재사용 시스템 개발을 위한 객체지향 검색 프레임워크

김 정 아[†] 문 중 렬[††] 김 승 태[††]

요 약

본 논문에서는 라이브러리의 하부 표현 구조에 관계 없이 재사용 라이브러리로 부터 소프트웨어 부품을 일관성 있게 저장하고 검색할 수 있도록 지원해주는 검색 프레임워크를 객체지향 기법을 도입하게 제안하였다. 제안한 검색 프레임워크는 일관된 사용자 인터페이스를 가능하도록 하기 위하여 시각적 영역에서 미리 정의한 간단한 시각적 검색 오퍼레이션을 통해 라이브러리에 저장한 부품에 대한 정보 객체를 검색할 수 있도록 하였다. 이는 제안한 4I모델에 의해 가능하다. 또한 새로운 검색 메카니즘이나 분류기법을 쉽게 추가할 수 있다. 본 논문에서는 객체지향 프레임워크 개발을 위한 4I모델을 제안하고 이를 구성하는 각각의 구성 요소를 객체로 모델링하고 구현하였다.

Object-Oriented Retrieval Framework to Construct the Reuse-Supporting Systems

Jung A Kim[†], Chung Ryeal Moon[††] and Seung Tae Kim[††]

ABSTRACT

This paper describes an object-oriented retrieval framework that is generally designed to store and retrieve the reusable components from the library regardless of the underlying representation of the library. We propose a retrieval framework on visual space so that reuser can identify their location at the library without any previous information of library structure. They can decide the directions of retrieval with the results displayed on the visual space and interact with the library using the defined simple retrieval operation that can access the library information object. For doing this, 4I model was proposed. Librarian as well as reuser can easily construct the new library on the visual environment. It is the process to give the semantic of the information object. This paper discusses the basic concepts of our 4I model and explains each constituent of our model and shows a simple example of the system.

1. Introduction

Recent research efforts in reuse area have concentrated on exploring new approach for formalizing and generalizing the reuse-related activities or model and process. We can consider these approaches as two main streams : One is design for reuse, which related with the way to identify the components from existing software, the criterion to decide the fittable for reuse, area of software classification, location and representation method, etc. The other is design by reuse on which many rese arches have focused on retrieval, specialization or realization techniques for modifying the selected component, program synthesis from reusable components. Above all, classification in design for reuse and retrieval in design by reuse have been considered as a clas-

† 정 회 원 : 중앙대학교 컴퓨터공학과 연구원.
†† 정 회 원 : 중앙대학교 대학원 컴퓨터공학과 재학중

논문접수 : 1995년 3월 30일, 심사완료 : 1995년 8월 24일.

sical and critical problem in software reuse. So far different classification and representation scheme and retrieval model for each classification approach have been proposed for organizing software collection into library and for utilizing the query.

The approaches of library constructing are divided into four groups[5] : similar to the library of program language, that is subroutine library, library based on information retrieval and indexing technology, knowledge-based method, hypertext-based library. Among several approaches done in classification scheme, we could find four schemes : free-text indexing scheme, faceted index, AI-based, enumerative scheme.

According to the evaluation result from Proteus, enumerative and faceted classification mechanisms have no retrieval time because of their structure[4]. That report showed also that the time for structuring of faceted classification scheme is longest. Unfortunately, there was no more detail evaluation for each method. We think that it is necessary to select the proper classification scheme for our own application and our experience in that application. Then, what about the reuse system for reuse by design? To success and practice the reuse, especially in reuse by design, we need the automated reuse system. This automated reuse system must meet the extendibility, flexibility, usefulness, high interactive user interface, and smooth learning curve. Different library representation method and classification scheme lead to different reuse environment, especially retrieval system and user interface, as we saw in Proteus research.

In this paper, we proposed classification framework and retrieval environment to support the multiple underlying representation and cataloging system with single view, which meets the requirement of reuse system. It means new representation and classification can easily be added without new learning time for reuser. Of course librarian can construct the library easily by classification framework. We call our framework 4I. To provide the consistent view for multiple underlying structure and improve the interaction bandwidth between the library and user, applying the visual reasoning and visualization could be one solution. Extendibility and flexibility can be possible by constructing the framework and retrieval model based on object-oriented model, that is, data abstraction, information hiding, polymorphism. From the top level of the model, we defined one object for library by defining the different representation and classification methods as the data and visual retrieval operations as the operation to manipulating the data as well as single viewing way. We provided the classification framework and retrieval environment to handle the several representation methods and to encapsulate this information from the reuser and to allow designer of library to access these schemes. Using the consistent retrieval operations, reuser accesses the different library that has own classification scheme.

Section 2 describes the existing classification scheme and retrieval mechanism to find the common feature for out retrieval model. In section 3, we discuss our object-oriented retrieval model, 4I model. We give simple illustration of our environment in section 4 and conclude our research in section V.

2. Classification Scheme and Retrieval Mechanism

Initial approach to software component classification was based on the enumerative classification scheme[9]. This assumes the whole group of component divided into successively narrower groups that include all the possible compounded keywords. Keyword means accurate description of the software component's intention. Whole collection, which is general, is recursively decomposed into narrower groups, which is more specific. So, enumerative classification results in constructing the hierarchical relationship among the components. So, reuser can easily recognize the candidates of his desired component.

Enumerative classification method shows the clear relationship on the library but it lacks of extendibility. It means that enumerative approach doesn't cope well with changing the components and can be used in well-defined area. Of course, the comparison among the classification approaches is not the goal of this paper. Other approach, faceted classification scheme can handle this problem[9]. Original faceted classification approach was decomposition and synthesis technique. Librarian breakdowns the knowledge of components into facets that show the common characteristics of components stored in the library and selects the terms corresponding the facets. Synthesizing the term from each facet describes the component. Thesaurus makes possible to use the controlled vocabulary. A weakness of the original faceted classification is that they can't show the relationship among the components. Conceptual distance graph could solve this problem. Librarians construct the conceptual distance graph that shows the distance between the terms of each facet. Using the conceptual distance, we can get the similarity between the terms as well as the similarity between the components. Important thing is not the fact that faceted classification approach uses the conceptual distance graph and how they use. More importance thing is that it was necessary to describe the relationship among the components. Some approach suggested by Embley defined the relationship among the components such as "depends-on", "close-to", "generalize", etc. Reuser can describe the his desired component with the selected term from each facet and ask to retrieve the component using thesaurus and several vocabulary control utilities.

Recently, there were several tries to use the information retrieval(IR) and indexing techniques based on free-text because IR systems are able to cope with the unstructured information such as software component and associated document. First attempt for using IR is RSL system[7] that analyzes source code commented with information needed for classifying such as keyword, author, developing environment so on. The keyword comments provide a profile used in IR system. It was not considered as pure IR approach. CATALOG system[8] automatically extracts keywords from documentation and creates a profile that defines the characteristics of component. There was limitation to capture the semantic information from the profile since there was no semantic information related with index included in profile. GURU proposed the concept of lexical affinities among pairs of indices to express semantic information. It identifies a set of features for each component to compute functional similarity. GURU[3] also provides the browsing facility with hierarchical clustering technique to display the relationship among the components. Retrieval mechanism is very similar for classifying. It takes a query, describing the user's purpose, can be treated as software documentation and makes query profile to match with the profile stored in the library. As a result of retrieval, user can get the matching set and get further information through the browsing hierarchy.

Final is knowledge-based approach that

aims the understanding the functionality of components. This approach is possible when the reuse library has rich domain knowledge and semantic information, which can be provided manually. It is common feature of proposed knowledge-based reuse library to present the structural representation of knowledge. System[3, 6] proposed by Wood and Sumerville[1] has component knowledge representation consists of independent property (nominal) and its action and modifier for more detail description on nominal and action. Each of attributes is associated to each other by dependencies.

From these survey studies, we found that library contained not only reusable components but also classification information regardless of its representation method. Classification means the collecting large amount of components into groups so that one group contains related components. Librarian needs prior knowledge on collected components. These showed us the following common features. First, every scheme has own representation way that is retrieval clue. Faceted classification has pairs of term and facet. IR has profile consisting of indices. Second, there are relationship information for browsing and for improving the efficiency of retrieval regardless of its contents and range coping with. As we saw in faceted classification, they tried to improve the faceted classification scheme by managing the term dictionary and conceptual graph even original scheme couldn't catch the semantic relationship among the components. Third, we could easily find the factor being visualized to reuse. For example, enumerative classification scheme has the tree structure and faceted classification scheme can show the terms that each facet has. Fourth, it is important issue what initial point for retrieval is to determine the success of retrieval. Faceted classification has limitation that its success depends on the access point selected by reuser and it is difficult to choose the access point. Also, IR with controlled vocabulary has difficulties in handling the reuser's uncontrolled vocabulary. These commonalties and limitations were our motivation for proposing classification frame work and retrieval environment with visualization. It was important to recognize that classification and indexing of reusable components reflect the programmer's view to component.

3. 4I Model for Retrieval Framework

So far, many researches on classification scheme and retrieval techniques tried to improve the efficiency. Retrieval efficiency means how much reuse can satisfy the results of reuse system. In other words, we can say it is the criterion of quality of retrieval. Those approaches were text based environment so that it required the reuser to know the way to use the library and to be familiar with the library structure as well as classification scheme. This is result from the dependency between classification scheme and retrieval technique. It increases the learning time of new library structure even though retrieval phase can be preserved from changing classification scheme. If we have a single view of multiple representation, then reuser didn't spend their time for considering the library structure. It is the reason we propose our retrieval model and environment.

3.1 Basic Concepts

Before we start to describe our model in detail , we define our paradigm of retrieval.

1. We assume that visual/spatial framework and visual representation can help

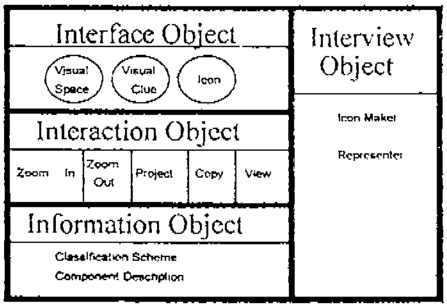
reuser to retrieve and navigate or browse the library rather than using formal query like SQL or compounded keyword. Visual framework allows reuser to retrieve components without the information of the way to store and represent[10].

2. Retrieval is not straightforward. It doesn't mean there is no start or access point. Our second presupposition is that we need two abstraction levels for retrieval corresponding the among of information showed to reuse. We arrange the component into two levels on visual space. Initial view provides the highest level so it gives the global information of library to reuses who want to decide his first access point. Next are detail views that return more detail information to show where reuser is located at the library and what are the results of his action. It helps to decide whether he can go further or go back more abstract view to modify the path. Our retrieval process is not straightforward but iterative between more detail level and more global level.

3. User interaction for retrieval may be simple. In other words, there are a few operations for retrieval. Let us consider that we want to find good play to rest and we already have global information. Using this global information, we can decide what we want to know more, i.e., where is for climbing, where is for shopping so on. After deciding what for, for example climbing, we need more information on that. We select more and get several groups more and more, so on. It means we can zoom-in some areas to arrive our destination . In contrast to zoom-in, sometimes we need to return where we start since our selection was wrong for our intention. It is zoom-out interaction. When we got some location, we need more information on the neighborhood place to decide. It doesn't need to go deeper we just move other point at the same level. It is project operation.

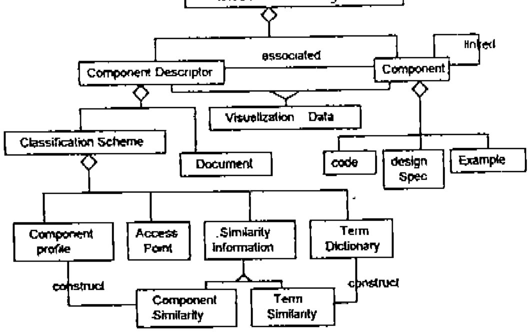
3.2 4I Model

Smalltalk uses MVC paradigm for smalltalk user interface[13]. MVC is the abbreviation for Model-View-Controller, which are data to be displayed, a way and style to display, and the relationship between both, respectively. Model can't send any message to View. View can't change the value of model. Both of them are clearly independent. We propose 4I (Information, Interface, Interaction, Interview) Model as single view retrieval model based on MVC Model.



(Fig. 1) 4I Model for Visual View for Retrieval

Information object consists of software components and associated information including classification knowledge as well as indexing information. Its layer provides all necessary information for retrieval. We modeled information object as(Fig. 2).



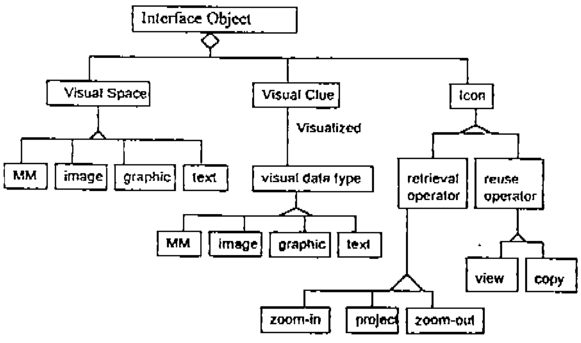
(Fig. 2) Information Object Model

To reuse a software component is only possible if what this software does is precisely and abstractly stated. It means that a description of this component has to be available. We consider the case of software components that are specified using component descriptor. A software library should contain a pair of component descriptor and component. We are not considering the reuse of software design or reuse of other model constructed through the software life cycle but reusability of code. Our information object model contains the design specification and example as the component that can help what associated software does. Component description has similar purposes and provides additional information for classifying. It contains classification scheme that gives the classification information to library system and document that describe the functionality of software in text. Among several objects defined in information object model, we define the classification scheme more precisely using algebraic specification suggested by Breu[11].

```
class spec Classification-scheme
  Uses : component-profile,  Access-Point,  Similarity-Infor-
        mation, Term-Dictionary, Component, Bool
  Opns
  Determine(Access-Point)→Classification-scheme
  Make(Component-Profile)→Classification-scheme
  Compute(Component-Profile)→Similarity-Information
  Construct(Term-Dictionary)→Classification-scheme
  Compute(Term-Dictionary)→Similarity-Information
  First(Classification-scheme)→Access-Point
  Select(Access-Point,Classification-Scheme)  →Component-
Profile
  Select(Similarity-Information,Access-Point,Classification-
Scheme) →Component-Profile
  Determine(Component-Profile)→Component
  Is-Leaf(Component-Profile)→Bool
Axioms
  td : Term-Dictionary, ap : Access-Point,
  cs : Classification-Scheme, co : component,
  si : Similarity-Information, cp : Component-Profile
  Select(a, c)=Select(First(c),c)
  First(Determine(ap))=Determine(First(cs))
  Select(si,ap,cs)=if is_leaf(cp) then Determine(cp) else cp
  Select(ap,cs)=if is leaf(cp) then Determine(cp) else cp
  if t2=(t1 c) then Construct(t1)=Contruct(t2)
End Class
Class Component-Profile
  sort : index, grade
  use : Bool
  opns :
  New() →component-profile
  Append(Component-Profile, index) →Component-Profile
  Insert(Component-Profile, index)→Component-Profile
  DetermineGrade(Component-Profile,  index)→Component-
Profile
  IsIn(Component-Profile, index)→Bool
axiom :
  cp : Component-Profile, i : index, g : grade
  Insert(cp,i)= =if IsIn(cp,i) then Determine Grade(cp, i)
else Append(cp,i)
End Class
```

Two illustrations define the most general object for Classification Scheme and Component-Profile, respectively. Using these general objects, we can define more detail objects for each representation such as Enum-Component-Profile, Facet-Component-Profile, IR-Component- Profile. These subclasses have own representation way to determine the semantics of representation.

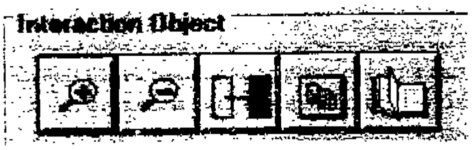
We suggest the visual space as retrieval environment. Interface object is visual space displaying the information on components. Our retrieval model shows graphical information on component so visual representations easily lead the user of library to arrive their goal. Library user can access the library without knowing the library structure and formal query if they used displaying informa-



(Fig. 3) Interface Object Model

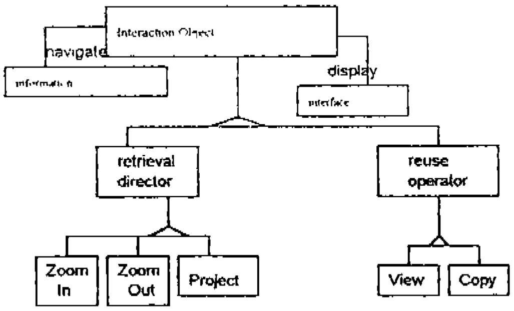
tion on visual space that sometimes shows detail information and sometime gives very abstract information.

(Fig. 4) shows the real interface icon for icon object. We named Zoom-in, Zoom-out, Project, Copy, View operator from left to right.



(Fig. 4) Interface Icon

These are interaction between the information and the interface. We define the retrieval operator as interaction object. Interview object is necessary to extend existing library structure and adding several visual representations. (Fig. 5) shows interaction object model.



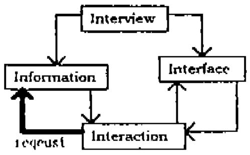
(Fig. 5) Interaction Object Model

This interaction model is corresponding the operations of whole retrieval object so that defined operations manipulate the data that are defined in information model and display the results to interface model. We defined interaction object as followings :

```
Class Interaction-Object
  use : Classification-Scheme, Access-Point,
        Similarity-Information, Component-Profile,
        Component
  Pons :
  Initial-View(Classification-Scheme)→Access-Point
  Zoom-In(Classification-Scheme, Access-Point, Component
        Profile)→Component-Profile
  Zoom-In(Classification-Scheme, Access-Point, Component
        -Profile)→Component-Profile
  Zoom-In(Classification-Scheme, Access-Point, Component
        -Profile)→Component
  Zoom-Out(Classification-Scheme, Component-Profile)→Co-
        mponent-Profile
  Zoom-Out(Classification-Scheme, Component-Profile)→Ac-
        cess-Point
  Project(Classification-Scheme, Component-Profile, Similar-
        ity-Information)→Component-Profile
  View(Component)→Code | Design Spec. | Example
  Copy(Component)→Code
  Axiom :
  cs : Classification-Scheme, ap : Access-Point, cp : Compo-
        nent Profile
  Zoom-Out(cs,cp,Zoom-In(cs,ap,cp)) = Zoom-In(cs,cp,
        Zoom-Out(cs,ap,cp))
End Class
```

In interaction object, we define each opera tion can handle toppest class of information object, such as classification-scheme, access-point, similarity-information so on. At operation time, this interaction object can bind with subclasses of top class that have own implementation.

We add interview object for extendibility. Interview object handles new classification scheme and visualization methods. Librarian can redefine the semantic of top class of information object and extend the visual clue and visual space with interview object. (Fig. 6) shows the interaction model for 4I model.

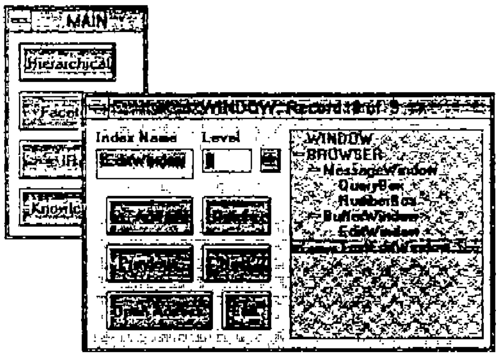


(Fig. 6) Interaction Model of 4I Model

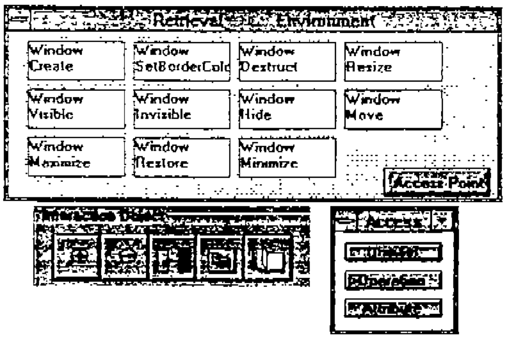
Librarians construct the library or extend the classification scheme and interface objects with interview objects that can access information object and interface object. Reuser can access the library with interaction object that navigate the information object and display the results of retrieval on interface object. Also, interface object and information object gives several information needed by interaction object such as constraint or features, that is the visual clue. Even the library structure has changed, interaction object can handle the subclasses of information without any change.

4. Realization of Retrieval Framework

Our retrieval framework was implemented on top of an event-driven mechanism as a part of CARS(Computer-Aided Reuse System) [12]. As we defined previously, our framework was made up with 4 object, information object(CScheme), interface object(CVisual), interaction object(CControl) and Interview object(CInterview). We implemented our framework to hold 4 instances of each object model and control the activities of these 4 instances based on our interaction model among 4 objects model. CScheme defines the interface that defines the protocol between CControl and CScheme as well as constraints of classification scheme. Therefore the CScheme can be subclasses that have detail semantic. CControl handles the interaction by sending out message according to the type of operator it receives and coordinates the activities of navigating and displaying. CVisual defines the appearance of visual date



(Fig. 7) Example of CInterview



(Fig. 8) Component Retrieval Environment

<Table 1> The results of evaluation

Criterion	RSL	Diaz	LaSSIE	OO Retrieval Framework
Extendibility	just add new item into template	extend the facets or add the terms to facet	extend the just frames	extend Cschem by defining new subclasses for new classification sheme
Classification Scheme	IR	Faceted	Knowledge-Based	support all
Query	Natural Language	Restricted Terms	template	Visual Clue
Retrieval	Keyword matching	Keyword matching	Keyword matching	Visual resoning
UI	Menu-Driven	Menu-Driven	Text-based	Visaul

and visual clue stored in CScheme by maintaining a collection of displayable objects. We manipulate the visual clue as text displayed in graphic objects not symbols. CInterview provides the way to modify and extend the semantic of CScheme and accept new visualization techniques. Also it insulates the changes of CScheme and CInterface.

Following (Fig. 7) shows the simple result of CInterview for constructing the library based on enumerative classification hierarchy and (Fig. 8) show an illustration of retrieval environment.

<Table 1> show the results of the evalutation of our framework. Becuase our model is a result from older classification scheme and the purpose of our model is the flexibility, our framework is adaptable into new classification scheme. The evalation crieterion were proposed in [8].

5. Conclusion

To reuse the retrieval system of reusable component, it has to be independent from the underlying classification scheme. Developed retrieval environments are tightly coupled with underlying component representation. There are already several component representation ways, free-text indexing, faceted index, and enumerative representation so on. These representation methods can be related with the library characteristics that is, subroutine library or object−oriented library with document or library without any description etc. Even librarian should use the different classification scheme according to the application domain, reuser wants to have a single interface with library. It reduces the learning time and consistent use of library. The lessons from the survey of several classifications and retrieval environment, we thought that retrieval model and library structure can be generalized. Regardless of the library structure, reuser has their first access point to start their retrieval and add their knowledge to the retrieval system using the result of their actions which retrieval environment answer to reuser's input. When reuser can't be satisfied with the result they remove their knowledge which reuser just inputted to return the previous retrieval point. Additionally, reuser wants to across the library with already inputted query for browsing. So, we suggested object-oriented retrieval framework with our 4I model. We defined library and classification scheme as the data to be handled several retrieval interaction operations. We put together these data and operations into one retrieval object with various interfaces. For extendibility, information objects and interface objects can be redefined and extend with inheritance and polymorphism mechanism by interview object. Basically, retrieval environment can be operated based on message passing mechanism that is defined as interaction model of 4I model. With our retrieval framework, it is possible to support the multiple underlying representations and single interface of reuse so that it meets the new requirement of reuse system, extendibility and flexibility.

References

[1] M. Wood & I. Sommerville, "An Information Retrieval System for Software Components", In Proceeding of ACM SIGIR Forum, Vol.22, No.3, 1988.

[2] P. Devanbu, et. al, "LaSSIE: A Knowledge-Based Software Information System", CACM, Vol.34, No.5, pp.34-49, 1989.

[3] R. Helm & Y. S. Maarek, "Integrating Information Retrieval and Domain Spe-

proaches for Browsing and Re- : Object-Oriented Class Librar Proceeding of OOPSLA'91, pp. -91.

akes & T.P. Pole, "Proteus:A Reuse Library System that Multiple Representation Meth- Proceeding of ACM SIGIR Vol.22, No.3, 1988.

gt. et.al. "Computing Similarity se Library System:An AI-Based ". ACM Transaction on Soft- gineering and Methodology, Vol. pp.205-228, 1992.

D. W & Woodfield. S.N., "A e structure for reusing abstract s", In Proceedings of the 9th Software Engineering Confer- .360-368, 1987.

Burton, et al, "The Reusable Sof Library", IEEE Software, pp. 25- -87.

[8] W. ces, B.A. Nejmeh, "An Information System for Software Reuse", In Proceeding of the Tenth Minnowbrook Workshop on Software Reuse, 1987.

[9] Ruben Prieto-Diaz, "Classification of Reusable Modules", IEEE Spftware Vol.4, No.1, pp.1-16.

[10] S.K.Chang, "Visual Reasoning for Information Retrieval from Very Large Database", IEEE Workshop on Visual Languages, pp.1-6, 1989.

[11] R.Breu, "Algebraic Specification Techniques in Object Oriented Programming Environments", Springer-Verlag,1991 .

[12] Jeong Ah Kim & Kyung Whan Lee, "CARS:Retrieval and Understanding supporting Environment", In Proceeding of InfoScience'93, 1993 .

[13] Lewis J. Pinson, Richard S. Wiener, An Introduction to Object-Oriented Programming and Smalltalk, Addison-Wesley Publishing Company, 1988.

김 정 아

1988년 중앙대학교 전자계산학과 졸업(이학사)
1990년 중앙대학교 대학원 전자계산학과 졸업(공학석사)
1994년 중앙대학교 대학원 전자계산학과 졸업(공학박사)
현재 중앙대학교 컴퓨터공학과 연구원(Post-Doc)
관심분야 : 객체지향 소프트웨어 공학, 형식명세의 재사용, 객체 모델의 검증, 객체지향 시각프로그래밍 언어

문 충 렬

1994년 중앙대학교 전자계산학과 졸업(공학학사)
1995년 현재 중앙대학교 대학원 컴퓨터공학과 재학중(석사과정)
관심분야 : 객체지향형식 명세 방법, 재사용, 객체 모델의 검증

김 승 태

1994년 중앙대학교 전자계산학과 졸업(공학학사)
1995년 현재 중앙대학교 대학원 컴퓨터공학과 재학중(석사과정)
관심분야 : 객체지향 소프트웨어 개발 방법론, CASE, 사용자인터페이스 생성기, 시각화 도구