leite [2], this paper looks at numerous aspects of requirements engineering and argues about their suitability for agile approaches. The aim is to elicit lessons from requirements engineering that agile methods might consider, if quality is a major concern. To highlight various important aspects of requirement engineering and describe a few such practices which may be suitable for agile approaches.

Frauke Paetsch et al [3], paper analyses commonalities and differences of both approaches and determines possible ways how agile software development can benefit from requirements engineering methods. To discuss current requirement engineering approaches. To describe agile methods from the perspective of requirement engineering. To evaluate the extent of improvement in agile methods by incorporating a few of requirement engineering techniques described in goal one.

Nelson Souto Rosa et al [4], To propose a comprehensive process oriented qualitative framework that integrates non-functional requirements into the process of software development. To illustrate the application of proposed methodology by taking examples of accuracy requirements in design phase and performance requirements in implementation phase for information systems.

Lawrence Chung et al [5], To review the current state of art on treatment of definition, classification and representation of non-functional requirements in software engineering. To highlight the future direction of research for non-functional requirements. Essentially a software system’s utility is determined by both its functionality and its non-functional characteristics, such as usability, flexibility, performance, interoperability and security.

Salam Farhat et al[6], To propose a methodology based on NFR framework for reasoning and refining about non-functional requirements by identifying four different types of non-functional requirements based on their development strategy and provide preliminary analysis on these non-functional requirements' development thereby enhancing software engineering process and promoting various software quality attributes. To illustrate and validate the proposed methodology using a case study.

Taehoon Um et al [7], To propose a light weight quality evaluation method that reveals quality attributes for enhancing non-functional features in an agile method enabling iterative evaluation of the quality attributes in each release, supports continuous consideration of quality issues by participants and better planning for improvements in quality.

Vikas Bajpai and Ravi Prakash Gorthi [8], To review importance, definition, specification of various non-functional requirements in present competitive software market to promote better understanding of non-functional requirements. To suggest future research areas for the same field like formal definition, specification and measurement of non-functional requirements; prediction and estimation of non-functional requirements in early stages that may improve customer satisfaction and maximize profit.

III. CONCLUSIONS

Light weight quality evaluation method that reveals quality attributes for enhancing non-functional features in an agile method enabling iterative evaluation of the quality attributes in each release, supports continuous consideration of quality issues by participants and better planning for improvements in quality. Non-functional requirements in software engineering. To highlight the future direction of research for non-functional requirements. Essentially a software system's utility is determined by both its functionality and its non-functional characteristics, such as usability, flexibility, performance, interoperability and security

REFERENCES

[1] B. Boehm, "Get Ready for Agile Methods, with Care," *Computer,* vol. 35, no. 1, pp. 64-69, 2002.

[2] K. Beck, "Embracing change with extreme programming," *Computer,* vol. 32, no. 10, pp. 70-77, 1999.

[3] K. Schwaber and M. Beedle, Agile Software Development with Scrum, Pearson Education International, 2002.

[4] J. Stapleton, DSDM, Dynamic Systems Development Method: The Method in Practice, Addison-Wesley, 1997.

[5] J. A. Highsmith, Adaptive Software Development: A Collaborative Approach to Managing Complex Systems, Dorset House, 2000.

[6] S. R. Palmer and J. M. Felsing, A Practical Guide to Feature-driven Development, Prentice Hall, 2002.

[7] A. Cockburn, Crystal Clear: A Human-powered Methodology for Small Teams, Addison-Wesley, 2005.

[8] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit, Addison-Wesley, 2003.

[9] D. J. Anderson, Kanban, Blue Hole Press, 2010.

[10] S. W. Ambler, Agile Modeling: Effective Practices for EXtreme Programming and the Unified Process, Wiley, 2002.

[11] S. W. Ambler, "The Agile Unified Process (AUP)," Ambysoft Inc., 2005. [Online]. Available: http://www.ambysoft.com/unifiedprocess/agileUP.html. [Accessed 04 01 2016].

[12] G. Kotonya and I. Sommerville, Requirements Engineering: Processes and Techniques, John Wiley & Sons, 1998.

[13] I. Inayat, S. S. Salim, S. Marczak, M. Daneva and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in human behavior,* 2014.

[14] L. Cao and B. Ramesh, "Agile requirements engineering practices: An empirical study," *Software,* vol. 25, no. 1, pp. 60-67, 2008.

[15] R. Balasubramaniam, C. Lan and B. Richard, "Agile requirements engineering practices and challenges: an empirical study," *Information Systems Journal,* vol. 20, no. 5, pp. 449-480, 2010.