How Practitioners Manage Quality Requirements in Rapid Software Development: A Survey

Lidia López
Universitat Politècnica de Catalunya
Barcelona, Spain
llopez@essi.upc.edu

Jari Partanen
Bittium Wireless Ltd
Oulu, Finland
Jari.Partanen@bittium.com

Pilar Rodríguez
University of Oulu
Oulu, Finland
pilar.rodriguez@oulu.fi

Silverio Martínez-Fernández
Fraunhofer IESE
Kaiserslautern, Germany
silverio.martinez@iese.fraunhofer.de

Abstract— **Software quality is an essential competitive factor for the success of software companies today. Increasing software quality levels of software products and services requires an adequate integration of quality requirements (QRs) in the software life-cycle, which is still scarcely supported in current rapid software development (RSD) approaches. RSD refers to the organizational capability to develop, release, and learn from software in rapid cycles. Q-Rapids (Quality-aware Rapid Software Development) research project aims to support decision-makers for QRs management in RSD. The goal of this survey is to explore how software development organizations using RSD manage QRs. The survey focuses on: (a) how data is gathered for assessing decisions related to product evolution, (b) how QRs are managed, and (c) the considered QRs in their products. We received 30 responses, from them we can conclude that most of the companies gather data both automatically and manually; QRs are managed together with functional requirements, sometimes or very often functionality gets priority over quality; and, the most reported QRs are reliability, performance, and security.**

Index Terms— **Quality Requirements, Non-functional Requirements, Agile, Rapid Software development, Survey**

I. INTRODUCTION

Software quality is an essential competitive factor for the success of IT companies today [1]. Increasing the software quality levels of software products and services requires an adequate integration of quality requirements (QRs) in the software life-cycle. However, QRs management is problematic in software development in general [2] and in rapid software development[1] (RSD) in particular [3]. As a response, the software engineering community has focused on the huge amount of data available about the software product, process and usage in order to perform software analytics detecting essential problems in real-time that can guide the management of QRs [4].

In order to understand how decision-makers manage QR management in RSD, we conducted a survey study to understand how QRs are managed in practice. We applied the GQM

---

[1] RSD refers to the organizational capability to develop, release, and learn from software in rapid cycles.

approach [6] to define the focus of our study. The goal of our survey is to *understand* (purpose) *the current way of working related to quality requirements management* (object) *from the point of view of practitioners* (viewpoint) *with practical experience in rapid software development* (context). This paper reports initial results of this survey.

This research falls into the Q-Rapids project (Quality-aware Rapid Software Development). Q-Rapids is a European research H2020 project, which defines an evidence-based, data-driven quality-aware rapid software development methodology in which QRs are incrementally elicited, refined and improved [5]. The Q-Rapids approach is based on data gathered from software repositories, project management tools, system usage and quality of service, and includes tool support [7].

Section II summarizes the research method. Section III reports the results of our study, which are discussed in Section IV. Finally, Section V presents our conclusions and a summary of our plans for future work.

II. RESEARCH METHOD

Conducting a survey is a complex activity that is endangered by many factors. Therefore, several authors have proposed methodologies with well-defined steps and protocols to reduce the likelihood of such dangers. In this survey, we have adopted the methodology proposed by Kasunic [8] that organizes the process for designing a questionnaire-based survey around the following seven steps: *identify the research objectives*, *identify and characterize the target audience*, *design the sampling plan*, *design and write the questionnaire*, *pilot the questionnaire*, *distribute the questionnaire*, and *analyse the results and write the report*.

A. Research Objectives

In order to achieve our main goal "to *understand the current way of working related to quality requirements management from the point of view of practitioners with practical experience in rapid software development*", in the context of Q-Rapids project, we identified the following three sub-goals:

- **G01:** *Identify* (purpose) *the current use & satisfaction* (focus) *of the data gathered from participants' project and processes* (object) *from the perspective of Enterprise architects, Product Manager, Product Owner* (viewpoint) *in the context of monitoring and supporting decisions related to product evolution* (context).
- **G02:** *Identify the current use & satisfaction of the participants' processes/practices/tools from the perspective of Enterprise architects, Product Manager, Product Owner, Risk Manager, Requirements Engineer, and Software Developers in the context of processes for managing QRs.*
- **G03:** *Identify the QRs considered in the participants' products from the perspective of Enterprise architects, Product Owners, Requirements Engineer.*

B. Target Audience

According to the research goals, the population is determined by the different roles described as viewpoint from organizations developing software products using RSD:

- *Enterprise Architects.* As responsible for the organisation's processes, they have information about the quality management processes.
- *Product Managers.* They have the responsibility of monitoring products' quality.
- *Product Owners.* They are involved in requirements prioritisation.
- *Requirements Engineer.* They are responsible of managing Quality Requirements.
- *Software Developers.* They are mainly the producers of the gathered data. They could visualise the impact of their work in the products' quality.

C. Sampling Plan

We used a convenience sample based mainly on the project partner's network to disseminate the survey. Q-Rapids consortium is composed by two Universities (Universtitat Politècnica de Catalunya and University of Oulu), one research institute (Fraunhofer IESE) and four industrial partners (Bittium, Softeam, Nokia, and iTTi) from five different countries (Spain, Finland, Germany, France, Poland).

D. Design and write the questionnaire

This survey was part of a broader survey as part of the Q-Rapids project's exploitation activities. From the exploitation point of view, the survey was conceived for identifying potential users of Q-Rapids. As the target audience was the same, we designed a survey combining both goals. The initial questionnaire contained a total of 39 questions (before piloting).

During the design, we identified the most relevant threats of internal, external, and construct validity as defined in [9]. They are described at the end of this section.

E. Pilot the questionnaire

Before opening the survey, we piloted the survey in two phases. First, we piloted the questionnaire internally, with the result of 23 questions selected and an optimised distribution of the questions. For the second pilot, we involved one person of each industrial partner belonging the project and two people external to the project. Finally, we ended up with the result of a final set of 26 questions[2].

F. Distribute the questionnaire

The electronic survey was open on July 2017. We mainly used the network of Q-Rapids to increase participation by direct e-mail. Additionally, we designed a poster that we used for in scientific venues such as PROFES 2017. The survey was implemented using the electronic survey tool LimeSurvey[3].

G. Threats to Validity

Internal validity is related to the extent to which other confounding aspects may influence the relationships that are identified in the study. We tried to make the survey as short as possible and we made all the questions optional to answer to minimise mortality. The sample was not selected randomly, but we tried to minimize the bias in the selection making the selection from all the partners' networks (universities, research institute, and industrial partners from five different countries).

External validity is related to the fact of generalizing the results to the target population. The sample was not selected randomly, thus, there is no statistical basis for generalising the results.

Construct Validity refers to the degree which the survey is capturing what is expected. In order to assure that we are asking what we needed, we defined the concrete set of goals to identify the concepts of interest. This helped us to have a specific idea of what we wanted to measure, minimising the fact that we can forget some questions and we are sure that the questions we have are measuring the concepts we wanted to measure (GQM+ approach). To minimise the bias that could have introduced by the knowledge we have about the project, we included in the piloting phase some people from outside the project consortium without any knowledge of the project.

III. Results

We obtained 30 responses to the survey. The questions were optional, leading some no-answer category in the results.

A. Respondents Characterisation

The experience of the participants in RSD is diverse, 47.6% more than seven years, 23.8% from four to six, and 28.6% less than three. Therefore, most of the participants have high experience in RSD. Referent to requirements management, 42.9% have more than ten years, 4.8% from six to ten, 19% from one to five, and 33.3% less than one year or no experience.

Two-thirds of the respondents (66.7%) use an exclusive development methodology (agile, waterfall or mixed). 61.9% considered that they use agile methodologies whereas few (4.8%) consider that they are using waterfall. Although the respondents are not using mixed methodologies, when we asked for mixing agile and waterfall, the answers related to agile/rapid concrete methodologies show that they use mixed

[2] http://www.essi.upc.edu/~qrapids/Q-Rapids_Survey.pdf
[3] http://www.essi.upc.edu/~e-survey/index.php?sid=38773

agile/rapid methodologies (Kanban; Scrum; Continuous delivery/Rapid releases; and DevOps). The majority uses more than one methodology (64%) and 32% of them mixes three. It is worth to mention that four respondents (18%) selected all the agile/rapids options provided (see Fig. 1). Only one respondent selected the others option answering "*Firm-specific agile and incremental*", not combined with any of the others.
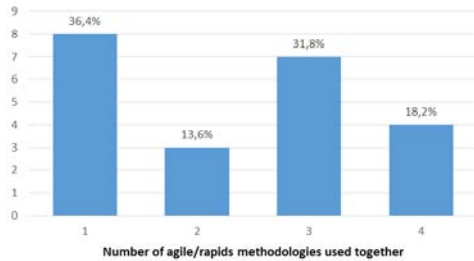

Fig. 1. How many agile/rapid methodologies lead your development process?

B. Data gathering (GO1)

We asked the respondents if they gather data automatically or manually for assessing quality (see Fig. 2). 67% of them reported a hybrid approach, gathering data both automatically and manually. However, organizations gathering most of the data automatically (38%) are greater than those gathering most of the data manually (29%). Organizations gathering all the data automatically are marginal (3%).
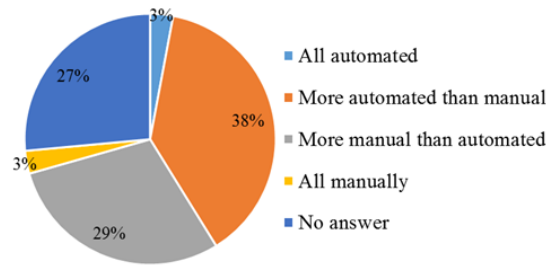

Fig. 2. Which statements describes better your data gathering approach?

In order to identify what kind of data can be used for taking decisions related to the QRs management, the respondents were asked for the data sources they considered. The data sources having more impact are *issue tracker systems*, *product behaviour*, *quality of service*, and *test tools*, being characterised mostly as "very much" (in an ordinal scale of marginally, slightly, somehow, significantly, and very much). The data sources considered as "somehow" affecting are *software code repositories*, *project management tools*, *product usage*, and *static code analysis*. The least considered sources, characterised as "marginally" or "slightly", are *documentation systems* and *communication tools*.

Table I provides some examples of the statements reported by respondents when we asked for examples of data gathered, which could help them to take their decisions. Some of the answers, not only include the data but also the purpose of the data, e.g. "*Team velocity is manually estimated and taken into account to prioritize stories*" or "*We manually review each other's code as part of the development pipeline*".

TABLE I. EXAMPLES OF DATA GATHERED

Data source	-Example
Issue trackers	Number of bugs, Number of critical open errors
Product behaviour	Logs are used for evaluate the user experience
Quality of service	Quality of service is key issue considering product usability
Test tools	The automatic deployment stops if any test fail
Code Repositories	Number of external components and LOC to understand the integration complexity
Project Management	Team velocity is manually estimated and taken into account in order to prioritize stories
Product usage	Feature usage
Code analysis	Found errors and too complex code should block integration to prevent contaminating mainline code

C. Quality Requirements management process (GO2)

When the respondents were asked about processes for managing QRs (see Fig. 3), most of them indicated that they follow some quality requirement process (50%), systematic and well defined or ad-hoc processes. Only one respondent selected the "Others" option, detailing that "*We have firm-specific practices that are not ad-hoc/unstructured, but don't necessarily follow any specific commonly-known methodology*".
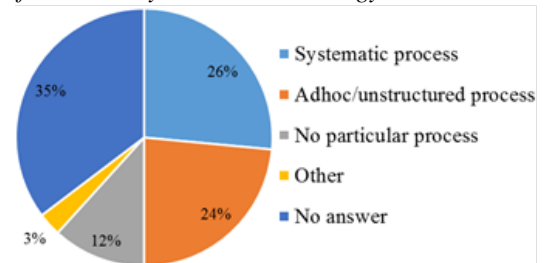

Fig. 3. Which statement describes better your QR management process?

Regarding how the functional and non-functional requirements (QRs) are treated, the majority of the respondents manage functional and non-functional together (41%). Only one respondent selected the option "Others", detailing that "*Depends on the quality attribute. Some of them are cross-cutting and, hence, managed separately. Others are specific to a functionality and managed as part of it*".

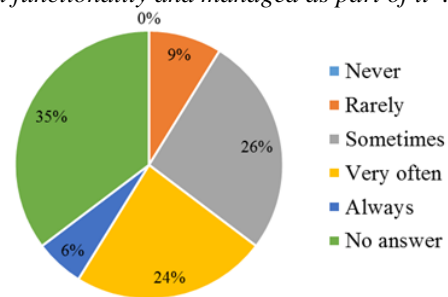Fig. 4. Functionality is first, quality comes later statement

D. Quality Requirements (GO3)

For the questions related to specific quality requirements we used open format question (i.e. free text).

For the question of quality requirements considered in the respondent's products, the quality requirements more reported are *reliability* (21%), *performance* (15%), and *security* (12%). Followed by *Customer/user experience*, *customer service*, and *scalability* (6%).

The survey includes a question asking explicitly for concrete quality requirements difficult to manage, the most mentioned is *usability*, some of the reasons are "*what is easy for me, might not be easy for the end user*" or "*Because it consumes many resources (time, financial, humans)*". *Maintenance*, *extensibility*, *reliability* and *scalability* are also mentioned. Some of the answers were not related to concrete quality requirement, but to difficulties that can apply to some of them, e.g. "*Extreme quality requirements (e.g. extreme security requirements, very high availability requirements, etc)*" or "*Following trend of things that require manual input (e.g. field testing) or complex system to automate data gathering (e.g. power consumption testing)*".

When we asked for easy-to-manage requirements, only *maintainability* and *customer satisfaction* have been answered explicitly. Most of the answers refer to requirement that can be easily monitored, e.g. "management *is easy for requirements of which follow up is integrated in an issue tracking system, test management tool or project management tool*" or "*Requirements you can automatically test*".

IV. DISCUSSION

Gathering data in RSD is a common practice in industry. 70% of the respondents gather data automatically and/or manually. The nature of these software development approaches, based on incremental and continuous development, contributes to produce huge amount of data, which properly used can help decision makers to guide software development, including decisions about product quality. Still, according to our survey results, despite companies consider useful to gather such data, many companies collect data manually.

Functionality is more important than QRs in real projects. The research community could support organizations to understand the impact of not considering QRs at early stages. QRs are managed together with functional requirements, thus, according to our results there is no separate process for QRs. Taking into account the nature of QRs and the fact that they are managed jointly to the functional requirements, their management is complex. This can be the reason because the management of QRs is not often well-structured or systematic. In the research community there is plenty of knowledge related to QR management, researchers should focus on how to transfer it.

The QRs reported can help the researchers to identify which QRs needs special attention. The ones more reported as considered in the organization products (reliability, performance, and security), and the ones reported as difficult-to-manage (usability). There is an interesting topic mentioned by one respondent "extreme quality requirements" (e.g. extreme security or very high availability requirements.

V. CONCLUSIONS AND FUTURE WORK

We presented the results of a survey about how practitioners manage quality requirements in the context of rapid software development.

The more remarkable results are that organizations gather both automatically and manually data to monitor and assess quality. The data sources affecting in a greater extent to quality are issue tracking systems, product behaviour sources, quality of service, and test tools. Quality requirements are managed together with functional requirements following systematic, well-defined or ad-hoc processes. Moreover, sometimes or very often functionality gets priority over quality. The most reported QRs are reliability, performance, and security. The QRs reported as difficult-to-manage are usability, maintenance, extensibility, reliability, and scalability. And easy-to-manage requirements maintainability and customer satisfaction.

For complementing these results about how practitioners manage quality requirements, we are conducting a systematic literature review (SLR) covering the reported research. We have also surveyed (in this survey and in the SLR) about challenges in managing quality requirements. Combining the results of both studies we will have a broader view of how quality management can be managed.

REFERENCES

[1] Capgemini: World Quality Report 2015-16, 7th Edition. Available at https://www.capgemini.com/thoughtleadership/world-quality-report-2015-16

[2] B. Nuseibeh, S. Easterbrook, "Requirements engineering: a roadmap", in ICSE 2000, pp. 35-46.

[3] E.M. Schön, J. Thomaschewski, M.J. Escalona, "Agile requirements engineering: a systematic literature review", in *Computer Standards & Interfaces*, 2017, Vol. 49, pp. 79-91.

[4] Y. Yang, D. Falessi, T. Menzies, J. Hihn, "Actionable Analytics for Software Engineering", in *IEEE Software*, 35(1), pp. 51-53.

[5] L. Guzmán, M. Oriol, P. Rodríguez, X. Franch, A. Jedlitschka, M. Oivo, "How Can Quality Awareness Support Rapid Software Development? - A Research Preview", in REFSQ 2017, pp.167-173.

[6] L. López, S. Martínez-Fernández, C. Gómez, M. Choras, R.l Kozik, L. Guzmán, A. M. Vollmer, X. Franch, A. Jedlitschka, "Q-Rapids Tool Prototype: Supporting Decision-Makers in Managing Quality in Rapid Software Development", in. CAiSE Forum 2018, pp. 200-208.

[7] V. Basili, G. Caldiera, D. Rombach, "The Goal Question Metric Approach", in: J.J. Marciniak (Ed.) *Encyclopedia of Software Engineering*, John Wiley & Sons, New York, 1994, pp. 528-532.

[8] Marc Kasunic, "Designing an Effective Survey". Report at Carnegie Mellon University, ref. CMU/SEI-2005-HB-004 (2005).

[9] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, A. Wesslen, "Experimentation in Software Engineering." Springer, 2012.