Enhancement Approach for Non-Functional Requirements Analysis in Agile Environment

Bahiya M.Aljallabi
Department of Computer Science
Faculty of Mathematical Science, University of Khartoum
Khartoum, Sudan
bahiyaj@gmail.com

Abdelhamid Mansour
Department of Computer Science
Faculty of Mathematical Science, University of Khartoum
Khartoum, Sudan
abhamidhn@gmail.com

Abstract— **Non-functional requirements (NFRs) are quality attributes that define how a software product will do its functions. They are important and critical to the success of any software in the market as they considered as the differentiating factor from other software products that provide similar functionality. Agile methodology became popular during the last few years, and although it improves the process of software development, it has a number of limitations regarding requirements analysis. Neglecting the non-functional requirements is one of the biggest limitations in Agile. It doesn't provide any widely accepted technique for elicitation and management of non-functional requirements. This paper summarizes two existing approaches that are currently used to analysis NFRs, and then it provides an enhanced approach for better analysis of NFRs. The new approach is better than the existing ones, because it combines their strengths and overcome their weaknesses.**

Keywords— ***Requirements Engineering; non-functional requirements; Functional requirement; Internal quality attributes; External quality attributes; Agile Methodology.***

I. INTRODUCTION

Software Requirements Analysis is a field within Software Engineering that deals with establishing the needs of stakeholders that are to be solved by a software product. The IEEE Standard Glossary of Software Engineering Technology defines a software requirement as "a condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document"[1].
Requirements are divided into two parts: functional requirements (FRs) and non-functional requirements (NFRs). The functional requirements define what the software product should do, while the non-functional requirements define how the software product will behave. For example: security, performance, availability, etc…

NFRs also known as technical requirements, quality attributes or quality of service requirements [2].

Non-Functional requirements (NFRs) analysis is one of the most important and critical activity for the success of any software development. NFRs are the constraints over the services that will be provided by the system and it defines the system quality. Through the process of elicitation and analysis of the requirements, the requirement engineer should collect and analyze both the functional and non-functional requirements. Customers are usually fully aware of the product's functional requirements but not so much so when it comes to its NFRs. As the software products complexity increasing and customers demand for higher quality software products, NFRs can no longer be considered as a secondary importance [3].

There are few approaches and methodologies that used to deal with the process of NFRs analysis. Agile methodologies play a great role in delivering products faster, with high quality, and satisfying customer's needs and expectations through its flexible principles of the agile software development [4]. Although Agile enhanced the process of software requirements, it still has a number of limitations in requirements engineering. Such as the minimal documentation of requirements which causes traceability issues, as well as the increase of input and feedback from customers wish increases the amount of rework. In addition, in agile software development budget and time estimation is often changed substantially as a result of requirements changes. Lastly, Agile methodology neglects the NFRs, and focuses only on functional requirements. Although these NFRs are defining the software quality, it doesn't provide any widely accepted technique for eliciting and managing non-functional requirements [5, 6].

There are not so many approaches to analysis the NFRs in an effective way. This paper will add value to NFRs analysis process by studying the existing approaches and propose an enhanced process oriented approach that overcomes existing limitations. Improving the quality of NFRs analysis will improve the overall software quality. As a result, this will make the software product more robust and a strong competitor and will lead to its success in the market.

II. Background

Software requirements engineering includes two major processes: requirements development and requirements management. It also has a number of types and levels [7].

A. *Requirements Engineering Process*

Requirements development includes elicitation, analyzing, specifying, and validating the requirements. In the other hand, requirement management mainly includes controlling changes requests and tracking the status of requirements.

- Requirements Elicitation is the first phase in requirements development process. It includes the activities of understanding, learning and gathering the requirements from each stockholder class and communicating these needs and requirements to the system developers [7,8].
- Requirements Analysis is the second phase in requirements development process. It includes refining and representing the requirements in various forms to make sure that, the received requirements from customers during requirements elicitation are consistent, complete and feasible [9].
- Requirements Specification is the process of formally documenting the refined requirements. It can be done in one document or may extend to a number of documents based on the project size [7].
- Requirements Validation is the last step in requirements development process. It is to make sure that requirements are complete, consistent, modifiable, unambiguous, concise, measurable, testable and traceable [7].
- Requirements Management is the process of working with stockholders to produce the baseline requirements and control them. This process involves a number of activities to control the changes requests, such as performing analysis on requested changes to evaluate their impact, and based on that analysis to decide whether to approve or disapprove these changes. And finally to make sure that approved requested changes are implemented. Requirements management also includes the activities used to ensure that requirements development process is consistent with the project plan, and the status of each requirement is updated as the project progresses through the software development process. [7].

B. *Levels and Types of Requirements*

Software requirement engineering includes three levels, Business level, user level and product level as illustrated below in figure I. In the business level, there is a Business Requirements Document (BRD) that defines the business problem which will be solved by the software product. In the user level, user requirements and quality attributes are defined under constraint of business rules. User Requirements Document (URD) defines tasks that the user will perform by the software product. Quality attributes are a description of software characteristics, and are used to drive the non-functional requirements (NFRs). There are two types of quality attributes, internal quality attributes and external quality attributes [10, 11].

In the product level there are two major types of requirements: Functional requirements that define the functions that will be provided by the product and non-functional requirements that describe “how well” the software product should behave. Both functional and non-functional requirements are documented in Specification Requirements Documents (SRS) [9].

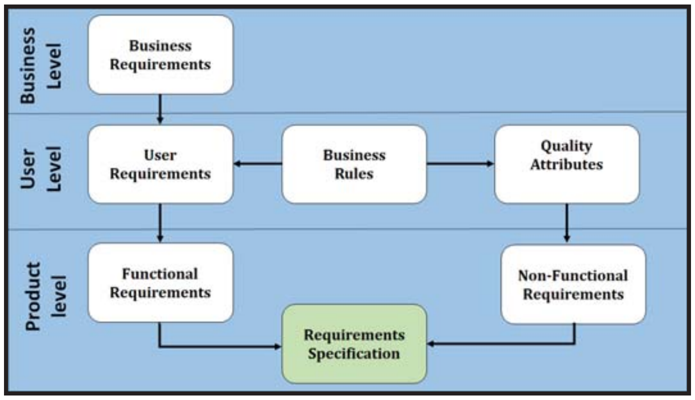

Fig 1. Levels and Types of Requirements

C. *Impact of Agile on Requirements Engineering*

The term “Agile Requirements Engineering” is used to define the Agile way of planning, executing and controlling requirements engineering Activities[12].The processes of requirements development and management used in TR (eliciting, analyzing, specifying, validating and management) are still the same in Agile, but Agile has adapted new ideas for the new environment [13].

Irum, Siti, Sabrina, Maya, Shahaboddin, [6] conducted a systematic review of literature published between 2002 and 2013 and identified 21 papers, that discuss agile requirements engineering. The review identified 17 practices of agile requirements. Some practice resolved issues of Traditional RE, while some others raised new issues. Below is a summary of these practices and their impact in the requirements engineering.

III. RELATED WORK

The proposed approach draws base on the results from investigations in two studies in the same topic. Below section is to summarize these studies.

A. Four Layered Approach to Non-functional requirements Analysis

Gopichand and Anandarao [3] notice that there are not so many approaches available to help in the eliciting and analysis of non-functional requirements, so they proposed a new approach called "Four Layered Approach for Non-functional Requirements". They have also provided a check list and matrix to validate the identified requirements and ensure their completeness.

This approach divides the process of identifying the NFRs into four steps as following: Step one is to identify the key stakeholders of the system. Step two is to generate goals from Stakeholders. Step three is to decompose the goal into sub goals. Step four is to identify non-functional requirements for each sub goal. After identify the NFR, they used a check list to validate the approach, which includes questions such as: Does each requirement have a source? Is each requirement bounded and unambiguous? And many other similar questions. The last check is to ensure that the NFR is complete. For this check, a matrices to measure the completeness of NFR is proposed, which is MCR = n_c / [n_c+n_{nv}].Where n_c is number of correct and valid NFR and n_{nv} is number of not yet validated NFR or not correct. The closer the MCR value to 1, the closer the requirements to completeness.

Although this approach has advantages presented in layering the process of defining NFRs, but it also has a number of limitations. For example, it depends on the customer to define the NFR. But the customer doesn't always have the proper knowledge to state his/her requirements. Customers may miss some important features. Also customers are not fully aware about internal quality attributes such as maintainability or portability. Above all, this approach ignores the FRs in defining the NFRs while they are directly linked together. Because FRs define the need for certain NFRS, and similarly, meeting certain NFR may negatively impact the meeting of certain FR.

B. Quantitative Assessment of Non-functional Requirements

M. Kassab, M. Daneva, and O. Ormandjieva [14] proposed a process-oriented and qualitative decomposition approach for dealing with NFRs in requirements engineering (RE). A cornerstone of this framework is the concept of the "soft goals", which is used to represent the NFR. A soft goal is a goal that has no clear-cut definition or criteria to determine whether or not it has been satisfied. In fact, the framework speaks of soft goals being "satisfied" rather than not satisfied, to underscore their ad hoc nature, both with respect to their definition and to their satisfaction. One drawback of the soft goal approach implied in the NFR framework becomes apparent when we look at solution architecture tradeoffs.

Deploying the framework in practice implies executing the following interleaved tasks, which are iterative: (1) Acquiring knowledge about the system's domain, FRs and the particular kinds of NFRs of a particular system; (2) Identifying NFRs as NFR soft goals and decomposing them into a finer level; (3) Identifying the possible design alternatives for meeting NFRs in the target system as operationalizing soft goals; (4) Dealing with ambiguities, tradeoffs, priorities and interdependencies among NFRs and operationalization; (5) Selecting operationalization; (6) Supporting decisions with a design rationale; (7) Evaluating the impact of operationalization selection decisions on NFR satisfaction.

The approach deal with some limitations that had been shown in the previous study such as including FRs in the process.

Table 1 below summarizes the key points and the drawback of each approach.

TABLE I. SUMMERY OF THE KEY POINTS AND THE LIMITATIONS OF EXISTING APPROACHES

Study	Key points	limitations
Four Layered Approach to NFR analysis	It proposes a way to validate and measure the completenes s of NFR, and provide matrices to find the critical NFR.	-It is assumed that the customer has all the needed knowledge to tell his NFRs. - It focuses on the- - relationship between product goal and NFRs rather than relationship between FRs and NFRs
Quantitative Assessment of NFRS	Study existing interdepende ncies between the various NFRs	- this process is Only carried out informally, leaving the knowledge and the rationale that led to decisions undocumented.

IV. The Enhanced Approach

This section will propose an approach to enhance the analysis process of the NFRs, starting from elicitation of the requirements from the customer until producing a baseline NFRs document. In addition to the proposed approach validation discussion, and results.

A. Non-functional Requirements Analysis Approach

Non-Functional Requirements Analysis Approach is a process-oriented approach. The purpose of this approach is to guarantee that the NFRs are correctly defined considering all factors and different point of views. This goal can be achieved through better elicitation and analysis of NFRs in terms of internal and external quality attributes, by studying the relationship and conflicts between these attributes from one side and between the functional requirements and quality attributes from other side.

The non functional requirements identification process can be divided into below phases:

Phase 1: FRS elicitation from the customers using one or more of the techniques used to collect requirements that are suitable for the project. In this step we will focus only on collecting the FRs from the customer keeping in mind that these FRs should be aligned with the business objectives.

Phase 2: preparing mapping sheet (matrix) between FRs and external quality attributes by system developer (table II). Before the developer can give the customer this sheet, she/he should define the related external quality attributes to be included in the sheet and the other attributes that will be excluded from the sheet according to a clear justification based on the developer knowledge. This step helps taking the advantage of the developer's knowledge about technical details of the system to lead the customer through the process of eliciting the NFRs. Also this step will overcome the customer's missing knowledge.

Phase 3: Eliciting the external quality attribute. After collecting the functional requirements from customer, and preparing the mapping sheet, the customer should fill this sheet to map each functional requirement with its required external quality attributes according to customer's desired requirements. This step helps define the internal quality attribute, its values and priority.

Phase 4: Eliciting the external quality attribute. External quality attributes decided internally between project members and the customer has no input regarding them. The external attributes can be defined as technical standards.

Phase 5: Studying conflicts between internal and external quality attribute. Developers should prepare mapping sheet (matrix) between internal and external quality attributes (table III), to find out if there are any conflicts between these attributes. The aim of studying these conflicts areas is to try to find out the most important attributes that must be met and to make some balance between them.

Phase 6: Negotiating the matrixes with the customer until both (customer and developer) agrees on the final attributes that will be the baseline for NFR. Fig II below show the process workflow.

TABLE II: MAPPING BETWEEN FRs AND EXTERNAL Q-ATTRIBUTES

Functional Requirements	external quality attribute				
	Usability	Reliability	Performance	Latency	…
FUN1			√	√	
FUN2	√	√	√		
FUN3	√	√	√		
…					
Priority	1	2	3	1	

.

TABLE III: MAPPING BETWEEN EXTERNAL AND INTERNAL Q-ATTRIBUTES

Internal quality attributes	external quality attributes			
	Usability	Reliability	performance	Accuracy
Maintainability			√	
Flexibility	√			
Portability		√		
…				

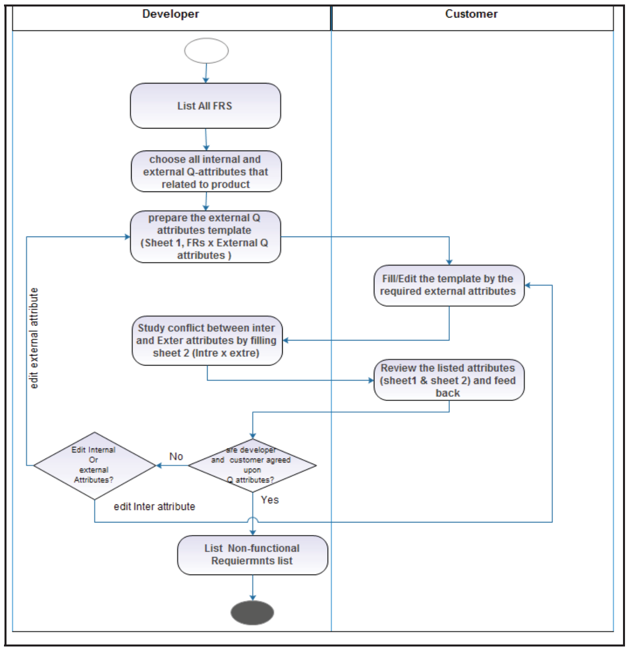
Developer
Customer
List All FRS
choose all internal and external Q-attributes that related to product
prepare the external Q attributes template (Sheet 1, FRs x External Q attributes)
Fill/Edit the template by the required external attributes
Study conflict between inter and Exter attributes by filling sheet 2 (Intre x extre)
Review the listed attributes (sheet1 & sheet 2) and feed back
edit external attribute
Edit Internal Or external Attributes?
No
are developer and customer agreed upon Q attributes?
Yes
edit Inter attribute
List Non-functional Requiermnts list

Fig II: NFRs Analysis approach workflow

B. *Approach validation*

The proposed approach is based on studying the existing approaches that were summarized in chapter two. It retains their strengths points and tries to overcome their limitations. So, to validate the new approach we made a comparison between existing approaches and the new approach according to a defined set of criteria. As shown in table IV below.

TABLE IV: NEW APPROACH VALIDATION

	Existing approaches	**New approach**
Documentation	All previous approaches carried out informally, leaving the knowledge and the rationale that led to decisions undocumented.	Provide a ready to use template to ease the analysis and treatability process and to document every step formally.
Accuracy	It focuses on the relationship between product goals and NFRs rather than relationship between FRs and NFRs.	For more accuracy, it depends on FRs from elicitation with alignment to business goals, then to study the relationship between each FRs and each NFRs using matrix. It decomposes the NFRs into internal and external quality attributes, and deals with each group according to its own specificity. It also considers the interdependencies between internal and external attributes and the need for trade-off.
Reliability	Existing approaches depends only on the customer to define NFRs. This makes the list of MFRs defined according to one point of view.	Is more reliable because both developer and customer work together to produce the NFRs, by this we involve two different point of views which result in more reliable result. Because the customer does not always have enough knowledge to define the requirements, this approach makes the developer leads the NFRs identification process.
Ease of use		It helps the customer a lot because it leads him through the entire process.

V. DISCUSSION AND RESULT

From the above comparison we can notice a number of strength points. According to applicability, the new approach gives a practical step by step method to analyze the NFRs which makes it better than the "Effect of the NFRs on the structural architecture" approach. According to documentation, all previous approaches carried out informally, leaving the knowledge and the rationale that led to decisions undocumented while the new approach provide a ready to use template to ease the analysis and treatability process and to document every step formally. Accuracy wise, the new approach depends on FRs that are defined with alignment to business goals, to define the NFRs, then to study the relationship between each FRs and each NFRs using a matrix. Also it decomposes the NFRs into internal and external quality attributes, and deals with each group according to its own specificity. It also consider the interdependencies between internal and external attributes and the need for NFRs trade-offs and prioritization. The new approach provides iteration process of editing, reviewing, and then feedback between the customer and the developer until they both agree on critical attributes. While the previous approaches, don't support the iteration concept. In addition to considering the customer and the developer point of view, this makes it more reliable.

Because the customer alone may not always have enough awareness to define the requirements, and usually focus on the external quality (Usability, Efficiency, Reliability, Integrity, Adaptability, and Accuracy), and they don't really care about the internal quality (Maintainability, Flexibility, Portability, Re-usability, Readability, and Testability) of the system, this approach makes the developer and customer work together to define the NFRs, prioritize them and make critical decisions. The developer's role is actually to lead customer through all these process.

As a result of all the above points and according to the defined criteria that used in the comparison, using this approach is enhancing the process of NFRs analysis because it overcome a major part of the previous approaches limitations and keeps their strengths point.

VI. CONCLUSION

Non-functional requirements are the differentiating factor from other products that provide similar functionality, and the success of any software product in the market depends on how this software is meeting the non-functional expectations. As there are not so many approaches in Agile that deal with the non-functional requirements gathering, analysis and documentation, this thesis introduced an enhanced NFRs analysis approach by merging the strength points of existing approaches that summarized in the literature review and overcoming their limitations. The new approach pays attention to customer knowledge in defining requirements, decomposes NFRs to internal and external quality attributes and defines and analyses each group according to its own specificity using pre-defined matrix. Also it provides a ready to use template to document the analysis and prioritization process. SOA impact quality attributes. Some impact is positive like in Interoperability and Reliability, some other are negatively impacted like in Usability, Security, Performance, and Testability. While there are attributes that need more techniques like Availability and Usability.

VII. FUTURE WORK

For future work, this approach can be applied in real projects and then enhanced even more according to the results. Also, future work can include further investigation in the software quality attributes to analyze trade-offs between multiple conflicting attributes to satisfy user requirements. Regarding provided template, it can be developed as a system rather than just a template.

REFERENCES

[1] Product Line Practice, Version 5.0", [Online]. Framework for Software Product Line Practice, Version 5.0. Available: http://www.sei.cmu.edu/productlines/frame_report/ [Accessed: Nov. 12, 2014]

[2] Mario Cardinal , "*Addressing Non-Functional, Requirements with Agile Practices*",[online document], 2014.Available: Agile tour, http://at2015.agiletour.org/ [Accessed: Nov. 12, 2014]

[3] Rao, A. Ananda, and Merugu Gopichand. "Four layered approach to non-functional requirements analysis." arXiv preprint arXiv:1201.6141 (2012).

[4] Batool, Asma, et al. "Comparative study of traditional requirement engineering and agile requirement engineering." *Advanced Communication Technology (ICACT), 2013 15th International Conference on*. IEEE, 2013.

[5] Asghar, Sohail, and Mahrukh Umar. "Requirement engineering challenges in development of software applications and selection of customer-off-the-shelf (COTS) components." *International Journal of Software Engineering* 1.1 (2010): 32-50.

[6] Inayat, Irum, et al. "A systematic literature review on agile requirements engineering practices and challenges." *Computers in Human Behavior* (2014).

[7] Westfall, Linda. "Software requirements engineering: what, why, who, when, and how." Software Quality Professional 7.4 (2005): 17.

[8] Sean Isserman, "Challenges of Requirements Elicitation", Louis University of Missouri–St. Louis . [online document], 2001. Available: http://www.umsl.edu/~sir3b/Sean_Isserman_Requirements_Elicitation_Home.html

[9] Karl E. wiegers, *Software Requirements*, 2nd ed , Washington: Microsoft Press, 2003.[e-book] Available: http://www.amazon.com/Software-Requirements-2-Karl-Wiegers/dp/0735618798#reader_0735618798

[10] B.Thomas. "Meeting the challenges of Requirement Engineering." News at Software Engineering Institute. 2009. Website: http://www.sei.cmu.edu/library/abstracts/news-atsei/spotlightmar99pdf.cfm. Access date: March 2015.

[11] Asghar, Sohail, and Mahrukh Umar. "Requirement engineering challenges in development of software applications and selection of customer-off-the-shelf (COTS) components." International Journal of Software Engineering 1.1 (2010): 32-50.

[12] Barbacci, Mario, et al. *Quality Attributes*. No. CMU/SEI-95-TR-021. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 1995.

[13] Sillitti, Alberto, and Giancarlo Succi. "Requirements engineering for agile methods." Engineering and Managing Software Requirements. Springer Berlin Heidelberg, 2005. 309-326.

[14] Kassab, M., M. Daneva, and O. Ormandjieva. "Early quantitative assessment of non-functional requirements." (2007).