PAPER Special Section on Knowledge-Based Software Engineering

Improving Reliability of Spectrum Analysis for Software Quality Requirements Using TCM

Haruhiko KAIYA[†,††a)], **Masaaki TANIGAWA**[†], **Shunichi SUZUKI**[†], **Tomonori SATO**[†], **Akira OSADA**[†], ***Nonmembers***, ***and*** **Kenji KAIJIRI**[†], ***Member***

SUMMARY Quality requirements are scattered over a requirements specification, thus it is hard to measure and trace such quality requirements to validate the specification against stakeholders' needs. We proposed a technique called "spectrum analysis for quality requirements" which enabled analysts to sort a requirements specification to measure and track quality requirements in the specification. In the same way as a spectrum in optics, a quality spectrum of a specification shows a quantitative feature of the specification with respect to quality. Therefore, we can compare a specification of a system to another one with respect to quality. As a result, we can validate such a specification because we can check whether the specification has common quality features and know its specific features against specifications of existing similar systems. However, our first spectrum analysis for quality requirements required a lot of effort and knowledge of a problem domain and it was hard to reuse such knowledge to reduce the effort. We thus introduce domain knowledge called term-characteristic map (TCM) to reuse the knowledge for our quality spectrum analysis. Through several experiments, we evaluate our spectrum analysis, and main finding are as follows. First, we confirmed specifications of similar systems have similar quality spectra. Second, results of spectrum analysis using TCM are objective, i.e., different analysts can generate almost the same spectra when they analyze the same specification.

key words: *requirements analysis, quality requirements, non-functional requirements*

1. Introduction

Software quality requirements of a system are specifications for defining how well functions of the system are accomplished. Defining quality requirements has more problems than defining functional ones, and there was a special issue about quality requirements in IEEE Software. In its guest editors' introduction [1], the following three problems are mentioned: implicit understanding of quality requirements by stakeholders, trade-offs among quality requirements and difficulty of measuring and tracking quality requirements.

There are several techniques for resolving one or more problems above, and we proposed a simple and general technique called "spectrum analysis for quality requirements" [2] for measuring and tracking quality requirements. A wave such as sound or light can be decomposed into several regular (or sine) waves each of which has different cycle (or wavelength) and power (or amplitude). Spectrum analysis in optics is based on this fact. In spectrum analysis for quality requirements, a quality characteristic such as suitability, accuracy, and interoperability is regarded as wavelength, and the power of the characteristic as its importance as shown in Fig. 1. By using a quality requirements spectrum of a system, stakeholders can identify relative attention to quality requirements in a software engineering artifact such as a requirements specification or a design document. Such relative attention enables stakeholders to validate quality requirements defined in such software engineering artifact. Suppose a power of security is larger than one of usability in a quality spectrum of a requirements document for a system. If a stakeholder regards usability is more important than security, he can easily suspect one of his quality requirements could not be reflected in the document.

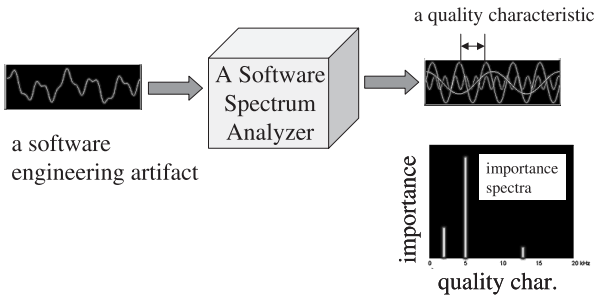


Fig. 1 Basic idea of spectrum analysis for quality requirements.

There are two systematic comparative analyses for quality spectrum analysis. One is comparison among spectra of similar systems to identify mandatory and optional quality characteristics. There are a lot of similar systems for each application domain, e.g., a lot of web browsers, painting tools, e-learning systems and so on. Systems in the same domain usually have similar quality spectrum, and such similarity shows mandatory quality requirements in such a domain [2]. On the other hand, differences among spectra of the systems in the same domain show optional or specific features of each system. Although different segments or different price ranges of the same domain do not always have similar spectrum, we may regard each segment or each range as a sub-domain and may compare spectra of parts in a segment or spectra systems in a range with each other. Comparison among spectra of similar systems is one of the main issues of this paper.

Another is comparison among spectra of a system in

Manuscript received July 4, 2009.
Manuscript revised October 19, 2009.

[†]The authors are with the Graduate School of Science and Technology, Shinshu University, Nagano-shi, 380–8553 Japan.
[††]The author is with GRACE Center, National Institute of Informatics, Tokyo, 101–8430 Japan.
a) E-mail: kaiya@cs.shinshu-u.ac.jp

DOI: 10.1587/transinf.E93.D.702

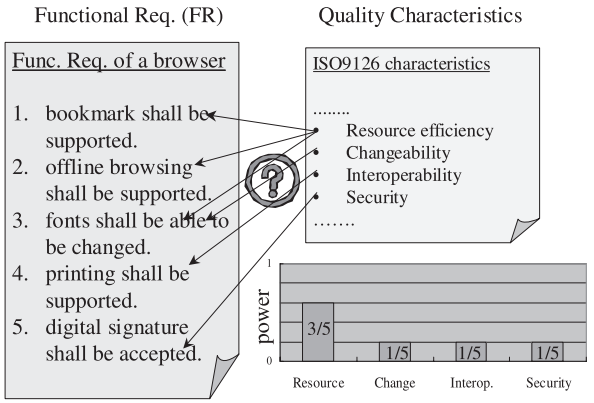


Fig. 2 An example of the old method.

different development phases, e.g., requirements, design, implementation and so on. Quality requirements should be inherited along the progress of development, but it is not easy to track such inheritance during the progress of such development. Quality requirements spectrum enables developers to track such inheritance. Comparison among spectra of a system in different development phases is one of our future issues, thus it is out of scope in this paper.

There is a serious problem in our early quality spectrum analysis [2]. As shown in Fig. 2, the power of each quality characteristic is calculated based on the number of relationships between an element of an artifact, e.g., a sentence in a requirements document, and each quality characteristics. Making such relationships largely depends on the expertise and subjective decision of an analyst. Therefore, it takes a lot of efforts to perform quality spectrum analysis and is hard to perform the analysis (semi-) automatically. In this paper, we will introduce an improved version of quality spectrum analysis for resolving this problem. In addition, we show a prototype of a CASE tool that supports quality spectrum analysis.

The rest of this paper is organized as follows. In the next section, we briefly explain the original quality spectrum analysis method (called "the old method" in this paper) [2], and clarify its problems. We then introduce the improved method (called "TCM method" in this paper) by using domain knowledge called term-characteristic map (TCM). In Sect. 3, we evaluate TCM method with respect to the following three points: results of TCM method inherit those of the old method, spectra of several different systems in a same domain are similar with each other and results of TCM method are objective, i.e., results of TCM method do not depend on analyst's subjective decision. In Sect. 4, we will show a supporting tool to perform TCM method. Finally, we briefly review related works, summarize our current results and show the future issues.

2. A Method for Generating Quality Spectrum

2.1 Requirements Specification and Similar Systems

We first discuss what are written in a SRS (Software Requirements Specification). Each company or organization usually has its own template for SRS, and there are several standard or public templates for SRS [3], [4]. In [3], a SRS is divided into three chapters; introduction, overall description and specific requirements. The first two chapters are very important because contexts of the software such as goals, environments, assumptions and constraints are specified. The third chapter contains requirements in detail in the sense that some stakeholders (users, operators or external systems) can externally perceive functions of the software. At least, inputs to the software, outputs from it and functions related to the inputs and outputs should be written in a requirement. In our spectrum analysis, we only focus on such requirements in the third chapter because spectrum analysis focuses on the specifications of software itself.

Such requirements are normally categorized and structured for readability and understandability, thus requirements are normally described in more than a list of sentences. Actually, [3] proposes several samples for structuring such requirements in its appendix. Because such kinds of categorization put several requirements into one description, granularity of such descriptions is not uniform thus it is difficult to compare one description to another. To avoid such a granularity problem in this paper, we focus on externally perceivable requirements before they are structured, i.e., a list of sentences. In [5], a technique to transform an i* model (which is a highly structured description of requirements) into the list of natural language sentences, each of which is an externally perceivable requirement. Therefore, we will be able to convert a structured description into the list of externally perceivable requirements.

As mentioned in introduction, we focus on comparison among spectra of similar systems, thus identifying similar systems is one of the critical issues. In [6], such similarity is identified based on the structure of a use case diagram for each system. Because a use case diagram specifies the context of the system as well as externally observable functions of the system, such context and functions are main information for identifying system similarity. According to [7], systems can be categorized based on eight dimensions (dimensions related to state, object, goal and so on), and 13 concrete categories of the object dimension (e.g., resource returning, resource supplying, object sensing, domain simulation and so on) are proposed in the literature. By using such dimensions and concrete categories in [7], we can also identify similarity among systems. Practically, we have already had web sites of software applications[†] and such sites have already categorized a lot of applications. We may choose one of the categories out of the list on such sites according to the system to be analyzed, and use several existing systems in the chosen category.

[†]For example, http://www.vector.co.jp/magazine/softnews/, http://sourceforge.net/index.php

2.2 The Old Method, Its Usage and Its Problems

As mentioned in introduction, we call the procedure to generate quality spectrum in our previous paper [2] as "the old method" in this paper. By using Fig. 2, we will explain how to perform the old method, the usage of the output of the method and problems of the method.

The inputs of the old method are a list of requirements and a list (catalog) of quality characteristics. Actual requirements specifications are not the list of requirements but we have already discussed how to convert such specifications into the list of sentences in Sect. 2.1. In Fig. 2, five requirements are listed in the list and quality factors in ISO9126 [8] are used for the catalog. ISO9129 contains one of the famous catalogs of quality characteristics. Such kind of catalogs helps requirements analysts to find missing quality requirements. A quality model in ISO9126 categorizes software quality attributes into six characteristics (functionality, reliability, usability, efficiency, maintainability and portability), which are further subdivided into subcharacteristics such as resource efficiency, changeability and so on in Fig. 2.

The output of the old method is a value of each quality characteristic. The value corresponds to the power in spectrum analysis for optics, thus we also call this value as "the power" of a quality characteristic. Powers of all quality characteristics is a kind of vector, and this vector is called "quality spectrum" in our previous paper [2]. We design the old method so that the power of a quality characteristic shows the importance of the quality characteristic. The scale type of powers should be at least ordinal because we want to know whether a quality characteristic is more important than another. In Fig. 2, quality characteristics "resource (efficiency)", "changeability", "interoperability" and "security" take powers 3/5, 1/5, 1/5 and 1/5 respectively. Therefore, we can decide that "resource efficiency" is more important characteristic than others based on the result.

Steps in the old method are as follows.

1. An analyst makes relationships between a requirement and a characteristic subjectively. To make such relationships, the analyst takes into account whether a quality characteristic is mentioned in a requirement.
2. The number of requirements related to each characteristic is counted respectively.
3. The numbers are normalized into 0 to 1 based on the total number of requirements, and the normalized numbers are powers of quality characteristics.

In Fig. 2, four characteristics, resource efficiency, changeability, interoperability and security are related to three, one, one and one requirement(s) respectively. Because the total number of requirements is five, the values are normalized into 3/5, 1/5, 1/5 and 1/5 as powers of these four characteristics. For convenience, the powers are visualized as a bar chart at the bottom right in this figure.

We want to identify the importance of each quality characteristic in a requirements document, and we assume important characteristics should be written enough in the document. The usage of the old method is thus checking whether a requirements document has just enough quantity of quality requirements description. The old method is thus performed after eliciting requirements and documenting them. The concrete usages are as follows. First, as mentioned in introduction, we can check whether important quality characteristic(s) are documented enough or not by comparing a power in a quality spectrum to another. Suppose a power of security is larger than one of usability in a document. If a stakeholder regards usability is more important than security, he can easily suspect one of his quality requirements could not be reflected in the document. Second, stakeholders will also be able to check whether a requirements document has just enough quantity of quality requirements description by comparing a quality spectrum of the document and quality spectra derived from documents of existing similar systems. These kinds of concrete usage have a possibility to enforce useless documentation on a requirements analyst, e.g., writing redundant qualifiers to functional descriptions. We thus should use the result of the old method not as the quantitative target of quality requirements but as the trigger to explain why each quality requirement is documented so much. As mentioned in Sect. 2.1, actual requirements documents can be reformatted with respect to uniform granularity. In addition, the old method makes only one relationship between a quality characteristic and a sentence in a requirements list even if the sentence contains a lot of terms related to the characteristic. Therefore, reliability of the old method does not depend on the amount of terms related to quality characteristics.

One of the serious problems of the old method is the step to make relationships between requirements and characteristics. The old method largely depends on the expertise of an analyst performing the method. As a result, it takes a lot of hours to perform the old method even if the analyst has enough expertise such as domain knowledge of both the application and the quality characteristics. In addition, it is hard to reuse experiences performing the old method.

Second problem is the definition of a power of a quality characteristic. Apparently, a power of a quality characteristic is derived from the quantity of requirements related to the quality characteristic in the old method, and we do not have formally confirmed such quantity is related to the importance of the quality characteristic. However, such quantity is intuitively related to the importance because no one can take care of a quality requirement without its explicit description in a requirements specification. To improve the validity of the power of a quality characteristic, we want to revise how to derive the power by taking other factors into account, e.g., priority among requirements.

Third problem is that a power of a quality characteristic can be regarded as enough value even when requirement(s) that actually require the characteristic are not related to the characteristic. Suppose an analyst makes a relationship between requirement 5 and "resource efficiency" and he does not make a relationship between requirement

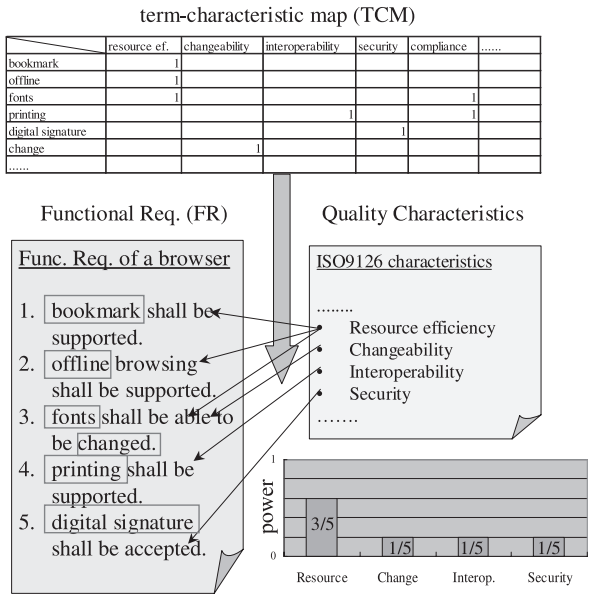


Fig. 3 An example of TCM method.

2 and "resource efficiency" based on relationships in Fig. 2. Even if such change is performed, a quality spectrum of the requirements list in the figure is not changed. This is an essential limitation of our quality spectrum analysis because the analysis focuses on the macroscopic viewpoint over the requirements list. To avoid this problem, an analyst has to review requirements related to each quality characteristic. To mitigate this problem, an analyst has to apply our quality spectrum analysis to the part of requirements, e.g., not all requirements of a Web browser but requirements only for the personal settings of the Web browser.

2.3 TCM Method

To overcome the first problem in the old method, we introduce a term-characteristic map (TCM) as domain knowledge for making relationships between documents and characteristics. Figure 3 shows an extended example of Fig. 2. TCM is a simple mapping between terms and characteristic, that tells potential relationships between them. In Fig. 3, terms in "RL of a browser" can be looked up in TCM, and an analyst can easily make relationships between requirements and characteristics. TCM merely tells potential relationships. In addition, the possibility whether a term is related to a characteristic depends on the contexts of the term usage. Therefore, relationships between requirements and characteristics cannot be made automatically and the analyst should make some subjective choice. In this example, the analyst does not use some mappings between several terms and a characteristic "compliance". Currently, we simply fill 1 or 0 value (blank if the value is 0 in Fig. 3) in cells of TCM to show whether there is potential relationship or not, but we would like to introduce some ordinal or ratio values to show the degree of its potential.

Steps in TCM method are almost the same as the steps in the old method mentioned in Sect. 2.2 except the first step. In the first step of TCM method, an analyst also makes relationships between a requirement and a characteristic subjectively. However, TCM method recommends the analyst for such relationships based on the occurrence(s) of term(s) in the requirement, thus the analyst may take such recommendation(s) into account.

Typically, experiences of performing the old method are gathered to develop TCM. Therefore, TCM method normally cannot be applied if no similar systems were analyzed using the old method beforehand. The steps to develop and to extend TCM are typically as follows. If an analyst is confident of systems in a domain, he may develop TCM without the following steps.

1. An analyst picks up a TCM of systems similar to a system which is currently analyzed. If there is no such TCM, the analyst creates new TCM with no mappings. This step may be performed by another analyst who is the expert of such systems instead of the analyst.
2. During the first step of the old method or TCM method, the analyst may pick up term(s) from a requirement related to a characteristic. The analyst may also pick up existing mapping(s) in the TCM as anti-candidate(s) if the mapping(s) seem to be irrelevant. If the analyst is not confident in his decision, he does not have to pick up any terms or mappings.
3. For each picked up term, a mapping between the term and the characteristic becomes a candidate to be added to the TCM.
4. Expert(s) of similar systems that are currently focused evaluate the candidates, and some candidates are added to the TCM. They also evaluate the anti-candidates and some anti-candidates are removed from the TCM.

Although some domain expert or an analyst himself should develop TCM beforehand, TCM will be able to be reused and be improved among similar systems in the same application domain. We would like to confirm this point in the future.

Apparently, steps for developing TCM above contains a lot of subjective decisions, e.g., what kinds of candidates or anti-candidates are chosen and which of them are added or deleted. Therefore, even if TCM is developed according to the steps above, TCM itself cannot be objective, i.e., all people cannot develop the same TCM of a domain according to the steps.

The usage of outputs of TCM method is the same as the old method in Sect. 2.2. In addition, TCM method still has second and third problems mentioned in Sect. 2.2.

3. Evaluation

We evaluate TCM method mentioned in the last section with respect to the following three points.

- Results of TCM method inherit those of the old method.

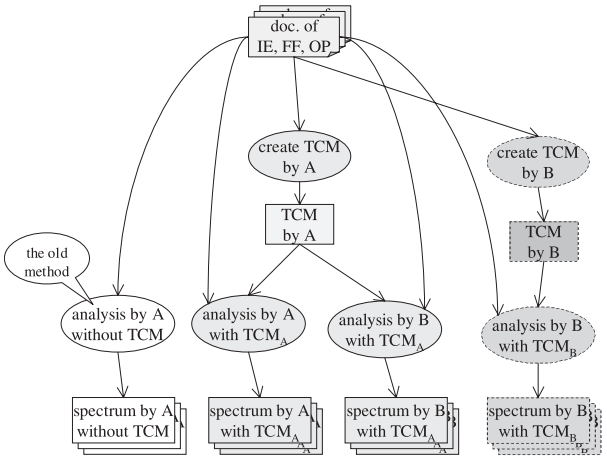


Fig. 4 Data gathering.

- Spectra of several different systems in the same domain are similar with each other.
- Results of TCM method are objective, i.e., different analysts can generate similar spectra of a system.

3.1 Data Gathering and Evaluation Method

To evaluate TCM method with respect to three points above, we need the following kinds of spectra: a spectrum generated by using the old method, a spectrum generated by using TCM method, a spectrum generated by another analyst using TCM method with the same TCM and a spectrum by the analyst with his own TCM. Figure 4 shows the outline how to gather such spectra data for our evaluation. This figure is written in data flow diagram, where boxes and notes correspond to data and ovals correspond to processes. We had two subjects called subject A and B, and we asked them perform spectrum analysis to documents of three browsers, Internet Explorer (IE), Fire Fox (FF) and Opera (OP), respectively. Subject A well knew this application domain as a user, and subject B was an average user.

At the left side of Fig. 4, subject A generated a spectrum without TCM. Note that this spectrum was generated before TCM method was proposed. Subjects A and B developed TCM of web browsers' domain respectively by using documents of web browsers according to the steps in Sect. 2.3. TCM developed by subject A was more appropriate than another because of the following reasons. First, one of authors and subject A together reviewed and discussed the result of the old method by subject A before subject A performed TCM method. Second, they also reviewed and discussed the TCM together. Subjects A and B then performed TCM method respectively by using the same TCM developed by subject A as shown in the middle of Fig. 4. Subject B also developed another spectrum by using his own TCM as shown in the right side of Fig. 4. Subject A developed spectra of another types of systems mentioned in the next sub section.

Both subjects used general spreadsheet to perform old or TCM method. Especially, subjects performing TCM method did not use a supporting tool mentioned in the next section because we decided to develop the tool based on the results of this evaluation.

Because a quality spectrum is a kind of vector, we use cosine similarity (cossim) to decide whether two spectra are similar with each other. The definition of cosine similarity between $\vec{a}$ and $\vec{b}$ is as follows.

$$cossim(\vec{a}, \vec{b}) = \frac{a_1 * b_1 + \cdots + a_n * b_n}{\sqrt{a_1^2 + \cdots + a_n^2} * \sqrt{b_1^2 + \cdots + b_n^2}}$$

When two vectors are completely the same, the value is one. Because quality spectrum never has negative value in its vector, cosine similarity between two quality spectra varies from 0 to 1. Therefore, we may regard two quality spectra are similar if their cosine similarity is close to 1. For example, $cossim(\vec{a}, \vec{b})$ is 0.99 when $\vec{a}$ is (0.00 0.08, 0.15, 0.25, 0.00, 0.00, 0.01, 0.15, 0.20, 0.85, 0.04, 0.09, 0.01, 0.25, 0.00, 0.00, 0.01, 0.00, 0.01, 0.01) and $\vec{b}$ is (0.00, 0.05, 0.18, 0.21, 0.00, 0.00, 0.01, 0.18, 0.12, 0.86, 0.02, 0.08, 0.01, 0.18, 0.00, 0.00, 0.03, 0.00, 0.00, 0.02). Note that $\vec{a}$ corresponds to a spectrum by A with TCM A in Fig. 5, and $\vec{b}$ corresponds to a spectrum by B with TCM A in the same figure.

3.2 Inheritance from the Old Method

Because TCM method is one of the improved version of the old method in our previous paper [2], a quality spectrum generated by the method should be similar to one by the old method. Figure 5 shows four quality spectra corresponding to the outputs in Fig. 4. Note that each spectrum in Fig. 5 is the average of spectra of three web browsers; IE, FF and OP. Horizontal axis of this figure shows quality characteristics used in our quality spectrum analysis. Because we have no explicit users of web browsers, we cannot identify objectives of such users. Therefore, we do not use a quality characteristic "suitability" in ISO9126 during this evaluation. Vertical axis shows the values of spectrum for each quality characteristic. Because documents of web browsers are analyzed and browsers are highly interactive system, "operability" has the highest value in a spectrum. "Security" has also higher value because of a lot of threats over the Internet. According to the definition of cosine similarity, similarity value between first and second spectra is 0.91, and the value between first and third spectra is 0.92. Therefore, we may regard TCM method inherits analytic ability from the old method.

3.3 Different Systems in the Same Domain

Quality spectrum is used to identify mandatory and optional quality requirements in an application domain, and this usage is based on the fact that different systems in the same domain have similar quality spectrum [2]. We confirm this fact by using TCM method.

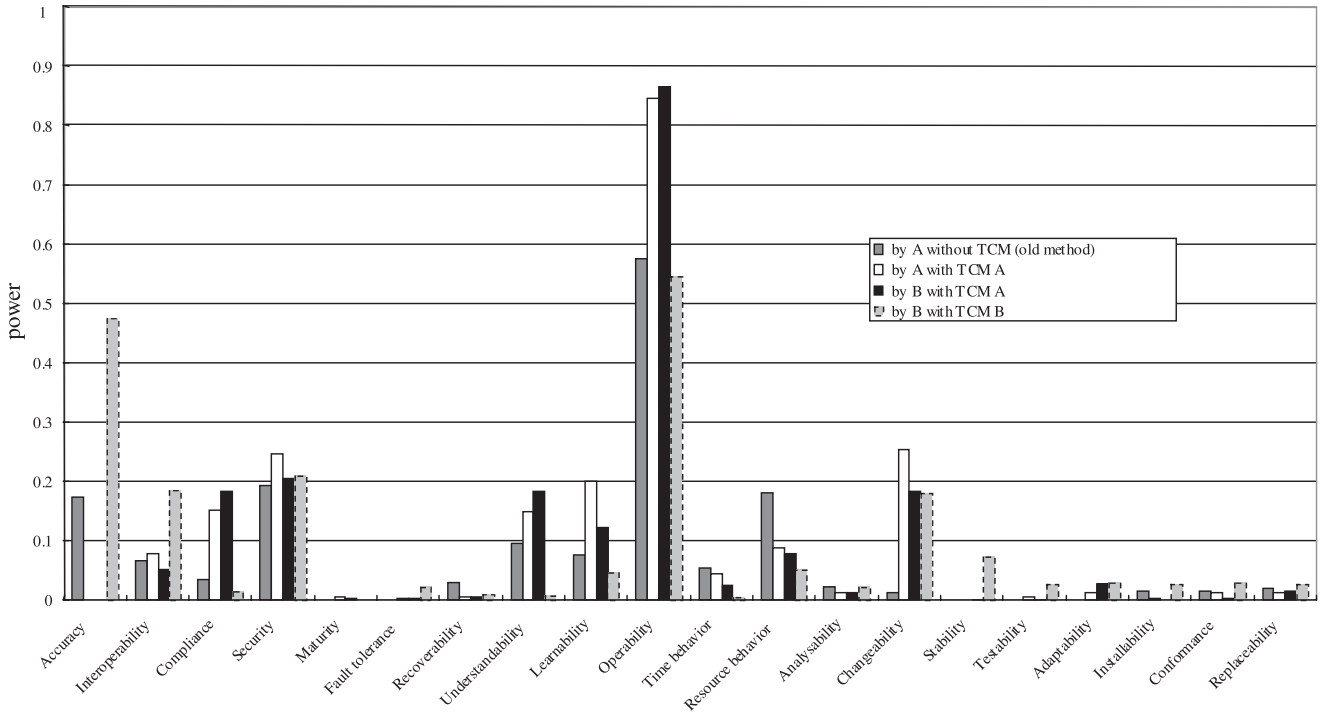


Fig. 5 The average of spectra of three Web browsers for four types of analysis.

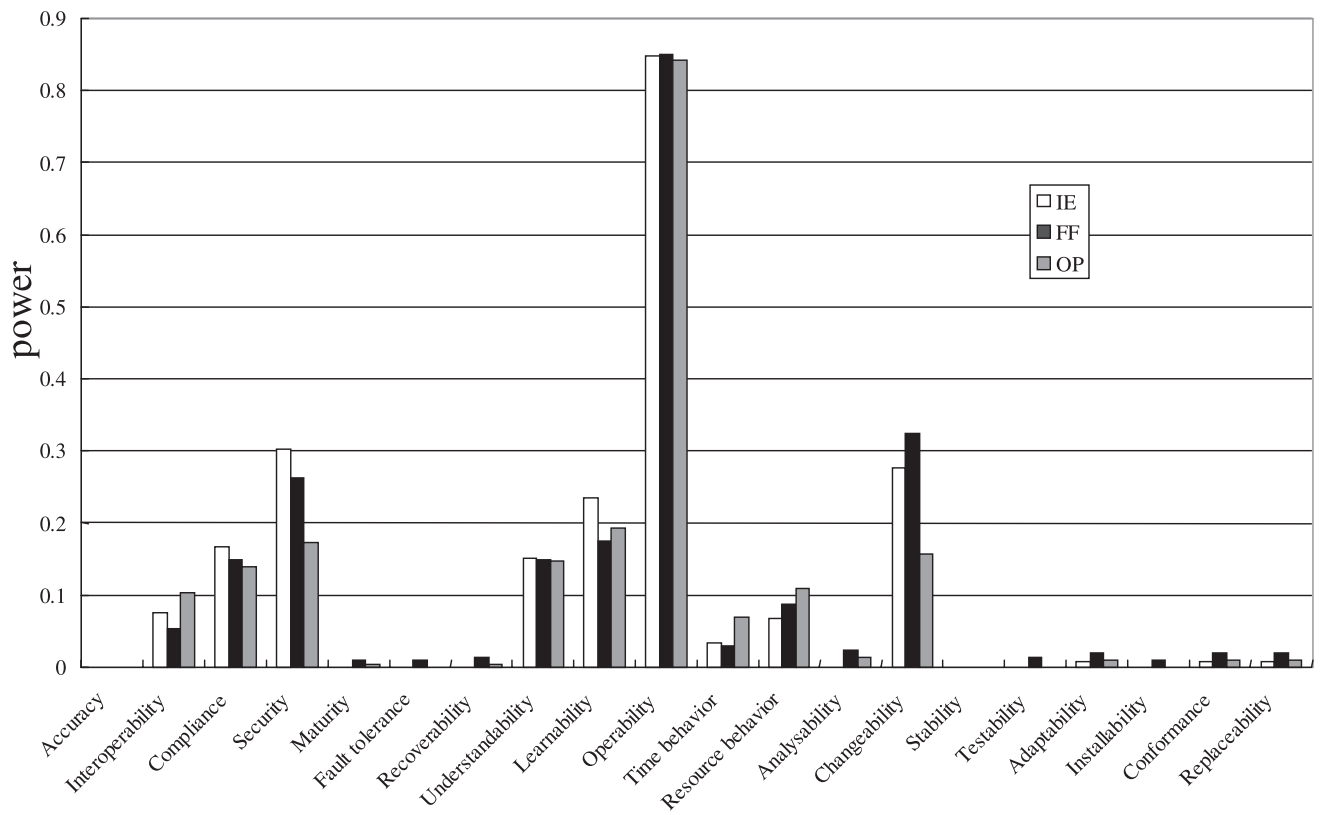


Fig. 6 Spectra of three Web browsers analyzed by subject A.

Figure 6 shows three spectra for each browser by subject A ("spectrum by A with TCM_A" in Fig. 4). Figure 7 shows three spectra by subject B ("spectrum by B with TCM_A" in Fig. 4). As mentioned in last subsection, A and B used the same TCM developed by subject A. As shown in these figures, the spectra for each subject are similar. In the

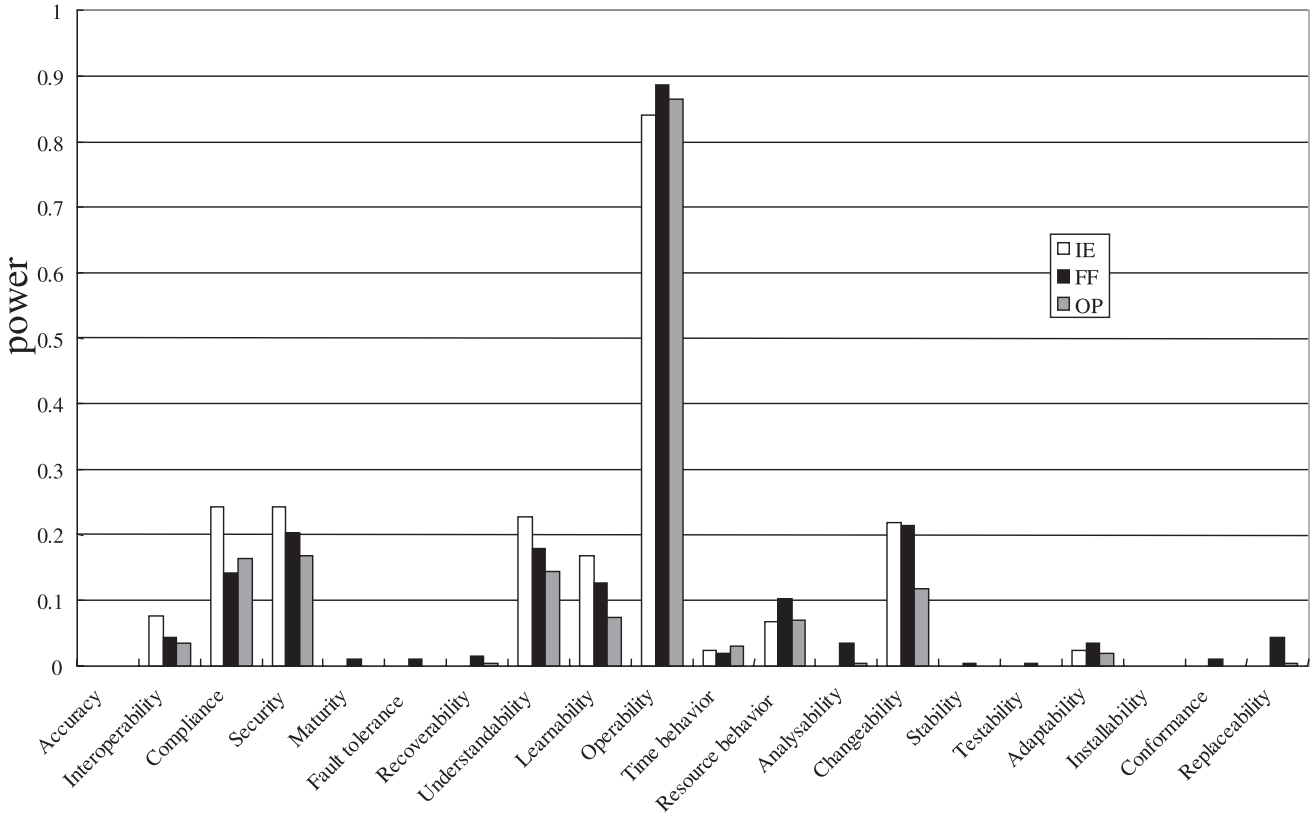


Fig. 7 Spectra of three Web browsers analyzed by subject B.

case of subject A in Fig. 6, cosine similarity between two out of three browsers are 0.99, 0.98 and 0.98. In the case of subject B in Fig. 7, cosine similarity between two out of three browsers are 0.98, 0.97 and 0.99. Therefore, we may regard different systems in the same domain have similar quality spectrum generated by TCM method.

We also have quality spectra of software systems other than browsers, and we show bar charts of both browsers and other types of systems in Fig. 8. A system labeled "NEWS" is a kind of a proxy system for feeding news articles to a specific intranet, and another system labeled "DB" is a kind of a document management system. Documents for both systems were published by our government [4]. Both systems are neither browsers nor interactive systems. Therefore, the spectra of NEWS and DB should be different from spectra of browsers. As shown in Fig. 8, spectra of NEWS and DB are clearly different from spectra of browsers. Cosine similarity between NEWS or DB and each browser is almost 0.5. If a system is interactive and security focused system but the system is not a kind of Web browsers, the quality spectrum of the system can be similar to spectra of web browsers. Therefore, we cannot say systems in different domains have different quality spectra. However, comparison of quality spectra of systems in the same domain is meaningful because systems in the same domain have similar quality spectrum and different types of systems, e.g., browsers and a proxy system like "NEWS", have different quality spectra.

As shown in Figs. 6 and 7, the power (amplitude) of some characteristics is different within three browsers. For example, powers of "interoperability" and "changeability" are different with each other. Because these three browsers are different with respect to its license (open source software or not), its platform (multi-platform including mobile devices or not) and so on, quality characteristics such as "interoperability" and "changeability" will be differently focused. On the other hand, powers of "operability" are similar because this characteristic is important for interactive systems such as browsers in general. Comparison here is the typical example of the usage mentioned in Sect. 2.2.

3.4 Different Analysts

Because one of the expected advantages of TCM method is that the result is more objective than the old method. To confirm this advantage, we compare a spectrum by subject A with TCM_A and another by B with TCM_A in Fig. 5. Because cosine similarity between these two spectra is 0.99, we may regard results by using TCM method is almost the same. On the other hands, cosine similarity between a spectrum by B with TCM_A and another spectrum by B with TCM_B is 0.75. We may also regard these two spectra are slightly different with each other. As a result, sharing TCM seems to help analysts to analyze requirements documents objectively. As mentioned in 3.2, subjects performing TCM method only use general spreadsheet. Therefore, they have to achieve

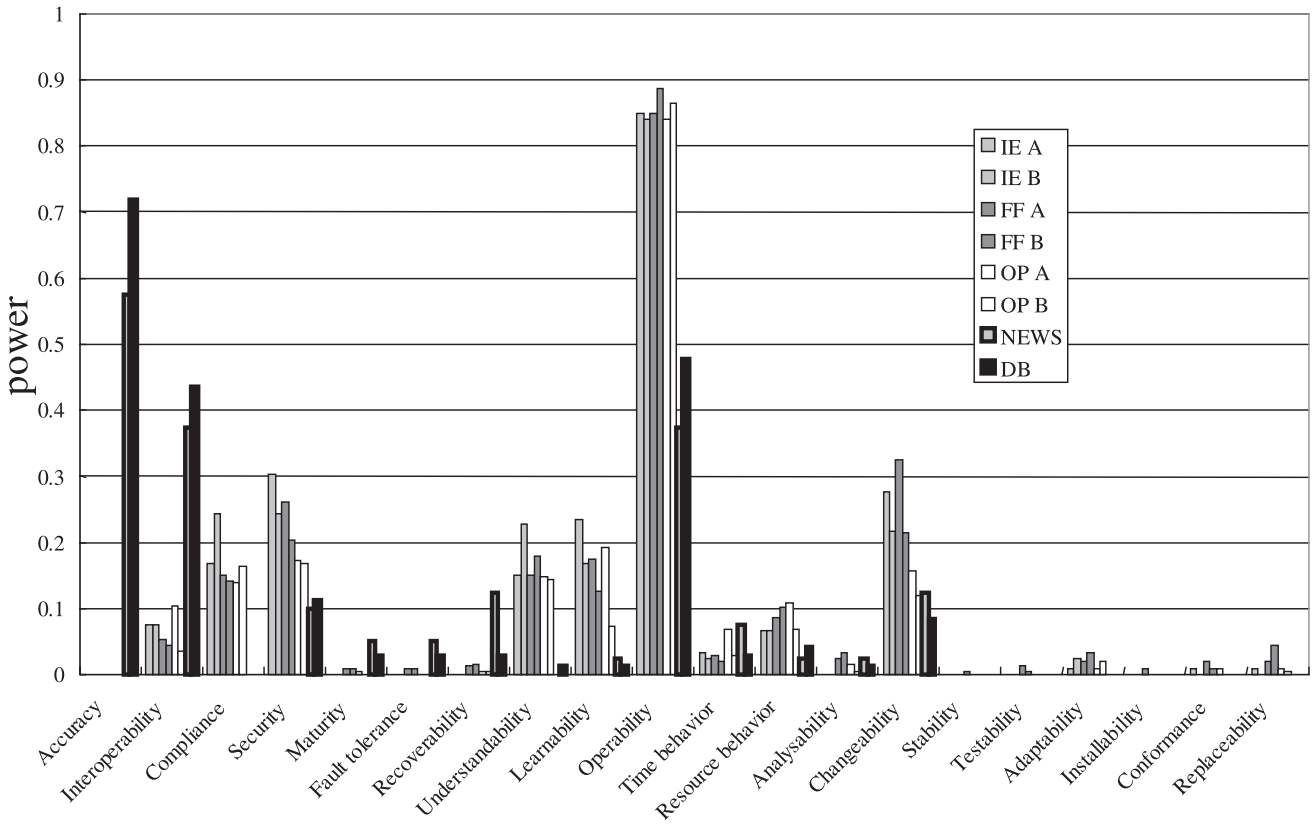


Fig. 8 Spectra of three Web browsers (IE, FF and OP) analyzed by subject A and B respectively and spectra of other types of systems (NEWS and DB).

tedious tasks that can be performed automatically because no supporting tools existed.

There can be a threat to validity whether each subject performed spectrum analysis with TCM objectively. As shown in Sect. 2.3, TCM method contains several subjective decision. The amount of such decision increases if given TCM is not appropriate. For example, an analyst has to make a lot of subjective decision if TCM contains a lot of mistakes. In this case, both subjects used TCM developed by subject A, and not only subject A but also one of the authors together reviewed and discussed the TCM itself and the results of the old method by subject A. Therefore, the TCM was appropriate. As a result, there is no such a threat to validity.

4. A Supporting Tool for TCM Method

Through evaluation in the last section, we can confirm TCM method seems to work well. To improve the efficiency of the task using TCM method, we are developing a supporting tool as shown in Fig. 9. In an example in this figure, NEWS system mentioned in Fig. 8 is analyzed. As stated in Sect. 2.3, TCM method includes both subjective and automatic tasks. Therefore, its supporting tool should be interactive one.

Before performing quality spectrum analysis, someone especially domain expert has to perform the following task.

1. To identify the domain of a system which requirements are analyzed.
2. To create or choose TCM of a domain.

An analyst then analyzes requirements with the help of following automatic tasks.

3. To look up terms appearing in each requirement in TCM, and to look up characteristics related to each term.
4. To generate quality spectrum by counting the number of requirements related to each quality characteristic.

The analyst finally performs the following tasks.

5. To choose quality characteristics actually related to each requirement based on TCM, contexts of each term and his expertise.

The tool in Fig. 9 supports its users to perform last four tasks above in the following ways.

1. The domain analyst can manually generate TCM by using “Term Characteristic Map (TCM)” tab in this figure (the contents of the tab are not shown in this figure). He first enumerates terms usually appearing in an application domain, and fill the checkboxes corresponding to quality characteristics for each term. By using this tool, candidates of terms can be automatically extracted and enumerated in “Term Characteristic Map (TCM)” tab from text documents. Therefore, the analyst can pick

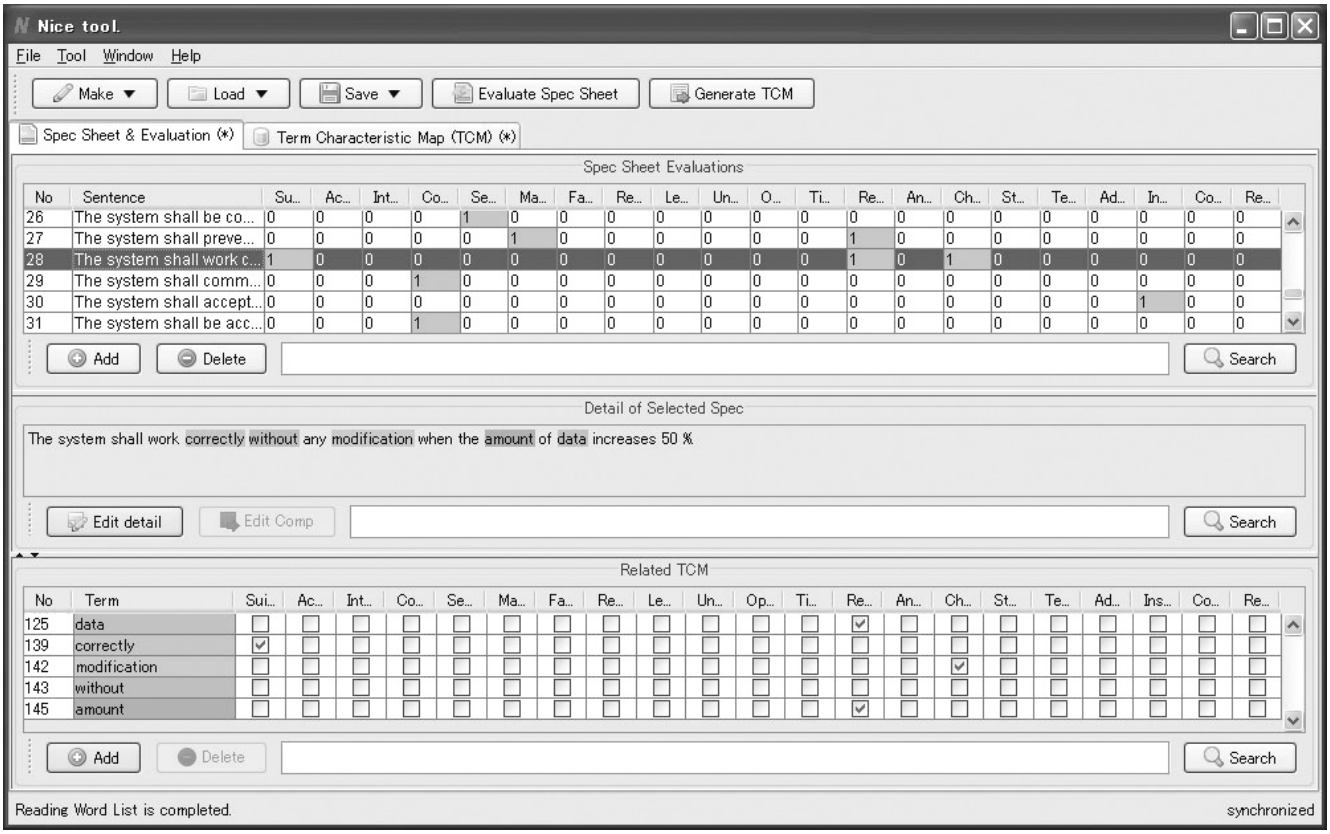


Fig. 9 A snapshot of a supporting tool.

terms out from such candidates.

2. As shown in the center area "Detail of Selected Spec" of the figure, terms are automatically identified according to the terms in pre-loaded TCM. A subset of the TCM is then shown at the bottom table "Related TCM" in the figure. In this example, five terms "correctly", "without", "modification", "amount" and "data" are looked up.
3. According to the checks on the table "Related TCM" in the figure, related quality characteristics are automatically accumulated at the top table of this figure "Spec Sheet Evaluations". Based on the checks of characteristics for each requirement (wrote "sentence" in this tool), the tool will visualize or output quality spectrum.
4. An analyst may freely change the value of checkboxes on the table "Related TCM" in this figure. Because this table is a copy of original TCM (generated in the first task), the original TCM gets no effects according to the changes.

5. Related Works

There are several studies how to define each quality requirement. In ISO25021 [9], concrete examples how to measure quality requirements are shown, and these examples help analysts to make quality requirements measurable. Donald Firesmith gives some format to specify quality requirements rigorously [10]. In ATAM (Architecture Tradeoff Analysis Method) [11], [12], a template for quality requirements called "quality attribute scenario" is provided to support stakeholders writing quality requirements.

Studies mentioned above focus on the *micro-view* of quality requirements because they focus on each requirement. Quality spectrum analysis [2] and its extension in this paper rather focus on the *macro-view* because it focuses on distribution of quality requirements in an artifact such as a requirements specification. We think both views are important to improve the quality requirements analysis, but there are few studies with macro-view. Studies (e.g., [13]) about the quality of requirements documents, e.g., completeness, correctness, and so on mentioned in IEEE 830 [14], also focus on the macro-view, but this kind of studies is not directly related with studies of quality requirements.

In an article by Ozkayad et al. [15], an empirical data of the most common quality attributes was shown based on the ATAM. The idea to analyze this kind of data is similar to quality spectrum analysis [2], but comparative analysis mentioned in introduction is not proposed in the article [15]. In DDP (defect detection prevention) [16], [17], the relationships among requirements, risks and their mitigations are visualized. Although this visualization shows a macro-view of quality requirements, trade-offs between risks and their mitigation costs are mainly focused.

Basic idea about relationships between requirements

and quality characteristics seems to be imported from QFD (Quality Function Deployment) [18]. TCM as domain knowledge is imported from a probabilistic model among terms, documents and queries in a paper by Cleland-Huang et al. [19].

6. Conclusion

In this paper, we improved quality requirements spectrum analysis proposed in our previous paper [2] by introducing term-characteristic map (TCM) as domain knowledge. Quality requirements spectrum analysis is a technique for measuring and tracking quality requirements over a requirements document written in natural language. TCM helps analysts to derive amplitude of each characteristic in a quality spectrum because TCM plays a role of domain knowledge for finding quality characteristics related to each requirements statement. Through several experiments, we evaluate quality requirements analysis method with TCM, and confirmed TCM method inherited some features of the old method in our previous paper [2] and results by TCM method became more objective.

Even if TCM for a domain can be reused, it is still hard to develop and maintain TCM for each domain. A technique called LSA (Latent Semantic Analysis) [20] seems to be used for developing and maintaining TCM because terms with the similar meaning can be found automatically by using LSA. In addition, LSA seems to be used to look up quality characteristics in TCM because the semantic similarity of a requirement sentence and a quality characteristic with its description can be calculated based on term occurrences in such sentences and descriptions.

Requirements documents written in natural languages are only analyzed now, but we would like to apply quality spectrum analysis to other types of artifacts. In a paper by Zhang et al. [21], UML notation is extended for representing quality attributes. In a paper by Chowdhury et al. [22], detailed characteristics about security in source codes are identified. These studies can be used to develop methods for quality spectrum analysis for design and source codes.

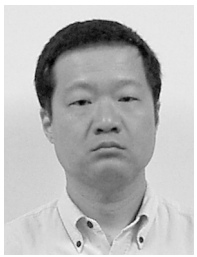
As mentioned in 2.3, we do not specify the degree of relationships between terms and characteristics. In the same way as shown in a paper by Cleland-Huang [19], frequency of relationships among requirements documents in the same domain can be used for specifying such a degree. We do not also specify the degree of relationships between requirements and characteristics. Frequency of terms in each requirement can be used for specifying such a degree. In addition, types of requirements representation can be used. In an article by Glinz [23], several different types of representations, e.g., qualitative, by example, quantitative and so on, are proposed. By using such types, we can give higher degree to a requirement sentence if a quality requirement is represented not qualitatively but quantitatively, for instance.

Currently, we only focus on coarse-grained quality as characteristics in a system, but we may use another kind of characteristics scattered over the system, e.g., fine-grained quality or an attention for each type of stakeholders. In an article by Ozkayad et al. [15], fine-grained quality characteristics about security can be found, and they can be used for quality spectrum analysis. In a book by Macaulay [24], types of stakeholders are shown such as designer, financial person, maintainer and users, and each stakeholder is interested in different part of requirements. We can perform "stakeholder spectrum analysis" over a document based on such types. That is the reason why we regard spectrum analysis over software artifacts is general.

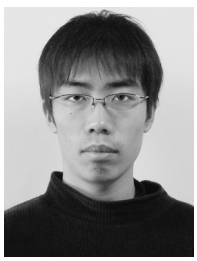
References

[1] J.D. Blaine and J. Cleland-Huang, "Software quality requirements: How to balance competing priorities," IEEE Softw., vol.25, no.2, pp.22–24, March/April 2008.
[2] H. Kaiya, T. Sato, A. Osada, N. Kitazawa, and K. Kaijiri, "Toward quality requirements analysis based on domain specific quality spectrum," Proc. 23rd Annual ACM Symposium on Applied Computing 2008, vol.1 of 3, pp.596–601, ACM, Fortaleza, Ceara, Brazil, March 2008. Track on Requirements Engineering.
[3] "IEEE recommended practice for software requirements specifications," 1998. IEEE Std. 830-1998.
[4] Minister of Economy, Trade and Industry, Japan, "http://www.meti.go.jp/feedback/data/i30728aj.html"
[5] C. Ncube, J. Lockerbie, and N.A.M. Maiden, "Automatically generating requirements from * models: Experiences with a complex airport operations system," REFSQ, pp.33–47, 2007.
[6] H. Kaiya, A. Osada, and K. Kaijiri, "Identifying stakeholders and their preferences about NFR by comparing use case diagrams of several existing systems," IEICE Trans. Inf. & Syst., vol.E91-D, no.4, pp.897–906, April 2008.
[7] N.A.M. Maiden and M. Hare, "Problem domain categories in requirements engineering," Int. J. Hum.-Comput. Stud., vol.49, no.3, pp.281–304, 1998.
[8] International Standard ISO/IEC 9126-1, "Software engineering — Product quality — Part 1: Quality model," 2001.
[9] International Standard ISO/IEC 25021, "Software engineering — Software product quality requirements and evaluation (SQuaRE) — Quality measure elements," Oct. 2007.
[10] D. Firesmith, "Quality requirements checklist," J. Object Technology, vol.4, no.9, pp.31–38, Nov.-Dec. 2005.
[11] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and J. Carriere, "The architecture tradeoff analysis method," IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), pp.68–78, 1998.
[12] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 2nd ed., Addison-Wesley, 2003.
[13] A. Bucchiarone, S. Gnesi, and P. Pierini, "Quality analysis of NL requirements: An industrial case study," IEEE International Requirements Engineering Conference (RE 2005), pp.390–394, 2005.
[14] "IEEE recommended practice for software requirements specification," Oct. 1998. IEEE Std 830-1998, ISBN 0-7381-0332-2 SH94654 (Print).
[15] I. Ozkayad, L. Bass, R.S. Sangwan, and R.L. Nord, "Making practical use of quality attribute information," IEEE Softw., vol.25, no.2, pp.25–33, March/April 2008.
[16] S.L. Cornford, M.S. Feather, J.C. Kelly, T.W. Larson, B. Sigal, and J.D. Kiper, "Design and development assessment," Proc. Tenth International Workshop on Software Specification and Design (IWSSD'00), pp.105–114, 2000.
[17] M.S. Feather, S.L. Cornford, K.A. Hicks, J.D. Kiper, and T. Menzies, "A broad, quantitative model for making early requirements decisions," IEEE Softw., vol.25, no.2, pp.49–56, March/April 2008.

[18] G. Herzwurm, S. Schockert, and W. Pietsch, "QFD for customer-focused requirements engineering," Proc. 11th IEEE International Requirements Engineering Conference, pp.330–338, Sept. 2003.
[19] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, and S. Christina, "Goal-centric traceability for managing non-functional requirements," International Conference on Software Engineering (ICSE), 2005.
[20] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," J. Society for Information Science, vol.41, no.6, pp.391–407, 1990.
[21] Y. Zhang, Y. Liu, L. Zhang, Z. Ma, and H. Mei, "Modeling and checking for non-functional attributes in extended UML class diagram," Annual IEEE International Computer Software and Applications Conference (COMPSAC2008), pp.100–107, 2008.
[22] I. Chowdhury, B. Chan, and M. Zulkernine, "Security metrics for source code structures," International Workshop on Software Engineering for Secure Systems (SESS2008), pp.57–64, 2008.
[23] M. Glinz, "A risk-based, value-oriented approach to quality requirements," IEEE Softw., vol.25, no.2, pp.35–41, March/April 2008.
[24] L.A. Macaulay, Requirements Engineering, Applied Computing, Springer, 1996.



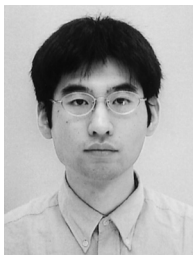
Haruhiko Kaiya is an associate professor of Software Engineering in Shinshu University, Japan. He is also a visiting associate professor in NII. http://www.cs.shinshu-u.ac.jp/˜kaiya/



Masaaki Tanigawa is a graduate school student in Shinshu University, Japan.

Shunichi Suzuki is a graduate school student in Shinshu University, Japan.

Tomonori Sato was a graduate school student in Shinshu University, Japan.



Akira Osada is a researcher in Shinshu University, Japan.

Kenji Kaijiri is a professor of Software Engineering in Shinshu University, Japan.