A Verification Method of Time-response Requirements

Yuuma Matsumoto and Atsushi Ohnishi
Department of Computer Science, Ritsumeikan University, Kusatsu, Japan

Keywords: Non-functional Requirements, Time-Response Requirements, Requirements Frame, Verification of Non-functional Requirements.

Abstract: In order to verify the correctness of functional requirements, we have been developing a verification method of the correctness of functional requirements specification using the Requirements Frame model. In this paper, we propose a verification method of non-functional requirements specification, especially time-response requirements written with a natural language. We establish a verification method by extending the Requirements Frame model. We have also developed a prototype system based on the method using Java. The extended Requirements Frame model and the verification method will be illustrated with examples.

1 INTRODUCTION

Software requirements should hold some characteristics in IEEE Std830 (IEEEstd98). We have developed a requirements frame and requirements language named X-JRDL based on the requirements frame to improve the characteristics of software functional requirements(Ohnishi1996). We can detect lack of indispensable cases and wrong noun types using X-JRDL, but since this language aims to specify functional requirements, we cannot improve the characteristics of non-functional requirements.

In this paper, we propose an extended requirements frame and a verification method of non-functional requirements based on the extended requirements frame model.

In the next section, we illustrate the original requirements frame model and the requirements language. In Section 3, we describe an extended requirements frame model and a verification method of non-functional requirements specification with examples. In Section 4, related works will be described. In Section 5, we conclude our research.

2 REQUIREMENTS FRAME MODEL

Consider requirements of a library system of book retrieval as below.

> *There exist users, cards of retrieval of books, and Identifier (ID) number of each book. Users are human-type objects. Cards and ID are data-type objects. Cards are classified into authors-cards that are sorted by author's name in alphabetical order, and title-cards sorted by title. A user can retrieve books with these cards.*

A requirements definer first identifies objects (nouns), object types (attributes) in a target system. Secondly he defines operations among objects (verbs) and roles of the operations (cases), and then constructs requirements sentences. The "cases" mean concept about agents, objects, goals of the operations (Shank1977). Thus, a requirement sentence includes nouns and verbs as its components, and there exist roles of objects as relations among the components. A particular functional requirement may be defined with several sentences. Our requirements model named Requirements Frame Model has been developed to easily represent the above structures. It involves two kinds of frames. These are a frame of noun level and a frame of sentence level (Ohnishi1996).

2.1 Noun Frame

The first frame is the Noun Frame, a frame whose components are nouns and their types. Table 1 shows the noun types provided to specify file-oriented software requirements. A new noun appearing in a requirements description will be classified into one of these types.

Matsumoto, Y. and Ohnishi, A.
A Verification Method of Time-response Requirements.
DOI: 10.5220/0005973101490156
In *Proceedings of the 11th International Joint Conference on Software Technologies (ICSOFT 2016) - Volume 1: ICSOFT-EA*, pages 149-156
ISBN: 978-989-758-194-6

Table 1: Noun types of the Noun Frame.

Type of noun	Meaning
human	active and external object
function	active and internal object
file	passive object of information set
data	passive object of a single information
control	passive object for control transition
device	passive object of an instrument

Table 2: Concept of the Case Frame.

Concept	Meaning
DFLOW	Data flow
FLOW	Control flow
ANDSUB	And-tree structure
ORSUB	Or-tree structure
GEN	Data creation
RET	Retrieve a record in a file
UPDATE	Update a record in a file
DEL	Delete a record in a file
INS	Insert a record in a file
MANIP	File manipulation
EQ, NE, LT, GT, LE, GE	Logic operators

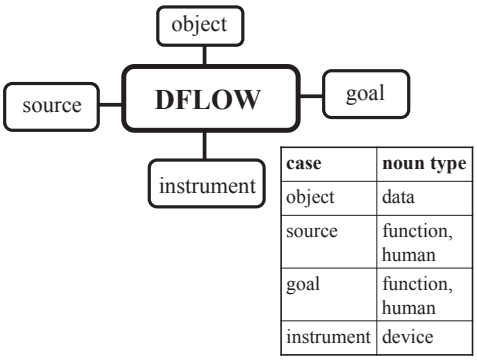
2.2 Case Frame

The second frame is the **Case Frame**, a frame whose components are nouns, verbs and cases. We provide seven different cases; *agent, goal, instrument, key, object, operation* and *source*. We also provide 16 different concepts including *data flow, control flow, data creation, file manipulation, data comparison,* and *structure of data/file/function.* There are several verbs to represent one of these concepts. For example, to specify a concept *data flow*, we may use *input, output, print out, display, send*, and so on. A requirements definer can use any verbs as far as it can be categorized in these 16 concepts provided.

We prepare these concepts to specify requirements of a file-oriented software domain. When a user wants to write requirements of another domain, he may need a verb not categorized into these concepts. In such a case, he can use a new verb if he defines its case structure. Since a newly defined verb, its concept, and its case structure can be registered in the verb dictionary, he can use his own verbs as well as provided verbs.

The 16 concepts (10 verb type concepts and 6 adjective type concepts) are shown in Table 2.

The Case Frame defines case structures of these concepts. For example, the *data flow* (DFLOW) concept has *agent, source, goal,* and *instrument* cases. The agent case corresponds to a data which is transferred from the source case object to the goal case object. So, an object assigned to the agent case should be a data type object. An object in the source or goal



case	noun type
object	data
source	function, human
goal	function, human
instrument	device

Figure 1: Case Frame of the Concept, "DFLOW".

Table 3: Analysis of a requirement sentence.

"A user enters a retrieval command with a terminal."

Concept	DFLOW
agent	a retrieval command
source	a user
goal	** undefined **
instrument	a terminal

cases should be either a human or a function type object. If and only if a human type object is assigned to source or goal cases, some device type object should be specified as an instrument case. These are illustrated in Fig. 1. Each concept has its own case structure.

The Case Frame enables to detect illegal usage of data and lack of cases. Suppose a requirement sentence, *"A user enters a retrieval command with a terminal."* Since the objective is *"a retrieval command"* that is data type noun, *"enters"* should be categorized into the DFLOW concept. With the Case Frame of the DFLOW, this sentence will be analyzed as shown in Table 3.

In this sentence the goal case object is omitted. The case structure of DFLOW says the goal case should be a noun of function type or human type. Previously specified nouns of the type become candidates of the omitted case. In this way, a requirement sentence is transformed into an internal representation named **CRD** (Conceptual Requirements Description). CRD is exactly based on the Noun Frame and the Case Frame.

2.3 Requirements Language: X-JRDL

We have developed a text-base requirements language named **X-JRDL**. This is based on the Requirements Frame model.

In X-JRDL, compound sentences and complex sentences will be divided into simple sentences each

Table 4: Verbs in the dictionary.

Concept	part of registered verbs
DFLOW, CFLOW	pass, move, receive, input
ANDSUB,ORSUB	subpart, part, construct
GEN	generate, produce, make
RET	retrieve
INS	insert, add
UPDATE	update
DEL	delete

of which has just one verb for analysis with the Case Frame. These simple sentences are transformed into CRD. We adopted X-JRDL as a requirements language in the course of Information Systems, graduate school of Information Sci., Kyoto University. Students specified SRSs of 50-500 sentences with this language. In this course we found 85 % of sentences were interpretably accepted by the X-JRDL analyzer and others needed to be pre-edited.

2.4 Analysis of X-JRDL Description

An X-JRDL description is analyzed through three interpreters. Since X-JRDL allows compound sentences and complex sentences, a surface interpreter divides them into simple sentences. Another interpreter, word interpreter, fulfills a case structure consulting with dictionaries. Since a noun is interpreted with its type, the noun dictionary holds a name and its type. A verb (or an adjective) is interpreted with its corresponding concept. In the case of pronoun and omission of nouns, its type will be guessed with the Case Frame. A sentence interpreter transforms a simple sentence transformed into CRD with checking lacks of indispensable cases.

X-JRDL allows using pronouns and omission of nouns. We frequently come across such features in Japanese sentences. The X-JRDL analyzer automatically assigns a concrete word into a pronoun or a lacked case.

Conjunctions are used to write down compound sentences and complex sentence. The analyzer divides such a sentence into a set of simple sentences.

The X-JRDL analyzer has a dictionary of nouns, verbs and adjectives. When a requirements definer uses a word which is not appeared in the dictionary, the analyzer guesses a type of new noun and a concept of new verb and adjective with the Requirements Frame. Table 4 shows registered verbs. Really these verbs are Japanese verbs. The analyzer can treat inflection of these verbs.

A same requirement can be described differently. For example, the previous requirement sentence *"A user enters a retrieval command with a terminal"* can be expressed as "A terminal receives a retrieval command from a user" or "A retrieval command can be passed from a user to a terminal." The CRDs of these three sentences are exactly the same. In other words, if the CRDs are the same, the meanings of requirement sentences are the same.

Since X-JRDL is focused to express functional requirements, non-functional requirements (NFRs) cannot be described.

Table 5: Keywords related to time-response requirements.

Keywords related to time-response requirements
response, respond, turnaround,
within(in) x (milliseconds/microseconds) seconds

3 VERIFICATION OF TIME-RESPONSE REQUIREMENTS

By extending the Requirements Frame, we can analyze sentences of NFRs and transformed into CRDs. In this paper, we focus on time-response requirements. This means our approach is limited to verification of time-response requirements.

Saito et al. propose a machine learning approach to evaluate the clarity of NFRs described in a Request For Proposal (RFP) written in a natural language (Saito2013). In this method, keywords related to NFRs are extracted from a RFP, and mapped to each NFR category. Then, the clarity of NFRs is modeled by the random forest with weight factors based on appearance frequency and context vectors. As a result of an experiment to evaluate the clarity (low, mid or high) of many NFR categories in 70 RFPs, the proposed method showed 69.8% match to the expert's decision. They clarified 18 keywords related to time-response requirements. We select 4 keywords as shown in Table 5.

With these 4 keywords, we retrieve requirements sentences in 20 RFPs. The results are shown in Fig. 2. In this figure, the same sentences are merged into one.

The seventh sentence is not a time-response requirement, because the agent (the presenter) is not a system or a function that responds. We do not regard a sentence whose responder is a human as a time-response requirement. We manually checked whether there are any other time-response requirements, but we could not find. So, with these keywords it is enough to retrieve time-response requirements.

1. The response time shall be within three seconds at normal time and within five seconds at peak time.
2. The standard response time shall be within 3 seconds.
3. The response time shall be same or less than the response time of the as-is system.
4. The server shall respond within a reasonable time.
5. The minimum time a bank card must be inserted to guarantee it is recognized in 200 milliseconds.
6. The time to generate a dial tone once a caller's phone is detected off hook should not exceed one second.
7. The presenter can respond to questions.

Figure 2: Retrieved time-response requirements.

3.1 Requirements Frame for Time-response Requirements

We introduce a new requirements frame to represent time-response requirements. We provide one action, that is, "respond " and its three cases, that is, "agent case," "goal case," and "condition case." The agent case corresponds to a noun that responds. The goal case corresponds to performance objective in response. So, goal case should be quantitative attributes. The condition case corresponds to condition or environment in response. Table 6 shows manually transformed into internal representation of the first six retrieved time-response requirements in Fig. 2 using this requirements frame. In the transformation, surficial representation such as "The response time (of the system) shall be within three seconds" can be transformed into conceptual representation "(The system) responds within three seconds." In this Figure, "-" means corresponding word is missing.

In (ISO/IEC/IEEE29148:2011), examples of requirement syntax are proposed as shown in Fig. 3. Requirement sentences based on the syntax in (ISO/IEC/IEEE29148:2011) can be transformed into internal representations based on the extended Requirements Frame in Table 6, because the agent case in the extended Requirements Frame corresponds to "Subject" in Fig. 3, and the goal case corresponds to "Constraint."

3.2 Verification Method of Time-response Requirements

We focus on the following qualities of time-response requirements.

1. non-redunancy
2. unambiguity
3. consistency
4. completeness

We can detect redundant requirements if there exist multiple requirements whose agent cases, whose goal cases, and whose condition cases are same nouns, respectively.

We can detect ambiguous requirements in terms of time-response if there exists a requirement whose goal case is missing or qualitative.

We can detect inconsistent requirements, if there exist two or more requirements whose agent cases are same and whose condition cases are same, but whose goal cases are different.

We can detect lack of time-response requirements if there exists a noun that should respond but there is no time-response requirement whose agent case is the noun.

We can detect potential errors if there exist two or more requirements whose agent cases are same and whose goal cases are same, but whose condition cases are different. In such a case, these requirements should be merged into one requirement by logically merging their condition cases.

Verification procedures of time-response requirements are shown in Fig. 4.

In Table 6, the first three requirements have no agent cases. The first two requirements are not inconsistent, because the condition cases are different, although the missing agent cases are same and goal cases are different. The fourth requirement is ambiguous if the response time of the as-is system cannot be referred. The fifth requirement is ambiguous, because the goal case is qualitative and not quantitative.

We can detect the ambiguity, the inconsistency, the completeness, and the redundancy of time-response requirements with our method. Actually, the redundancy of requirements is not an error, but in case of modification if one requirement is changed and the other is not changed, this modification may cause the inconsistency.

3.3 Prototype

Fig. 5 shows the whole system of verification of time-response requirements. We use an existing text-based retrieval system as the Retrieval system in this figure. Retrieved time-response requirements are automatically processed by a natural language processor or manually processed by a user, and then transformed into internal representaion based on the extended Requirements Frame. The error detecting system gets

Table 6: Transformed time-response requirements in Fig. 2.

agent	action	goal	condtion
-	respond	within three seconds	normal time
-	respond	within five seconds	peak time
-	respond	same or less than the response time of the as-is system	-
-	respond	within three seconds	-
the server	respond	within a reasonable time	-
-	respond	in 200 milliseconds	bank card insertion
-	respond	one second	dial tone generation once a caller's phone is detected

[Condition][Subject][Action][Object][Constraint]

EXAMPLE: When signal x received [Condition], the system [Subject] shall set [Action] the signal x received bit [Object] within 2 seconds [Constraint].

or

[Condition][Action or Constraint][Value]

EXAMPLE: At sea state 1 [Condition], the Radar System shall detect targets out to [Action or Constraint] 100 nautical miles [Value].

or

[Subject][Action][Value]

EXAMPLE: The Invoice System [Subject], shall display pending customer invoices [Action] in ascending order [Value] in which invoices are to be paid.

Figure 3: Examples of requirement syntax (ISO/IEC/IEEE29148:2011).

1. Retrieve time-response requirements in an SRS using the four keywords.
2. User checks whether each of retrieved requirements is time-response requirements or not, and omits non-time-response requirements.
3. The time-response requirements will be analyzed with a natural language processor in order to analyze the syntax of the sentences.
4. Transform the analyzed results by the natural language processor into internal representation based on the extended requirements frame model.
5. Detect the redundancy, inconsistency, incompleteness, or ambiguity of the time-response requirements.

Figure 4: Verification Procedure of time-response requirements.

the internal representation and produce a report of detected errors.

We have developed a prototype system of error detection using Java with Eclipse 4.4 Luna. The number of source code lines is about 500. This is a 2 man-month product. This system supports the first, the fourth and the fifth steps of the verification procedure in Fig. 4.

Figure 6 and 7 show snapshots of detecting an error with the prototype system. Original messages of the prototype system are in Japanese, so we add messages in English for English readers.

Using the prototype system, we can detect two ambiguous time-response requirements in 20 RFPs. These are the 3rd sentence and the 4th sentence in Fig. 2. The 4th sentence is really ambiguous, because the time-response requirement is defined qualitatively. In contrast, the 3rd sentence may not be ambiguous, if the response time of as-is system is quantitatively specified in other SRS. This requirement may be ambiguous, if the response time of as-is system is qualitatively specified in other SRS or not clearly specified.

4 DISCUSSION

Our method enables to detect the redundancy, ambiguity, inconsistency, and incompleteness of time-response requirements. We suppose a time-response requirement is expressed as a single sentence and we can analyze a time-response requirement sentence using the extended Requirements Frame described in Section 3.1. However, a time-response requirement

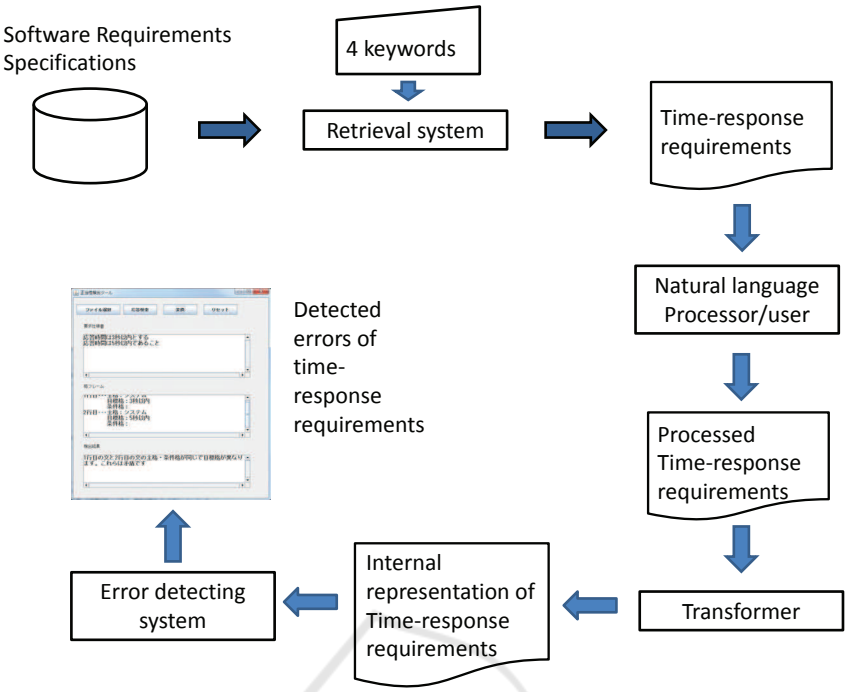


Figure 5: The whole system of verification of time-response requirements.

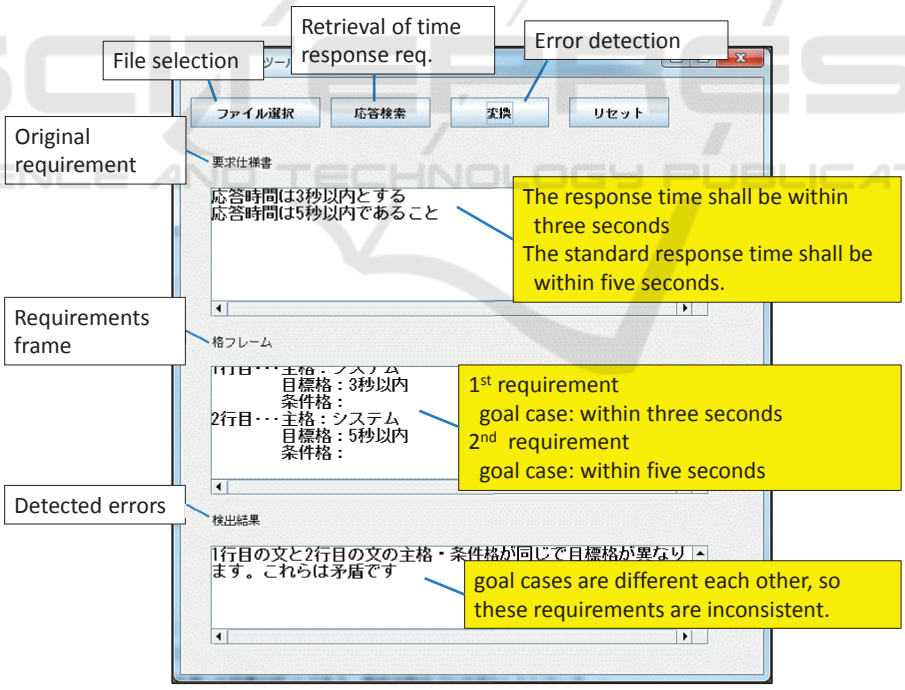


Figure 6: Snapshot of detecting the inconsistency between two requirements.

may be expressed as multiple sentences or expressed as a sentence and a table like the second sentence in Fig. 8.

In Fig. 8, each time-response requirement is expressed as non-single sentence. In such a case, it is difficult to analyze a time-response requirement with the extended Requirements Frame model. However such requirements can be retrieved with the keywords shown in Table 5. By transforming such requirements into a single sentence by hand, we can apply

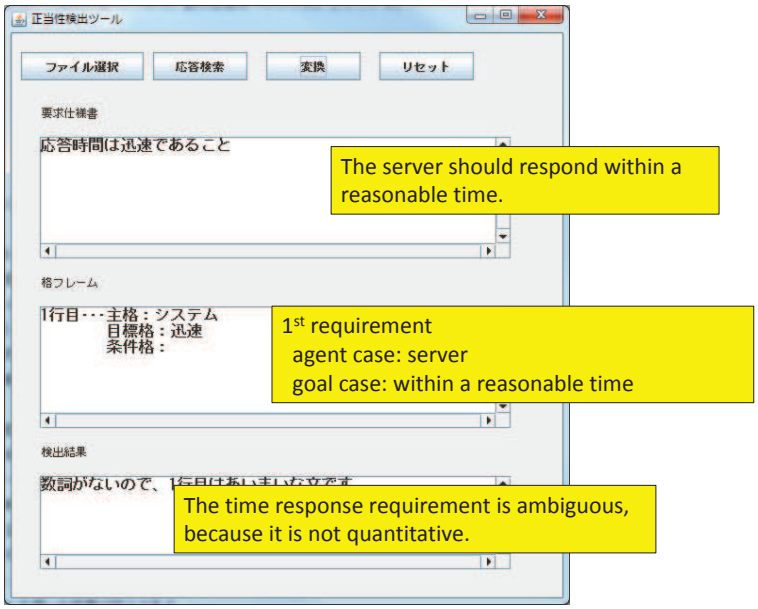


Figure 7: Snapshot of detecting an ambiguous requirement.

a) The response times of the system are shown as follows. At peak-time it should be within five seconds. At non-peak-time it should be within three seconds.
b) The response time of the system is shown in Table X.
c) The response time of the new system should be same as the response time of the current system.
d) The response time for normal use of the system should be specified. It is desirable that the response time is within three seconds.

Figure 8: Time-response requirements with non-single sentence.

our method to the transformed sentence.

5 RELATED WORKS

The NFR-frame work is a goal-oriented analysis method for non-functional aspects of a target system (Chung1999). Our method aims to verify the characteristics of an SRS, instead.

In (Cin2000), performance requirements written with a structured language will be transformed into Petri-net model and then the inconsistency and ambiguity will be checked. Our method enables to detect not only the inconsistency and ambiguity, but also the incompleteness and redundancy in requirements written with a natural language.

In (Cysneiros2005; Fatwanto2008), use-case oriented verification method of NFR is proposed. This method is useful to check scenarios or use-case descriptions, but is not suitable to check SRSs.

In (Kaiya2011), rule-based checking method of NFR is proposed. This method checks whether NFRs are specified or not and do not check the characteristics of NFRs.

In (Bo2011), a formal verification method of NFR is proposed, but it is not suitable for SRSs in a natural language.

In (Irfan2011), the importance of quantitative requirements is claimed, but they do not verify the characteristics of NFRs.

6 CONCLUSION

In this paper, we propose a verification method of the unambiguity, the consistency, the completeness, and the redundancy of time-response requirements in SRS written with a natural language. We can detect ambiguous, inconsistent, incomplete, and/or redundant time-response requirements with our method. We also developed a prototype system based on the method.

Evaluation of the method by applying to other SRSs, and the establishment of verification method of other NFR are left as future works.

ACKNOWLEDGEMENTS

We thank to Associate Professor Hiroya Itoga, Assistant Professor Takayuki Omori, under graduate student Shouta Kasai, and other members of Software Engineering laboratory, Department of Computer Science, Ritsumeikan University for their contributions to the research. This research is partly supported by Grant-in-Aid for Scientific Research, Japan Society for the Promotion of Science.

REFERENCES

Bo, W., Bin, Y., Zhi, J., Zowghi, D.: "r sigma: Automated reasoning tool for non-functional requirement goal models," Proc. 19th International Requirements Engineering Conference (RE2011), 2011, pp.337-338.

Chung, L., Nixon, B.A., Yu, E., Mylopoulo,s J.: "Non-Functional Requirements in Software Engineering," Springer, 1999.

Cin, M, D.: "Structured Language for Specifications of Quantitative Requirements," Proc. The 5th IEEE International Symposium on High Assurance Systems Engineering (HASE), pp.221-227, 2000.

Cysneiros, L.M., Sampaio, P., Leite, J.C.S.P.: "Non-Functional Requirements: from Elicitation to Conceptual Model," IEEE Transaction on Software Engineering, IEEE, Vol. 30 No. 5, 2005, pp.328-350.

Fatwanto, A., Boughton, C.: "Analysis, Specification and Modeling of Non-Functional Requirements for Translative Model-Driven Development," Proc. International Conference on Computational Intelligence and Security (CIS'08), 2008, pp.405-410.

IEEE Standards Board: Section 4.3 "Characteristics of a good SRS," "IEEE Recommended Practice for Software Requirements Specifications," IEEE830-1998, 1998.

Irfan, M., Hong, Z.: "Key role of value-oriented requirements to develop real-time database systems," Proc. IEEE 2nd International Conference on Computing, Control and Industrial Engineering (CCIE), 2011, pp.405-408.

ISO/IEC/IEEE: Systems and software engineering - Life cycle processes- Requirements engineering, International Standard, first edition, 2011, pp.10-11.

Kaiya, H., Ohnishi, A.: "Quality Requirements Analysis Using Requirements Frames ," Proc. 11th International Conference on Quality Software (QSIC 2011), 2011, pp.198-207.

Ohnishi, A.: "Software Requirements Specification Database Based on Requirements Frame Model," Proc. 2nd International Conference on Requirements Engineering (ICRE '96), 1996, pp.221-228.

Saito, Y., Monden, A., Matsumoto K.: "Evaluation of RFPs Based on Machine Learning (in Japanese)," Vol.2013-SE-179, No.5, SIG Technical Report, IPS, Japan, pp.1-7, 2013.

Shank, R.: "Representation and Understanding of Text," *Machine Intelligence* 8, Ellis Horwood Ltd., Cambridge, 1977, pp.575-607.