A Study on Efficient Classification of Pattern Using Object Oriented Relationship between Design Patterns

Gui-Jung Kim
Department of Biomedical Engineering,
KonYang University, Nonsan, Korea

Jung-Soo Han
Division of Information & Communication,
BaekSeok University, Cheonan, Korea

ABSTRACT

The Clustering is representative method of components classification. The previous clustering methods that use cohesion and coupling cannot be effective because design pattern has focused on relation between classes. In this paper, we classified design patterns with features of object-oriented relationship. The result is that classification by clustering showed higher precision than classification by facet. It is effective that design patterns are classified by automatic clustering algorithm. When patterns are retrieved in classification of design patterns, we can use to compare them because similar pattern is saved to same category. Also we can manage repository efficiently because of storing patterns with link information.

***Keywords**: Design pattern, Clustering Algorithm, Pattern Features, Object Oriented Relationship.*

1. INTRODUCTION

In order to use object-oriented methodology, we need practical concept and standardized pattern, and selection and application of the right objects in each context. It's design pattern that is standardized concretely for application to programming without stopping only by suggesting object oriented design concept theoretically[1][2][3]. Design pattern is a solution in realization course to maximize reusability and modularity, the biggest advantage of object-oriented methodology. There are hundreds of design patterns announced and known publicly by PLoP(Pattern Languages of Programs), design pattern conference in the USA, and Euro Plop in Europe[4][5]. Efficient management of components is necessary to improve reusability of these increasing patterns[6].

Existing clustering is mostly done between classes or in classes, so this kind of clustering has used cohesion and coupling of class or module[7]. However, this way is not efficient to cluster design pattern focusing on relation and structure between classes. Therefore this study suggests pattern clustering algorithm for clustering design pattern consisted of relation between classes. Pattern clustering process takes two-course classification for pattern's quality. In the first course, pattern is classified by function and in the second course, by structure. Pattern algorithm is used in classifying with pattern structure. Suggested clustering of design pattern classifies pattern by pattern clustering algorithm and saves it using link information of pattern, for this helps us to manage repository efficiently and redesign system using pattern information.

* *Corresponding author. E-mail: gjkim@konyang.ac.kr*
Manuscript received Aug. 18, 2006 ; accepted Sep. 08, 2006

2. RELATED WORK

2.1 Gamma's Pattern Classification

Gamma classified design pattern into creational pattern, structural pattern and behavioral pattern according to pattern's role. Creational pattern, a pattern to provide comprehensive solution for deciding creation method of objects, suggests method to organize and capsulate class definition and creation method of objects. Structural pattern, a pattern giving solution for method to compose class and objects in much bigger structure, provides general method to organize them when objects with different function play a role through cooperation. Behavioral patter, a pattern used to organize, manage and combine object's behaviors, is mostly used in algorithm performance like dividing function between objects [2].

2.2 Facet Classification System of GTE corp.

The system of GTE corp. proposed by Diaz is a representative reuse system using facet classification method [8]. This method, by which expansion of components, a weak

point of classification method by enumeration was improved, demonstrates one component with several facets after compounding component's common quality and expressing one facet. This method expresses only foundation class of components, so classification is simple and easy to understand and expand[9][10]. Query can be corrected or newly asked using synonym management method for extracting components with similar function at the time of retrieval failure. However, there's a limit that it's fixed once classification is designed. It's difficult to detail their relationship and to deal synonyms when facet items are increasing. There's another weak point that retrieval time can be long. Also, facets classification method that uses the concept of abstract pattern's quality is very arbitrary and subjective classification in clustering design pattern focused on relation and structure between classes.

2.3 Concept Analysis for Pattern Extraction

Concept analysis makes it possible to group objects with common attributes. In application of concept analysis, object is a group of class and attribute is relation between classes. The first view of concept analysis is context - collection of objects, collection of attributes and binary relation between object to determine attribute of each object and attribute[11]. Each relation links binary relation of class couple for application here. Concept is collection of objects with common attributes. Namely, it's a group of all objects sharing attributes. In general, concept is made by (X, Y).

$$X = \{o \in O \mid \forall a \in Y ; (o,a) \in P\} \quad (1)$$

$$Y = \{a \in A \mid \forall o \in X ; (o,a) \in P\} \quad (2)$$

O is an object. A is attribute. P is binary relation. X is concept extent. Y is concept intent. The main point for using concept analysis is that design pattern is concept generally. Actually, concept is composed of class group sharing relation in common. This class group is acquired by pattern found in source code or design while the number of attributes determines complexity of pattern [12].

3. PATTERN CLUSTERING

Pattern clustering is process to cluster design pattern efficiently that is increasing in number. UML is essential to adapt design patterns to design phase easily. But adequate patterns are not selected because systematic classification and community of patterns are not accomplished. This problem can be solved in constructing efficient patterns library. And it is necessary to provide pattern structure together with pattern information. Facet method and Gamma method have problems in classification design patterns efficiently. To reuse a large number of patterns, Gamma method that retrieves using pattern name only and 3 category classifications according to role of pattern has trouble in searching and adapting patterns that user request. Pattern classification that offers suitable experience automatically is necessary. Also, efficient retrieval method and detail information of patterns must be provided. Facet method is difficult to maintain objectivity because facets are set and classified according to designer's thinking. Also it is difficult to grasp class structure and relationship in module because facet items are presented common features of components. In this paper we propose pattern clustering algorithm using class structure of design patterns is made relationship between classes.

3.1. Pattern Clustering Process

Pattern Clustering Process is taken with two stages in Fig.1. In the first stage, functional classification is done by pattern's role. In the second stage, structural classification is done using relation between classes of patterns. A user can form user pattern with class diagram of UML, and then save pattern in pattern library through clustering process. In clustering using pattern structure, it's difficult to define as related pattern though a pattern structure has a part to correspond. Therefore, the stage of functional classification should be taken before clustering with pattern structure.

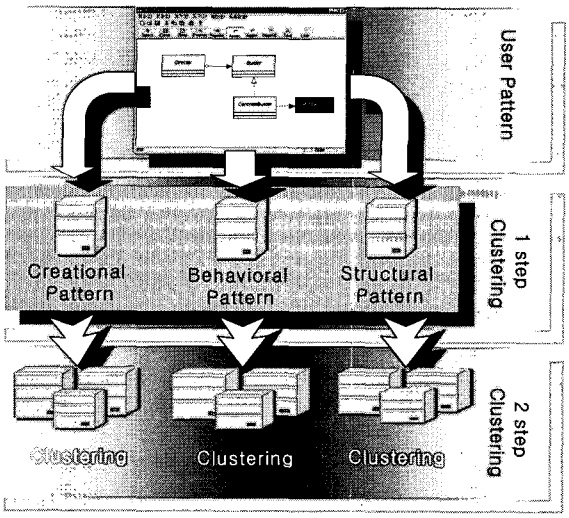


Fig.1. Pattern Clustering

In functional classification stage, pattern is classified into three kinds according to pattern's role. The pattern giving a comprehensive method to decide creation way of objects is classified into creational pattern. The pattern showing a general method to organize objects is classified into structural pattern. The pattern doing a role to classify function between objects is classified into behavioral pattern. In structural classification stage, clustering is done using pattern's structural form. We used clustering algorithm for clustering one of three foundation patterns of Gamma and pattern with the same structure among added pattern by a user to the same category.

3.2. Pattern Clustering Algorithm

Pattern clustering algorithm clusters to the same category when there's corresponding pattern with added pattern in foundation patterns. Pattern's structure is expressed with relation between classes in class diagram of UML. To compare patter's structure, one pattern is transformed into a group of order pair.

$$P = \{R_k(i,j) \mid (i,j) \in R, i \in C, j \in C, 1 : k : n\} \quad (3)$$

P is patter's order pair. R is relation of class. Proxy pattern in Fig.2 can be expressed with order pair, P=G(2,1), S(3,2). In each order pair, an alphabet letter means relation of class diagram and numbers mean two classes. Table 1 shows abbreviated words to express relation of class diagram. An added pattern by a user in Fig.3 can be expressed with order pair, P=S(1,2), G(1,3), G(4,3), S(5,4), G(6,5), G(7,5). An added pattern shows expanded proxy pattern. In this pattern, algorithm to locate proxy pattern is completed by comparing order pair of two patterns.

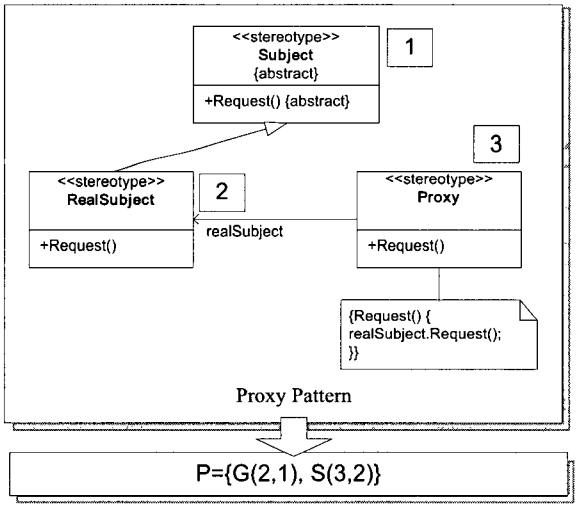


Fig.2. Proxy pattern's order pair

Table 1. Class Relationship

Relationship	Symbol
Association	S
Generalization	G
Aggregation	E
Composition	C
Dependency	D
Realization	R

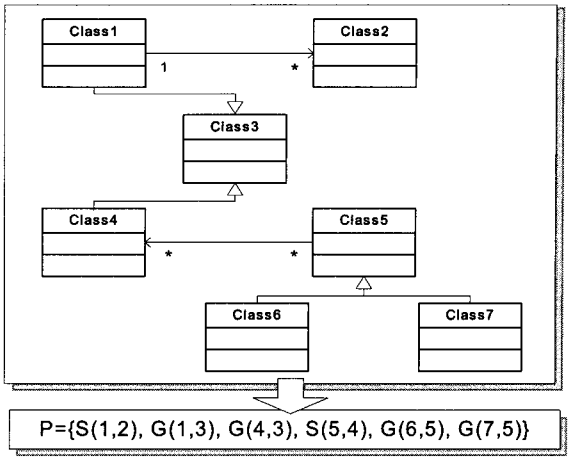


Fig.3. Added pattern order pair

Initial Step $P_1 = \{R_1=(i_1, j_1) \mid a \rightarrow i_1, b \rightarrow j_1\}$
Step $2 \le k \le n$
$P_1 = \{R_k(i_k, j_k) \mid$ *if* $i_k=i_1$, *then* $i_k \rightarrow a, j_i \rightarrow x$
if $j_k=i_1$, *then* $i_k \rightarrow x, j_i \rightarrow a$
if $j_k=j_1$, *then* $i_k \rightarrow x, j_i \rightarrow b$
if $i_k=j_1$, *then* $i_k \rightarrow b, j_i \rightarrow x\}$

Fig.4. Pattern order pair transform algorithm

In order to compare foundation pattern expressed by order pair with an added pattern; first, it's necessary to transform class expressed by numbers. Fig.4 shows pattern order pair transformation algorithm. Transformation should be done to prevent from bringing a different result without comparing each other whenever order pair changes because it's possible class number is optional, and compare regardless of changing order. The transformation process changes the first order pair into any letters. Here, the first class expressed in order pair is changed into a, and the second order pair is changed into b. In the rest order pair, the same number with one of the first class in the first order pair is changed into a, and the same number with one of the second class is changed into b. A Class coupling with class changed in order pair is expressed with x because there's no important meaning in comparison even though it is expressed with any number.

P=G(1,3), G(4,3), G(6,5), G(7,5), S(1,2), S(5,4)
⇒P=G(a,b), G(x,b), G(6,5), G(7,5), S(a,x), S(5,4)

Like above, class of the first order pair is transformed into (a,b), 1 into a and 3 into b. In other classes, 1 into a , 3 into b, and class coupling with order pair(a,b) into x because there's no relation in comparing with any class number.

```
//foundation pattern transformation
Pf={Pf1,Pf2,Pf3,....Pfk} | Transformation algorithms
        of Foundation Pattern
//pattern order pair comparison
for(i=0; i<n; i++)
 {
  //additional pattern transformation
    for comparison
  Pa={Pa1,Pa2,Pa3,....Pak}|Transformation
      algorithms of addition Pattern
  //if addition pattern include foundation
    pattern, save and exit
  Pf ⊂ Pa : Break Addition();
  //if pattern structure not same,
    change order pair
  Pa1 → Pa0, Pak → Pa(k-1)
 }
```

Fig.5. Pattern comparison algorithm

In Fig.5, pattern comparison algorithm, P_f is foundation pattern, P_a is added pattern after transformation. In order to compare added pattern with foundation pattern, a user transforms foundation pattern and added pattern with transformation algorithm and then compares if these two patterns is same. To find a pattern related with foundation pattern among some order pair showing relation of added pattern, these order pairs are transformed in order. After this transformation, in order pair satisfies a conditional formula (4) below, added pattern, P_a is clustered into foundation pattern, P_f. If order pair doesn't satisfy a conditional formula (4) even after transformation of all order pair, added pattern is compared with other foundation pattern.

$$P_f \subset P_a \tag{4}$$

Fig.6 is a concrete diagramming of comparison process of proxy pattern and added pattern. Proxy pattern is composed of one association relation and one inheritance relation. Proxy pattern, G(1,3) is transformed to G(a,b) to search for association relation and inheritance relation of the same pattern among user patterns. In other order pair, 1 is transformed into a, 3 into b. After transformation, order pair of proxy pattern becomes G(a,b), S(x,b). A user pattern is compared through transformation process and has 4 association relations. Among these relations, to find the one with proxy pattern's structure, one of them will be compared with order pair transformed from proxy pattern through transformation process. If there are order pairs having structure of G(a,b),S(x,b), a user pattern is clustered to the group of proxy pattern. Otherwise, proxy pattern and added pattern can't be called the same structure pattern and comparing will be continued over again.

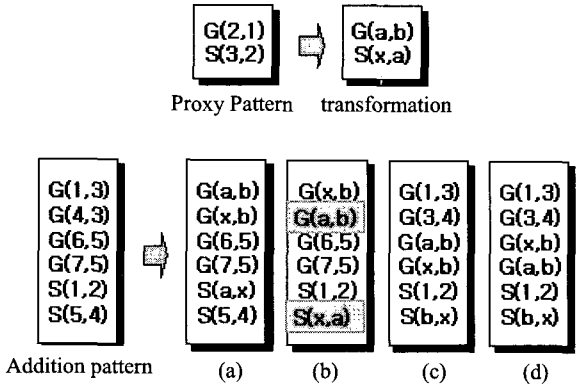


Fig.6. Pattern comparison process

4. PATTERN REGISTRATION AND MANAGEMENT

4.1 Pattern Information DB

Design pattern information was designed to check information on pattern at the time of pattern retrieval. strCargegory is supposed to be marked in the middle of classifying 3 kinds, creational pattern, structural pattern, and behavioral pattern by function. docDocument is a field for explaining pitfall, hint, and technique needed to be recognized at the time of pattern application. strPart is a foundation pattern including pattern classified by clustering. UserID is an ID of a user who creates and registers a pattern. Table 2 demonstrates a table of pattern information DB.

Table 2. pattern information DB

field	data type	Mean
strName	varchar	pattern name
strCartegory	varchar	category by function classification
docDocument	text	document for pattern
strPart	varchar	pattern classification base
UserID	varchar	pattern registration name

4.2 User Interface

A use can create a pattern; register it in pattern library and search for pattern from pattern DB. Like Fig.7, a pattern is created using UML diagram on main screen. To register it, pattern items should be selected on the menu. Pattern items are composed of pattern registration and pattern DB. If you select pattern registration item, a dialog box of order pair for comparing patters is shown, and relation and class is expressed. One pattern chosen among 3 patterns is saved in pattern library through pattern clustering proceeding inside. Pattern DB item is

used to check result of classification or search for pattern's structure. Like Fig.8, pattern DB shows you the result of classification by clustering. Patterns were classified on the basis of 24 Gamma's patterns. The list of patterns clustered among Gamma's patterns are shown to check pattern's structure. Various patterns can be compared and details of patterns can be informed.

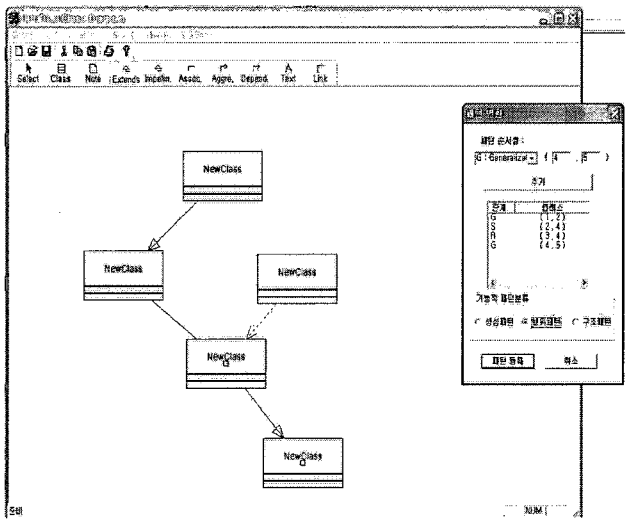


Fig.7. Pattern main screen

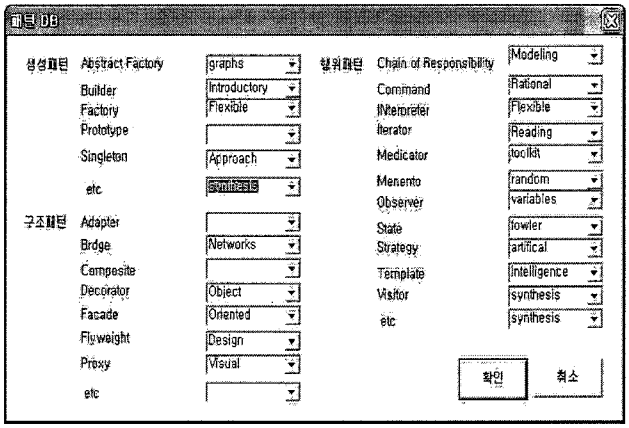


Fig.8. Pattern DB

5. PERFORMANCE

In this paper, we used 30 patterns announced at the PLOP(Pattern Languages of Programs) conference for an efficiency test. We classified 30 patterns by method suggested in this paper and facet method. Facet method is a method to classify facet items after components' common qualities are made into several facet items. Precision of suggested method was measured checking relationship between classification result by clustering and real information of patterns.

Table 3. pattern classification

index	pattern	# of class	PC	C	FC	C
1	Order/Shipment Pattern	6	memento	o	memento	o
2	The Stock Manager analysis Pattern	8	prototype	o	prototype	o
3	Rule object [PLOP2000]	13	mediator	o	mediator	o
4	The Object Filter and Access Control Framework	18	strategy	o	strategy	O
5	Legacy Wrapping[PLOP2000]	6	adapter	o	adapter	O
6	modifier[PLOP2000]	8	prototype	x	factory method	x
7	Protocol System Architecture	15	bridge	o	proxy	x
8	Strategized Concurrency	5	state	o	state	o
9	Reductor[PLOP2000]	9	command	o	command	o
10	Phrasebook Pattern	4	builder	o	builder	o
11	Two Phase Commit [PLOP99]	7	decorator	o	decorator	o
12	Composite Transaction [PLOP99]	7	composite	o	composite	x
13	Matcher-andler[PLOP99]	6	observer	o	observer	o
14	Autherntcator Pattern	5	prototype	x	Builder	x
15	Alternator[PLOP99]	6	state	o	state	o
16	Mayfly[PLOP99]	10	interactor	o	interactor	o
17	ACEE architecture [PLOP99]	5	observer	o	observer	o
18	Abstract Machine [PLOP99]	11	abstract factory	o	abstract factory	o
19	Grafcet Pattern[PLOP99]	5	command	x	interactor	o
20	Composite Calls[PLOP99]	10	composite	o	composite	o
21	Data Filter Architecture	9	proxy	o	proxy	o
22	Emissary Pattern[PLOP98]	5	strategy	o	state	x
23	Courier[PLOP98]	6	mediator	o	mediator	o
24	Override Current Processing	5	command	o	command	o
25	Substitution Pattern[PLOP98]	5	strategy	o	strategy	o
26	Delegation Pattern[PLOP98]	4	proxy	o	proxy	o
27	Reliable Hybrid Pattern	16	composite	o	composite	o
28	The Dynamic Template Pattern	6	observer	o	observer	o
29	Distributed Proxy [PLOP97]	9	proxy	o	proxy	o
30	Virtual Proxy[Larman98]	6	proxy	o	proxy	o

C - classification
PC - proposed category
FC - facet classification category

Table 4. Classified pattern precision

pattern	proposed category	facet classification category
precision	90%	83.33%

In Table 3, existence of relation was surveyed after classifying in two ways, suggested method and facet method. Table 4 shows that precision is about 90% in category of suggested method and about 83% in facet method. This means that precision by facet method is lower than one by suggested method. In facet method, selection of facet items is optional according to pattern's quality, so there may be difference of precision according to choice of exact facet items. In pattern clustering classification, patterns are classified by forms expanded from foundation pattern, for patterns in the same category can diminish the size of repository as repeating classes with only link information on foundation pattern when you try to save structures of expanded patterns. Table 5 shows the number of classes saved in repository. Fig.9 is the size comparison of repository.

Table 5. # of class for pattern repository

Pattern repository	10	20	30
existing system(# of class)	92	164	235
proposed system(# of class)	48	80	110

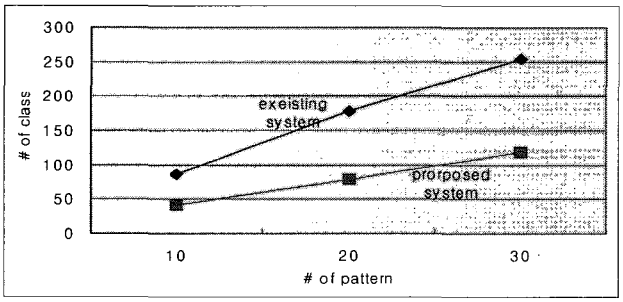


Fig.9. Comparisons for pattern repository size

Therefore, suggested method in this paper enables UML modeling, automatic classification by clustering and similar pattern retrieval in category like retrieval by string matching.

6. CONCLUSION

This paper proposes pattern classification using pattern's structure for efficient management of pattern. A pattern is classified into one among 23 categories by pattern clustering algorithm and then stored when pattern's structure and information is input. Pattern clustering algorithm is the method to use class structure of patters. By this method, added pattern is expressed with order pair and then clustered to one group if there's corresponding part with foundation pattern. Similar pattern retrieval is also possible by searching for patterns classified in category.

Previous clustering method is not efficient, so we propose pattern clustering algorithm for design pattern classification. Classification by clustering expressed higher precision than classification by facet and saved information repository as the number of repeating classes with only link information at the time of storing patterns.

From now on, the study of expanded clustering method for when added pattern by a user includes more than two foundation patterns is demanded. Research on various methods that make it possible for a user to add a foundation pattern when a pattern isn't classified to Gamma's foundation patterns should also be done.

REFERENCES

[1] http://www.omg.org/

[2] E.Gamma, R.Helm, R.Johnson, and J.Vlissides, "Design Pattern : Elements of Reusable Object-Oriented Software", **Addison-Wesley**, 1995

[3] Jeffrey H. Kingston and Benjamin Yin-Sun Lynn, "A Software Architecture for Timetable Construction", **Lecture Notes in Computer Science 2079**, 2001, pp.342-352.

[4] http://hillside.net/plop/

[5] http://hillside.net/europlop/

[6] Gui-Jung Kim, Young-jae Song, "Design Pattern Based Component Classification and Retrieval using E-SARM", **The KIPS Transactions**, vol. 11-D, No.5, Oct. 2004, pp.1133-1142.

[7] Nicolas Anquetil and timothy c. Lethbridge, "Experiments with Clustering as a software Remodularization Method", **Proceedings of the 6th Working Conference on Reverse Engineering**, 1999, pp.235-255.

[8] R.Prieto-Diaz and P.Freeman, "Classifying Software for Reusability", **IEEE Software**, Vol.4, No.1, Jan. 1987, pp.6-16.

[9] Gui-Jung Kim, Young-jae Song, "Efficient pattern classification and retrieval supporting design pattern reuse", **Proceeding of the IASTED International Symposia**, Feb. 2001, pp.511-517.

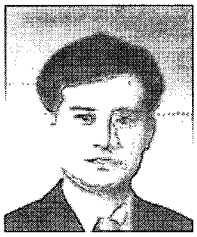
[10] Gui-Jung Kim, Jung-Soo Han, Young-Jae Song, "Efficient Management of Pattern Information for Similar Design Pattern Retrieval", **Proceeding of the SNPD01 International Conference**, 2001, pp.444-449.

[11] F.A.Grootjen, and Th.P. van der Weide, "Conceptual Query Expansion", **ICIS report NIII-R0406**, Jan. 2004.

[12] Paolo Tonella and Giulio Antoniol, "Object Oriented Design Pattern Inference", **Proceedings of the IEEE International Conference on Software Maintenance**, 1999, pp.230-238.

Gui-Jung Kim

She received the B.S. degree and the M.S. degree in computer engineering from Hannam university, Korea, and the Ph.D. degree in computer engineering from Kyunghee university, Korea in 2003. Since 2001, she has been a professor in department of Biomedical Engineering, Konyang University, Chungnam, Korea. Her main research interests include CASE and component reuse.



Jung-Soo Han

He received the B.S. degree, the M.S. degree and the Ph.D. degree in computer engineering from Kyunghee university, Korea. Since 2001, he has been a professor in division of Computer Science, Baekseok University, Chungnam, Korea. His main research interests include component management and CBD.