Proactive Reviews of Textual Requirements

Vard Antinyan  
Computer Science and Engineering  
University of Gothenburg  
Gothenburg, Sweden  
vard.antinyan@cse.gu.se

Miroslaw Staron  
Computer Science and Engineering  
University of Gothenburg  
Gothenburg, Sweden  
miroslaw.staron@cse.gu.se

Abstract—In large software development products the number of textual requirements can reach tens of thousands. When such a large number of requirements is delivered to software developers, there is a risk that vague or complex requirements remain undetected until late in the design process. In order to detect such requirements, companies conduct manual reviews of requirements. Manual reviews, however, take substantial amount of effort, and the efficiency is low. The goal of this paper is to present the application of a method for proactive requirements reviews. The method, that was developed and evaluated in a previous study, is now used in three companies. We show how the method evolved from an isolated scripted use to a fully integrated use in the three companies. The results showed that software engineers in the three companies use the method as a help in their job for continuous improvements of requirements.

Keywords—metric, measure, requirement review, technical risk

I. INTRODUCTION

A central activity in requirements engineering is reviewing of textual requirements. Software design without requirements’ reviews triggers technical risks of late design modifications, which takes a considerable amount of the cost of development. Manually reviewing of requirements is one technique that software engineers use nowadays to mitigate risks [1], [2], however, manually reviewing itself is a tiresome and effort-intensive activity [3], [4]. Particularly, large software development companies increasingly rely on continuous software engineering principles, where all the activities are carried out in a continuous fashion, and thus companies do not want to disrupt the continuity of development by a lengthy review process.

When identifying challenges associated with large number of requirements, Kauppinen, et al. [5] reports that the engineers in industry find the tool support has a crucial role for an efficient reviewing process. The need for tool support and automation come naturally as the amount of requirements grows. There could be three major benefits by using automated reviews:

1. Substantially reducing effort spent on reviewing requirements
2. Mitigating technical risks of late design modifications
3. Getting feedback “just in time” of writing requirements, in order to facilitate the improvement process

There are a few studies, which have created methods for automating requirements’ review process. Known examples are [6] and [7]. These studies could achieve remarkable results in terms of assessing textual requirements’ quality to a good precision. However, since textual requirements are sophisticated constructs, the adequacy of measures from company to company can be different. Additionally, practitioners need help in understanding how the measures can be integrated in their “ways of working” so the reviewing process can be enhanced as much as possible.

Two years ago the authors of this paper also conducted research to create measures for textual requirements. The aim was automating reviews in three large companies [8], [9]. Achieving encouraging results the collaborating companies aimed at creating integrated measurement systems, in order to automate the reviewing process. At the same time, we (researchers) found that it would be valuable for practitioners to know the challenges and success factors when integrating an automated method within the software engineers’ working environment. Pursuing this aim, we conducted a follow-up study presented in this paper. The research question we address is:

How can we proactively review textual requirements in practice?

The key contribution of this study is to present how our previous method was integrated and used in three large companies. Action research method was applied to address the research question. After the adoption and use of the method for several months, we got mainly positive feedback from software engineers. An experienced software engineer from Volvo Cars Group had the following remark: “the use of this tool can save us hundreds of hours per year”.

If the reader might wonder why this evaluation results were not reported in the previous study, we inform that we needed approximately two years of time to adopt and observe the use of the method in the companies.

II. A SUMMARY OF THE PREVIOUS WORK

In this section we present a brief overview of the four measures and measurement based method that we created in our previous work for requirements’ internal quality assessment. Table 1 presents the name and definitions of the four measures of textual requirements.
The method, Rendex, is based on the four measures presented in the table. The method defines a single internal quality index ($Q_{IR}$) for a requirement and then measures the internal quality indices for all requirements. $Q_{IR}$ represents the weighted sum of the four measures. It is calculated as follows:

$$Q_{IR} = NC + NV + NR + 5NRD$$  \hspace{1cm} (1)

Formula (1) is derived from regression analyses in three companies. It is the best generic combination of the four measures to assess the requirements quality in companies. The $Q_{IR}$ number, which we get for a requirement, does not show any absolute value of quality, it rather shows whether or not one requirement has better quality than another requirement.

### III. COLLABORATING COMPANIES

The three collaborating companies were Grundfos, which produces pumps for variety of purposes, Volvo Car Groups, which manufactures cars, and Volvo Group, which manufactures trucks and construction equipment. All of the companies develop electronic control units for their products which were empowered by software functionality. All of the companies had thousands of text-based requirements for their software.

### IV. RESEARCH METHOD

This study employed action research principles [10] adjusted for information sciences as described by Baskerville [11]. In order to understand the most beneficial way that the method could be used in the collaborating organizations, we formed reference groups in each of the organization. The reference groups were composed of 4-5 software engineers to discuss the choices that we made for using the method. Figure 1 presents the action research cycle applied in each of the organizations. The detailed description of the typical five consecutive steps in the action research cycle is as follows:

1. In the inception of the research we wanted the method to be integrated and used in the most effective and convenient way.
2. In the next step we discuss possible alternatives for using Rendex with the reference group engineers. We understood engineers’ needs in review process, ways of working, organizational habits, tooling solutions, and common issues. We then proposed alternative solutions and discussed them. The best alternatives were chosen for the initial application in the organization. In the third step the reference group engineers deployed the method or modified according the feedback of the previous action research cycle.
3. In the fourth step researchers and reference groups met to evaluate how the alternative uses of the method are received by the reference group engineers and the rest of the organization.
4. In the last step we examined and documented the success stories and challenges.

![Figure 1 Action research cycle applied in the collaborating organizations](image-url)

### V. ADOPTING RENDEX IN THREE ORGANIZATIONS

In this section we describe how Rendex was adopted for organizational use in the three companies.

#### A. Using Rendex in VG and VCG

In these two companies initially the method was used by individual engineers for generating quality reports on requirements. Typically an independent script was run on the target set of requirements, which generated a report, where the requirements were ranked by the quality index $Q_{IR}$. Then a responsible person for the requirements quality took the report, select requirements according to development areas, and sent them to the engineers that are responsible for given areas.
Even though this approach was better than the old ways of working with manual reviews, we understood that there are two major issues:

- The feedback that requirements’ analysts got was not ultimately proactive, because it was up to the assigned person to generate and send the reports.
- Some of the requirements analysts felt that the report is a control mechanism over their ways of working, which made them unhappy with the solution.

To this end we decided to integrate the method within the RMS, and recommend engineers to use the method by themselves.

In case of VCG the RMS was developed by a supplier, so we needed to negotiate the development with them. Here the problem was that if the method is integrated inside the system, the researchers cannot tune the measurement method in the later research. So the suppliers developed an interface for the method inside the system, however, the system needed to call a locally developed script, in order to conduct the measurement. The script could be modified by us, the researchers, but the presentation of measures was not possible to change.

Figure 2 presents a snapshot of requirements in the requirements’ management system (RMS) at VCG. Every “REQPROD” item in the figure is a requirement. When clicking on any of the requirements, we can get the requirement’s text and all the related information. In the lower part of the figure we can see that there are several attributes of the requirement, such as “owner”, “outbox”, “version comment”, etc. Among this attributes we can find “quality analysis” attribute. Clicking on it we get the analysis results below. The analysis also provides a general quality index based on formula (1), which is outlined with a dashed line on the lower left side of the figure.

We were also asked to prepare a wiki page with brief information about what measures indicate and what the decision criteria were. At this point, according to the engineers of reference group, most of their colleagues were happy with the designed solution and considered the tool helpful in their daily job. The tool provided instantaneous feedback on requirements internal quality and no one controlled their work.

At VG we had a similar scenario. The only difference was that VG used a locally developed RMS so the integration of the method was much easier.

A snapshot of the measurement is presented in Figure 3. Requirements and their QI\(_R\)-s are displayed in the RMS in the left side of the figure. In the right side of the diagram we can see the requirements’ names and QI\(_R\)-s per requirement. Since engineers found it somewhat difficult to interpret QI\(_R\)-s we defined decision criteria for them. Based on our experience of a year, we had learned some approximate thresholds, which can be useful in practice.
Table 2 presents the defined thresholds and corresponding indicators for action taking. The indicators are colour-coded fields for every requirement. Green and yellow indicate “absolutely no problem” and “minor issues” correspondingly. Depending on how much effort the organization can allocate for reviewing and improving requirements, red and orange distinguish “the most urgent” and “urgent” needs for improvements correspondingly.

<table>
<thead>
<tr>
<th>QI₆ threshold</th>
<th>Indicator</th>
</tr>
</thead>
<tbody>
<tr>
<td>QI₆ ≤ 3</td>
<td>Green</td>
</tr>
<tr>
<td>3 &lt; QI₆ ≤ 7</td>
<td>yellow</td>
</tr>
<tr>
<td>7 &gt; QI₆ ≤ 12</td>
<td>Orange</td>
</tr>
<tr>
<td>QI₆ &gt; 12</td>
<td>Red</td>
</tr>
</tbody>
</table>

Later the engineers found that color coding over the entire view of requirements is disturbing. Hence, we removed color coding for the whole view and left it only for individual requirements.

B. Using Rendex in Grundfos

Before introducing Rendex to engineers of Grundfos, they had already started a massive process of requirements improvements. Their aim was to define a complete list of patterns for requirements, so that any requirement must fulfil one of the predefined patterns.

Their wish was to find such requirements that do not comply with the predefined patterns. They were developing an automation tool for indicating whether a requirement complies with predefined patterns. When introducing Rendex, engineers of the reference group found that using two of our measures is helpful in understanding whether a requirement does not comply with predefined patterns. The two of the measures were NC and NV. Our measures were strongly related to the way they define the requirements’ patterns. For example, one of the rules for defining requirements’ patterns was that a requirement should be atomic. Our NC measure was a good indicator for atomicity. If the number of conjunctions are too many (usually NC > 3), the relation of actions in a requirement description becomes complex for understanding and testing.

Based on the measures we provided, they improved their predefined patterns. Then they integrated this rules in their RMS so they could get instantaneous feedback when writing requirements. Figure 4 shows a snapshot of the report on requirements quality in Grundfos. In the figure we can see the requirements’ names on the leftmost side and the likely problems in them on the rightmost side. In Grundfos engineers had chosen more strict rules for requirements’ texts, therefore, instead of displaying QI₆ numbers they indicated the measured key word for additional manual check. For example, in the third row of the rightmost column in the figure, the indication of “Multiple Keywords” means that more than one conjunction is used in a requirement.

C. Overall Results

The automation of reviews, according to the overall feedback that we got, saved several hundreds of hours of working time in every product release (6 months). The adoption of Rendex in the collaborating companies made a remarkably new step towards proactive reviews of requirements. Furthermore, as the software engineers noticed, getting instantaneous feedback on newly written requirements helped requirements’ designers to improve the requirements much efficiently, because the feedback was received “just in time” of writing requirements, when they remember requirements content the best.

When it comes to adopting Rendex in companies, there were important similarities and differences between the choices that the companies made, and between the difficulties we encountered in companies. The following points summarize the similarities in the three companies:

- The best choice of using the method was to use it as a part of the RMS, because it enabled the most proactive feedback on requirements, and was perceived as a natural part of requirements management process.
- The method should not be used by an assigned person to give personal feedback to others. Even though the authorized person might have a clear intension of helping engineers, his actions might be perceived as intrusive.
- The method should be used directly by software engineers who write and communicate requirements. They can be guided or trained on how to use the method, however, no one else but only themselves should run analysis, monitor results of quality assessment, and improve requirements.
- Decision criteria should be established to help engineers in action planning. If there is no decision criteria, than they postpone improvement activities.

Color-coding was a powerful visual decision criteria for conducting actions.

The following points summarize the differences between the companies:

- Integrating the method with the RMS takes much more time if the RMS is not a local tool. Since the companies are big, there were political maneuvers, legal issues, price discussions, and maintenance...
difficulties. Especially difficulties in maintenance can be a problem, because action research projects require a continuous tuning of the method based on continuous feedback.

- Decision criteria can be different from company to company depending on several factors, such as types of requirements, the foreseen effort for improvements, engineers’ choice, etc.

We would like to extra-emphasize the care for not creating an impression of intrusive action towards engineers’ old ways of working. The key activities for achieving this are to involve several engineers in the discussions and consider their comments. It is always a good practice to suggest the use of the method to a smaller team of the organization, who are more open to using new methods and can give valuable feedback on what can be calibrated.

VI. VALIDITY THREATS

**Recoverability** of our research is the guarantor for external validity [12]. This means that, when replicating the study in different companies we documented the similarities of results for general conclusions, and differences across the companies as necessary information for anyone who intends to use the content of this paper in other organizational contexts.

This study cannot generalize how exactly the method should be applied. Different organizations have different ways of working and habits, so it is always relevant to do an initial investigation on how the particular organization can benefit most from adopting a new method. Action research is one methodology that permits understanding effects when introducing new methods.

VII. RELATED WORK

There are a few studies who proposed automated methods for requirements’ internal quality assessment. For example Natt och Dag et al. [13] present an approach based on textual analysis of requirements to find similarities, linkages or business connections between different requirements, in order to facilitate requirements management process. Glinz [14] presents an approach for evaluating not only external but also internal quality attributes of requirements. Glinz put the emphasis on the criticality of requirement, which should be considered, when doing risk assessment. Sharma and Kushwaha [15] propose a software complexity measurement approach directly from the textual requirements. To the best of our knowledge there is only one study which focused not only on creating automated methods but also investigating its use in practice [7]. The authors developed a tool (RQA) for automated analysis, which was introduced to several organizations.

VIII. CONCLUSIONS

In this paper we showed how a requirements’ measurement method can be integrated with software development organizations, enabling proactive reviews of requirements. Getting feedback “just in time” of writing requirements facilitates requirements review process. The use of this method helped the collaborating organizations to be proactive in the review process and take action based on predefined decision criteria.

REFERENCES