설계패턴의 효율적 분류와 관리

Efficient Classification and Management of Design Patterns

한정수, 김귀정*
천안대학교 정보통신학부,
건양대학교 컴퓨터공학과*

Han Jung-Soo, Kim Gui-Jung*
Division of Information & Communication, Cheonan Univ.
Department of Computer Engineering, Konyang Univ.*

요약

본 논문에서는 디자인 패턴을 분류하기 위해 패턴구조의 특성을 가지고 분류하였다. 그리고 클러스터링에 의한 분류는 패싯 분류에 의한 방법보다 높은 정확도를 보여주었다. 따라서 자동화된 분류방법인 클러스터링 알고리즘을 사용하여 디자인 패턴을 분류하는 것이 효과적이라 할 수 있다. 디자인 패턴의 분류는 검색 시 유사한 패턴들이 같은 카테고리에 저장이 되므로 유사 패턴을 비교하여 사용할 수 있으며, 패턴 클러스터링에 의해 분류되고, 패턴의 링크정보를 이용하여 저장하므로 저장소를 효율적으로 관리할 수 있다.

Abstract

In this paper, we classified design patterns with special quality of pattern structure. Classification by clustering had expressed higher correctness degree than classification by facet. Therefore, can do that it is effective that classify design patterns using clustering algorithms that is automatic classification method. When we are searching design patterns, classification of design patterns can compare and analyze similar patterns because similar patterns is saved to same category. Also we can manage repository efficiently because of using and storing link information of patterns.

Ⅰ. Introduction

In order touse 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]. 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 U.S. and Euro Plop in Europe. Efficient management of components is necessary to improve reusability of these increasing patterns.

Existing clustering is mostly done between classes or in classes, so this kind of clustering has used cohesion and coupling of class or

module. 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 proceeding 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 design pattern clustering classifies pattern by pattern clustering and saves it using link information of pattern, for this helps us to manage repository efficiently and redesign system using pattern information.

Ⅱ. Related Study

2.1 Gamma's Pattern Classification

Gamma classified design pattern into creational patter, 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 andobjects 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 [3]. 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. 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.

Ⅲ. Pattern Clustering

3.1 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.

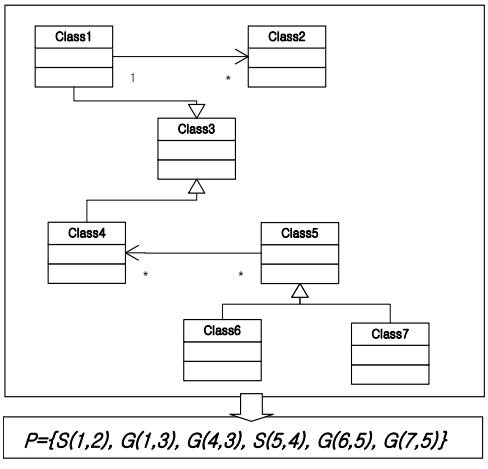
$$P = R_k(i,j) | (i,j) \in R, i \in C, j \in C, 1 \le k \le n \quad (1)$$

P is patter's order pair. R is relation of class. and C means classes. <Table 1> shows abbreviated

words to express relation of class diagram. An added pattern by a user in Figure 1 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. In order to compare foundation pattern expressed by order pair with an added pattern; first, it's necessary to transform class expressed by numbers.

[Table 1] Class Relationship

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



▶▶ Figure 1. User additional pattern order pair

$$
\begin{aligned}
&\textit{Initial Step}\ P_1 = \{R_1 = (i_1, j_1) |\ a \rightarrow i_1, b \rightarrow j_1\} \\
&\textit{Step}\qquad 2 \le k \le n \\
&\qquad P_1 = \{R_k(i_k, j_k) \mid \text{if } i_k = i_1,\ i_k \rightarrow a,\ j_i \rightarrow x \\
&\qquad\qquad\qquad\qquad \text{if } j_k = i_1,\ i_k \rightarrow x,\ j_i \rightarrow a \\
&\qquad\qquad\qquad\qquad \text{if } j_k = j_1,\ i_k \rightarrow x,\ j_i \rightarrow b \\
&\qquad\qquad\qquad\qquad \text{if } i_k = j_1,\ i_k \rightarrow b,\ j_i \rightarrow x\ \}
\end{aligned}
$$

▶▶ Figure 2. Pattern order pair transform algorithm

Figure 2 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 proceeding 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 the'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 ther's no relation in comparing with any class number.

3.2 Pattern comparison algorithm

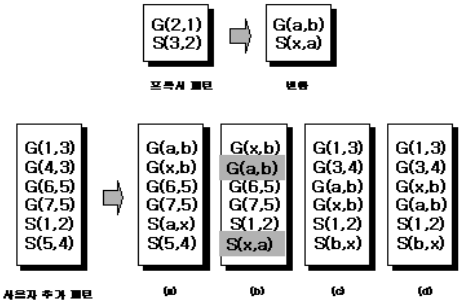
```
Pf=Pf1,Pf2,Pf3,....Pfk) | Transformation algorithms
            of Foundation Pattern
// pattern order pair comparison
If(i=0 ; i<n ; i++)
 //additional pattern transformation for comparison
  Pf=Pa1,Pa2,Pa3,....pak) | Transformation algorithms
            of Foundation Pattern
  Pf ⊂Pa : Break Addition();
// if pattern structure Not same, change order pair
  Pa1 → Pan, Pak→Pa(k-1)
EndIf
```

▶▶ Figure 3. Pattern comparison algorithm

In Figure 3, pattern comparison algorithm, Pf is foundation pattern, Pa 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 proceeding, in order pair satisfies a conditional formula (2) below, added pattern, Pa is clustered into foundation pattern, Pf. If order pair doesn't satisfy a conditional formula (2) even after transformation of all order pair, added pattern is compared with other foundation pattern.

$$P_f \subset P_a \qquad (2)$$

Figure 4 is a concrete diagramming of comparison proceeding of proxy pattern and added pattern. Proxy pattern is composed of one association relation and one inheritance relation. Proxy patter, 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 proceeding 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 proceeding. 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.

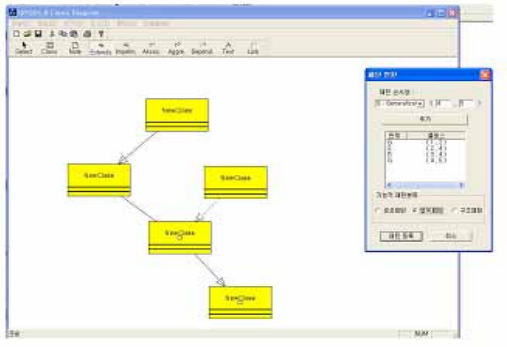


▶▶ Figure 4. Pattern comparison process

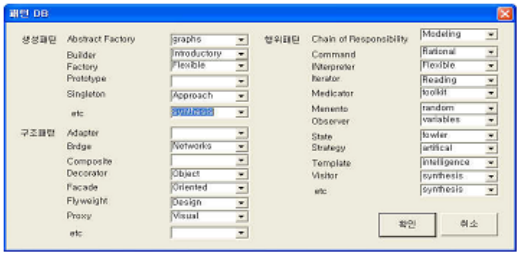
Ⅳ. Pattern Registrations and Management

A use can create a pattern; register it in pattern library and search for pattern from pattern DB. Like Figure 5, a pattern is created using UMLdiagram 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 Figure 6, 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 is shown to check pattern's structure. Various patterns can be compared and details of patterns can be informed.



▶▶ Figure 5. Pattern main screen



▶▶ Figure 6. Pattern DB

V. Efficiency Test

In this paper, we used 30 patterns announce ate the PLOP(Pattern Languages ofPrograms) conference for an efficiency test [4]. We classified 30 patterns by method suggested here and facet method [5]. 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 2] classified pattern precision

pattern	proposed category	facet classification category
precision	90%	83.33%

<Table 2> 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 inthe 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 3> shows the number of classes saved in information repository.

[Table 3] # of class for pattern repository size

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

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

VI. Conclusion

This paper proposes pattern classification suing 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.

▌참고문헌 ▐

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

[4] http://st-www.cs.uiuc.edu/~plop/.

[5] Gui-Jung Kim, Jung-Soo Han, Young-Jae Song, "Efficient Management of Pattern Information for Similar Design Pattern Retrieval," Proceeding of the SNPD'L01 International Conference, pp.444-449, 2001.