Web-based Requirements Elicitation Supporting System using Requirements Categorization

Youngjoong Ko, Sooyong Park, Jungyun Seo

Department of Computer Science, Sogang University
Seoul, 121-742, Korea
kyj@nlpzodiac.sogang.ac.kr, {sypark,seojy}@ccs.sogang.ac.kr

Abstract

As software becomes more complicated and large-scaled, it is important for software engineers to understand user's requirements correctly. In typical large-scaled system development, there are various number and types of users, who locate in various places. To gather requirements from this kind of environments, web could be a good resource. One of the major problems using web is that software engineers can not understand and analyze the collected requirements well because they are unorganized and large amount at the initial phase of requirements elicitation.

This paper proposes a web-based requirements elicitation supporting system that offers effective requirements elicitation and analysis in distributed environments. The proposed system provides to gather requirements through web and automatically categorizes the collected requirements into selected subject fields by similarity measure technique. It relieves a burden on understanding a large amount of unorganized requirements in the initial stage of requirements elicitation. This paper proposes a web-based requirements elicitation supporting system, which automatically categorizes the collected requirements into selected subject fields using similarity measure technique. It relieves a burden on understanding a large amount of unorganized requirements in the initial stage of requirements elicitation. This paper proposes a web-based requirements elicitation supporting system, which automatically categorizes the collected requirements into selected subject fields using similarity measure technique.

1. Introduction

As software becomes more complicated and large-scaled, requirements analysis play an increasingly important role in a software development. Typically, in complicated and large-scaled software developments, there exist various user groups in different locations. Therefore, a requirements elicitation and analysis through their efficient participation is very important success factor in the software development[1][2]. An efficient requirements elicitation from various user groups needs to be worked in collaborative environment. Internet could be a useful resource for this need.

Due to the growth of internet, web has become a convenient interface for various working groups to collaborate for developing software in distributed environment. The use of web in order to collect requirements from various users in a requirements elicitation process could be an efficient way of overcoming spatial and temporal barriers[3][4]. In comparison to traditional requirements elicitation techniques[5] including scenario, questionary, interview and conversation, web may create some difficulties in requirements analysis because requirements are collected from distributed environment in disorganized manner. If various users offer their requirements through web, their efficient analysis becomes very difficult because their requirements present only one view about the system. For example, If each 40 user provides 50 requirements, we will get a total of 2,000 requirements. Then, it will be very difficult to analyze these requirements because their view points could be different, and their content itself could be dispersed and redundant. Therefore, if we can automatically categorize the collected requirements into each subject field, we will offer a good basis for analysis and we will remove difficulties of the initial analysis stage from unorganized requirements[6].

In this paper, we propose a web-based requirements elicitation supporting system, which automatically categorizes the collected requirements into selected subject fields using similarity measure technique. The system will offer a basis to analyze requirements. It will remove the spatial and temporal restrictions, which are generated from participation of various working groups in the elicitation and analysis stages of Requirements Engineering as shown in Figure 1.
The rest of this paper is organized as follow. Section 2 describes related works. In section 3, we propose a web-based requirements elicitation supporting system and explain similarity measure technique, which is used for sentence categorization. In section 4, we report experimental results. After showing the implementation of our system in section 5, Section 6 concludes with a discussion of future works and a summary.

2. Related Works

The automatic classification of requirement documents have used simple similarity measure using linguistic features such as reiteration and collocation of words[7]-[10]. Parmar proposed the TTC(Two-Tired Clustering) algorithm which indexes and clusters requirements specifications[9]. First, it identifies the verbs and keywords of each requirement, then it clusters these requirements by functionality. Next, each cluster is subdivided using cosine similarity among clusters. And Yang proposed an automated decomposition tool supporting requirements clustering, which is based on graph model[10]. As the method makes requirements located in each cluster according to each cohesion value, it can support requirements clustering analysis. However, these are not efficient methods of categorizing the requirement sentences collected in distributed environment, because the processing unit of our method is not a document but a sentence. Although the methods using a thesaurus have better performance in accuracy, it is difficult, domain specific, and time-consuming task to construct and manage the thesaurus for each application domain.

There are researches to apply natural language processing techniques to requirements elicitation. Lecoeuche described a framework for requirements elicitation using a theory of natural language dialogue[11]. Palmer embodied Mutigroup Decision Support System(MDGSS) and used it for efficient requirements elicitation and analysis from various user groups[12]. If these systems are embodied in web, they will provide more comfortable and useful environments for collaborative works.

Therefore, we propose a web-based requirements elicitation supporting system. It can automatically categorize collected requirements using similarity measure technique without the thesaurus.

3. Requirements Categorization

3.1 Overview of the proposed requirements categorization system

The proposed requirements categorization system consists of two modules as shown in Figure 2: a module to extract example sentences for each subject, and a module to measure word and sentence similarities. Subject example is the sentence that contains a subject word. Keyword example is the sentence that has a keyword. These examples are extracted from the collected requirements. We regard these examples as sentences that reflect the special features of any subject field.

![Figure 1: Requirements analysis process using the proposed system](image)

![Figure 2: Architecture for the proposed categorization system](image)
The sub-module to extract examples finds subject examples and keyword examples in the collected requirements. To support this function, the system makes use of a POS (Part-Of-Speech) tagger. The sub-module to measure the similarity of the sentences computes the similarities of unclassified requirement sentences to these examples. We define the unclassified requirement sentences as sentences that are not chosen as subject examples and keyword examples. And then, each unclassified requirement sentences is assigned into the subject field of the example with the highest similarity value.

3.2 Extracting subject examples and keyword examples

We extract the sentences that contain sufficiently the special features of each subject field. These sentences are extracted by using system engineer’s input subject words and keywords. They are subject examples and keyword examples. A keyword list consists of words that are highly similar to each subject word. In traditional research, a thesaurus and MRD (machine-readable dictionary) are used for generating the keyword lists[9][13]. However, the feature of words in requirement sentences is more dependent on an application domain than that of words in general corpus. In that sense, we can not expect high performance when using the thesaurus and MRD. Also, it is very difficult and time-consuming task to construct a thesaurus or MRD for each application domain.

3.2.1 Generating content word lists of requirement sentence

To extract subject examples and keyword examples, we have to extract content words, which contain features of sentence. In Korean, there are active-predicative common nouns, which become verbs when they are combined with verb-derivational suffixes (‘do (-지다)’, ‘become (-거나)’ etc.). Also, active-predicative common nouns become adjectives when they are combined with adjective-derivational suffixes (‘-하’ etc.). These cases appear frequently in Korean, and they are classified into nouns from Korean POS tagger. Verbs and adjectives without these cases are non-informative words in many cases, especially in requirements. In order to verify them, we analyzed our experiment data (180 requirement sentences). The total number of verbs and/or adjectives in our data is 449. As expected, 60% of them, i.e. 267 out of 449, are derived verbs and/or adjectives whose heads are active-predicate common nouns and/or stative-predicative common nouns, respectively. 86 (47%), out of the rest 182 verbs and/or adjectives, are non-informative words (‘be (있다)’, ‘not be (없다)’ etc.). Therefore, in this paper, we extract content words using noun according to Korean’s linguistic features and research domain.

There are non-informative words in these content words because they are universally presented in many sentences. To give an example, there are ‘management (관리)’, ‘part (부분)’ etc. To solve this problem, we make a stop word list and we eliminate non-informative words from content words using it.

3.2.2 Extracting example sentences

First, Subject examples, which have a subject word in their content words, are extracted from collected requirement sentences. Next, we extract keyword examples which have keywords. At this time, sentences with two or more subject words and keywords are excluded from examples. Figure 3 shows the process of extracting subject example.

3.3 Measuring word and sentence similarities

Unclassified requirement sentences are categorized into subject fields through measuring similarities of unclassified requirement sentences to subject examples and keyword examples. As similar words tend to appear in similar contexts, we compute the similarity by using contextual information[7][13]. In this paper, words and sentences play complementary roles. That is, a sentence is represented by the set of words it contains, and a word by the set of sentences in which it appears. Sentences are similar to the extent that they contain similar words, and words are similar to the extent that they appear in similar sentences. This definition is circular. Thus, it is applied iteratively using two matrices as shown in Figure 4.
In Figure 4, each subject field has a word similarity matrix $WSM_n$ and a sentence similarity matrix $SSM_n$. In each iteration $n$, we update $WSM_n$, whose rows and columns are labeled by all content words encountered in examples for each subject field and unclassified requirement sentences. In that matrix, the cell $(i,j)$ holds a value between 0 and 1, indicating the extent to which the $i$th word is contextually similar to the $j$th word. Also, we keep and update a $SSM_n$, which holds similarity values among sentences. The rows of $SSM_n$ correspond to unclassified requirement sentences and the columns to subject examples and keyword examples.

To compute the similarities, we initialize $WSM_n$ to the identity matrix. That is, each word is fully similar (1) to itself and completely dissimilar (0) to other words. The following steps are iterated until the changes in the similarity values are small enough.

1. Update the sentence similarity matrix $SSM_n$, using the word similarity matrix $WSM_n$.
2. Update the word similarity matrix $WSM_n$, using the sentence similarity matrix $SSM_n$.

### 3.3.1 Affinity formulae

To simplify the symmetric iterative treatment of similarity between words and sentences, we define an auxiliary relation between words and sentences, which we call affinity. A word $W$ is assumed to have a certain affinity to every sentence, which is a real number between 0 and 1. It reflects the contextual relationships between $W$ and the words of the sentence. If $W$ belongs to a sentence $S$, its affinity to $S$ is 1. If $W$ is totally unrelated to $S$, the affinity is close to 0. If $W$ is contextually similar to the words of $S$, its affinity to $S$ is between 0 and 1. In a similar manner, a sentence $S$ has some affinity to every word, reflecting the similarity of $S$ to the sentences involving that word.

Affinity formulae is defined as follows[13]. In this formulae, $W \in S$ means that a word belong to a sentence:

$$\text{aff}(W,S) = \max_{w \in S} \text{sim}(W,W_i)$$  
(1)

$$\text{aff}(S,W) = \max_{w \in S} \text{sim}(S,S_i)$$  
(2)

In the above formulae, $n$ denotes the iteration number, and the similarity value are defined by $WSM_n$ and $SSM_n$. Every word has some affinity to the sentence, and the sentence can be represented by a vector indicating the affinity of each word to it. But, note that affinity is asymmetric as follows:

$$\text{aff}(W,S) \neq \text{aff}(S,W)$$  
(3)

### 3.3.2 Similarity formulae

The similarity of $W_1$ to $W_2$ is the average affinity of sentences that include $W_1$ to $W_2$, and the similarity of a sentence $S_1$ to $S_2$ is a weighted average of the affinity of the words in $S_1$ to $S_2$. Similarity formulae is defined as follows[13]:

$$\text{sim}_n(W_1,S_2) = \sum_{W \in S_1} \text{weight}(W) \cdot \text{aff}(W,S_2)$$  
(4)

if $W_1 = W_2$

$$\text{sim}_n(W_1,W_2) = 1$$

else

$$\text{sim}_n(W_1,W_2) = \sum_{W \in S} \text{weight}(S,W) \cdot \text{aff}(S,W_2)$$  
(5)

The weights in Formula 4 are computed following the methodology in the next section. The sum of weights in Formula 5, which are a reciprocal number of sentences that contain $W_1$, is 1. These values are used to update the corresponding entries of $WSM_n$ and $SSM_n$.

### 3.3.3 Word weights

In Formula 4, the weight of word is a product of three factors. It is used to exclude the words that are expected to be given unreliable similarity values. The weights do not change in their process of iteration.

1. **Global Frequency**: Frequent words in total requirements are less informative of sense and of sentence similarity. For example, a word like ‘management ( khôan vi)’ frequently appears in total requirements. The formula is as follows[13]:

$$\max \left\{ 0.1 - \frac{\text{freq}(W)}{\max_{S} \text{freq}(S)} \right\}$$  
(6)
In (6), \( \max_{x} \sum \text{freq}(x) \) is the sum of the five highest frequencies in total requirements.

2. Log-likelihood factor: In general, the words that are indicative of the sense usually appear in subject and keyword examples more frequently than in total requirements. The log-likelihood factor captures this tendency. It is computed as follows[13]:

\[
\log \frac{\Pr(W_i | W)}{\Pr(W)} \tag{7}
\]

In (7), \( \Pr(W) \) is estimated from the frequency of \( W_i \) in the total requirements, and \( \Pr(W_i | W) \) from the frequency of \( W_i \) in subject examples and keyword examples. To avoid poor estimation for words with a low count in subject examples and keyword examples, we multiply the log-likelihood by (8) where \( \text{count}(W_i) \) is the number of occurrences of \( W_i \) in subject examples and keyword examples. For the words which do not appear in subject examples and keyword examples, we assign weight (1.0) to them. And the other words are assigned weight that adds 1.0 to computed value.

\[
\min \left\{ 1, \frac{\text{count}(W_i)}{3} \right\} \tag{8}
\]

3. Part of speech: Each part of speech is assigned a weight. We assign weight (1.0) to proper noun, non-predicative common noun and foreign word, and assign weight (0.6) to active-predicative common noun and stative-predicative common noun.

The total weight of a word is the product of the above factors, each normalized by the sum of factors of the words in the sentence as follows[13].

\[
\text{weight} = \sum_{W \in S} \frac{\text{factor}(W, S)}{\text{factor}(W, S)} \tag{9}
\]

In (9), \( \text{factor}(W, S) \) is the weight before normalization.

4. Experimental Evaluations

4.1 Experiment data

To evaluate the efficiency of the requirements categorization method, we set up an experiment as follows. For the experiment, we collected requirements from real fields. The data consisted of 10 documents with 180 sentences. Next, we divided the collected requirement sentences into five subject fields (‘attendance (근태)’, ‘salary (급여)’, ‘personnel (인사)’, ‘employment (채용)’, ‘appraisal (평가)’). We regard them as correct answers. Table 1 lists subject words and keywords used in our experiment.

Table 1: Subject words and keywords lists used in our experiment

<table>
<thead>
<tr>
<th>Subject Words</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>attendance (A1)</td>
<td>absence (인근), night duty (야근), vacation (휴가)</td>
</tr>
<tr>
<td>salary (S)</td>
<td>remuneration (보수), benefit (수당), salary class (급등), annual salary (연봉), monthly salary (월봉)</td>
</tr>
<tr>
<td>personnel (P)</td>
<td>promotion (진급), discipline (장계), rank (직급), position (직위), continuous service (근속)</td>
</tr>
<tr>
<td>employment (E)</td>
<td>screening (선발), interviews (면접), pass (합격), examination (시험)</td>
</tr>
<tr>
<td>appraisal (A2)</td>
<td>grade mark (점수), past records (실적), ability (능력), adjustment (조정)</td>
</tr>
</tbody>
</table>

4.2 Primary results

We use precision and recall to evaluate experimental results. Formulae for them is as follows[14].

\[
\text{precision} = \frac{\text{category sentences found and correct}}{\text{total category sentences found}} \tag{10}
\]

\[
\text{recall} = \frac{\text{category sentences found and correct}}{\text{total category sentences correct}} \tag{11}
\]

Table 2 lists precision (P) and recall (R) for each subject field that is described in Table 1. And we compare these values in Figure 5.

Table 2: Comparison of precision and recall for each subject field

<table>
<thead>
<tr>
<th></th>
<th>A1</th>
<th>S</th>
<th>P</th>
<th>E</th>
<th>A2</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>96.88</td>
<td>86.44</td>
<td>74.00</td>
<td>91.67</td>
<td>85.19</td>
<td>85.00</td>
</tr>
<tr>
<td></td>
<td>(31/35)</td>
<td>(51/59)</td>
<td>(37/50)</td>
<td>(11/12)</td>
<td>(23/27)</td>
<td>(153/180)</td>
</tr>
<tr>
<td>R</td>
<td>79.49</td>
<td>87.93</td>
<td>84.09</td>
<td>100.00</td>
<td>82.14</td>
<td>85.00</td>
</tr>
<tr>
<td></td>
<td>(31/39)</td>
<td>(51/58)</td>
<td>(37/44)</td>
<td>(11/11)</td>
<td>(23/28)</td>
<td></td>
</tr>
</tbody>
</table>
5. Implementation

We implemented the web-based requirements analysis supporting system on Solaris 2.5.1 using C, CGI and JAVA. Figure 7 shows a architecture of the proposed system.

Figure 7: Architecture for the proposed web-based requirements elicitation system

Requirements engineering process for requirements analysis is displayed in Figure 8. In this process, process 120, process 130, and process 140 in Requirements Elicitation and Classification stage are supported by the proposed system.

Figure 8: Requirements engineering process
For process 120, the main functions to store and analyze requirements are executed in the server. The functions, that show data from the server and send data from user to the server, are provided in the client. The users of the proposed system consist of a project manager, system engineers and general users. They can log into the system and they can execute functions based on their pre-defined restrictions. The project manager generates a new project, and he registers users of the project with restrictions. Typically, general users are restricted to issue and review requirements, and system engineers can perform more functions such as organization, traceability, and change impact analysis[7]. The functions provided to system engineers focus on analyzing requirements, while user’s functions focus on elicitation of requirements. Figure 9 shows the initial user interface of the proposed system.

![Figure 9: Initial user interface to input user's requirements](image)

For process 130, a system engineer selects ‘Input Subject’ in menu lists, and then determines subject fields to categorize requirements by inputting subject words and keywords. Based on process 130, the requirements categorization, which is matched with process 140, is processed. The categorization results are depicted in Figure 11.

![Figure 10: User interface to input subject words and keywords](image)

Figure 10 shows the user interface to input subject words and keywords. A system engineer inputs the number of subject fields, and then inputs each subject word and keyword.

![Figure 11: User interface to verify results of categorization](image)

Figure 11 shows the user interface to verify the results of categorization. After the system engineer analyzes the results, he/she can revise subject words and keywords to make better results.
6. Conclusion and Future Works

In this paper, we proposed the web-based requirements elicitation supporting system, which can be used to produce a basis for efficiently analyzing the requirements collected in distributed environment. The developed system can reduce a amount of works done by hand at the initial stage of requirements analysis, because it automatically categorize the requirements. The system enables rapid and correct requirements analysis out of messy requirements. Due to the growth of the internet, works in a distributed environment is more widespread. In this trend, application of our system could be increased.

The developed system efficiently supports the requirements analysis through categorizing requirements, which contain different view points and dispersed contents. It is important to build up the structure of categorization. It can be built up through analyzing a development domain. After the structure of categorization builds up, we can categorize requirements into the structure.

As a future research, we hope to automatically extract the categorization structure of development domain by reusing previously developed structures. And we plan to develop techniques to assist users to select keywords, because the selection of keywords has an important effect on the performance of the system. Enhancement of our system, by more experiment and application to industry project, also left as a future research.

Acknowledgment

This work was supported in part by BK21 project.

References