Framework for Quantitatively Evaluating the Quality Requirements of Software System

Yuki Terawaki

Research Center for Computing and Multimedia Studies,
Hosei University, 3-7-2 Kajinocho, Koganei, Tokyo 184-8584, Japan
yuki.terawaki.dc@k.hosei.ac.jp

Abstract. Quality requirements (QR) are a description which indicates how well the software's behavior is to be executed. It is widely recognized that quality requirements are vital for the success of software systems. Therefore, to define the quality requirements and to check the quality attributes carefully is necessary for bringing good- quality software and ensuring quality of the service. This paper proposes a framework that measures the quality attributes in the requirements document such as SRS. The effectiveness of this framework was briefly described, we discuss approach was to enrich the representative quality corpora.

Keywords: Requirements Engineering, Quality Requirements, Non-Functional Requirements, text-mining.

1 Introduction

Quality requirements (QR) are a description which indicates how well the software's behavior is to be executed. It is widely recognized that quality requirements are vital for the success of software systems. So, it is necessary to check the quality requirements carefully to bring good-quality software for the user and to ensure the quality of service. However, in the requirements acquisition phase, functional requirements are highly focused, quality requirements are not necessarily sufficiently defined [1]. Despite the importance of QR, it is generally acknowledged that QR are difficult to capture and specify. Several studies [2] [3] [4] [5] have identified challenges of QR as: difficult to gather, often poorly understood, general stated informally in a non-quantifiable manner, where should QR document, and difficulties to get attention for QR.

To define quality requirements adequately and sufficiently, we must know how much QR are stated in the software requirements specifications (SRS). It will make a good base to explore ways for eliciting, representing and implementing QR. For that purpose, this paper proposes a framework to identify where in a SRS quality requirements are stated and which characteristics class each requirement belongs to. The proposed framework can analyze QR found in an SRS in terms of their volume, balance and structure. This framework can analyze the SRS written in natural language,

S. Yamamoto (Ed.): HIMI/HCII 2013, Part I, LNCS 8016, pp. 383–392, 2013.

Japanese [icons ref]. Today, it's a fact that SRS are mostly written in natural language. In this framework, natural language processing techniques, particularly the text similarity detecting methods, are employed to measure the degree of quality factors in each requirement sentence.

The paper is organized as follows. In section 2, the background and related work are presented. In section 3, the proposed framework and the implementation of tool are described. Section 4 introduces case studies briefly. Section 5 presents the discussion. In section 6 conclusion and future works are provided.

2 The Previous Practice in Industry and Academic

Today, though software development requires quick delivery, it is not unusual for development documents (such as SRS or Request for Proposal: RFP) to be over several hundred pages long. As the scale of the SRS gets bigger, the structure of the SRS becomes complicated. At present, despite of the increasing number of documents which should be inspected, shortening of development time is desired. To respond these demands of the present age, some templates of reader-friendly SRS have been proposed. These templates are often recommended to write Functional Requirements and QR separately. For example, FR and QR are to be described in separate chapters in IEEE Std. 830-1998 [6]. Wiegers's book [1] introduces another template adapted from the IEEE 830, which also separates FR and NFR. Lauesen [7] said the SRS written in industry were inspired by the IEEE830 guidelines, but when it came to the specific requirements, they were bewildering because IEEE 830 suggested no guidance. Moreover, Lauesen's book introduces SRS of good example. These SRS are not similar to IEEE 830 structure, and it is more instructive to grasp how QR's are distributed over the document and how they are mixed with FR or not. Particularly, visualization of the distribution is helpful.

The following researches are developing the tool which detects the defect of requirements. William M. Wilson et al proposed the Automated Requirements Measurement (ARM) [8]. The Quality Analyzer for Requirements Specifications (QuARS) was proposed by A. Fantechi et al [9]. These researches aim at pointing out the inaccuracy of the requirement specification document written by natural language. The advantage of this research over these researches is as follows. This research provides stronger support function for quality requirements. This framework gives the evaluation criterion of quality requirements (development documents) to the author of RS. The author of RS can focus on improving the quality requirements.

Moreover, the following researches are related to specification, classification, and measurement of QR. Grimshaw and Draper [10] found that QR are often overlooked and there is a lack of consensus about quality requirements. Johansson et al. found that reliability was identified by a multitude of stakeholders to be the most important QR[10]. Our research objective is to investigate an SRS, what kind of QR are actually written, how they are distributed over quality characteristics as categorized in the ISO/IEC 25010 standard. There are already some works that attempt to discern

and classify QR in SRS. H. Kaiya et al. [12] calls the distribution of requirements sentences across QR characteristics as 'spectrum'. In Kaiya's research, the policy of distinguishing QR from FR is not clear, so, it is required to classify requirements quality characteristics by hand. That implies making up the spectrum they needs keyword-to-quality matrix. The matrix is constructed for each system by a human, although there is a possibility of reusing the existing one if the application domain is the same. Svensson et al. [13] analyzed how quality requirements are specified and which types of requirements exist in a requirements specification from industry by manual work. By contrast, in our approach, a text mining technique is used to filter out QR statements from SRS, classifying them into the ISO/IEC 25010 quality characteristic categories at same time. So, the original of our approach is to identify where in a SRS quality requirements are stated and which characteristics class each requirement belongs to. We can identify how each QR characteristic scatters over the document, i.e. how much in column and in what way.

3 Proposed Framework

3.1 Our Approach

When we manage QR, We can apply a sentence of requirements to each characteristic of the software quality attributes of ISO/IEC 25010 and can check the requirements for quality. However, it is difficult to evaluate correspondence with attribute and requirements for a general reason. This is because some quality requirements overlap two or more quality attributes. Also, when identifying attributes and requirements, human judgment may change over time. Thus, it is difficult to review every quality requirements in terms of coherent thinking. However, if the quality attributes can be quantitatively measured, then they could potentially help the author of SRS decide if a revision is needed. Additionally, almost all development documents are described in natural language. If the quality requirements needed are written to the document created to the upper process, quality can be measured at the time of the acceptance inspection. However, it is difficult to review every quality attributes in terms of time. These problems bring deterioration in the quality of development document and play a role in the failure of the project. So, text-mining technique is employed.

The Requirements Process includes 4 processes. There are requirements elicitation, evaluation, specification (documentation) and quality assurance. This process is iteration on successive increments according to a spiral model [14]. In the spiral process, when requirements document will become elaborate, the error of requirements may be made. In spiral process, the revised document (SRS) can check without spending hours using text-mining technique. The quality attributes contained in SRS are showed quantitatively because the text mining analyzes where quality attributes are contained, and how much. The rate of documentation of quality attribute can be showed using the output of text mining. Thus, the quality requirements are checked by coherent thinking. Therefore, the workload for verification of SRS will be decreased.

3.2 Overview of Framework

We propose a framework for mining QR in SRS. This framework can be use to improve the quality of SRS through the requirements definition process. As criteria for evaluating the quality requirement, the quality model of ISO/IEC25010 [15] is used. The proposed framework can specify the statement related to quality attribute of ISO/IEC 25010. The framework is conceptually composed of two parts. One is the QR mining mechanism and the other is its usage in the RE process. Figure 1 shows the conceptual diagram of this framework.

The QR mining mechanism analyzes the similarities of the quality content of a given piece of requirements text between the similarity with a corpus of typical requirements sentences that state the target quality characteristic such as Performance Efficiency (PE), Compatibility (Co), Usability (U), Security (Se), etc. A detailed discussion of the calculation method of similarity can be found in [19].

In Kaiya et al.'s work [12], keywords are used to link requirements with quality attributes. In comparison our approach is not keyword-based but based on text similarity metrics using an associative search engine. It is more suitable to our purpose than a keyword-based approach, because it is not affected strongly by a particular choice of keywords.

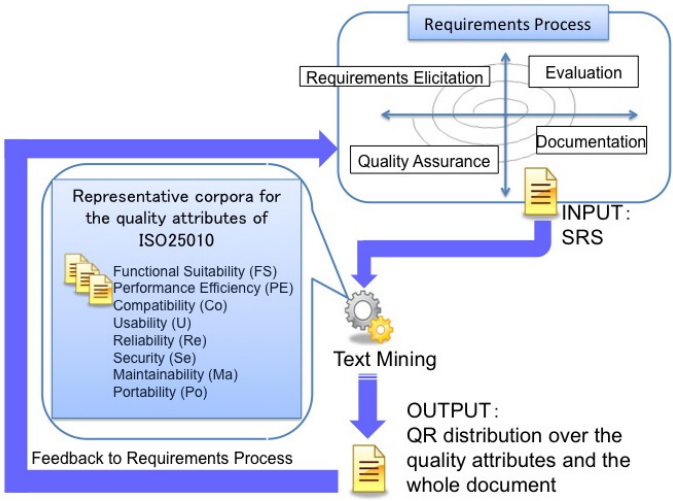

Fig. 1. Conceptual diagram of QR Mining Framework

Regarding the usage in the RE process, the output of tool can contribute to different phases of the RE process cycle. For example:

- Documentation phase: As unfinished requirements documents can be analyzed in this framework, findings obtained from QR mining can be used to give a feedback when writing an SRS. The authors of SRS can find the redundant description and the overlapping statements more effectively and quickly by using the output of tool.

- Quality Assurance phase: Probably, the most natural usage of QR mining is in the phase of evaluation. When the SRS is completed and passed to a quality assurance team for reviewing, the output of the QR mining will be informative in analyzing and improving the SRS. And in inspection process [20] the output of the QR mining helps inspectors examine the work product to identify possible defects.

3.3 Tool

We developed a tool *QRMiner* that supports the proposed framework. This tool was developed as a CUI application by using Java programming language and shell scripts. The Japanese morphological analyzer Sen [16] is employed for morphological analysis. The Japanese WordNet [18] is used to add Synonyms and extracted through a method that first acquires a lower level of the word group against the original words and then acquires the sum of the sets of each upper level word group. For the Quality-to-term matrix file creation and search operations, GETA associative engine is employed [21]. GETA, an acronym for "Generic Engine for Transposable Association", is an efficient tool that accepts a group of queries in text and returns highly related documents from the designated repository in their relevance order. It is open to public use. Such as mkw command, a part of GETA commands are provided only as the C library, so we developed an execute format to wrap the I/O for the connection by using the Java program and standard I/O. A detailed discussion of the tool can be found in [22], [19], [23].

4 Case Studies

This case study presents the results of analysis of SRS that collected from the public sector of universities and governments. After an early analysis of the SRS, a paper [19] was presented at conference. This paper extends our previous report on findings with more description of the SRS and account of idea of future works.

4.1 Overview

Total of nine SRS's are analyzed, which are consists of two groups; six are systems for universities and three are for governments. The nine SRS was analyzed by QRMiner and manually by the three experts. Moreover, to evaluate the usefulness of QRMiner quantitatively, we calculated the precision and recall between the results by QRMiner and those decided by the human experts for all the SRS. Precision and recall in total are 0.60 and 0.62, which is not so bad, the evaluation of the tool in terms of precision and recall was satisfactory. The followings are findings of the previous study [19].

QR structure: The QRMiner can show the structure of SRS. As noted before chapter of this paper, a well-structured SRS, where QR and FR are written in separate chapters. When they are mixed apparently randomly, the SRS commonly have got the following problems.

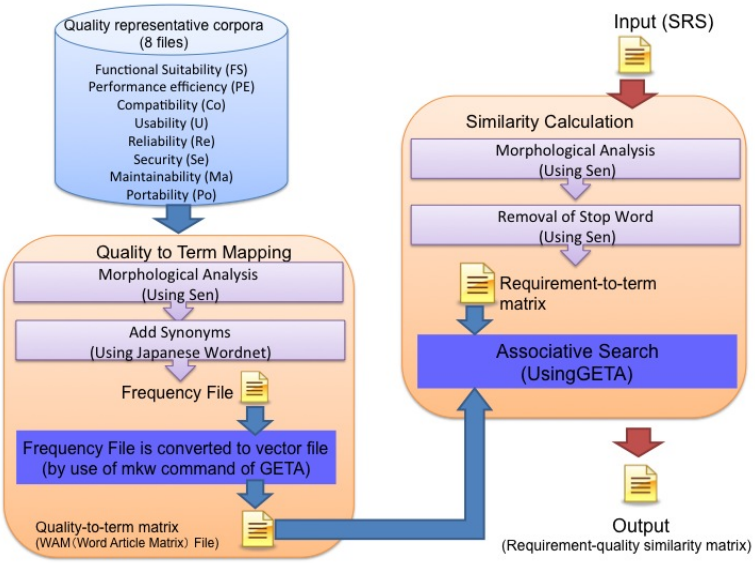

Fig. 2. Structure of QRMiner

Redundancy: The QRMiner output detects the same quality requirements in the early part and also in the later part of the document. Inspecting those parts manually, it was suggested that they are rather redundant and had better be merged in one place.

- Overloading: The QRMiner detects that some sentences are found to have multiple QR overloaded within. By inspection, it was suggested to divide the sentence to enhance the readability.

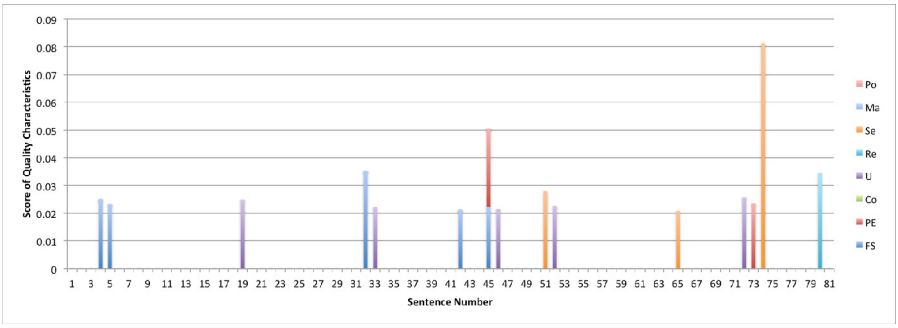

Fig. 3. Distriburion of quality characteristics over the whole document

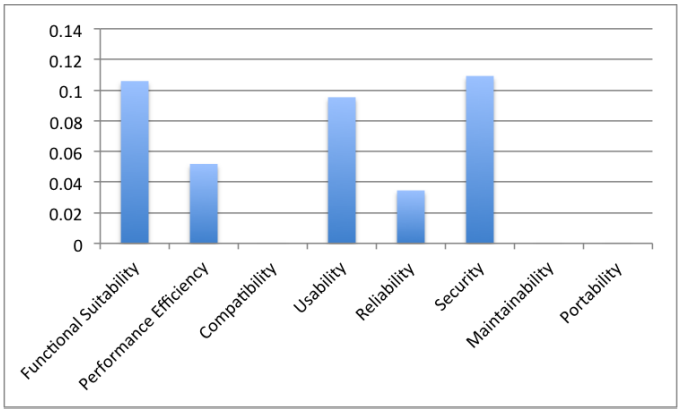

Fig. 4. Result of QRMiner

4.2 Difference between University and Government Systems

As regarding the evaluation of precision and recall, there is no particular difference found between the university systems and the government systems. However, the SRS that we used have distinct characteristics of " Request for Proposal (RFP)". The SRS that the public sector wrote have some features.

1. The requestor does not have much technical knowledge. The requestor often gets a suggestion from competitor several times, and then the requestor perfects the SRS through the suggestion.
2. The requirements are not only for software but integrated with those for hardware and includes rules for administration of system.
3. The upper limit budget is determined but usually hidden to the tenders. A proposal with cost estimation under the budget limit and with high values will obtain the contract.

5 Discussion

This research concentrates on analyzing QR. The proposed framework in terms of precision and recall is satisfactory. However, such as SRS for procurement, when the SRS that mentioned chapter 4.2 are treated, it is required to have a point of view.

5.1 Aspect of Procurement

The software requirements specifications are often referred to as the information system procurement specifications or specification requirements for information system procurement at universities or government agencies. This means requirements specifications are made on the essential basis. When you aim to accomplish quality improvement on requirements specification made in universities or government

agencies, it is desirable to focus on not only system requirements characteristics but also analyzing requirements comprehensively. The requirements related to system are specifically as follows;

- Handling of Precondition or reference for software /system structure
- Service agreement among software providers and contractors
- Handling of indeterminate requirements before completing an order

5.2 System Requirements Categorization and Service Level Agreement

It is required to increase the quality representative corpora to accumulate sentences, to deal with problems above with the proposed framework in this study.

ISO/IEC25030 [24] and ISO20000-2 are helpful to increase the quality representative corpora.

This tool registers corpus sentences according to ISO/IEC25010. ISO/IEC25010 is an international standard stated in system and software product quality requirements and evaluation (SQuaRE) series. The SQuaRE series consist of some standards. For example, ISO/IEC25030 describes that a system consists of number of interacting elements and they can be defined and categorized in different ways. And it defines classes of requirements. ISO/IEC 25010 deal with the gray part (**Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.**) of classes for requirements. It is efficient to accumulate corpus sentences according to ISO/IEC25030 to manage any requirements connected to Precaution for operation, system running, and maintenance conditions.

ISO/IEC2000-2 is a standard guideline for concluding Service Level Agreement. There are 20 items listed. When you accumulate corpus sentences connected to support system and scope of work, for example, emergency contact during any trouble occurrence, job commission to system engineers who is continuously presence.

<table>
<tr><td rowspan="6">System requirements</td><td rowspan="5">Software requirements</td><td rowspan="3">Software product requirements</td><td rowspan="2">Inherent property requirements</td><td>Functional requirements</td><td></td></tr>
<tr><td rowspan="3">Software quality requirements</td><td>Quality in use requiremens</td></tr>
<tr><td>Extermal quality requiremens</td></tr>
<tr><td>Assigned property requirements</td><td colspan="2">Managerial requirements including for example requirements for price, delivery date, product future, and product supplier</td></tr>
<tr><td rowspan="2">Software development requirements</td><td colspan="3">Development process requirements</td></tr>
<tr><td colspan="3">Development organisation requirements</td></tr>
<tr><td>Other system requirements</td><td colspan="4">Include for example requirements for computer hardware, data, mechanical parts, and human business processes</td></tr>
</table>

Fig. 5. System requiremens categorisation (ISO/IEC 25030)

Lately, in Japan, the basic guideline for procurement of information systems and handling of government procurement of information systems were established. There exists the improvement we found with this case, and assistance with this framework that correspond with the government basic guideline. Therefore, this framework can make a contribution to improve the quality of requirement specification by enhancing corpus sentences with using ISO/IEC25030 and ISO/IEC2000-2.

6 Conclusion and Future Work

We propose a QR mining framework and developed a tool QRMiner that supports the framework. The effectiveness of this framework was briefly described, we discuss approach was to enrich the representative quality corpora. We will prepare representative corpora by hand using ISO/IEC25030 and ISO/IEC2000-2. Another different approach is the use of a learning mechanism. However, in our first attempt will be to collect more documents and to extend the scope of target system types. In near future, we may find a better way of a learning mechanism.

References

1. Wiegers, K.E.: Software Requirements. Microsft Press (2003)
2. Chung, L., Nixon, B.A., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Academic Publishers (2000)
3. Berntsson Svensson, R., Gorschek, T., Regnell, B., Torkar, R., Shahrokni, A., Feldt, R.: Quality Requirements in Industrial Practice – an extended interview study at eleven companies. IEEE Transaction on Software Engineering (2011) (in print)
4. Borg, A., Yong, A., Carlshamre, P., Sandahl, K.: The Bad Conscience of Requirements Engineering: An Investigation in Real-world Treatment of Non-functional Requirements. In: Third Conference on Software Engineering Research and Practice in Sweden (SERPS 2003), Lund, Sweden (2003)
5. Chung, L., do Prado Leite, J.C.S.: On Non-Functional Requirements in software engineering. In: Borgida, A.T., Chaudhri, V.K., Giorgini, P., Yu, E.S. (eds.) Conceptual Modeling: Foundations and Applications. LNCS, vol. 5600, pp. 363–379. Springer, Heidelberg (2009)
6. IEEE, Recommended practice for software requirements specifications, IEEE, Tech. Rep., IEEE Std830-1998 (1998)
7. Lauesen, S.: Software Requirements styles and Techniques. Addison-Wesley (2002)
8. Wilson, W.M., et al.: Automated Analysis of Requirement Specifications. In: Proceedings of the International Conference on Software Engineering ICSE 1997, pp. 161–171 (1997)
9. Fantechi, A., et al.: Application of Linguistic Techniques for Use Case Analysis. In: Proceedings of the IEEE Joint International Conference on Requirements Engineering, pp. 157–164 (2002)
10. Grimshaw, D., Drapper, G.W.: Non-functional requirements analysis: Deficiencies in structured methods. Information and Software Technology 43, 629–634 (2001)
11. Johansson, E., Wesslen, A., Bratthall, L., Host, M.: The importance of quality requirements in software platform development –a survey. In: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (2001)

12. Kaiya, H., Sato, T., Osada, A., Kitazawa, N., Kaijiri, K.: Toward quality requirements analysis based on domain specific quality spectrum. In: SAC 2008 Proceedings of the 2008 ACM Symposium on Applied Computing (2008)
13. Olsson, T., Sevenssion, R.B., Regnell, B.: Non-functional requirements metrics in practice – an empirical document analysis. In: Workshop on Measuring Requirements for Project and Product Success (2007) (a full version is to appear in the Information and Software Technology)
14. Kotonya, G., Sommerville, I.: Requirements Engineering: Processes and Techniques. John Wiley & Sons Ltd. (1997)
15. ISO/IEC, ISO/IEC 25010 Systems and software engineering- Systems and software Quality Requirements and Evaluation (SQuaRE) -System and software quality models (2011)
16. Sen (December 2010), http://ultimania.org/sen/
17. GETA (December 2010), http://geta.ex.nii.ac.jp/e/index.html
18. 日本語 WordNet (December 2010), http://nlpwww.nict.go.jp/wn-ja/index.en.html
19. Terawaki, Y., Tamai, T.: A Framework for Mining Quality Requirements in Software Requirements Specifications. In: The International Symposium on Requirements Engineering (RE) (submitted)
20. Fagan, M.: Design and Code Inspections to Reduce Errors in Program Development. IBM Systems Journal 15(3), 182–211 (1976)
21. Takano, A.: Association computation for information access. In: Grieser, G., Tanaka, Y., Yamamoto, A. (eds.) DS 2003. LNCS (LNAI), vol. 2843, pp. 33–44. Springer, Heidelberg (2003)
22. Terawaki, Y.: Supporting of Requirements Elicitation for Ensuring Services of Information Systems Used for Education. In: Smith, M.J., Salvendy, G. (eds.) HCII 2011, Part I. LNCS, vol. 6771, pp. 58–65. Springer, Heidelberg (2011)
23. Terawaki, Y., Tamai, T.: A practical approach to Quality Requirements Handling in Software Systems Development. In: The Eighth International Conference on Systems, ICONS 2013, pp. 160–163 (2013)
24. ISO/IEC, ISO/IEC 25030 Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Quality requirements, 1 edn. (2007)