Non Functional Requirement in Agile Software Development

Y. Aziz[1], T. Aziz[2], M. I. Malik[3], M. K. Baig[4], M. Z. Ali[5], M. Baqer[6]

[1,3,4,6]Computer Engineering Department, UCE&T Bahauddin Zakariya University, Multan
[2]School of Information and Communication Engineering, Beijing University of Posts and Telecommunication, China
[5]Electrical Engineering Department, UCE&T Bahauddin Zakariya University, Multan
[1]engr.yasiraziz@gmail.com

Abstract-Agile Software Development (ASD) is very popular worldwide for developing software applications efficiently according to dynamic user's requirements. Previously, Traditional Software Engineering (TSE) was used but unplanned changes in customer's requirements raised a need for agile software development technology. Both mentioned techniques of software development have significant pros and cons like: ASD bases on modeling rather than documentation while TSE bases on documentation rather than modeling. In order to overcome the drawbacks, this paper particularly focuses on analysis of some methods and approaches like Non-functional Requirements Model for Agile Process (NORMAP), Method of Elicitation, Documentation & Validation (MEDoV) and Dual Application Model so that benefits of both TSE and ASD can be achieved.

Keywords-Agile Requirements Engineering, Dual Application Model, Functional Requirements, MEDoV, Non Functional Requirements, NORMAP, Traditional Requirement Engineering, UML Use Case

I. Introduction

Requirement engineering is a software process focused on identification of user requirements, their in-depth analysis, documentation & validation for the proposed system. As software industry has grown, software community has continually attempted to develop easier, faster and less expensive technologies to build and maintain high-quality computer programs. Understanding customer's requirement is among the most difficult tasks because even if customers or end-users are explicit in their needs, those needs will change throughout the project development. Broadly, requirement engineering is divided in two types: Functional Requirements (FR) and Non-Functional Requirements (NFR). FR is the goal or model making after collecting data from users. Alternatively, Unified Modeling Language (UML) Diagrams are included in Functional Requirements. NFR covers maintenance, performance, security, and testability segments. Agile focuses on FR and TSE focuses on NFR. This paper is subdivided into three parts: In first part, comparison of ASE and TSE is discussed. In second part, various approaches and their drawbacks are discussed for including NFR in Agile Requirement Engineering. In last part, different methods are described for combining functional and non functional requirements in a single ASD model.

II. Literature Review

Due to change in business environment as well as change in system requirements the demand of agility in software development increases. Many firms replace their method of development with more adaptive agile approach. System Dynamic (SD) model was first developed by Jay Forster. SD is a modeling technique which uses feedback control system. SD principles and techniques are used to explain the dynamic behavior of organization. It is also used to create a connection between different elements in a system.

A. Agile Vs Traditional Requirement Engineering

Requirement engineering in traditional technique is complicated because documentation is constructed by collecting all the client requirements [i]. It becomes cumbersome when clients propose changes in constructed requirement document. Agile Requirement Engineering (ARE) is more flexible and quicker because it provides an iterative and incremental framework which helps developers in rapid delivery of products. In ARE customers are satisfied because of constant communication with the developers. Agile is successful due to its visibility of system to its clients and face to face communication. Comparison is precisely elaborated in the form of diagram.

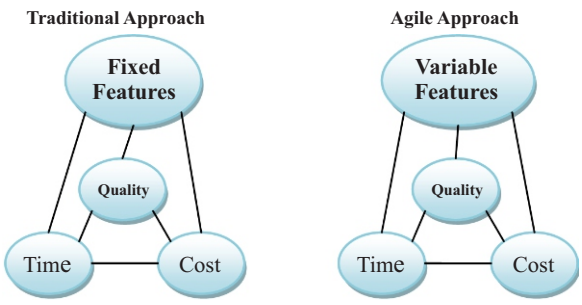


Fig. 1. Traditional Vs Agile Approach

B. Traditional Requirement Engineering

It involves some characteristics of elicitation, analysis, documentation, validation and requirements management [ii] [iii].

Requirement Elicitation: requirements are primarily discovered from stake holders. Different techniques are used for requirement elicitation like brainstorming, prototyping, interviews, use cases etc.

Requirement Analysis: checks either requirement elicitation is completed and feasible. Negotiation process is carried out in-order to refine the requirements. Requirements should be prioritize with the help of stake holders and requirements are modeled by using the different techniques like data flow model, semantic data model and object oriented approaches.

Requirement Documentation: requirements are written in a reviewable way. These documentations are for all types of functional and non functional requirements. For this purpose SRS template is used.

Requirement Validation: Customer continuously provides support regarding requirements and mainly validate satisfaction of his need. He must also check either requirement satisfaction level is achieved or not? Test cases are used for ambiguities, if there are problems in requirement validation then test cases can be used for better results.

Requirement Management: all the facts provided about the requirements are successfully done by the requirement management team. This is a modification control board which consists of pivotal stakeholders that establish all the diversity.

C. Agile Development

By using the agile method products are delivered quickly. Initially agile development was used for small projects but later on successful outcomes of agile resulted demand for agile methods for the large projects. The following diagram explains the Agile Development Methodology.

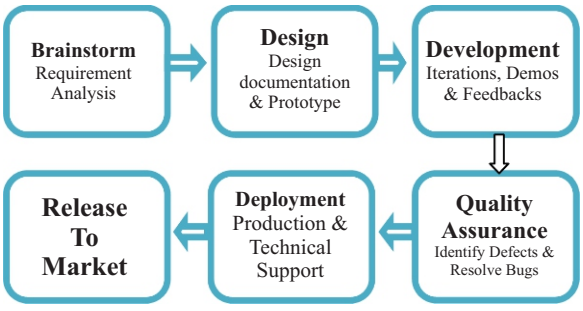
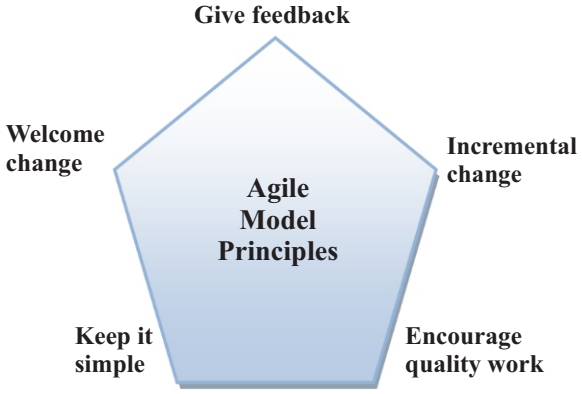


Fig. 2. Agile Development Methodology

Agile Principles: are slightly divergent but they work on the clone principles as pointed out in the agile manifesto. Agile Model Principles are elaborated in Fig. 3.



Agile Values: For every agile methodology there are same set of values provided by agile. Agile Manifesto is explained in the form of diagram in Fig. 4.

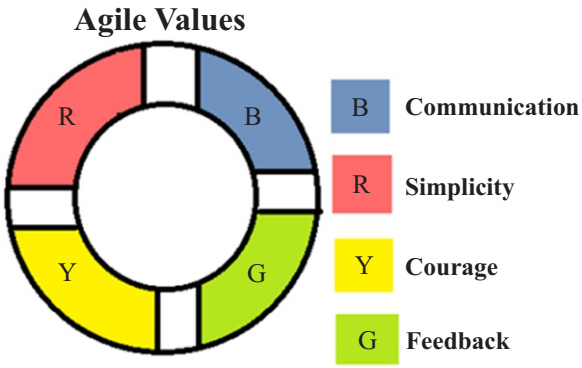


Fig. 4. Agile Values

Traditional approaches like waterfall model concentrate on planning while agile methods are flexible and depend upon the user expectation, so the customer satisfaction increases in agile. Agile requirement engineering easily captures the evolving requirements at any stage of software development life cycle (SDLC). But this is not the case in traditional RE. In traditional RE we prepare huge documentation while in agile RE we don't. By using the agile methods products deliver in time, which will satisfy the client's expectation.

D. Tools for Requirement Description

Agile methodologies like Scrum and XP are becoming more and more popular. They modernize our thinking power about the software development. Agility just does not have impact only on coding but also on the activities of requirement engineering. A case study was done in 2014 in Poland shows that use cases are best tool for requirement description rather than user stories, because use cases provide zooming but this scenario can be different for non-functional requirements. So UML use case diagrams are the best tool for requirements description.

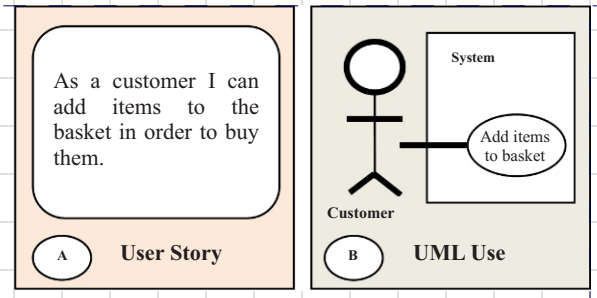


Fig. 5. User Story & Use Case

E. Agile Requirement Issues

There are three major issues found in agile requirement [iv] and are shortly discussed here:

- Missing RE Activities: as there is no technique available to create an activity specification document.
- Missing Requirement Interface: as nobody knows the whole system requirement and this can cause missing requirement interface.
- Non-Functional Requirement Elicitation: in agile after every iteration product is handed over to customer to make sure his/her requirements are met.

F. Requirement Gathering Approaches

Joint Application Development (JAD) is a requirement gathering technique. In JAD project manager conducts a meeting with all executive sponsors & other stakeholders for requirement gathering. This document contains use cases & business models through which a customer communicate with system. Catalog having all known requirements about product is called **product backlog**. For prioritization of gathered requirements, **VIEWPOINT** approach is used. Viewpoint approach is used for getting the viewpoint of a system in terms of customer. By view point approach we gathered information from stakeholders related to system & organization. By using Viewpoint approach we verify Increments. We get view point of customers in terms of system. Scrum is a part of agile methodology. It is used when requirements vary with time. In scrum work is divided into small modules known as "sprints", where each sprint has a limitation of 2 to 4 weeks. Scrum methodology is mostly used for small projects. However A large projects can be divided into sub projects and these sub projects handled by a scrum team.

In agile requirements engineering approach several architectural related issues arises. Common issues observed are:

- Incomplete requirements elicitation
- Incorrect prioritization of user stories
- Lack of focus on non functional requirement.

III. Proposed Methodology

Author has divided proposed methodology into several phases. Each phase is discussed briefly:

A. Inception Phase

In this phase customer is briefed about main phases, activities & techniques. The basic activities of Initial phase are:

- Defining responsibilities of each role
- Assignment of roles to team members
- Identification of various users of system
- Briefing customers and users about proposed methodology
- Identification of goal and mission of the product.

B. Feature's List Identification

In this phase following parameters are covered:

- Preparation: Rearrange Meeting with stakeholders & major feature of system.
- Elicitation: Brainstorming & Open-ended session with stakeholders.
- Validation & Estimation: Development team discuss identified features with customers.
- Prioritization: Customers priorities identified in this step.

C. Feature Grouping

Feature grouping consists of:

- Preparation
- Grouping
- Validation & Estimation

D. Group Prioritization

Group prioritization consists of:

- Preparation
- Prioritization

E. Non Functional Requirement Identification

Non-functional requirements identification consists of:

- Elicitation
- Validation
- Prioritization

F. Architecture Envisioning

Architecture envisioning phase involves:

- Must be familiar to the problem & proposed solution
- Over all view of issues about system
- Increase collaboration between whole team

G. Task Identification

In this phase user stories are defined. If multiple teams are involved then each team define its own user stories.

- Each story must be developed by more than one developer
- Ensure that developer properly take detail.

H. Task Development

Test Driven Development (TDD) is an agile practice for code writing. Fundamental principle of agile is to deliver demanded product to the customer so TDD & Customer Acceptance Testing are used in this phase.

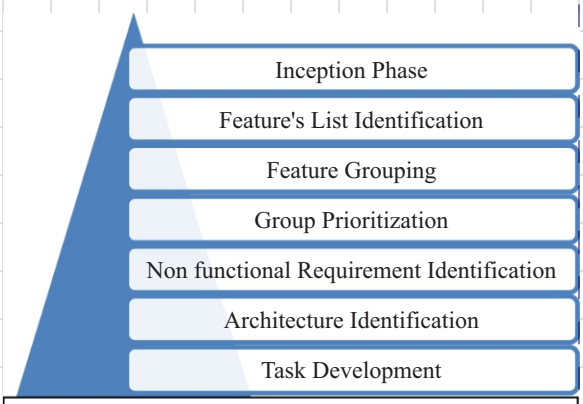


Fig. 6: Relation between Phases

IV. Non Functional Requirements in Agile

A method named Non Functional Requirement Modeling for Agile Process (NORMAP) has significant importance in linking non functional requirement with functional requirements [v]. Another technique named as Method of Elicitation, Documentation and Validation of Software User Requirement (MEDoV) have significant importance in linking NFR to FR [vi] [vii].

A. NORMAP

Main purpose of this methodology is to handle non functional requirements and design an easily miscible framework for agile development. Basic idea of NORMAP is to combine NFR and FR in single model. NORMAP framework can be used manually and automatically. Manual framework is called Non-functional Requirements Modeling for Agile Manual (NORMANUAL). Automatic framework is called Non-functional Requirement Modeling with Agile Automatic (NORMATIC) [vii]. Keeping NFR on first priority a new W^8 Story Card model is proposed to be used. W^8 model captures functional requirements with Chung's Color coded NFR framework [v] resulting new techniques of dealing both NFR and FR with minimized risks.

NORMAP framework proposes three objectives. First one is W^8 Story Card Model in which there are 8 'W' and each 'W' represent different word.

TABLE I
Eight W'S of W^8

Who (Actor)
What (What user wants)
Why (Business justification according to user requirement)
Within (Estimated time of project completion)
While it is nice to have (NFR)
Without ignoring (functional requirements)
With a priority of (According to user requirements)
Which may impact (List of impacted requirements)

Second objective is Agile Requirement Classification in which both FR (Agile USE Case - AUC) and NFR (Agile Choose Case - ACC) are classified.

Third objective is "PointCut" operators in which "before", "after", "override" and "wrap" keywords are used for combining both FR and NFR. These points are also used for linking ACC to AUC.

B. Phases of NORMAP Methodology

Main phases of NORMAP development are:

- Selected NFR and initial data collection (Select NFR according to dataset, there are 161 NFR in framework).
- Initial Data Pre-Processing (tool for selecting synonym and antonym of selected NFR and store in database).
- Automatic parsing of requirements statements (For checking grammar and sentences and populate W^8 Story Card Model.
- Modelling Agile Use Cases (Design according to user stories).
- Modelling Agile Loose Cases (All NFR are kept in Loose Case)
- Modelling Agile Choose Case (Like ALC it also check NFR and their linking).
- Requirement Implementation Sequence Planning (NFR planning phase and check risk here)

NORMAP methodology was validated by utilizing Predicator Models in Software Engineering (PROMISE). Collected data was from 15 different software projects of DE Paul University. Those requirements which were not part of 25 NFR were eliminated and out of 635, 607 requirements were used. In this case success rate was 87.15%.

In another case 26 requirements were used from which NFR were identified by using NORMAP methodology. This case study utilized European Union (EU) e-Procurement Online System. Eight (viii) steps

were performed in this case study using Non-Functional with Agile Automatic (NORMATIC). Result of this case study was 87.71%.

C. *MEDoV 2nd Model*

Plan Driven Software Development is for companies whose requirements remain same till end of project and have best technology but it is not good for all companies that's why Agile Software Development is used for gathering exact information from user and make software in the way that if user requirement change at some point modifications can be done easily. Lean Software Development is getting exact requirements from user and understanding problems from customer's point of view [vii].

There were some flaws in Agile and Lean Software Development like no connection between requirements, short documentation, and non-functional requirements for this purpose new Method of Elicitation, Documentation and Validation of Software User Requirement (MEDOV) method [viii] [ix] is introduced and by this model Stake Holders have to deal with following decisions like Quality Decision, Preference Decision, Classification Decision and Property Decision. This model helps to Stake Holders in making right decisions and implementing these decisions in good way.

Phase I observes the system and set priorities according to user requirements. Initially Key Performance Indicator (KPI) is used by the software developers to focus on important parts of software. This phase is restricted to be performed in the start of development only.

Phase II is working on requirement documentation and for this purpose Event-Driven Process Chain (EPC) and Unified Modeling Language (UML) are used. EPC is warmly welcomed by business users because of user friendliness and easy modeling, while UML diagrams are used widely by developers. MEDoV uses top-down approach for traversing high priorities to lowest one. Business information is safe in business process repository. Business process repository contains Conventions (Some way for doing work in project like graphic conventions), Business Process Modeling (EPC use) [x] and Business Rules, Computational Models (EPC and UML diagram)

Phase III performs requirement analysis, specification and validation; it is named as "To Be" model, means work done in last.

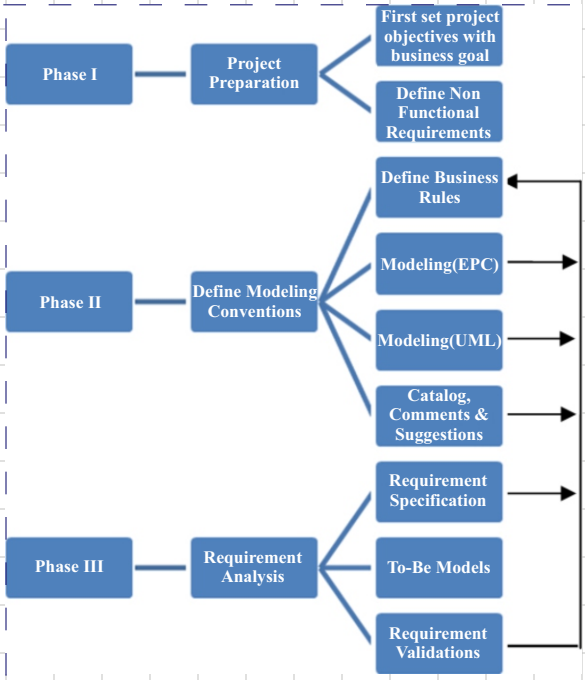


Fig. 7. Phases of MEDoV

D. *Dual Application Model*

Dual Application Model refers to two applications in single model. These two applications are logical and physical Application. Logical Application (LA) is a logical model of domain solution / software problem. It is a system in which there are no non-functional requirements or implementations. Physical application (PA) is the one in which first LA and then non functional requirements are implemented. This model also covers the problems in Agile software and develop a new technology namely Agile Software Engineering in which the best of Traditional Software Engineering (TSW) and best of ASD has been added.

V. Conclusion

Quality is worthy of serious considerations whenever software engineering practices are applied. For quality product it is necessary to combine both NFR and FR in one model. Both Agile and Traditional software engineering principles are widely used by the software developers for producing high-quality systems. Yet, there is a need for combining agile and traditional models into one model so that both Functional and Non Functional requirements can be compensated. If we just focus on modeling technique like Agile, the non functional requirements are snubbed resulting unavoidable security & maintenance issues.

References

[i] H. Saiedian and R. Dale, "Requirements engineering: making the connection between

the software developer and customer," Information and Software Technology, 42(6), 2000,:ppAI9-428

[ii] A. D. Lucia and A. Qusef "Requirements Engineering in Agile Software Development", Journal of Emerging Technologies in Web Intelligence, Vol. 2, No. 3, August 2010.

[iii] F. Paetsch, A. Eberlein, F. Maurer, "Requirements Engineering and Agile Software Development", Twelfth IEEE International Workshop, 2003, pp. 308-313.

[iv] J. Nawrocki et al., "Extreme Programming Modified: Embrace Requirements Engineering Practices," Proc. IEEE Joint Int'l Conf. Requirements Eng. (RE 02), 2002, IEEE CS Press, pp. 303–310.

[v] L. Chung, B.A. Nixon, E. Yu, and J. Mylopoulos, Non-functional Requirements in Software Engineering, Boston, MA, Kluwer Academic Publisher, 2000.

[vi] W. M. Farid and F. J. Mitropoulos, "Novel lightweight engineering artifacts for modeling non-functional requirements in agile processes", Proc. IEEE SoutheastCon 2012 (SoutheastCon 2012), Mar. 2012.

[vii] W. M. Farid and F. J. Mitropoulos, "NORMATIC: A visual tool for modeling non-functional requirements in agile processes", Proc. IEEE SoutheastCon 2012 (SoutheastCon 2012), Mar. 2012.

[viii] M. Poppendieck, and T. Poppendieck, "Lean software development", Addison-Wesley, 2003.

[ix] S. Dragicevic, S. Celar, and L. Novak, "Roadmap for requirements engineering process improvement using BPM and UML", Advanced in Production Engineering & Management (APEM), Vol. 6(3), Sept. 2011, pp.221-231.

[x] S. Dragicevic, and S. Celar, "Method for elicitation, documentation and validation of software user requirements (MEDoV)", 18th IEEE International Symposium on Computers and Communications (ISCC 2013), 7-10.07.2013, pp. 956-961.