A Quality Attributes Evaluation Method for an Agile Approach

Taehoon Um, Neunghoe Kim, Donghyun Lee, Hoh Peter In*
Department of Computer Science & Engineering, Korea University
Seoul, South Korea
{ompa, nunghoi, tellmehenry, hoh_in}@korea.ac.kr
*Corresponding Author: Hoh Peter In

***Abstract*—Rapid change of customer requirements has propelled the emergence of an agile approach due to such merits as fast release and simplified documents. An agile approach aims at maximizing productivity and effectiveness. Participants in an agile project usually focus on functionality of a system during the process. However, there are few considerations for non-functional requirements (or quality requirements) such as performance, security and accuracy. This causes huge costs, and wastes efforts for later changes. A novel approach is required to strengthen non-functional aspects in an agile approach. We propose a lightweight quality evaluation method that reflects quality attributes to enhance non-functional features in an agile approach. Our approach enables iterative evaluation of quality attributes in each release. It supports participants to continuously consider quality issues. In addition, participants are able to make plans on quality improvements.**

Keywords: agile approach, quality attributes, quality attributes evaluation method, non-functional aspects

I. INTRODUCTION

As customer requirements are constantly changed, an agile approach has been recently emerging, since it is possible, with the methods, to quickly satisfy customer needs. An agile approach aims at fast delivery through minimization of documents and reduction of unnecessary practices under tightened time constraints. Stakeholders of an agile project usually concentrate on software functionality. Since functionality is focused on, critical non-functional requirements are likely to be ill-defined and neglected [1][2]. Existing agile approach is capable of satisfying customers in the short run. If changes ever occur as to quality requirements in the latter half of the development cycle, however, change costs become tremendous, compared with those related to functional requirements [3]. Most of the previous researches on quality issues of an agile approach mainly focused on code quality or on reformation of agile practices to maximize productivity. However, those studies are not about non-functional aspects (or quality attributes: QAs). Thus, a novel approach is required to enhance non-functional aspects by means of quality attributes. Although the ISO 9126 series provides quality measurement metrics and various quality evaluation guidelines on a general software project, these approaches are on the basis of well-defined documents such as software requirement specification and test report. Hence, the ISO 9126 series is inappropriate for an agile approach, which requires minimal documents. We suggest a lightweight quality evaluation method that requires a handful of documents. Also, the proposed method helps participants identify quality issues in the early phase of a project, using categories of quality attributes. Finally, participants are able to obtain feedbacks for quality improvement. The overall process of evaluating quality attributes is explained in Section 2.

II. EVALUATION OF QUALITY ATTRIBUTES IN AN AGILE PROCESS

We adopted the following 6 major QAs from ISO 9126-1 [4] as main characteristics for practical evaluation:

Major Quality Attributes:

Functionality, Reliability, Maintainability, Usability, Efficiency, Portability

Figure 1 depicts how our proposed model evaluates quality attributes in an agile approach. Firstly, critical QAs of a system are defined (S1), and then prioritized (S2). S1 and S2 are executed in the early phase of a process. As the development cycle gets started, quality requirements are elicited from related functional requirements (S3). Then, QAs are evaluated by quality requirements (S4). Lastly, feedbacks are obtained from evaluation results and reflected in the next release (S5). S3, S4 and S5 are repeatedly executed until a final product is released. Each step of QA evaluation is explained below.

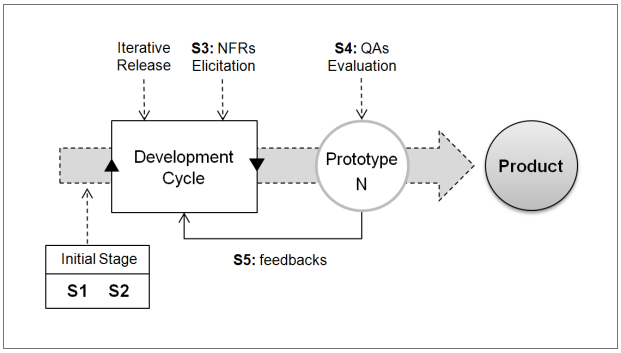

Figure 1. Iterative Quality Evaluation in an Agile Process

978-0-7695-4417-5/11 $26.00 © 2011 IEEE
DOI 10.1109/CNSI.2011.93

(1) Identifying Critical QAs: S1

Participants in an agile project identify QAs of a system. To do this, a team meeting or QAW [5] is recommended. Participants define the most significantly-considered QAs, and also define sub QAs. Each QA consists of several sub QAs. For instance, adaptability, installability and replaceability are sub QAs of portability. QAs are introduced in the quality model of the ISO 9126-1 [4]. We propose use of a story card consisting of QA categories to reflect QAs and sub QAs. The proposed story card is shown in Figure 2. Identified QAs are written on the quality category.

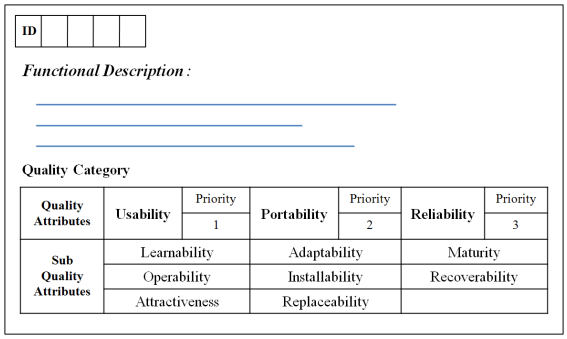

Figure 2. QA-based Story Card

In addition, participants set up the 'Required Level,' which functions as a target goal in assessing completeness of each sub QA. It is used in S5 to check how QAs are satisfied in each release.

(2) Prioritizing QAs: S2

The identified QAs are prioritized by their importance. The importance is defined, based on characteristics of a system to be developed or a goal of an organization. The priority of QAs affects the implementation order of quality requirements in the next release. Quality requirements related to the most critical QAs are implemented earlier, based on the relevant priority.

(3) Eliciting Quality Requirements Based on QAs: S3

A story card serves as a tool for eliciting additional requirements from functional requirements through communication among stakeholders. Prior to evaluating QAs, participants also use a story card to elicit quality requirements related to QAs. Sub QAs help participants effectively elicit quality requirements with specific characteristics. At the beginning of the development cycle in each release, quality requirements are elicited from related functional requirements written on the story card. The elicited quality requirements are added to an evaluation list.

(4) Evaluating Quality Requirements: S4

The quality evaluation is executed with identified QAs. At the end of the development cycle in each release, executable prototypes are produced and used for evaluation in place of well-define documents. We suggest a 2-way approach below to assess QAs:

- Customer's View: Basically, an agile approach is customer centric. Customers should be involved in QAs evaluation. A customer uses executable prototypes to check how much he or she is satisfied with implementation of quality requirements.
- Developer's View: Developers check the implementing progress of quality requirements. Developers should also take into account the seriousness of errors and the impacts of anomalies as evaluation criteria.

Participants evaluate quality requirements with their own views (a customer and a developer). Then, evaluation results are written on the 'Current Status' category.

(5) Obtaining Feedbacks from Quality Evaluation: S5

Through comparison of the 'Required Level' with the 'Current Status,' participants figure out which QAs are improved. With these evaluation results, participants obtain feedbacks, and build up strategies on how to enhance quality in the next release.

III. CONCLUSION

We proposed a lightweight quality evaluation method for an agile approach to reflect non-functional aspects. Our approach supports early identification of non-functionality, and helps participants consistently keep concentrating on quality attributes. Participants obtain feedbacks for the next release by the evaluation results of indicating unsatisfied quality attributes in each release. Furthermore, participants are able to make plans on quality improvement. However, participants have their own criteria when they conduct evaluation with prototypes. Therefore, the proposed evaluation method could be subjective. As a result, it is needed to create a quantitative evaluation model, using measurement metrics to help participants have more trust in the results.

ACKNOWLEDGMENT

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency (NIPA-2011-(C1090-1131-0008); and was also supported by Mid-career Researcher Program through NRF grant funded by the MEST (No.2010-0000142). For more information, contact Hoh P. In at hoh_in@korea.ac.kr

REFERENCES

[1] Lan Cao, B. Ramesh, "Agile Requirements Engineering Practices: An Empirical Study", IEEE Software, Jan / Feb 2008

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

[3] Len Bass, Paul Clements, Rick Kazman. "Software Architecture in Practice - Second Edition", pp.71

[4] ISO / IEC 9126-1.2: Information Technology - Software Product Quality - Part1: Quality Model 1998

[5] CMU / SEI TR-016, Quality Attribute Workshops(QAWs), Third Edition, Aug 2003