# A Statistical Learning Method for Identification of Analysis Classes from Requirements in Korean

Hyoungil Jeong[1], Jungyun Seo[1,2] and Soojin Park[3]

*{[1]Department of Computer Science and Engineering, [2]Interdisciplinary Program of Integrated Biotechnology, [3]Sogang Institute of Advanced Technology},*
*Sogang University, 121-742, Korea*
*{hijeong, seojy, psjdream}@sogang.ac.kr*

***Abstract***

*Identification of analysis classes is a critical design decision to be made early in the design phase in software development. Although incorrect identification of analysis classes can diminish the quality of a whole software design, it still heavily relies on the expertise and experience of the developer and has been ad-hoc. The majority existing works on identification of analysis classes focus on the rule-based approaches. However, the rule-based approaches which are used for analyzing sentence structures cannot be adopted for the language, which has very flexible word order like Korean. In this paper, we proposed a statistical learning method for identification of analysis classes from requirements sentences in Korean. The approach is evaluated using the precision and recall of the automatically extracted candidate classes from real requirements sentences in Korean. The result shows that we can promise numerically measurable enhancement of performance on solving the automatic identification problem of analysis classes using statistical methods, in the real use case specifications of a banking system.*

***Keywords:*** *Automatic Analysis Class Identification, CRFs Classifier, Phrase Chunking, Natural Language Processing*

## 1 Introduction

Design is the first phase in software development where an abstract solution is contrived after problem analysis of requirements. In designing a solution, one of challenging tasks requiring high creativity is to identify analysis classes from requirements documents. In general, identification of analysis classes heavily relies on the developer's experience and knowledge about the application domain, which is hard to be acquired through training [1]. Moreover, it is also a difficult task to automate using hand-craft rules or patterns because it needs human intuition. These kinds of attributes are similar to many problems in natural language processing (NLP) that must be addressed to identify specific semantics in a natural language.

A number of researchers have adopted various methods for NLP in the analysis of software requirements. Their methods have used some pre-defined rules about analyzing the structure of the requirement sentences, and extracting the candidates for analysis classes or use cases. These approaches can be effective in the strict word order of languages. But in Korean language, the word order is flexible, and omissions and abbreviations of components occur frequently. This makes it difficult to parse Korean sentences automatically [2]. Even if sentences can be parsed perfectly, the rule-based methods are hard to apply because the target language must have a strict word order. To solve this, we applied statistical information

extraction techniques to the automatic generation of analysis classes. This technique has many advantages; you do not need to parse sentences and it can be ported to other domains.

In this paper, we tried to solve this problem by adaption of the statistical information extracting techniques of NLP. We propose a method for processing semantics by extracting classes with three step tasks. In the first step, each sentence in a raw corpus is annotated with parts-of-speech (*POS*) tags and begin-inside-or-outside (*BIO*) [3] tags. In the second step, a model used for the identification of analysis classes is learned by the annotated corpus obtained in the first step and by a conditional random fields (CRFs) classifier [4]. This model is used to define feature functions of morpheme and *POS* and find the optimal weight for each feature. In the third step, the proposed system extracts the analysis classes and measures itself. For the measurement, we inputted the requirement documents that were not learned by the CRFs classifier and measured them for completeness and correctness.

The purpose of this paper is to propose a method for the identification of candidate analysis classes which can work on requirements documents even in the languages with a flexible word order, such as the Korean language. To the best of our knowledge, our work is the first one to report the numerical performances for the automatic identification of candidate analysis classes, and also the first to use the statistical methods, even in English. We hope that our research can contribute to lessen human effort in the whole process of software development.

The rest of this paper is organized as follows: In Section 2, we discuss related works about automatic identification of analysis classes, and in Section 3, we introduce important NLP techniques. In Section 4, we explain a proposed mechanism for the identification of analysis classes, and we present the quantitative experimental result that verified the performance of this mechanism. Section 5 discusses the conclusions of this study.

## 2. Related Work

Researchers have been studying methods for automatic identification of analysis classes. For identification of classes, most studies have used methods that apply handcrafted rules by using the result of parsing sentences or that search for a handcrafted dictionary about classes [5, 6, 7, 8, 9]. There are some studies that have extracted relationships of classes using departmentalized rules for requirement sentences [10, 11] or have generated a sequence diagram that can show collaboration of classes by formal restriction of input sentences [12, 13].

Wahono proposed an object identification process [5]. This process consists of three steps. The first step analyzes the subject, verb, and object of a sentence and considers them as tentative objects. The second step eliminates spurious objects by using rule-based reasoning (RBR) [14], and the third step identifies relevant objects by using case-based reasoning (CBR) [15]. There are some studies that generate a use case model, not classes. Kumar and Sanyal proposed a system that can identify an actor and use cases and find the relationships between them [6]. Their system identifies the subject of sentences as the actor and the predicate as the use case, and finds the relationship by using prepositions. Subramaniam, *et al.*, introduced a method that can automatically extract actors and use cases in a similar manner as in Kumar and Sanyal's study [7]. However, their method has different aspects; it analyzes the parse tree of sentences, distinguishes between ten types of sentences, and identifies elements of sentences as actors and use cases according to the type of the sentence. Vinay proposed a system that inputs the requirement sentence and extracts the actor from nouns and classes from the stems of verbs automatically by applying handcrafted rules [8]. Giganto and Smith proposed a rule-based system that parses input sentences, extracts actors

and classes in subjects or objects according to specific words or phrases, and finds relationships if there are no relationships between the actors and classes [9]. The most of these researches are staying in the level that can extract elements of use case model. This level is considered relatively simple. Their methods for elicitation of semantics need a premise that can parse sentences.

Liu, *et al.*, extended the scope to the extraction of transitions of classes, wherein the transition of each object is extracted on the basis of a guideline [10, 11]. They constructed a glossary that is used in each domain to extract objects, and this glossary is applied to semantics for the extraction of objects. They used a method that can extract class models by using language patterns for parsing sentences. However, their study required preprocessing by parsing sentences, too.

Li proposed a study that generates sequence diagrams, wherein a semi-automatic approach is used to translate use cases to sequence diagrams [12]. He applied eight-step rules for parsing sentences, and these rules are used to generate sequence diagrams. These rules must be applied by hand. Segundo et al. studied a system that can automatically transform a use case description in natural language into a sequence diagram [13]. Input sentences must be constructed corresponding exactly to the seven grammatical rules presented in this paper. Strictly speaking, the input sentences were in a formal language, not natural language.

## 3. *B-I-O* Tags and CRFs Classifier for Phrase Chunking

Rhamshaw and Marcus considered phrase chunking a problem of tagging [3]. They proposed *BIO* tags: *B* is "Beginning of chunk," *I* is "Inside of chunk," and *O* is "Outside of chunk." An example of *BIO* tags for chunking a noun phrase (NP) for a sentence is shown in Figure 1.

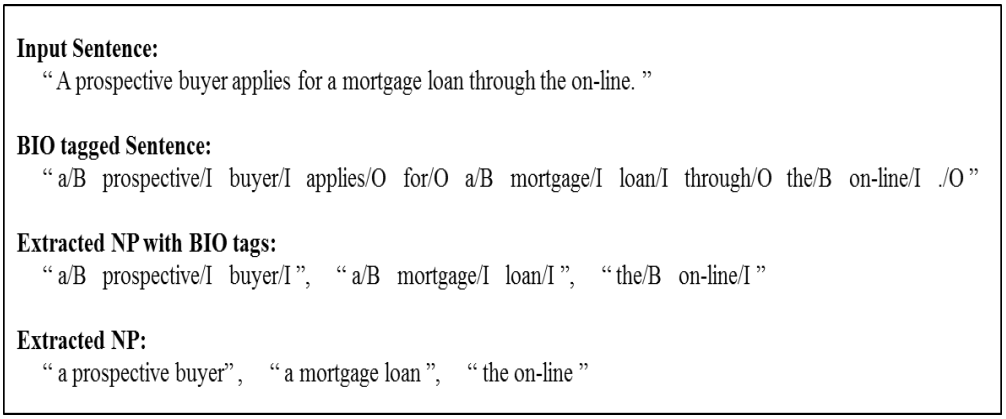


**Figure 1. An Example of BIO Tags in NP Chunking**

In this example, "a prospective buyer", "a mortgage loan", and "the on-line" are NPs. In "a prospective buyer", "a" is tagged *B* because that is a morpheme for the beginning of an NP, and "prospective" and "buyer" are tagged *I* because that is a morpheme for the inside of an NP. "Applies" is tagged *O* because that is a morpheme for the outside of an NP.

In decisions on categories for each word, *BIO* tags are influenced by context. We construct a system using conditional random fields (CRFs) classifier [4] that shows good performance in classification for continuative labels in machine learning methods. A CRFs classifier is a model that mitigates independent assumptions and overcomes the label bias problem of the hidden Markov model (HMM) [16] that is often used in classification for continuative labels.

In our proposed system, the input sentence s is a supposed word sequence $W=\{w_1,\ldots,w_n\}$ [17], and classification using *BIO* tags for each morpheme is the goal. So we calculate the conditional probability for a given word sequence by using the CRFs classifier in (1).

$$P(T \mid W) = \frac{1}{Z(X)} \exp\left(\sum_{i=1}^{N} \sum_{k} \lambda_k f_k(t_{i-1}, t_i, W, i)\right) \quad (1)$$

In (1), $t_i$ means a *BIO* tag for i-th word, and *W* means a word sequence. The $f_k(t_{i-1}, t_i, W, i)$ are feature, and $\lambda_k$ are weights of each feature. We introduce the set of features for the proposed system in Section 4.2.

## 4. Process for Identification of Analysis Classes

Our proposed system consists of three tasks that are annotating corpus, the learning, and the extracting and testing, as seen in Figure 2. In the annotating task, there are two main activities. The one is automatic morphological analysis and *POS* tagging for input sentences from a raw corpus, and the other is the *BIO* tagging by experts for learning and testing the proposed system. The learning task consists of two activities: the extraction of feature vectors in input sentences and the learning of the statistical model by using the CRFs classifier. Finally, in the extracting and testing task, the system extracts *BIO* tags and classes by using the CRFs' model, and the precision and recall of the result is evaluated.

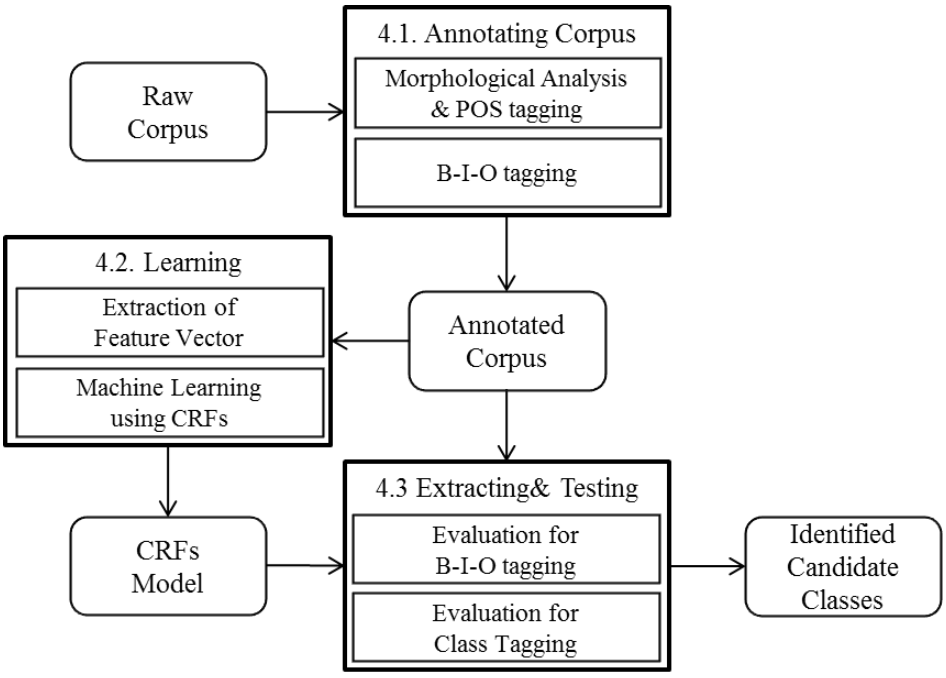


**Figure 2. The Approach Overview**

### 4.1 Annotating Corpus

The raw corpus is constructed by sentences that are present basic flow about normal activity for system on the description of a use case expressed in natural language. And they are not dependent in specific domain of application. In this paper, we collected 554 sentences with 472 classes with 29 descriptions of use cases in a banking operation. To raise the performance of a statistical classifier, one needs a large corpus. Unfortunately, the raw corpus as mentioned above is rather small. However, we cannot describe additional requirements to extend the corpus for description of a use case. Therefore, we suggest an alternative of collect additional sentences from the Web. To expand our corpus, we used Google [18] as a Web searcher, classes as queries, and sentences of snippets searched as additional sentences. We collected an additional 9,763 sentences with 18,300 classes. The sentences in the original raw corpus have a frequency of classes of 0.85, and that is very small. The extended raw corpus has a frequency of 1.87, and that is noticeably high. This new raw corpus has low quality, but it can be used as an open system because the recall of the statistical is enhanced by the increase in the number of objects that can be extracted as patterns about classes.

We then needed to annotate our raw corpus. Unlike the parsing process, automatic morphological analyzers have accuracies of more than 90 %, even in Korean. We adopted a morphological analyzer to grasp the structure of a sentence. We used SMASH [19] as an automatic Korean morphological analyzer and POS tagger. It uses a corpus and a set of POS tags from the 21st Century Sejong Project [20], and the method of left-longest matching and HMM.

We wanted to construct data for learning, so we added some information to elicit classes to the corpus. We added manually *BIO* tags - *B* is "Beginning of class," *I* is "Inside of class," and *O* is "Outside of class" - for each morpheme.

### 4.2 Learning

In this paper, the proposed method for the identification of classes use only morphemes and POS tags as features for elicitation of semantics in sentences. These morphemes and *POS* tags are not dependent on a specific domain. This characteristic has the advantage that the system needs only a statistical classifier and a morphological analyzer. The proposed method is portable.

The feature for the CRFs model for the classification of *BIO* tags is extracted in a window that has a size of w around a target morpheme. In this window, morphemes of n-grams and *POS* tags of m-grams are extracted [21]. If the time t of target morpheme set to zero, then the window is $-(w-1)/2 \leq time\ t \leq (w-1)/2$. In a corpus with a small volume, if a target morpheme is used to elicit features, then the statistical model can work like a dictionary and its evaluation can be unfair. We did not use a target morpheme as a feature.

We proceeded to select features for the identification of as many correct classes as possible. In this experiment, we constructed many combinations of features, and evaluated each combination, and adapted it to the system. To evaluate performance, we chose *Recall*, *Precision*, and *F2_score*. *Recall*, in (2), is the rate of the number of correct tags in the system divided by the number of correct tags in the corpus. That is, it explains how many classes are identified compared to the whole set of classes that should be identified. *Precision*, in (3), is the rate of the number of correct tags in the system divided by the number of tags in result of system. In other words, this rate means how many correct classes are identified among the classes that identified by the system. $F_{\beta}\_score$, in (4), is a harmonic means between *Recall* and *Precision*. In general, $F_1\_score$ is most used that has a rate of 1:1 for *Recall* and *Precision*. But, we thought that the most important factor is the identification of maximum

number of classes because our goal is the reduction of human intervention. We used $F_2$_*score* that has a rate of 2:1 for *Recall* and *Precision*.

$$Recall = \frac{the\_number\_of\_correct\_tags\_in\_result\_of\_system}{the\_number\_of\_correct\_tags\_in\_corpus} \quad (2)$$

$$Precision = \frac{the\_number\_of\_correct\_tags\_in\_result\_of\_system}{the\_number\_of\_tags\_in\_result\_of\_system} \quad (3)$$

$$F_\beta\_score = (1+\beta^2)\frac{Recall \times Precision}{Recall + \beta^2 \times Precision} \quad (4)$$

We evaluated each feature vector that was set to a different sizes of window, n-grams of morphemes, and m-grams of *POS* tags. In Table 1, we show some features and their performances. Feature3 was chosen because it has the highest value F2_score what extracts a unigram and bigram for both morphemes and *POS* tags in window with size of 5. We tried to experiment in other various features, but they did not reach at these performances in Table 1. In Section 4.3, we use Feature3.

**Table 1. The Features and Their Performances**

| Feature ID | Size of window | Lexicon | POS | *Recall* | *Precision* | *F2_score* |
|---|---|---|---|---|---|---|
| Feature1 | w=5 | n=1 | m=1 | 0.7380 | 0.8455 | 0.7573 |
| Feature2 | w=5 | n=1,2 | m=1 | 0.8076 | 0.8934 | 0.8234 |
| *Feature3* | *w=5* | *n=1,2* | *m=1,2* | *0.8381* | *0.8955* | *0.8490* |
| Feature4 | w=5 | n=1,2 | m=1,2,3 | 0.8286 | 0.8887 | 0.8399 |
| Feature5 | w=3 | n=1,2 | m=1,2,3 | 0.8387 | 0.8831 | 0.8472 |
| Feature6 | w=7 | n=1,2 | m=1,2,3 | 0.7155 | 0.8579 | 0.7400 |

We learned the statistical model that can classify *BIO* tags by using a CRFs classifier. We used CRF++ v0.54 [22] as the classifier, and L-BFGS [23] as the learning algorithm.

Figure 3 shows the feature template by Feature3 for CRF++. In this template, U01~U03 are unigrams of morphemes, U04~U08 are unigrams of POS tags, U09~U10 are bigrams of morphemes, and U11~U14 are bigrams of *POS* tags. A target morpheme x[0,0] was not used because it would have been unfair in this evaluation. Lexical features are less than *POS* features, as shown in Figure 3.

```
U00 : %x[-2,0]
U01 : %x[-1,0]
U02 : %x[1,0]
U03 : %x[2,0]
U04 : %x[-2,1]
U05 : %x[-1,1]
U06 : %x[0,1]
U07 : %x[1,1]
U08 : %x[2,1]
U09 : %x[-2,0]/%x[-1,0]
U10 : %x[1,0]/%x[2,0]
U11 : %x[-2,1]/%x[-1,1]
U12 : %x[-1,1]/%x[0,1]
U13 : %x[0,1]/%x[1,1]
U14 : %x[1,1]/%x[2,1]
```

**Figure 3. The CRFs Feature Template (w=5, n=1,2, m=1,2)**

The problem about identification of classes is that the goal is to extract as many classes as possible. So although the probability of the *O* tag is bigger than the probability of the *B* or *I* tag, we choose these tags when these are bigger than *threshold* $\theta$. If both of their probabilities are bigger than $\theta$, the larger one is chosen in both. If the system adopts the concept about *threshold* $\theta$ for *B* and *I*, the size of the elicited classes can be maximized. There is an issue as to how to decide the proper value of *threshold* $\theta$. We observed the performance of the system when *threshold* $\theta$ is changed. And we set *threshold* $\theta$ to 0.4 with the highest $F_2$*_score* as the result of this experiment, as shown in Table 2.

**Table 2. The Performance by Change in *threshold* $\theta$**

| $\theta$ | *Recall* | *Precision* | $F_2$*_score* |
|---|---|---|---|
| *no_threshold* | 0.8381 | 0.8955 | 0.8490 |
| 0.5 | 0.8381 | 0.8942 | 0.8487 |
| *0.4* | *0.8862* | *0.8842* | *0.8858* |
| 0.3 | 0.9016 | 0.8269 | 0.8456 |
| 0.2 | 0.9278 | 0.7338 | 0.8812 |
| 0.1 | 0.9574 | 0.6753 | 0.8836 |

### 4.3 Extracting and Testing

For evaluating the performance of the extraction of *BIO* tags and classes, we proceeded with 5-fold cross validation wherein we separate the raw corpus into 5 equal parts and repeat learning and testing with a rate of 4:1. We used *Precision*, *Recall*, and $F_2$*_score*. We used the Feature3 set the *threshold* $\theta$ to 0.4. As a result, in Table 3, the system has an *F2_score* of 0.8857 for *B* and *I* tags.

**Table 3. The Performance for the Extraction of *BIO* Tags**

| ***BIO* tag** | ***Recall*** | ***Precision*** | ***F2_score*** |
|---|---|---|---|
| *B* | 0.8163 | 0.8379 | 0.8206 |
| *I* | 0.9560 | 0.9306 | 0.9508 |
| Average | 0.8862 | 0.8842 | 0.8857 |

We want to extract class, not *BIO* tags, so we evaluated the performance of the extraction of classes that has combined morphemes by using *BIO* tags for each morpheme. A way of combining is to reverse the definition of *BIO* tags. By this definition, the first morpheme of the class should have *B* tag, and the next morphemes should have *I* tag. This extraction rule can be represented by a regular expression like (5).

$$Class = B + I * \tag{5}$$

The extraction of classes corresponded to (5) in sequences of *BIO* tags. For example, in case of *O-O-O*, there are no elicited classes. In *O-B-I-I-O*, *B-I-I* can be elicited as the class. And in *O-I-I-I-O*, no classes are elicited by (5) although - in reality - *I-I-I* is possible as a class. So, we try to increase *Recall* by the addition of another definition that is a transformed (5). This extraction rule is (6) that reflects an unconventional approach for a sequence of *BIO* tags.

$$Class = ( B \mid I )+ \tag{6}$$

**Table 4. The Performance for Identification of Classes**

| **Extract Rule** | ***Recall*** | ***Precision*** | $F_2$***_score*** |
|---|---|---|---|
| (5) | 0.7944 | 0.8283 | 0.8010 |
| (6) | 0.8771 | 0.7925 | 0.8588 |

Table 4 shows the performances when (5) and (6) are adopted respectively. F2_score was 0.8010 and 0.8588, respectively. In (5), the performance for the elicitation of classes was lower than the performance of the extraction of BIO tags because the system can pick up the wrong boundary of classes. (6) increases the F2_score more than 5% absolutely and more than 29% relatively.

## 5. Discussion and Conclusion

In this paper, we propose a system that can automatically elicit analysis classes from requirement sentences in a natural language, without using the rule-based methods of previous systems. We used a statistical method to elicit classes automatically. We focused on the distinctiveness of Korean language in this study and on the fact that the construction of a complete rule set that can substitute for human intuition is very difficult. We constructed a statistical model for information extraction that has been used in natural language processing, and we tried to validate the effectiveness of our proposed system in the identification of classes.

Previously, authors have tried to validate their approach in their own way. However, there has been little research quantitative performance. Therefore, we can say that our system can elicit classes more than 80% of the time by any measure of Precision, Recall, or F2_score.

Surely, there are more issues. One is evaluation to maximize recall according to the threshold for probability of *BIO* tags and the formula for identification of classes. This may be evaluated by porting to the various software applications. Then, the construction of learning data needs the human intuition. This data should adapt to the type of semantic web that can be reused in a similar environment. Our future works will be studies on the reuse of semantics for minimizing human efforts in preprocessing and the validation or adjustment of feature, threshold, etc. in learning.

## Acknowledgements

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency)(NIPA-2012-(H0301-12-3004)).

## References

[1] D. Svetinovic, D. M. Berry and M. Godfrey, "Concept Identification in Object Oriented Domain Analysis: Why Some Students Just Don't Get It", Proceedings of the 13th International Conference on Requirements Engineering, **(2005)** August 29–September 2; Paris, France.
[2] H. Chung, "Statistical Korean Dependency Parsing Model based on the Surface Contextual Information", Ph.D. Thesis, Korea University, Seoul **(2004)**.
[3] L. A. Ramshaw and M. P. Marcus, "Exploring the Statistical Derivation of Transformational Rule Sequences for Part-Of-Speech Tagging", Proceedings of the Balancing Act: ACL 1994 Workshop on Combining Symbolic and Statistical Approaches to Language, **(1994)** June 27-30; New Mexico, USA.
[4] J. Lafferty, A. McCallum and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data", Proceedings of 18th International Conference on Machine Learning, **(2001)** June 28-July 1; Massachusetts, USA.
[5] R. S. Wahono, "OOExpert: An Agent Based System for Identifying and Refining Objects from Software Requirements Based on Object Based Formal Specification", Proceedings of the 12th Scientific Meeting Indonesian Student Association, **(2003)** September 6-7; Osaka, Japan.
[6] D. D. Kumar and R. Sanyal, "Static UML Model Generator from Analysis of Requirements", Proceedings of 2008 Advanced Software Engineering and Its Applications, **(2008)** December 13-15; Hainan, China.
[7] K. Subramaniam, B. H. Far and A. Eberlein, "Automating the Transition from Stakeholders' Requests to Use Cases in OOAD", Proceedings of 17th IEEE Canadian Conference on Electrical and Computer Engineering, **(2004)** May 2-5; Niagara Falls, Canada.
[8] S. Vinay, S. Aithal and P. Desai, "An Approach towards Automation of Requirements Analysis", Proc. of the 2009 Int'l MultiConference of Engineers and computer Scientists, **(2009)** March 18-20; Hong Kong, China.
[9] R. Giganto and T. Smith, "Derivation of Classes from Use Cases Automatically Generated by a Three-Level Sentence Processing Algorithm", Proc. of the 3rd Int'l Conf. on Sys., **(2008)** April 13-18; Cancun, Mexico.
[10] D. Liu, K. Subramaniam, B. H. Far and A. Eberlein, "Automating Transition from Use-Cases to Class Model", Proceedings of 16th IEEE Canadian Conference on Electrical and Computer Engineering, **(2003)** May 4-7; Montreal, Canada.

[11] D. Liu, K. Subramaniam, B. H. Far and A. Eberlein, "Natural Language Requirements Analysis and Class Model Generation Using UCDA", Springer-Verlag, Berlin, LNCS, vol. 3029, **(2004)**, pp. 295-304.
[12] L. Li, "A Semi-automatic Approach to Translating Use Cases to Sequence Diagram", Proceedings of the 30st Technology of Object-Oriented Languages and Systems, **(1999)** June 7-10; Santa Barbara, USA.
[13] L. M. Segundo, R. R. Herrera and K. Y. P. Herrera, "UML Sequence Diagram Generator System from Use Case Description Using Natural Language", Proceedings of the 4th Electronics, Robotics and Automotive Mechanics Conference, **(2007)** September 25-28; Cuernavaca, Mexico.
[14] D. Frye, P. D. Zelazo and T. Palfai, "Theory of Mind and Rule-based Reasoning", Cognitive Development, vol. 10, **(1995)**, pp. 483-527.
[15] S. Slade, "Case-based Reasoning: a Research Paradigm", AI Magazine, vol. 12, no. 1, **(1991)**.
[16] L. R. Rabiner, "A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", Proceedings of the Institute of Electrical and Electronics Engineers, vol. 77, no. 2, **(1989)**.
[17] T. Jo, "Representation of Texts into String Vectors for Text Categorization", Journal of Computing Science and Engineering, vol. 4, no. 2, **(2010)**.
[18] Google, http://www.google.com.
[19] J. Yang, "Korean Morphological Analysis for Mobile Devices with Limited Hardware Resources", Master Thesis, Master Thesis, Sogang University, Seoul, **(2009)**.
[20] 21th Century Sejong Project, http://www.sejong.or.kr/eindex.php.
[21] P. Banerjee and H. Han, "Language Modeling Approaches to Information Retrieval", Journal of Computing Science and Engineering, vol. 3, no. 3, **(2009)**.
[22] CRF++: Yet Another CRF Toolkit, http://crfpp.sourceforge.net.
[23] H. Matthies and G. Strang, "The Solution of Non-linear Finite Element Equations", International Journal for Numerical Methods in Engineering, vol. 14, no. 11, **(1979)**.

## Authors

**Hyoungil Jeong**

BS degree in Computer Science at Sogang University in 2006. MS in Computer Science and Engineering at Sogang Universyty in 2008. Ph.D. Candidate in Computer Science and Engineering at Sogang University from 2008.

**Jungyun Seo**

BS degree in Mathematics at Sogang University in 1981. MS and Ph.D. in Computer Science at the University of Texas, Austin in 1985 and 1990. Professor of the Computer Science Department of Korea Advanced Institute of Science and Technology (KAIST) from 1991 to 1995. Professor of the Department of Computer Science of Sogang University from 1995.

**Soojin Park**

BS degree in Computer Engineering at Kyoungbook National University in 1995. MS in Information Processing at Sogang University in 2000. Ph.D in Computer Engineering at Sogang University in 2008. Software engineer at LG-CNS Systems from 1995 to 1999. Technical consultant at Rational Software Korea from 2000 to 2002. Assistant Professor of the Sogang Institute of Advanced Technology at Sogang University from 2010.