컴포넌트 검색을 위한 정보저장소 구조에 관한 연구

A Study on a Repository Structure for Components Retrieval

김귀정
건양대학교

Kim Gui-Jung
Konyang University

요약

본 논문은 컴포넌트 검색을 위한 정보저장소 구조를 설계하였다. 정보저장소는 컴포넌트 검색을 위한 구문분석, 검색방법, 뷰어, 재사용으로 연결시킬 수 있도록 데이터, 프로세스, 뷰어로 구분하여 구성하였다. 컴포넌트 검색은 개선된 스프레딩 액티베이션 알고리즘을 이용하여 그 결과를 뷰어를 통하여 보여준다. 검색 결과는 우선순위가 높은 순서로 검색되기 때문에 재사용의 효율성이 높다.

Abstract

In this paper, we designed a repository structure for components retrieval. The repository is consisted of data, process, and view for syntax analyzer, retrieval method, viewer, and reuse. We used enhanced spreading activation algorithm for components retrieval and showed the retrieval results. The retrieval result shows the components in order of high priority. Thus components reusability is improved.

I. 서 론

Software reuse is a technique of reducing developing time and cost and enhancing the quality and productivity by reusing the software library commonly witnessed in many cases of system development. Sharing the information and development method of the system, it also reduces the burden of development on the part of the developer, and the concentration on the expandability of the software and its maintenance enhances the reliability, usability, and efficiency of the software[1].

For reuse, however, techniques for efficiently storing, managing, and retrieving objects are required. software components, through the analysis of the existing software, are reused after deriving, classifying, and storing the reusable components and their information, i.e, the code, class, inheritance relationship, and diagram information in the repository[2]. The existing reuse system, depending on the component retrieval for object reuse and document information, has been designed on the basis of limited experiences on the actual reuse. Although RSL[3], a retrieval method for reuse, uses queries in a natural language form, it does not have functions of restructuring the meaning of key words and queries and browsing, which are used in retrieval. Diaz classification mechanism[4] facet-classifies the modules and retrieve with queries after defining the area model.

This paper constructs the repository for the object-oriented reuse with the information of the software components-class name, member function, attribute, and inheritance. With the input of the query, the components obtained through E-SARM(Enhanced Spreading Activation Retrieval Method) algorithm and their similar sets are retrieved in order of their activation value and the retrieved components are represented in a graph of class hierarchy though the function of Viewer.

II. Related Work

This Section surveys the merits and demerits of the past component retrieval systems and then the reusing methods of components and SARM algorithm.

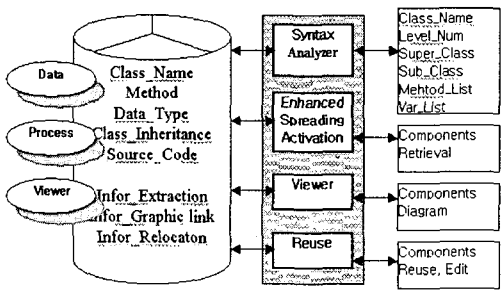
CATALOG [5] can process unstructured data and each of the modules within the established database derives information necessary for indexing from header descriptor. Retrieval provides the menu interface and search interface centering on order words for professionals. It allows Boolean query and partly, string matching techniques. Although meaningless words are processed through the stoplist, it is such problems as the library expansion, the understanding of components, and merging through the selection and modification of components.

SARM retrieval system operates for the stable activation vale or up to the maximum number of cycles the user designated. But SARM has the problem of delaying the retrieval time as it measures the similarity through repeated computation using activation values. This makes the retrieval for a greater number of components more difficult as the number of computation exponentially increases in proportion to the increase of component Thus, SARM is an efficient retrieval method that can cover components not directly indexed, and it greatly reduces the cost as it spares indexing for every item when establishing components in the repository. Therefore, the ideal method will be the one, which enhances the retrieval speed while retaining the capacity of SARM retrieval system.

III. Component Retrieval System

1. The Structure of Reuse System

The system for the software component reuse saves the information for components in the repository through the syntax analyzer, as shown in [Fig. 1]. The repository consists of Data, Process, and Viewer; when a query is input though the retrieval viewer; the components appear as a list in order of the activation value after the activation values are computed through E-SARM.



▶▶ Fig. 1 Repository Structure

This can be achieved by altering the information of inheritance of the subclass by reference to the name of the superclass in inheritance relationship. Modification for the reusable components is necessary for each of the functions of new software even if the components are moved on the hierarchy diagram. The present system makes the modification of the components possible by providing an editor. The editor also supplies the functions of methods and attributes for the modification of components. The selection of the class to modify on the class hierarchy diagram activates the editor with the source code and after modifying and saving the code, the code information of the component is altered in the repository and the reuse of the component is achieved.

2. Component Analysis

Components for reuse form a repository containing the information on the name of the class, the member function, properties, variables, the inheritance relationship between the superclass and subclass extracted through syntax analysis[6]. The information of class extraction, the information of link based on the inheritance information, and the location information of each class are collected for the representation of the hierarchy diagram for each of the components. The location information is extracted according to the label within the hierarchy diagram as in the hierarchy diagram the location information of each class should be represented equivalently with the superclass at the center.

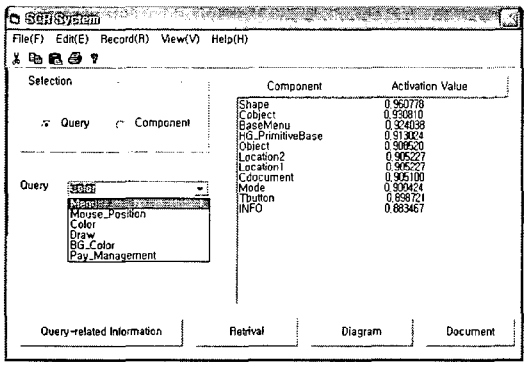
3. Component Viewer

Viewer is a function for showing the analyzed information of the components as a graphic, represented as an object-oriented diagram using the number of subclasses in inheritance relationship with each class, with the topmost class at the center. The schema information is represented as a diagram after designating the object and inheritance relationship by computing the depth and width of the object, the numbers of classes and subclasses, and the number of classes in each label. The name of each class is shown along with its location. The the viewer makes it possible to view the diagram, class inheritance relationship, and the code information of each class using the component information saved in the repository. The important function of the viewer rearranges the increased or decreased classes in each label by reallocating the location of each class according to the number of increase or decrease of the classes in the process of insertion or deletion of the whole components or a portion of classes.

IV. Component Retrieval

Components directly connected to the query and similar ones indirectly connected to it can be retrieved through E-SARM. Every data is saved in the repository and such a process is repeated until the activation value connected to the query is stabilized with the completion of computation. The topmost class of the whole components, the Base Class is retrieved within the standard range of the activation value. With the increase of reference the activation value increases and

the similar components are also retrieved through the reference function. With such retrieval the options for selecting the reusable components expand as there are many components provided, and the understanding of the components in terms of the query is enhanced. Our system is designed with the interface for component and query input, query-related information, retrieval, document, the representation of component diagram, and the component reuse function.



▶▶ **Fig. 2 Component Retrieval Result**

In the process of component input, retrieval is achieved by the addition of components to the components already existing in the database. The input interface identifies the relationship between the new components and the query. And in the representation of the relationship between the query and the component, by showing the permissible queries in the list, errors can be prevented. [Fig. 2] shows the retrieval result of the components in order of the activation value with the input of the query color. As a result of the component retrieval for the input of the query Color, 11 components have been retrieved centering on the component ball with the greatest number of reference. Components directly connected to Color are those directly connected up to the top five, Object and the rest are those indirectly connected. The retrieved components include the classes with the function of the query Color. The Document function for each of the components makes the access to the detailed function information of components possible. In order to earn the information of the class structure of each component, select the component and the diagram for the component is represented through the hierarchy diagram. As a result, information may be obtained for the understanding of the retrieved components and with the selection of components for reuse; direct reuse on the diagram is possible.

V. Conclusion

In this paper, we designed a repository structure for components retrieval. The repository is consisted of data, process, and view for syntax analyzer, retrieval method, viewer, and reuse. We used enhanced spreading activation algorithm for components retrieval and showed the retrieval results. The retrieval result shows the components in order of high priority. Thus components reusability is improved. And this paper has improved SARM for retrieval of the object-oriented components so as to retrieve similar components, and established in object-oriented software component reuse system that allows the reuse of the components directly on the hierarchy diagram. Components

are syntactically analyzed for their reusability and the interface is used to define the component-query relationship.

References

[1] Carma McClure, "The Three Rs of Software Automation: ReEngineering, Repository, Reusability," Prentice Hall, 1992, 221-230.

[2] James Petro and Michael E. Fotta, "Model-Based Reuse Repositories-Concepts and Experience," IEEE Computer Society Press-Technical Council on software Eng., 1995, 60-69.

[3] B.A.Burton, R.W. Argon, S.A. Bailey, K.K. Koehler and L.A. Mayers, " The Reusable Software Library,"IEEE Software, July, 1987, 25-33.

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

[5] W.B.Frakes and B.A.Nejmeh, "An Information System for Software Reuse, Proceedings of the Tenth Minnowbrook Workshop on software Reuse, 1987.

[6] Jun-Soo Han, Young-Jae Song, "Extraction and Comprehension of Objects for Class Components Reuse", The Transactions of Korea Information Processing Society, Vol.6, No.4, April, 1999, pp.941-951.