SOPView : A Visual Query and Object Browsing Environment for SOP OODBMS

Seong-Woo Chang
(swchang@gaston.snu.ac.kr)

Suk-Ho Lee
(shlee@snucom.snu.ac.kr)

Hyoung-Joo Kim
(hjk@im4u.snu.ac.kr)

Department of Computer Engineering
Seoul National University
Seoul, Korea 151-742

Abstract

SOPView is a graphical user environment for SOP OODBMS [1]. SOPView supports the frame-based visual specification of object-oriented database query elements such as target classes, projections, conditional predicates, and path expressions. SOPView also provides a browsing method which consists of components such as visualization structure, synchronized browsing, and intermediate reference set query in order for users to understand and access complex reference hierarchy in object-oriented databases with ease. [2]

1 Introduction

Object-oriented databases provide powerful data modeling features such as class inheritance, reference, and set attribute. Therefore, object-oriented databases are suitable for advanced applications, for example, multimedia, CAD, and GIS. In object-oriented databases, the unit of storing and retrieving information is an object. Usually, the object contains a lot of reference information. The reference information represents the inter-relationship between objects, and plays an important role to express rich semantics. However, the richness of concepts in the object-oriented data model imposes the complex structure for information modeling. Especially, the cross-reference of objects' relationships and references to set of objects make information structure more complex. Due to this complexity, it is difficult to understand and access the object-oriented databases. Therefore, in order to get easy and correct understanding of information on object-oriented database, an object and its relationships with other objects need to be combined and presented together. Also, it is required to clearly visualize whether referenced object is a single object or a set of objects. In order to support this kind of work, we need systematic and easy-to-use paradigms.

Currently, many GUI systems have been developed for the purpose of browsing objects on object-oriented databases[9][10][11][12][13][14][15][16]. Most OODB GUIs provide the functions of displaying content of both an object and referenced object, and representing reference relationship. Especially, KIVIEW[9] and OdeView[13] provide *synchronized browsing* to release the burden of navigating every reference path during browsing objects. This technique displays both an object and referenced object at the same time, and, if current object is changed to the next object, the object which is referenced by the next object is displayed automatically. However, these systems have the shortages that 1) there is no clear visualization structure for the case that the referenced object is not a single object but a set of objects, and 2) they cannot display several objects simultaneously in a screen. As the other approach to avoid navigating every reference path, J. H. Cha[16] offers a browsing method via composite icon. This method displays result objects of query using icons, and shows the attributes of referenced objects together at the same time. With this simultaneous presentation, users can see and compare the attributes of result objects and referenced objects. But, this method 1) permits only comparison at root object level, and 2) has too simple structure to handle

[1] SOP OODBMS stands for SNU OODB Platform Object-Oriented Database Management System, which has been developed for four years 1992 to 1995 in the OOPSLA laboratory of Seoul National University.

[2] This research was partially supported by the Ministry of Trade, Industry, and Energy of KOREA under project 943-20-4, "Implementation of Design Tools for Object-Oriented Database".

0730-3157/96 $5.00 © 1996 IEEE

a set of referenced objects.

To overcome the shortages stated above, we designed and implemented a graphical user environment called SOPView which visualizes the reference hierarchy and the set of objects clearly, and supports the synchronization and comparison of objects in every stage of reference path. It provides facilities such as frame-based visual query specification and execution, visualization structure for complex reference hierarchy, browsing method that let users freely search and compare information following the reference path, and intermediate reference set query. In this paper, we present design concepts and implementation details of SOPView for SOP OODBMS. SOP supports ODMG(Object Database Management Group) specification[1] which is a de-facto standard for object-oriented data model.

In section 2, we give a brief explanation about visual query interface, and describe about browsing interface in section 3. In section 4, we demonstrate the execution of SOPView. In section 5, we explain related works about GUIs for OODBMS. Finally, we conclude and present future works in section 6.

2 Visual Query Interface

In this section, we explain briefly the concept of frame-based visual query interface of SOPView. Due to the size limitation of this paper, only the main features are explained here. A more complete and precise treatment of visual query interface including nesting, function composition and formalization of visual query language will be provided in a forthcoming paper[17].

2.1 Frame-Based Structure

The frame-based structure for visual query interface is like figure 1. Visual query interface consists of both *schema window* and *query window*. Schema window displays the schema of database. User can select target class from this schema window. Query window helps specification of query with visual manner. Users can specify components of query such as classes which can be targets of query, attributes in target classes which users want to project, conditional predicates to locate objects of interest, and output type like struct, array or list. These components can be specified on the panes in the query window such as *target class pane, projection attribute pane, condition pane,* and *output type pane* respectively. User can write visual query by filling these panes out. The visual query

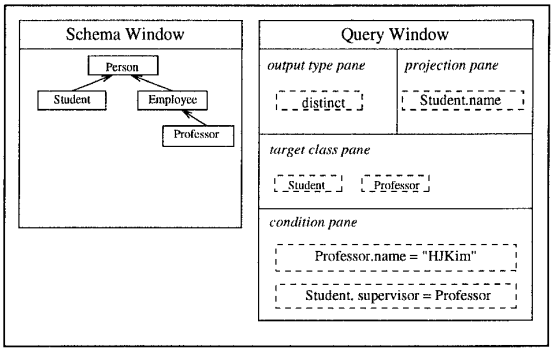


Figure 1: Frame-based structure for visual query interface

specification is converted into ODMG OQL[1] expression, and executed on SOP query processor.

2.2 Visual Query Specification Method

Target Class and Projection For the specification of target class, user first has to choose and place class icon from schema window to target class pane on query window using 'drag&drop'. To specify projection, user first has to click target class button. Then, the attributes of target class will be shown up on pull-down menu. Among this, user can select attribute for projection and place it onto attribute projection pane also using 'drag&drop'.

Conditional Predicate For the specification of conditional predicate, user first has to click the button in condition pane. Then, the dialog for input of conditional predicate will appear. As a next step, user has to specify left operand by 'drag&drop' attribute from target class, select operator, and later specify right operand by 'drag&drop' attribute or inputting value.

Path Expression For the specification of path expression, user has to click the target class button which will be the starting point of the path expression. Then, the attributes of target class will appear as a form of pop-up dialog. Now, click the attribute which is the element of path expression. If user arrives at the final element of path expression by doing this kind of job continuously, 'drag&drop' the last attribute onto the pane of interest. The whole path expression is created automatically by the system.

3 Object Browsing Interface

In this section, we present the detailed explanation about each component of browsing interface.

3.1 Visualization Structure

Figure 2 shows the visualization structure for browsing interface.

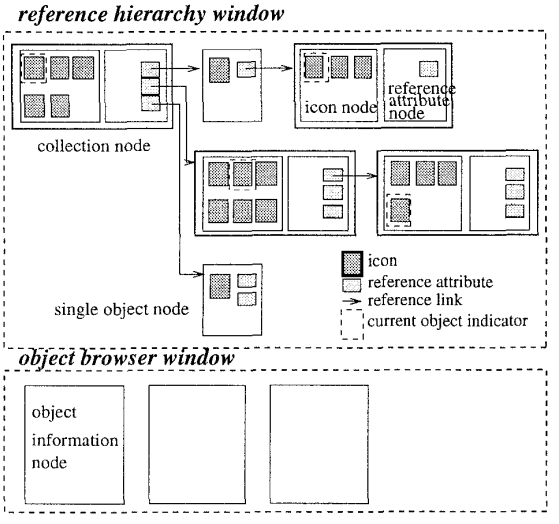


Figure 2: Visualization structure of browsing interface

The visualization structure consists of the following components :

- *Object node* is an abstract super class for both collection node and single object node, and represents an unit of reference structure.
- *Collection node* displays a set of objects and represents that reference is a set of referenced objects. It consists of an icon node and a reference attribute node.
- *Single object node* shows a single object and consists of only one icon and reference attributes of a object.
- *Icon node* consists of icons that show the pictorial attribute values of objects,
- *Reference attribute node* displays the reference attributes of current object.
- *Reference hierarchy window* visualizes the structure of reference hierarchy. Here, the reference hierarchy can be represented by the connection of object nodes via links.
- Through *object information node*, user can see the whole information of object.
- *Object browser window* is an window to display the set of object information nodes and to control the display of them.

This structure offers the benefits to users such that : 1) it provides visual representation structure for complex reference hierarchy, 2) it displays clearly that whether a certain reference path is to a single object or to a set of objects, and 3) it supports both global visualization and specific representation of objects in this reference hierarchy structure

On this visualization structure, user can use the functions such as synchronized browsing and intermediate reference set query.

3.2 Synchronized Browsing

SOPView supports synchronized browsing[9][13]. To support more specified synchronization and help users understand the pattern of synchronization, we identify the following three types of synchronized browsing in figure 3.

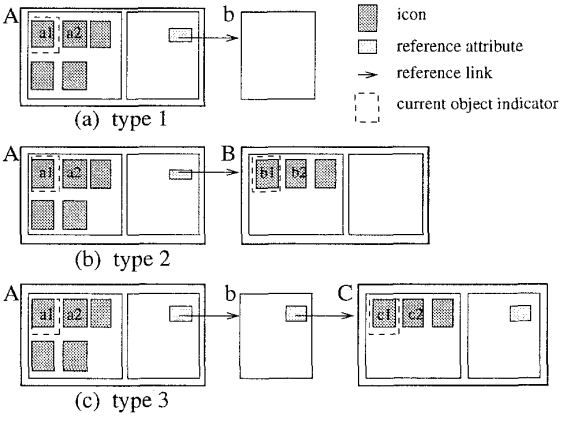


Figure 3: Three types of synchronized browsing

Type 1 In this type 1, current object a_1 in the set A of objects refers another object b. If current object is changed to object a_2, the referenced object also is changed automatically to object b' which is referenced by object a_2. This type includes the case that object b refers other single object c, and this reference to a single object occurs continuously. In this case, the synchronization is transferred to the final referenced single object.

Type 2 This type is for the case that the set A of objects refers the set B of referenced objects. In this case, if current object a_1 of the set A is changed to next object a_2, the set B' of objects which is referenced by object a_2 will be displayed automatically. And, in the set B', the first object b'_1 will be set as a current object in this set by default.

Type 3 This type shows the case that the current object a_1 in the set A refers the object b, and the object b refers other set of objects C. Now, if current object a_1 of the set A is changed to next object a_2, the object b' which is referenced by object a_2 will be displayed automatically. Later, the set C' which is referenced by object b' will also be displayed concurrently.

In the types above, every collection node can be a *synchronization control point.* Synchronization control point can 1) execute the synchronized browsing and 2) be a basis for discrimination of the type of synchronization. Any kind of composite synchronization can be represented as a combination of above three types. For example, in the visualization structure of figure 2, users can execute a synchronization of type 3 for upper reference path, of a combination of type 2 and type 2 for middle reference path, and of type 1 for lower reference path.

SOPView controls the synchronization of reference hierarchy using this categorization. Every reference path is represented internally by the link of synchronization structure which contains type and control information.

3.3 Intermediate Reference Set Query

When user browses the database, user can happen to get the necessity of querying the set of referenced objects. It is because query tends to be composed incrementally when user had little knowledge of database, and thereby user can run into some questions about this reference set. In this case, doing conditional query to reference set and browsing from restricted reference set which is the result of conditional query can be better than composing another query and browsing from the result of that query.

For these kinds of needs, this function provides the facility for specifying conditional predicate to reference set, getting restricted set of referenced objects which satisfy condition, and browsing from this reduced reference set further through the reference path. The collection node for result objects of this query has a label to show the condition. The demonstration will be shown in the next section.

When synchronized browsing occurs, the reduced reference set is also synchronized, i.e., the condition is applied to the synchronized set of objects and the result of that application is shown.

4 Demonstration of Visual Query and Browsing

This section demonstrates the information access method using visual query and object browsing interface.

4.1 Example Schema

Let us take a look at the example schema for universities in figure 4. This schema contains 'Professor',

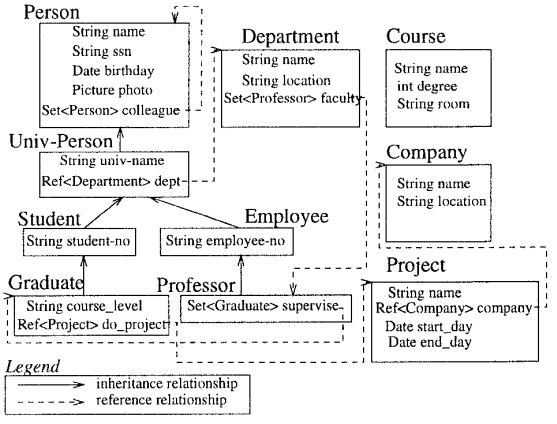


Figure 4: Example schema

'Graduate', 'Student', and 'Employee' classes to model members of universities. 'Person' and 'Univ-Person' classes are parent classes for these. Other classes are provided to model other informations such as membership or activities.

4.2 Specification of Visual Query

Now, let us execute the example query like this : **"Show all professors who belong to the department of computer engineering"**. Figure 5 shows the demonstration of visual query specification.

4.3 Browsing from Query Result Objects

The set of objects are shown up as a result of execution of example visual query. Now, let us follow

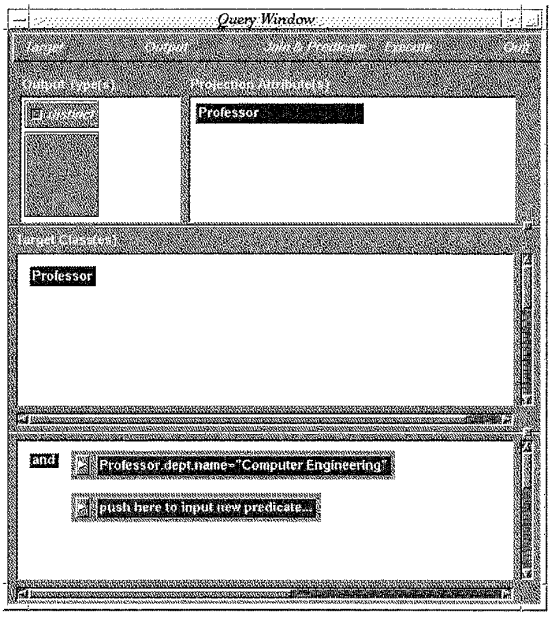
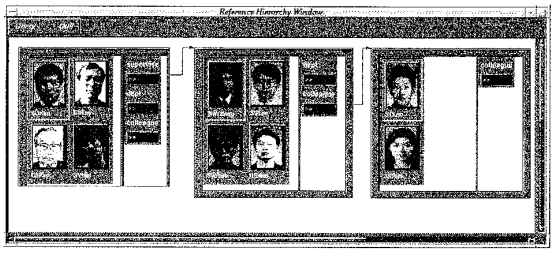
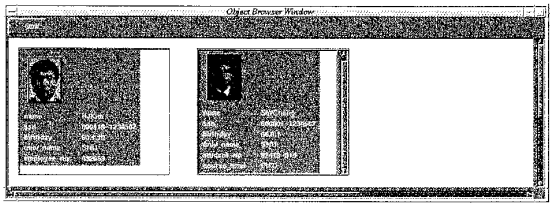


Figure 5: The demonstration of visual query specification



(a) the sets of result objects and referenced objects



(b) the contents of objects

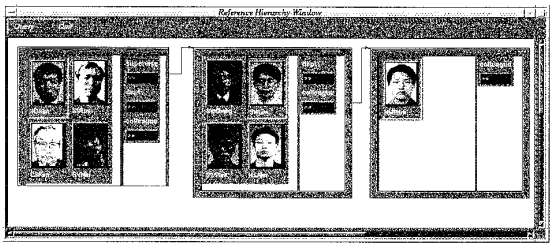
Figure 6: The demonstration of browsing the sets of result objects and referenced objects

the 'supervise' reference which 'HJKim' object in the set of result objects has. If we push the reference button, the set of referenced objects will appear. Next, let us continue to follow the 'colleague' reference of 'SWChang' object in the set of referenced objects. If 'colleague' reference button is to be pushed, another set of referenced objects will be shown up. The demonstration for this browsing are shown in (a) of figure 6.

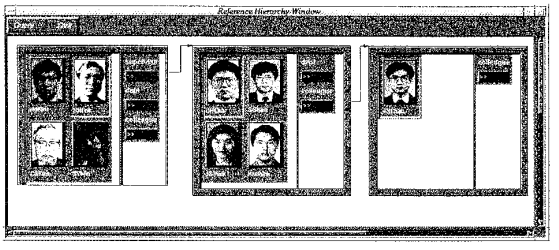
Then, let us look up the contents of 'HJKim' and 'SWChang' objects. If we double-click each icon button, object information nodes appear on the object browser window. The display of contents of two objects are shown in (b) of figure 6.

From now on, we can execute synchronized browsing on the reference hierarchy. First, if we move current object from 'SWChang' to 'SWLee' object on the collection node for 'Graduate' class, the synchronization of type 2 will be invoked like (a) in figure 7. Next, if we move current object from 'HJKim' to 'SHLee' object on the collection node for 'Professor' class, the combination of two type 2 synchronization will be occurred like (b) in figure 7.

Finally, let us examine the example of intermediate reference set query. From the set of objects of 'Graduate' class, let us find and look at only Ph.D course students. To get this results, user can give condition of "course_level = 'Ph.D course' ". As a result of this



(a) the synchronization of type 2 between the sets of objects of 'Graduate' and 'Person' class



(b) the synchronization of combination of type 2 between the sets of objects of 'Professor' and 'Person' class

Figure 7: The demonstration of synchronized browsing of type 2 and combination of two type 2

query, user can get results like figure 8.

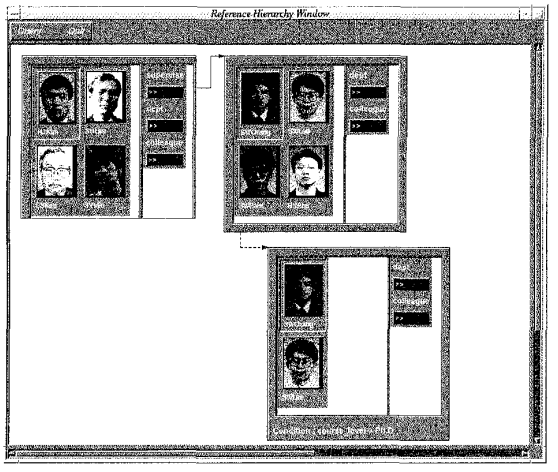


Figure 8: The demonstration of intermediate reference set query

4.4 Implementation

The SOPView prototype has been developed in a UNIX environment on SUN workstations. It consists of about 20,000 lines of C++[2][3][4] code. Motif[5][6][7][8] widget has been used to facilitate the implementation of the graphical interface. Currently, the visualization model, synchronized browsing, and intermediate reference set query have been developed.

5 Related Works

Many GUI systems for browsing informations on OODBMS have been proposed [9][10][11][12] [13][14][15][16].

KIVIEW[9] first proposed the synchronized browsing method to avoid repetitive navigation through reference path. Its internal model is characterized as a semantic network, its browsing session interleaves navigation and manipulation activities. However, it allows users to display only the immediate attributes, and it supports no visual query facility.

IconicBrowser[10] represents classes and objects which consist of database as icons, and provides facilities for querying visually and browsing result of query using these icons. However, it does not provide method for browsing reference objects and synchronized browsing.

MoodView[11] is a graphical front end for MOOD OODBMS. User can execute class definition and modification, object browsing, object management and visual query. Objects are represented through two modes of SID(Single Instance Display) and MID(Multiple Instance Display). It represents reference informations, but, does not support synchronized browsing.

SMARTIE[12] system is a form-based GUI for ITASCA OODBMS. It is composed of several abstract class methods that automatically configure visual representations based on domain information of each attribute in the class. These methods can handle the dynamic modification of any part of database schema. However, SMARTIE system does not support visual query interface, and only admits text-based query.

OdeView[13] is a system that mostly influenced SOPView. OdeView is a graphical interface for Ode OODBMS, and it provides facilities for examining schema, querying visually, browsing the database, and synchronized browsing. Especially, OdeView overcomes the limited representation of referenced objects in KIVIEW, and represents referenced objects using separate window. However, it does not support the representation of set of objects and comparison of attributes of interest.

GOMI[14] is a GUI for object-oriented database. GOMI provides functions such as object definition, object browsing, object manipulation and visual query. In GOMI, objects are represented together with schema, and reference relationship and referenced objects are also displayed. However, it does not support the synchronized browsing.

SUPER[15] is a graphical interface for ERC+ data model, which is a relational and object-oriented model. SUPER supports visual query interface and two modes of object browsing methods, one is form-based browsing and the other is graphic mode one. In graphic mode browsing, objects are represented as a set of objects on schema. This system represents reference hierarchy in graphic mode, and also supports synchronized browsing.

J. H. Cha[16] proposes a browsing method via composite icon. This method displays result objects from query execution at the same time using icons, and also shows the attributes of referenced objects together. With this simultaneous presentation, users can see and compare the attributes of result objects and referenced objects at the same time. However, this method enables only comparison at root object level, and has too simple structure to handle a set of referenced objects.

6 Conclusion

In this paper, we presented design and implementation of SOPView, a graphical user environment for SOP OODBMS. Major contributions of SOPView are as follows :

- A *visualization structure* to represent complex reference hierarchy has been proposed. This structure can visualize the set of objects to distinguish if the referenced information is a single object or a set of objects.
- The *synchronized browsing* method supports the synchronization on every stage of reference path. In order to support it, we categorized synchronization into three types, and present the execution model for each type.
- For effective browsing, conditional query to a set of referenced object called *intermediate reference set query* has proposed.

As a plan of future work, we consider the formalization of visual query interface, and the visualization method for schema structure.

Acknowledgements

We would like to acknowledge the comments of the reviewers which improved the presentation. First author especially thanks Y. S. Shin of Univ. of Michigan for his help in developing prototypes, and acknowledges S. W. Lee for a number of discussions.

References

[1] Cattel, R., "The Object Database Standard: ODMG-93, Release 1.1", Morgan Kaufman Publishers, San Mateo, California, 1994

[2] Bjarne Stroustrup, "The C++ Programming Language", 2nd Ed., *Addison Wesley,* 1992

[3] Margaret A. Ellis, and Bjarne Stroustrup, "The Annotated C++ Reference Manual", *Addison-Wesley,* 1990

[4] James O. Coplien, "Advanced C++ Programming Styles and Idioms", *Addison-Wesley,* 1992

[5] Marshall Brain, "Motif Programming", *Digital Press,* 1992

[6] Douglas A. Young, "Object-Oriented Programming with C++ and OSF/Motif", *Prentice Hall,* 1992

[7] Open Software Foundation, "OSF/Motif Programmer's Guide", *Prentice Hall,* 1993

[8] Open Software Foundation, "OSF/Motif Programmer's Reference", *Prentice Hall,* 1993

[9] A. Motro, A. D'Atri, and L. Tarantino, "The design of KIVIEW: an object-oriented browser", *Proc. of the 2nd Int'l Conf. on Expert Database Systems,* pp. 17-31, 1988

[10] K. Tsuda, M. Hirakawa, and T. Ichikawa, "IconicBrowser: an iconic retrieval system for object-oriented databases", *Journal of Visual Languages and Computing,* vol.1, no.1, pp.59-76, 1990

[11] Ismailcem Budak Arpinar, Asuman Dogac, and Cem Evrendilek, "MoodView: An Advanced Graphical User Interface for OODBMSs", SIGMOD RECORD, Vol.22, No.4, Dec., 1993

[12] R. V. Zoeller and D. K. Barry, "Dynamic self-configuring methods for graphical presentation of ODBMS objects", *Prof. 8th Int'l Conf. on Data Engineering,* pp.136-143, 1992

[13] R. Agrawal, N.H. Gehani, and J. Srinivasan, "OdeView: The Graphical Interface to Ode" *Proc. of Int'l Conf. on Management of Data,* San Diego, California, 1990

[14] Y. S. Jun and S. I, Yoo, "GOMI: a graphical user interface for object-oriented databases", *Int'l Conf. on Object-Oriented Information Systems,* 1994

[15] A. Auddino, E. Amiel, Y. Dennebouy, Y. Dupont, E. Fontana, S. Spaccapierta, and Z. Tari, "Database Visual Environments Based on Advanced Data Models", *Proc. Work. on Advanced Visual Interfaces,* pp. 156-172, 1992

[16] J. H. Cha and S. H. Lee, "Browsing Multimedia Objects via Composite Icons", *Proc. of IEEE Conf. on Systems, Man, and Cybernetics,* Vancouver, Canada, 1995

[17] S. W. Chang and H. J. Kim, "Visual Query Language for Object-Oriented Databases". *In preparation*