A Scenario-Based Object-Oriented Methodology for Developing Hypermedia Information Systems

Heeseok Lee, Choongseok Lee, and Cheonsoo Yoo
Graduate School of Management,
Korea Advanced Institute of Science and Technology
E-mail: hlee@msd.kaist.ac.kr

Abstract

This paper proposes an object-oriented methodology for developing hypermedia information systems. The methodology consists of six phases: domain analysis, object modeling, view design, navigation design, implementation design, and construction. Users' requirements are analyzed with a responsibility-driven technology using scenarios. Object-oriented views are generated as the result of object modeling, and then used for the subsequent navigation and implementation design. The implementation design phase deals with database schema, page structure and flow, and user interface. This methodology is effective for integrating enterprise databases with distributed hypermedia systems via Internet or Intranet.

1. Introduction

Hypermedia information systems enable users to share information through a variety of media such as text, video, image, and voice. Hypermedia systems can be widely used for World Wide Web (WWW) applications via Internet or Intranet [6].

Hypermedia extend the hypertext paradigm into multimedia. Hypertext is a nonsequential manner of looking at text-based information. Even though manuals, books, reports, or some other documents may be stored within a computer system in a sequential manner, hypertext provides an interface so that any number of nonsequential links may allow a user to access an item of interest even if it is not the next sequentially-stored item. Instead of navigation among text objects, hypermedia offer links among the whole spectrum of multimedia information [26].

Hypermedia development, especially on a commercial scale, often involves teams of developers who need to be managed and coordinated over an extended period of time. Formal systems development and project management techniques are needed to ensure that the hypermedia product meets its objectives, and is completed on time and within budget [14]. Therefore, it is not surprising to note that the research for hypermedia development methodologies is active (e.g. HDM [8], EORM [18], RMM [14], and OOHDM [24]).

HDM (Hypermedia Design Method) provides the first formal model for hypermedia applications. It is based on an object-oriented technique that encourages the use of different perspectives in presenting the same conceptual entity. EORM (Enhanced Object Relationship Model) is the first object-oriented hypermedia design methodology. EORM consists of three frameworks: class framework, composition framework, and graphical user interface (GUI) framework. RMM (Relationship Management Model) uses Entity-Relationship (E-R) abstractions; it enhances HDM by the use of additional access structures (conditional indexes and guided tours). RMM phases include E-R design, slice design, navigation design, conversion protocol design, user-interface design, runtime behavior design, and construction/testing. OOHDM (Object Oriented Hypermedia Design Model), another object-oriented methodology, adopts four design phases such as conceptual design, navigation design, abstract interface design, and implementation. A view based hypermedia design methodology (VHDM) has been proposed by Lee et al. [20]. Views are used to represent users' perception of hypermedia requirements. VHDM facilitates exploring the functionality of views in hypermedia applications.

Previous studies have several weaknesses. First, they employ relationships among data for determination of navigational paths. These data-oriented relationships in isolation may not reflect users' navigational requirements satisfactorily. Second, the integration of hypermedia with enterprise database is not sufficiently emphasized, in view of its importance for Intranet implementation. Third, research in the past adopt conceptual data models to capture users' requirements.

1060-3425/98 $10.00 (C) 1998 IEEE

Those models may not provide the flexibility that hypermedia systems require.

To solve the above problems, this paper develops a scenario-based object-oriented hypermedia design methodology (SOHDM). The paradigm of object orientation should extend to the hypermedia development environment as a whole [26]. SOHDM has several new features. SOHDM identifies requirements for hypermedia applications from the beginning of system development. Scenarios are used to enhance the expressive capability of modeling. In particular, object-oriented views are used as navigational units as well as logical user views.

2. Methodology Architecture

SOHDM consists of six phases: domain analysis, object modeling, view design, navigation design, implementation design, and construction. The framework of SOHDM is depicted in Figure 1. For the purpose of clearer presentation, feedback among phases is not depicted in Figure 1. However, feedback is important for refining analysis or design outputs in each phase.

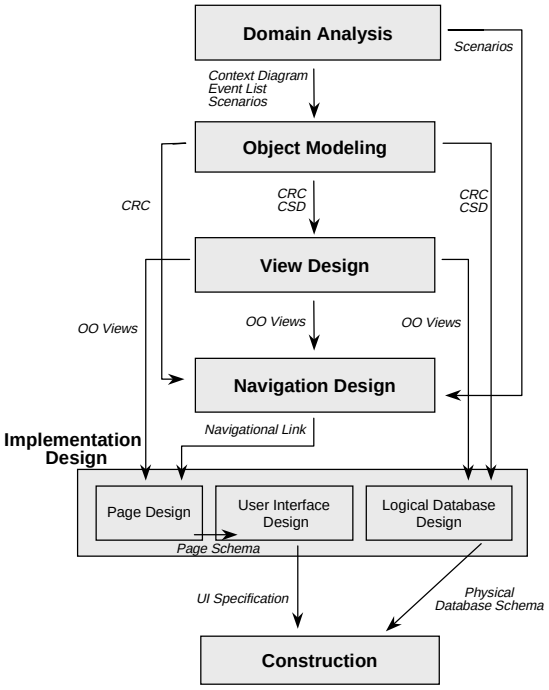

Figure 1. Architecture of SOHDM

In the domain analysis phase, a context diagram is drawn to represent the system boundary. In addition, scenarios are employed to identify users' requirements. The scenarios result in an object model within the framework of Class Responsibilities Collaboration (CRC) [30]. For better representation of relationships among object classes, a Class Structure Diagram (CSD) is drawn. In the view design phase, object-oriented (OO) views are extracted from CRC cards. OO views are navigational units in the navigation design.

The navigation design phase defines Access Structure Nodes (ASNs) and navigational links. ASN contains the access structure and path to OO views. For the sake of convenience, navigational links are presented in the form of a navigational link matrix. Implementation design phase defines users' windows (e.g., a set of HTML pages) and navigational flows from one page to another page. In addition, detailed user interfaces (UIs) are designed. OO views may be transformed into relational database schema. Clearly, database independence is guaranteed [4]. Finally, a hypermedia information system is built which depicts the resulting physical database schema and UI specifications.

3. Methodology Details

Here we explain each phase of SOHDM in further details, by using a real-life application system. The application has operated on a client/server architecture for one of the largest banks in South Korea. The bank is currently interested in an Intranet infrastructure, and thus SOHDM is applied for building a hypermedia system for this application.

3.1 Domain Analysis

First, a system scope diagram is drawn to delimit the hypermedia system to be developed. A well known data flow diagram (DFD) [31] is used. Typically, a context diagram (e.g., zero level DFD) is a good alternative. For our IS (Bank Account System), the system scope diagram is drawn as shown in Figure 2. The bank accounting system has three external entities, such as management, customer, and branch.

Events are identified for each external entity. An event is defined as the trigger which starts the system [15]. Four events are found, as shown in Table 1.

SOHDM uses scenarios to identify hypermedia applications requirements from the earliest opportunity. Scenario activity charts (SACs) are developed to describe scenarios. SAC describes business processes according to actors. An actor is an operator of specific activities, i.e., a creator of events. Notation of SACs is given in Figure 3. External entities are the primary candidates for actors in SACs. An event is a starting point, a trigger of a scenario. An activity is an operation by which an actor completes a scenario. An alternation

is an activity which enables for an actor to choose the next activity. An activity flow is a sequence of activities. Termination is the end of a scenario.

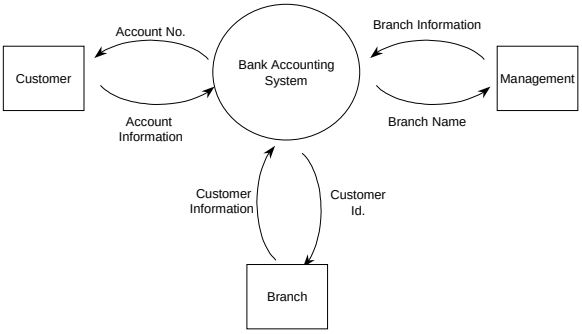

Figure 2. System Scope Diagram

Table 1. Event List

Source Entity	Event Name
Branch	Request Total Information*
	Request Customer Information
Customer	Check Account
	Identify Banking Product Type
Management	Request Total Information*

** The Same Event*

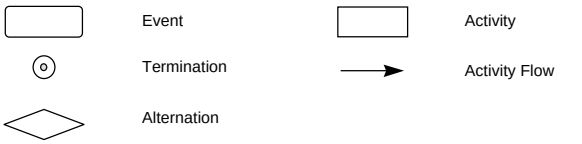

Figure 3. Notation of SAC

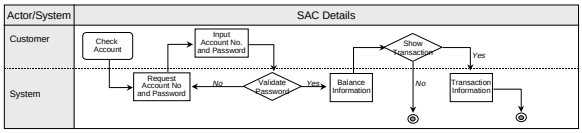

Figure 4. "Check Account" Scenario

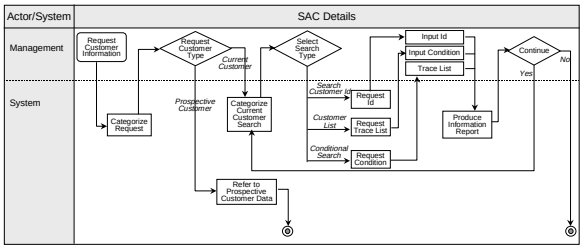

Figure 5. "Request Customer Information" Scenario

Four SACs are generated from the four events. For the sake of simplicity, three scenarios are illustrated in Figures 4, 5, and 6. For example, Figure 5 depicts the SAC for "Request Customer Information" event. Customers are grouped into two categories, such as current customers and prospective customers, and customer information is provided for each category.

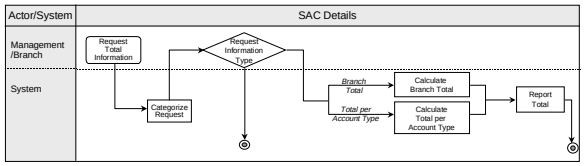

Figure 6. "Request Total Information" Scenario

3.2 Object Modeling

Scenarios in SACs are used for object modeling. Scenarios are transformed into objects in the form of CRC cards. The CRC cards are adopted because they have attractive informal appeal that helps make complex modeling tractable. Objects are generated from scenarios as follows.

First, actors are primary candidates for objects; then activities are considered. Scenarios (Figure 4,5, and 6) generate object candidates such as current customer, prospective customer, account, branch, and transaction. In particular, another object, customer, a superset of current customer and prospective customer object, is obtained.

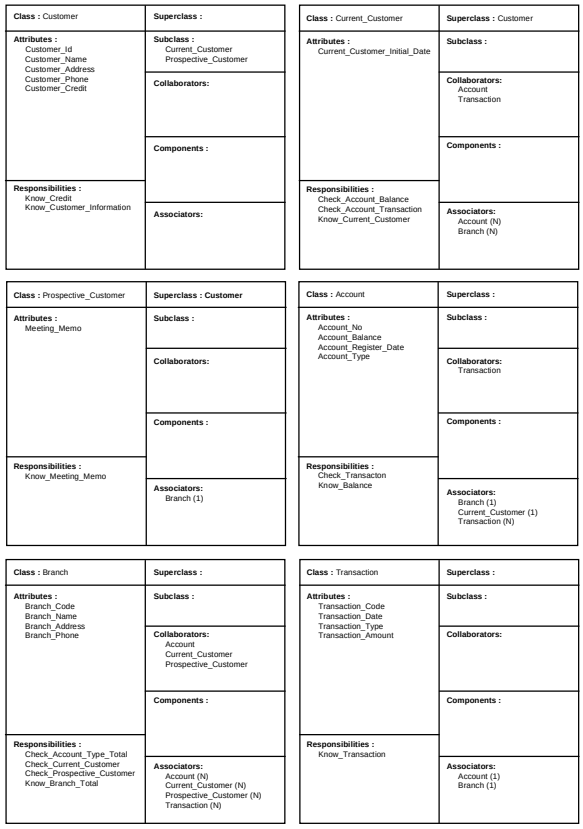

Figure 7. CRC Cards

Next, objects identified are documented in formatted index cards, CRC cards. Here, Wirfs-Borck et al.'s original CRCs are extended to include attribute lists and associators. In addition, cardinality of association is depicted in a bracket. The cards may be specified repeatedly for refinement. Six object classes (Customer, Current_Customer, Prospective_Customer, Account, Branch, and Transaction) are modeled as shown in Figure 7. For example, in the Current_Customer CRC, it is noted that its superclass is Customer and its subclass does not exist. Current_Customer class inherits attributes and responsibilities from Customer class. It has Current_Customer_Initial_Date as an attribute and Check_Account_Balance and Check_Transaction as responsibilities.

Four types of relationships are depicted in CRCs (superclass/subclass, collaborators, components, and associators). To present these relationships more effectively, a CSD shown in Figure 8 is prepared.

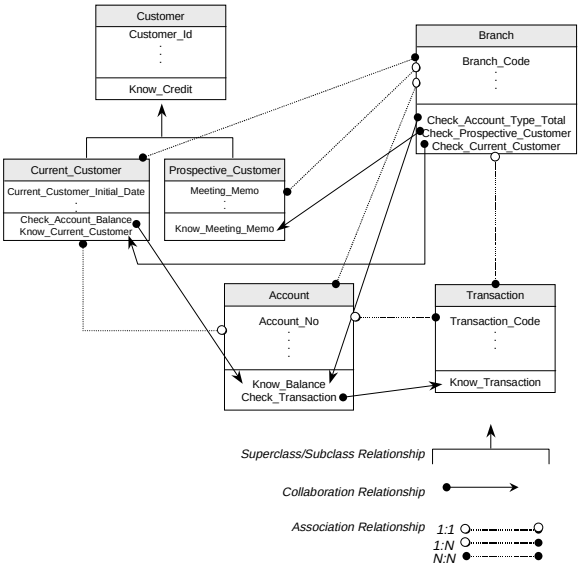

Figure 8. Class Structure Diagram

3.3 View Design

In the view design phase, objects are reorganized for navigational units. A navigational unit represents an OO view. The use of views in hypermedia application design has several advantages. First, views can support a number of users who have different requirements. Second, cognitive overhead can be effectively reduced, because heterogeneous attributes and responsibilities of objects are grouped into views. Third, hypermedia applications are easily extendable, since additional requirements of presentation or navigation can be accommodated.

We can extract OO views on the basis of responsibilities and attributes in CRC cards as well as their relationship in CSD. OO views are categorized into three types: base view, association view, and collaboration view. A base view is generated from a single object class. An association view is extracted from an association relationship. Similarly, a collaboration view is generated from a collaboration relationship.

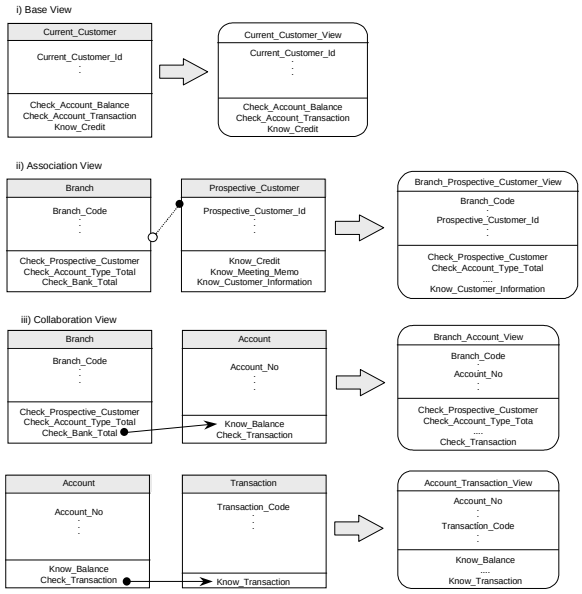

Figure 9. Generating Object Oriented Views

In Figure 9, the Current_Customer_View, a base view, is a subset of attributes and responsibilities in the Current_Customer class. Branch_Prospective_Customer_View, an association view, uses attributes and responsibilities in Branch class and Prospective_Customer class. The origin of responsibilities for association views differs from that for collaboration views. In an association view, the responsibilities related with attributes are extracted. However, in case of a collaboration view, the responsibilities required for collaboration are of importance.

3.4 Navigation Design

The navigation design phase deals with the manner in which users gather and use information on the basis of scenarios. In well-designed hypermedia applications, the way users explore the hypermedia is an important design issue, in order to avoid redundant information and prevent them from getting lost in the hyperspace [24]. Past studies base the navigation design on data models in isolation. However, we suggest that the use

of scenario as well as data model should improve the quality of the navigation design.

In navigation design, OO view and access structure node (ASN) are adopted for navigational units. ASN is used to implement the grouping, which is a menu-like mechanism that enables users to access other parts of hypermedia documents in RMM [14]. ASN provides users the access structures which users can use to navigate to different or detailed part of hypermedia application.

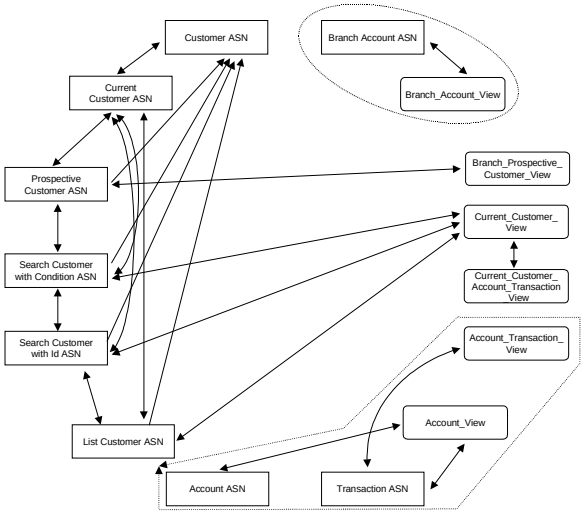

Figure 10. Navigational Link

The first step in the navigation design phase is to determine ASNs by using scenarios. In a scenario, activity flows that start from the system to the actor are the primary concern for determining ASN. An alternation or activity includes two types of activity flows such as activity inflow and activity outflow. An activity inflow may become an ASN. An activity outflow from an activity may be input fields of the ASN, and activity outflows from an alternation may be the menu of the ASN.

For example, in "Request Customer Information" scenario (Figure 5), the activity flow from "Categorize Request" to "Request Customer Type" may be an ASN entitled "Customer." The activity outflows from "Request Customer Type," such as "Current Customer" and "Prospective Customer" may be menus of the ASN. "Customer" ASN has two menus, "Current Customer" and "Prospective Customer."

Next, navigational links are built. In SOHDM, OO views and ASNs are navigational units. The ANS differs from OO view in that ASN contains only access paths to OO views, but OO view contains the actual information that users want to obtain. These OO views and ASNs correspond to nodes. HTML pages are implemented on the basis of these OO views and ASNs in the subsequent implementation design. A link denotes the relationship between source and target node. This source and target node may be OO view or ASN.

From scenarios in Figures 4, 5, and 6, eight ASNs are found and then navigational links are determined, as shown in Figure 10.

Figure 10 shows that the navigational units are categorized into three separate groups, as indicated by dotted lines. Each group is separated from others because it has no related information, i.e., does not have any navigational link that leads to other groups. For the sake of convenience, links may be summarized in a form of a navigational link matrix. This matrix is shown in Figure 11.

Navigational Unit / Navigational Unit	Customer ASN	Current Customer ASN	Prospective Customer ASN	Search Customer with Id ASN	List Customer ASN	Search Customer with Condition ASN	Current_ Customer_ View	Branch_ Prospective_ Customer_ View	Current_ Customer_ Account_ Transaction_ View
Customer ASN									
Current Customer ASN	◪								
Prospective Customer ASN	◪	◪							
Search Customer with Id ASN	◪	◪	◪						
List Customer ASN	◪	◪	◪	◪					
Search Customer with Condition ASN	◪	◪	◪	◪	◪				
Current_Customer_ View	◪	◪	◪	◪	◪	◪			
Branch_Prospective_ Customer_View	◪	◪	◪	◪	◪	◪	◪		
Current_Customer_ Account_Transaction_ View	◪	◪	◪	◪	◪	◪	◪	◪	

◪ If the navigational unit in the first column is source, and the navigational unit in the first row is target.

◪ If the navigational unit in the first row is source, and the navigational unit in the first column is target.

◪ If the navigational unit in the first column and the navigational unit in the first row are bi-directional.

Figure 11. Navigational Link Matrix

3.5 Implementation Design

The implementation design phase generates page structure, page flow, user interface, and logical database schema for construction in a particular development environment. Hypermedia application can be developed under a variety of system environments, including different DBMSs, and development tools such as CGI, HTML, Java [10], or Shockwaves. However, it is noted that our implementation design phase is independent of different environments.

First, the HTML page schema is designed by the organization of OO views, ASNs, and description details (text, image, sound, etc.). Figure 13 shows an example of the page schema. For example, the page "Current_Customer_001" entitled "Current_ Customer" is based on two views, Current_ Customer_ View and Account_Transaction_View. It has two anchors that lead to "Main_001" and "Prospective_ Customer_001" pages.

Page schema in Figure 12 is enhanced to UI specifications by the use of UI components. Figure 13 summarizes the notation for UI components. A caption is a text displayed in a component. An action is an

interactive procedure. An action is executed when a component is clicked. Items are a set of data included in a component. The key-based transition refers to the current state of the components, while the simple transition does not.

PID :Main_001	Title : Main Menu
•**Views**	
• **Description** Description of Main Menu	
•**Anchor** Current Customer / Current_Customer_001 Prospective Customer / Prospective_Customer_001 Management Information / MIS_001	
• Other Component Logo	

PID : MIS_001	Title : Management Information
•**Views** Branch_Account_View	
• **Description** Description of Management Information	
•**Anchor** Main Menu / Main_001	
• Other Component Logo	

PID : Current_Customer_001	Title : Current Customer
•**Views** Current_Customer_View Account_Transaction_View	
• **Description** Description of Current Customer, and Help Message.	
• **Anchor** Main Menu / Main_001 Prospective Customer / Prospective_Customer_001	
• Other Component Logo	

PID : Prospective_Customer_001	Title : Prospective Customer
•**Views** Branch_Prospective_Customer_View	
• **Description** Description of Prospective Customer, and Help Message.	
• **Anchor** Main Menu / Main_001 Current Customer / Current_Customer_001	
• Other Component Logo	

Figure 12. Page Schema

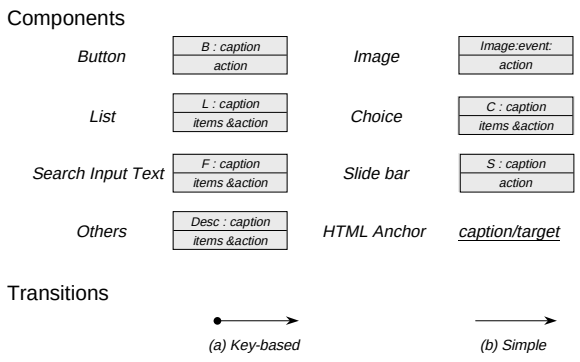

Figure 13. Notation of User Interface Component

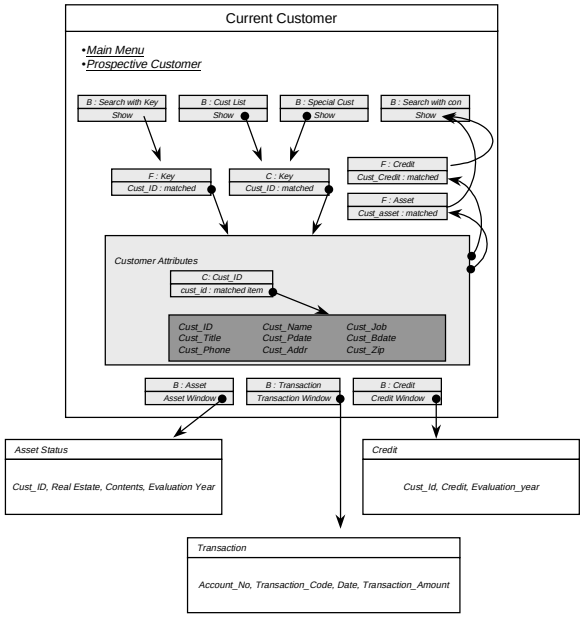

Figure 14. User Interface Design Specification

ASNs determine the UI components. ASNs are transformed into HTML pages or anchors. Direct links correspond to buttons or images. Index, direct query, or indexed query is transformed into a choice or a list. Guided tour is transformed into a slide bar or buttons with "Next" or "Previous." Figure 14 shows the resulting UI specifications for the "Current_Customer _001" page schema.

Next, object model is transformed to logical database schema. Note that the transformation may not be necessary if OODBMS is used. In many real-life cases, however, relational DBMS are the most useful systems. Here, designers need rules that map object model into relational schema. The transformation rules can be found in [3] or [5]. We adapt Blaha's rule. Blaha suggests three rules, as follows:

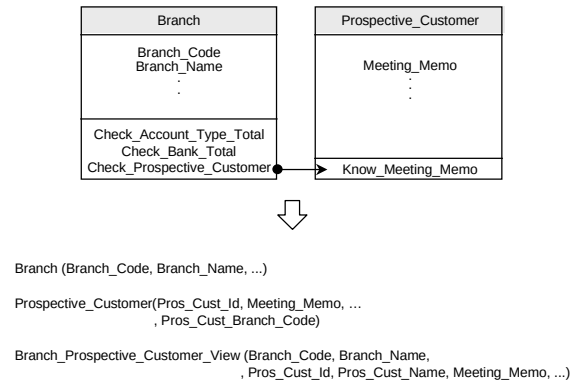

Figure 15. A Collaboration Relationship Transformation

First, each class maps directly to one table. Second, a generalization relationship is transformed into one

superclass table and multiple subclass tables. Third, many-to-many relationships map to distinct tables. One-to-one and one-to-many relationships may be mapped to distinct tables or merged with a participating class. In addition to these four rules, another transformation rule is considered for collaboration relationships of our CRCs. An additional view or stored procedure is required in case of the collaboration relationship. Collaboration relationship transformation to view is depicted in Figure 15.

Transforming CRCs in Figure 7 results in ten tables, as shown in Table 2.

Table 2. Relational Table

Table Name	Type	Attribute Name
Customer	Table	Cust_Id , Cust_Name, Cust_Addr , Cust_Credit
Current_Customer	Table	Cust_Id , Cur_Cust_Init_Date
Prospective_Customer	Table	Cust_Id , Meeting_Memo,Branch_Code
Branch	Table	Branch_Code , Branch_Name , Branch_Address
Account	Table	Acct_No , Acct_Type , Acct_Reg_Date , Acct_Balance , Cur_Cust_Id , Branch_Code
Transaction	Table	Tran_Code , Tran_Date , Tran_Type , Tran_Amount , Acct_No , Branch_Code ,
Branch_Prospective _Customer_View	View	Branch_Name , Branch_Code , Cust_Id , Pro_Cust_Name, Meeting_Memo
Branch_Account_View	View	Branch_Name ,Branch_Code ,Acct_No , Acct_Type , Acct_Balance
Account_Transaction_ View	View	Acct_No , Tran_Code , Tran_type , Tran_Amount
Current_Customer_View	View	Cust_Id , Cust_Name, Cust_Addr , Cust_Credit, Cur_Cust_Init_Date

3.6 Construction

In the construction phase, developers implement a physically running hypermedia application system in target environments. All of the products during the implementation design phase should be mapped to physical elements, as shown in Figure 16. A logical database schema generated in implementation design phase is transformed into physical database schema on target DBMS. User interfaces in running hypermedia are implemented by the use of user interface schema.

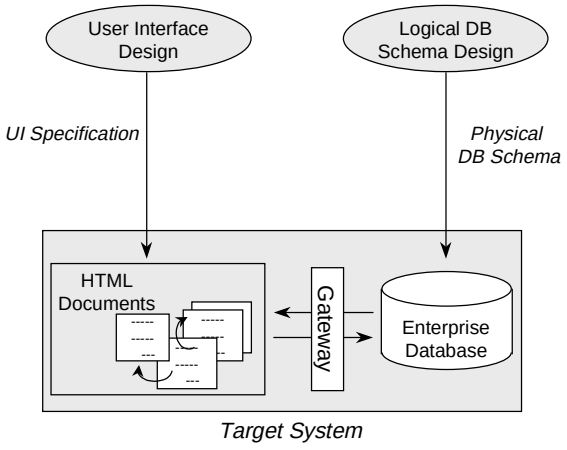

Figure 16. Construction Process of SOHDM

4. Prototype

A prototype is built by the use of Oracle RDMBS, Java, and HTML for the Bank Accounting System. The prototype has a TCP/IP LAN based client/server configuration as shown in Figure 17. Oracle Call Interface (OCI) is used as an interface gateway between Java and Oracle. The system is based on three-tier architecture, the first-tier for WWW clients, the second for the web server in Windows NT, and the third for Oracle database in UNIX.

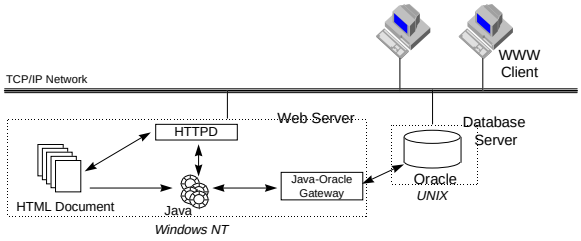

Figure 17. Target Hypermedia System Architecture

Figure 18 is the main menu screen. From the main menu icon (left side in Figure 18), "Customer Information System," one can navigate among submenus like "Current Customer," "Prospective Customer," "MIS," and "Service & Products." The main menu refers to Main_001 page schema in Figure 12.

Figure 19 is the HTML page for current customers. This page is implemented for the Figure 20 shows results from retrieving prospective customer information. Prospective customers to be interviewed are listed for each branch. This page screen is implemented for "Prospective_Customer _001" page schema. Clicking the "Branch" button and then selecting the branch name provides a new window. Prospective customers in the branch selected are shown in this window.

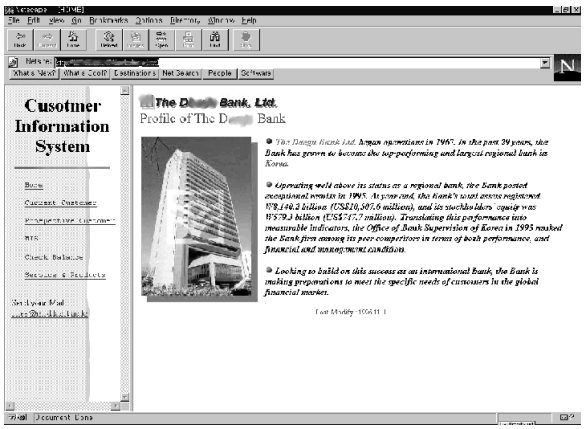

Figure 18. Main Menu Screen

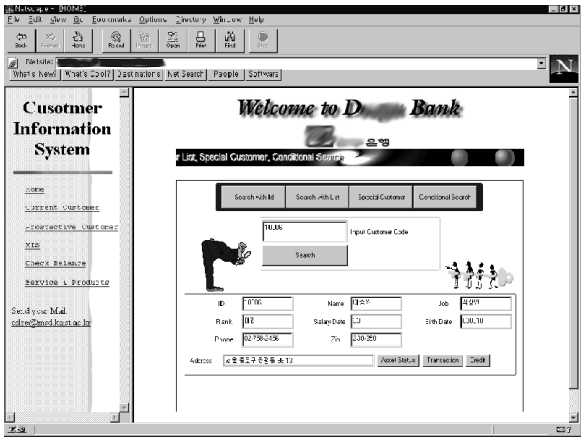

Figure 19. Current Customer Page Screen

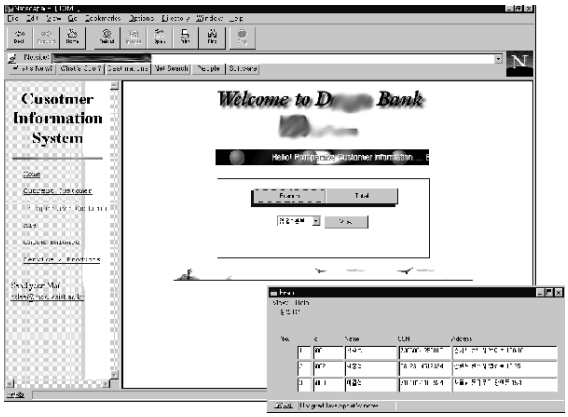

Figure 20. Prospective Customer Page Screen

Clicking the "MIS" anchor in the left side of prospective customer page screen results in the "MIS" page screen as shown in Figure 21. This page is implemented as "MIS_001" schema. It shows the deposit amounts (today and yesterday) by headquarters, or branches.

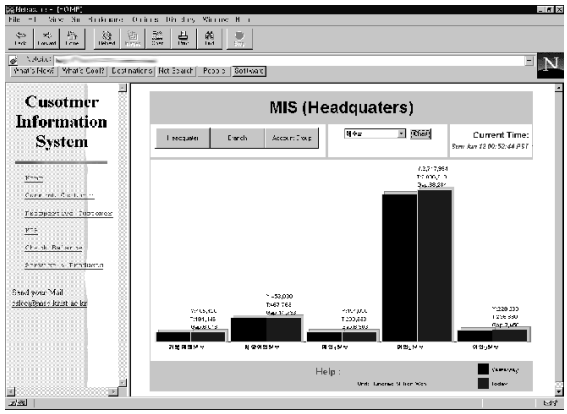

Figure 21. Customer MIS Page Screen

"Current_ Customer_001" page schema. It has a scrolled text field which highlights the current page, as well as four buttons -"Search with Id," "Search with List," "Special Customer," and "Conditional Search." The buttons act as anchors which lead users to embedded applications. If the "Search with Id" button is clicked, then the "Customer Search with Id"

Table 3. Comparison of Hypermedia Design Methodologies

Criteria \ Methodology	RMM (Isakowitz et al. 1995)	EORM (Langel. 1993)	OOHDM (Schwabe et al. 1995)	VHDM (Lee et al. 1995)	SOHDM
Key Modeling Technique	E-R	OO	OO	E-R	CRC Cards Scenarios
Phases	1. E-R Design 2. Slice Design 3. Navigational Design 4. Conversion Protocol Design 5. UI Screen Design 6. Run-time Behavior Design 7. Construction	1. Class Framework 2. Composition Framework 3. GUI Framework	1. Conceptual Design 2. Navigational Design 3. Abstract Interface Design 4. Implementation	1. Requirement Analysis 2. E-R Design 3. View Design 4. Navigational Design 5. Mapping 6. Implementation	1. Domain Analysis 2. OO Modeling 3. View Design 4. Navigational Design 5. Implementation Design 6. Construction
Documentation	E-R Diagram Slice Diagram RMDM Diagram	Class Structure GUI Design	OMT's OO Model Navigational Class Abstract Interface Design Model	E-R Schema View Schema Navigational Schema.	System Scope Diagram Event List Scenario Sets CRC Card Class Structure Diagram OO View Navigation Link Schema Page Schema UI Specification
Source of Navigation	E-R Relationship	OO Relationship	OO Relationship	E-R Relationship	Scenario and OO Views
Approach to Identifying Users' View	None	None	View	View	Scenario and OO Views
Semantic Richness	Relatively Poor	Relatively Rich	Relatively Rich	Relatively Poor	Relatively Rich

application is initiated. This application has an additional input field for search with customer Id. It has three buttons, "Asset," "Transaction," and "Credit" for additional information on the related customers.

5. Methodology Comparison

We compare features of SOHDM with those of four other major hypermedia design methodologies. This comparison is summarized in Table 3.

RMM and VHDM are based on E-R model and thus relatively simple to use. In contrast, SOHDM, as well as EORM and OOHDM adopt OO technologies to deal with rich semantics. Accommodating semantics may improve user satisfaction even with the price of analysis complexity.

In SOHDM, scenarios and CRCs are used for analyzing and modeling users' requirements. Specifying scenarios helps to capture ensure system flexibility from the earliest phase in the development process. In addition, the structure of CRC card promotes a behavioral approach to object modeling. The CRC card has an attractive informal appeal that helps make the emerging design tractable. SOHDM employs scenarios and OO views for finding navigational units. Concentrating on responsibility-driven scenarios and OO views is likely to capture navigational requirements better than static data entities.

6. Conclusions

This paper develops a scenario-based object-oriented methodology for developing hypermedia applications. To our best knowledge, our methodology is the first to use scenarios to capture hypermedia navigational requirements that are not shown in data relationship. The scenarios are transformed into object-oriented views. These views are used for designing hypermedia pages with navigational links. The use of scenarios is likely to improve the quality of hypermedia design to ensure flexibility from the earliest opportunity. The use of scenario as well as data model improve the quality of the navigation design.

The methodology is effective for integrating WWW hypermedia system with enterprise databases. A prototype is built to demonstrate the usefulness of the methodology.

The following research areas are further explored. First, a system for linking design phases in a flexible fashion is highly valued. It may reduce design time. Second, security and authority problems can be solved because Intranet hypermedia systems contain critical business applications and data. Third, a repository system may be useful to maintain and reuse the analysis and design outputs.

Acknowledgments

This research was partially supported by Wang Computer Korea Ltd. Research Grant GI90020.

References

[1] P. Balasubramanian, T. Isakowitz, and E. A. Stohr, "Designing hypermedia applications", Proceedings of the 28th Hawaii International Conference on System Sciences, 1994, pp.354-365.

[2] M. Bieber, and C. Kacmar, "Designing hypertext support for computational applications", Commun. ACM, Vo.38, No.8, August 1995, pp.99-107.

[3] M. R. Blaha, W. J. Premerlani, and J. E. Rumbaugh, "Relational database design using an object-oriented methodology", Commun. ACM, Vol.31, No.4, April 1988, pp.414-427.

[4] C. J. Date, An introduction to database systems, 6th edition, Addison-Wesley, 1995

[5] J. Fong, "Mapping extended entity relationship model to object modeling technique", SIGMOD Record, Vol.24, No.3, September 1995, pp.18-22.

[6] M. Frank, "Shifting gears", Internet Systems, Vol.1, 1996, pp.6-47.

[7] F. Garzotto, L. Mainetti, and P. Paolini, "Hypermedia design, analysis, and evaluation issues", Commun. ACM, Vol.38, No.8, August 1995, pp.74-86.

[8] F. Garzotto, P. Paolini, and D. Schwabe, "HDM: A model-based approach to hypertext application design", ACM Trans. Off. Info. Syst., Vol.11, No.1, January 1993, pp.1-26.

[9] F. Garzotto, P. Paolini, and L. Mainetti, "Navigational patterns in hypermedia databases", Proceedings of the 26th Hawaii International Conference on System Sciences, 1993, pp.370-379.

[10] J. Gosling, and H. McGilton, The Java Language Environment, White Paper, Sun Microsystems Computer Company, October 1995.

[11] F. G. Halasz, and M. Schwartz, "The Dexter hypertext reference model", Commun. ACM, Vol.37, No.2, February 1994, pp.30-39.

[12] A. T. F. Hutt, Object Analysis and Design - Description of Methods, A Wiley-QED Publication, John Wiley & Sons, Inc., 1994.

[13] T. Isakowitz, E. A. Stohr, and P. Balasubramanian, "Designing hypermedia applications", Proceedings of the 27th Hawaii International Conference on System Sciences, 1994, pp.354-365.

[14] T. Isakowitz, E. A. Stohr, and P. Balasubramanian, “RMM: A methodology for structured hypermedia design”, Commun. ACM, Vol.38, No.8, August 1995, pp.34-44.
[15] I. Jacobson, Object-Oriented Software Engineering - A Use Case Driven Approach, Addison-Wesley, 1995.
[16] W. Kim, Modern Database Systems - The Object Model, Interoperability, and Beyond, Addison Wesley, 1995.
[17] D. B. Lange, “An object-oriented design method for hypermedia information systems”, Proceedings of the 27th Hawaii International Conference on System Sciences, Maui, Hawaii, Vol.3, 1994, pp.336-375.
[18] D. B. Lange, “Enhanced relationships in object-oriented database modeling”, Proceedings of InfoScience ’93, Seoul, Korea, 1993, pp.296-304.
[19] D. B. Lange, “Object-oriented hypermodeling of hypermedia supported information systems”, Proceedings of the 26th Hawaii International Conference on System Sciences, 1993, pp.380-389.
[20] H. Lee, J. Kim, Y-G Kim, and S. H. Cho, “View based methodology for hypermedia applications”, Journal of Database Management (Forthcoming).
[21] ORACLE Co., Oracle 7 Server SQL Reference, Release 7.2, Oracle Corporation, April 1995.
[22] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, Object Oriented Modeling and Design, Prentice Hall, New York, 1991.
[23] D. Schwabe, and G. Rossi, “Building hypermedia applications as navigational views of information models”, Proceedings of the 28th Hawaii International Conference on System Sciences, 1995, pp.231-240.
[24] D. Schwabe, and G. Rossi, “The object-oriented hypermedia design model”, Commun. ACM, Vol. 38, No. 8, August 1995, pp.45-46.
[25] D. Schwabe, G. Rossi, and S. D. J. Barbosa, “Abstraction, composition, and layout definition mechanisms in OOHDM”, Electronic Proceedings of the ACM Workshop on Effective Abstractions in Multimedia, Sanfrancisco, California, 4 November 1995.
[26] A. Simon, Strategic Database Technology, Morgan Kumfmann Publishers, Inc., 1995.
[27] D. A. Taylor, Business Engineering with Object Technology, John Wiley & Sons, Inc., 1995.
[29] M. Thuring, J. Hannemann, and J. M. Haake, “Hypermedia and cognition: designing for comprehension”, Commun. ACM, Vol.38, No.8, August 1995, pp.57-66.
[30] R. Wirfs-Brock, B. Wilkerson, L. And Wiener, Designing Object-Oriented Software, Prentice Hall, Englewood Cliffs, NJ, 1990.
[31] E. Yourdon, Modern Structured Analysis, Prentice-Hall, 1989.