Organizational memory information systems have recently received considerable attention in information systems development. As yet little research has been reported on the methodology used for the analysis and design of organizational memory information systems. This paper describes an object-oriented method for modeling organizational memory information systems. Six concepts that support organizational memory information systems are discussed. They are: document, episode, cognizance, goal, mnemonic instrument, and integration. These concepts are harmonized into a unified object-oriented paradigm. The proposed modeling method extends the traditional information systems analysis into the context of organizational memory information systems. It is useful for the development of organizational memory information systems for organizational learning, surveillance, and decision making in general.

Organizational memory is the collective knowledge from the past which has influence on present activities within the organization (Walsh & Ungson, 1991). Organizational memory is vital for the organization's effectiveness and learning (Duncan & Weiss, 1979), integration of organizational actors, and lower transaction and turnover cost (Walsh & Ungson, 1991). Information technology supported organizational memory information systems (OMIS) are desirable to make organizational knowledge explicit, communicable, and integrated. (Stein & Zwass, 1995). The need to recognize and develop information systems for organizational memory is growing stronger. OMIS is becoming one of the most important issues in management information systems development because of the increasing weight of knowledge work and the capabilities of advanced information technology used by organizations.

The analysis and design of an enterprise’s OMIS are the tasks of information specialists in the organization with regard to the evolution of approaches to information systems design methodology (Sibley, 1986). In terms of modeling OMIS, a natural language is universal in the sense that it can describe anything that can be described. Yet natural languages have potential ambiguities. Natural language descriptions do not provide measures to map the structure of organizational memory to the information system world, especially, in the information system implementation stages. Conventional systems analysis methods, such as the data flow diagram method (DeMarco, 1978; Gane & Sarson, 1979), entity-relationship method (Chen, 1976) and the combination of these methods, emphasize descriptions of functional and data requirements within the context of information processing automation. In contrast, OMIS analysis must present many aspects of organizational memory, including knowledge integration and adaption, goal attainment, organizational pattern maintenance, and mnemonic functions (Stein & Zwass, 1995).

In recent years, the object-oriented approach has received attention from the computer and information systems communities (Wang, 1996a). A recent information industry
survey indicated that the adoption rate of object-oriented methods is increasing dramatically. Organizations that embraced the approach have experienced significant cost savings in the systems development area (Pei & Cutone, 1995). Research (e.g., Yoon & Guimaraes, 1992) indicated that the object-oriented approach is beneficial for developing knowledge-based expert systems for the organization. Nevertheless, little research into the integration of the object-oriented method with modeling OMIS can be found. As a result, there is a lack of integration of modeling OMIS with object-oriented information systems analysis. This lack of integration may cause difficulties in developing OMIS in practice for the specific applications and fitting OMIS into the enterprise information system as a whole. We contend that, if OMIS is to be constructed, choices must be made in approach and the corresponding structure.

Object-oriented Approach

The object-oriented approach (Martin, 1993; Rumbaugh et al. 1991) is still in the developmental stage. Using the object-oriented approach, analysts model the system being investigated by identifying a set of objects in conjunction with the attributes (i.e., data) and methods (i.e., internal operations and messages) that manipulate the object data or request services from other objects. Objects are grouped into classes which have common properties. Classes are organized into hierarchies in which the subclasses inherit properties, including data definitions and methods. The terms “object” and “class” are often interchangeable in systems modeling as few cases involve objects with specific actualized values. Interactions between objects are handled by means of message sending. The dynamic relationships between objects are built into the descriptions of the classes through the definition of message sending.

All of the characteristics of object-oriented methods make the approach more effective than the traditional data flow diagram method in information systems development (see (Korson & McGregor, 1990; McIntyre & Higgins, 1988; Heintz, 1991) for a detailed discussion). More importantly, the model can be implemented by a computer based information system using object-oriented programming without the requirement of a creative system logical design phase (Fichman & Kemerer, 1992). The methodology of object-oriented analysis is far from mature, although the object-oriented approach has been recognized for many years. A serious criticism of current development work is the piecemeal fashion of object-oriented systems development. This requires an investigation of an extension of the object-oriented approach into modeling OMIS.

There are a variety of tools and techniques for object-oriented analysis (see a recent survey in Eckert & Golder (1994). Coad-Yourdon’s method (Coad & Yourdon, 1991) was selected as a base for this study because of its simplicity. Two significant modifications were made to Coad-Yourdon’s method. First, cardinalities (0, 1, N) between classes were omitted. Cardinalities do not play an important role in modeling OMIS. The second modification was that data transmitted by messages between object classes were explicitly annotated. The concept of data flows passing in the object-oriented paradigm is quite different from data flows in the structured data flow diagram in that data flows must be associated with the messages between objects. The elements in object-oriented modeling are:

1. **Attributes**: Encapsulated data descriptions of the object class.
2. **Operations**: Processes that apply to the object class. There are two types of operations: method and message.
   1. (2.1) **Method**: An operation which manipulates the encapsulated data in the object.
   2. (2.2) **Message**: An operation procedure which requests service from another object(s). In the object-oriented paradigm, message sending from one object class to another makes dynamic connections between the object classes.
3. **Data flow**: A group of data elements that are associated with a message and outline the communication between the objects.
4. **Inheritance**: In a hierarchical relationship between object classes, subclasses inherit properties, including data definitions and operations, from their superclasses. Inheritance results in static connections between object classes. There are two types of inheritance relationships: assembly structure (has_a)

![Figure 1: The Diagram Representing Object Class, Inheritance and Message Sending](image-url)
and generalization structure (is_a).

These elements are shown in Figure 1. They provide a
generic instrument for object-oriented modeling. However, the
instrument does not offer much help to the systems modeler in identifying object classes. Actualizing object
classes is a task accomplished by the problem domain, but is
virtually the central issue in applications of object-oriented
techniques for a particular field.

The object-oriented approach is a powerful systems
modeling approach. However, without a domain analysis for
particular types of information systems, the object-oriented
approach remains a virtual philosophy, rather than a concrete
technique for systems development and software reuse. One
the other hand, effective OMIS must be supported by ad-
vanced information technology including the Internet, Intranet,
hypertext and multimedia. The use of object-oriented meth-
odology to model related components of OMIS is imperative.
A generic object structure for OMIS can help in general
aspects of systems development for the organization, espe-
cially when OMIS is incorporated into comprehensive informa-
sion systems analysis. This paper proposes a generic object
structure for OMIS based on a domain analysis of OMIS. The
domain analysis of OMIS identifies and formalizes funda-
mental objects and their relationships involved in OMIS
specification, especially intangible objects which are unique
to other types of information systems.

OMIS and Object-oriented Perspective

The difference between ordinary systems analysis and
modeling OMIS is significant. Ordinary systems analysis
addresses major functionalities and data requirements within
the organization. Its key concerns are data and processes. On
the other hand, OMIS operates to support the organization in
the integration of specialized knowledge into a common set of
tasks for the organization. Organizational knowledge proper-
ties are the major focus of modeling OMIS. Since OMIS and
ordinary information systems (i.e., data processing systems,
management report systems, and decision support systems)
normally overlap, the general components of ordinary informa-
sion systems such as structured hard data (databases),
models (model bases), and groupware for group decision
making can also serve as general components for OMIS
(Huber, 1990; Nunamaker et al., 1989). This study, however,
places its focal point on the unique knowledge components of
OMIS.

This endeavor for theoretical development of OMIS is
illustrated in Figure 2. As Figure 2 shows, this paper contrib-
utes to the step from Stein and Zwass's (SZ) conceptual
framework (1995) to the generic object-oriented structure for
OMIS. However, as will be seen later, this extension towards
the OMIS object-oriented structure is made based primarily
on domain analysis in the object-oriented paradigm and the
theories of organizational memory, but it does not pursue
SZ's propositions on the layers and subsystems because, in
our view, the partition of the layers and subsystems is func-
tion-oriented instead of object-oriented. To develop the
generic object-oriented framework for OMIS, the taxonomy
framework is radically restructured based on the theory of
object-oriented paradigm. The advantage of the object-ori-
ented framework over the taxonomy framework is that the
object-oriented framework provides an explicit logical schema
of OMIS, and can be tailored and actualized based on the
requirements of a specific OMIS, as illustrated in Figure 2.

According to the theory of organizational memory,
OMIS must be able to facilitate two basic types of informa-
tion: semantic and episodic (El Sawy et al., 1986). Semantic
information is general knowledge independent of specific
events. Handbooks, manuals, and standard operational proto-
cols are typical examples of semantic information. Episodic
information describes specific events, including event occu-
rence, context, and outcomes. This concept of organizational
memory is derived directly from human memory (Schacter,
1989; Tulving, 1983). However, organizational memory is an
instance of collective memory as a construct, rather than a
simple concept (Walsh & Ungson, 1991). This implies that
the content of organizational memory can range between
semantic and episodic contents. For instance, production
rules (IF-THEN rules) can be classified as episodic memory
if we emphasize the relationship between an event and its
outcome, or semantic memory if we stress the abstraction
feature of collective knowledge independent of specific events.
We refer to this type of information for organizational memory
as cognizance information. To avoid confusion, the word
"document" is used instead of "semantic" in our study,
because the term "semantic" is often overloaded and repre-
sents a variety of meanings in several fields.

In addition to documents, episodic memory, and cogni-
zance knowledge, an organization must possess goals which
assist actors of the organization in using these primary knowl-
edge for planning and control (Ariav, 1992). Goals are high
level knowledge abstracted from basic episodic, document
and cognizance memory. Goals influence all organizational
memory processes, and should be a component of organiza-
tional memory.

The attendant processes of memory need functional
support, including acquisition, search, reasoning and other
facilities, called mnemonic functions (Krippendorff, 1975).
From the object-oriented point of view, these mnemonic
functions are objects of knowledge handling instruments, and
constitute an independent component of organizational
memory.

The knowledge components of organizational memory
are shared by the entire organization. Specifically, docu-
ments, episodes, cognizance knowledge, goal information,
and mnemonic instruments are integrated in three dimen-
sions: time, space, and medium. The integration apparatus of
OMIS is therefore an imperative component of OMIS to
ensure that knowledge in the organizational memory is made explicit and is available for the users in the organization (Huber, 1990; Scott Morton, 1991). As shown later in this paper, the object-oriented approach made it possible to merge the integration functions of OMIS with the knowledge components and mnemonic instruments in a harmonious way.

In summary, OMIS analysis and design must describe six concepts of organizational memory which are unique to other ordinary information systems. They are document, episode, cognizance, goal, mnemonic instrument, and time-space integration. The set of the six concepts is a counterpart of SZ's conceptual framework, but provides an object-oriented view of the OMIS domain. Next, we describe each of these concepts in more detail, and harmonize these descriptions into a unified object-oriented paradigm. Given the fact that OMIS is a relatively new field, we have difficulty in finding a comprehensive practical OMIS case around us at this point. Due to this limitation, most of the examples (see Figure 3) for explanations of these concepts are rather hypothetical. The objective set for the use of the examples is to provide a general object structure for OMIS.

Document Descriptions

Two characteristics of object specifications make the representation of a document memory structure unique to other systems specification approaches. First, since a document is an object, it possesses its own operations such as retrieval and migration from a document to its subdocument. Second, a document object can send messages to trigger other objects and it can also be triggered by other objects. The dynamic feature of the object-oriented approach is especially effective in the hypertext and hypermedia environment by making the representation of a document structure more flexible than the traditional static method of designing the organizational memory. The document object structure and
an example of document object INCOME_STATEMENT are shown in Figure 3(a). As illustrated in the figure, a document can inherit its properties (e.g., descriptions and retrieve methods) from its parent document, and a document can have its subdocuments. For instance, an income statement can have its reference documents of legal expenses, stored in the microfiche form.

**Episode Descriptions**

OMIS are oriented toward the support of human cognitive activities. Research (e.g., Schank, 1990) has strongly suggested that human memory is story-based. Therefore, the core competence of an organization derives from episodic memory. In fact, case-based reasoning (CBR) is an approach to problem solving based on the retrieval and adaptation of cases, or episodic descriptions of problems and their associated solutions (Allen, 1994).

In the operation part, an episodic object is assumed to have a generic processor for retrieval. The operation part of each episodic object can also contain a set of messages for the creation or playback of multimedia objects. An object-oriented frame for episode descriptions and an example of episode object are shown in Figure 3(b). As illustrated in the figure, an episode can inherit its properties (e.g., descriptions and retrieve methods) from its parent episode, and an episode can have its sub-episodes. For instance, an important executive meeting can contain a specific meeting episode recorded on a video tape.

**Cognizance Descriptions**

Cognizance concepts are powerful appliances to symbolize abstract knowledge in all functional departments of the organization. For instance, human resource management of the firm requires soft knowledge about personal patterns from the organizational memory (Stein & Zwass, 1995). Personal patterns pertain to attitudes (satisfaction, equity, level of stress), culture (languages, symbols), values, norms, personal routines, capabilities, skills, aspirations, and so on. These descriptions of personal patterns can be stored in a frame. In fact, abstract knowledge is the foundation of the capability of executive information systems, supported by data warehousing (Inmon et al., 1997).

Figure 3(c) illustrates object-oriented representations of cognizance concepts, and an example of cognitive map CAUSES_OF_ORGANIZATIONAL_STRUCTURE as a cognizance object. In this example, a CEO has his/her own cognitive map regarding the relationship between organizational structure and influence factors such as group size and formalization. These factors can be described further for the object of the CEO's cognitive map. On the other hand, employees also have their own cognitive maps. For the organization, it might be useful to keep these two types of cognitive maps and examine the common characteristics as well as the differences between them.

**Goal Descriptions**

A central part of OMIS is the provision of goal descriptions for the organization. Technically, goal descriptions could be classified as a specific type of cognizance information as discussed in the last section. However, since the goal attainment function is vital for organizational actors in planning and control (Ariav, 1992), a goal is considered as an independent concept of organizational memory.

Modeling OMIS should be based upon principles that have increased the utility of system specifications. Therefore, if object-oriented system specifications are applied, a goal or a subgoal is an object, and a goal structure is a type of assembly structure. Since a goal (or subgoal) is an object, it possesses its own attributes and operations. Attributes of a goal object include goal states, annotated histories, and strategies to achieve the goal. Technically, the strategies of a goal are general statements which can be used as a guideline for the design of the operations of the goal object. Operations of a goal object include knowledge apprehension related to the goal and evaluation of progress in the direction of goal state. Knowledge apprehension retains the step-by-step behavior of the user of the OMIS in reaching the goal, and thus is a set of operators to access the organization memory. A general description of goal object and an example of a goal EFFECTIVE_BUDGETING is shown in Figure 3(d).

In the example, suppose the organization has a goal of effective budgeting. The goal must have its measurements and evaluation methods specified by the object attributes and operations. This goal in turn has its subgoals, say, satisfying financial constraints and satisfying departmental needs. To support the organization in achieving these subgoals, the OMIS must be able to trigger other objects in the organizational memory (e.g., documents in this case).

**Mnemonic Instrument Descriptions**

Some generic mnemonic functions, such as retention and retrieval of information, are encapsulated in individual objects. Nevertheless, OMIS needs several types of specific task-oriented mnemonic functions, such as knowledge acquisition (Ellofson & Konsynski, 1991) and case matching (Kolodner & Mark, 1992), which are independent of the knowledge objects. In contemporary information systems, these mnemonic functions are implemented in software packages, called mnemonic instruments. A mnemonic instrument is usually comprised of a user interface, tools and a repository.

The object-oriented description of mnemonic instruments and an example of mnemonic instrument FUZZY_SEARCH_ENGINE are shown in Figure 3(e). Re-
Figure 3: Document, Episode, Cognizance, Goal, and Mnemonic Instrument Objects
In the object-oriented context, the concept of integration is not an independent object, but a synthesis of the five categories of descriptions based on the contingency of knowledge sharing in the organization. If the first step of modeling an OMIS is to identify the five independent categories of object-oriented descriptions (document, episode, cognizance, goals, and mnemonic instrument), then the second step of the modeling is to synthesize the five categories of descriptions and implement the integration.

Synthesis of the five descriptions can be formalized into a structured procedure as follows.

1. A synthesis process starts with goal descriptions. Each of the operators of knowledge apprehension initiates messages to other types of objects.
2. Document, episode, and/or cognizance objects are retrieved in response to the messages sent by the goals.
3. Mnemonic instrument objects are then evoked by goal objects to process document, episode, and/or cognizance information required by knowledge apprehension.
4. Upon the completion of knowledge apprehension processes of a goal, the goal object updates the goal state. When the goal is finally accomplished, it sends a message to its super-goal to further evoke knowledge apprehension processes of the super-goal.
5. In principle, the synthesis process is successfully completed if the top goal of the OMIS is accomplished, otherwise, the descriptions for the OMIS need to be redefined.

Following these steps, the OMIS modeler can integrate the knowledge components to fulfill the requirements (knowledge apprehension) of the user of the organizational memory. Before an example of integration processes is demonstrated, knowledge integration in the three dimensions is further discussed as follows.

(1) **Temporal integration.** In the traditional data base
systems, there often is a lack of explicit recognition of temporal information integration. This is due in part to the difficulty in dealing with the dimension of time using the conventional data modeling methods. The object-oriented paradigm, on the other hand, provides a natural vision of the temporal property of information. Each of the stored document, episode, and cognizance objects has its temporal identification as an attribute value. The messages sent by the goals to these objects carry data flows which specify the requirements for temporal information retention. The responses to these messages actualize the temporal information integration for the organizational memory.

(2) Spatial integration. In contemporary information systems, information spatial integration across the organization as well as its environment is essentially the client-server issue. Detailed discussions on modeling client-server computing in the object-oriented paradigm are beyond the scope of the present study, but can be found in Wang (1996b). For our purposes, we simplify the structure of distributed processing by assuming that each of the stored document, episode, and cognizance objects has its spatial identification (access address) as an attribute value. The messages sent by the goals to these objects carry data flows which specify the access address for spatial information retrieval. The responses to these messages actualize the spatial information integration for the organizational memory.

(3) Hypermedia integration. Hypermedia systems allow the intermediate users (who design an application for basic end users) to organize knowledge in a desired network form. The (basic end) user of the network can choose his/her own path to move from node to node. The nodes can contain text and multimedia presentation including graphics, images, sound, and full-motion video. In fact, a hypermedia system is an object-oriented database software environment. The organization of the network by the intermediate user is a hypermedia integration process, and would better be modeled in the object-oriented paradigm.

An example with a small scale showing a synthesis process for modeling an OMIS is given in Figure 4. It shows an object model of a small portion of an information system designed for organizational learning pertaining to employee motivation that has been adapted from a large courier company case study (Palvia et al., 1992). This example is used merely to demonstrate the principles of OMIS modeling discussed above, but not for discussion of an OMIS for organizational employee motivation, which is a topic independent of this study.

The system facilitates the execution of an annual survey. The survey consists of a questionnaire related to the company and the supervisor of the employee. The employee

![Figure 4: An Example of the Integration Process in Modeling OMIS](image-url)
can also write letters to express concerns. The survey results provide time-trend data on the company’s overall performance as viewed by the employees. The managers and supervisors can review the survey results and hold a feedback meeting(s) with employees to discuss the employee responses and to develop an action plan to address the issues raised by the employees. The goal of the system is to provide information for the improvement of managers and supervisors’ performance in dealing with employee concerns. As shown in Figure 4, the document object in this case is a standard set of questionnaires. The episode objects are questionnaire survey data, original written letters in microfiche form, and feedback meetings in video form. The cognizance object is the abstract knowledge of industrial relation indices, leadership indices, and an overall index viewed by the employees based on the survey data. The mnemonic instrument is a tool which can generate a performance appraisal for the supervisor based on the survey data. In this simple case, the goal tree is a single goal object which specifies its knowledge apprehension procedures, including Review_Questionnaire, Review_Survey_Results (up to 5 past years), View_Meeting (up to last 5 meetings), View_Microfiche (up to 5 past folios), Appraise, Review_Indices, etc.

The synthesis is started with the knowledge apprehension of the goal object. Message 1, which is sent by Review_Questionnaire to the QUESTIONNAIRE document, represents the request for document review. Message 2.1, 2.2, and 4 from Review_Survey_Results, View_Meeting, and View_Microfiche, respectively, indicate episodic information retrieval in the hypermedia environment. When the user reviews the survey results, he/she can move from survey to microfiche and meeting video. These activities are represented by Message 2.1 and 2.2. Similarly, Message 5 symbolizes the interaction between the knowledge apprehension and cognizance object INDICES of a company’s performance. Message 6 requests the mnemonic instrument to assess the supervisor’s performance based on the survey data. Message 6.1 indicates the linkages between the mnemonic instrument and episode objects in this case. Note that these messages accentuate the dynamic relationships between the objects, but do not represent particular access paths.

As shown in this example, the synthesis of integration of the five categories of object descriptions actually builds the associations between objects by identifying the messages between them. It is also an elaboration process to check the completeness of the components for the organizational memory to accomplish a goal. In this sense, the modeling of a particular OMIS is an iteration of describing the five types of objects and then synthesizing them for integration. This modeling method is called object-oriented OMIS analysis.

Summary

In our view, conventional expert systems are part of organizational memory, and expert system models can fit the general object-oriented framework. For instance, the four major components of organizational knowledge-based systems suggested in Yoon and Guimaraes (1992) can be modeled in the present genuine object-oriented framework. Specifically, a knowledge base management system is a set of mnemonic instrument objects (i.e., inference engines and/or other intelligent software agents), corporate knowledge base is a set of cognizance objects for production rules and/or episode objects for cases, knowledge-based system bank and knowledge dictionary are mnemonic instrument objects (i.e., specific inference engines and search engines).

The proposed modeling method employs formal (scholastic) object-oriented modeling symbols. It might be practical to put this model into an information management perspective. Technically, the formal object-oriented modeling symbols can be substituted with desirable icons (or clip-art images) for presentational purposes. The object structure depicted in icons can imitate the organizational memory in a way favored by the CIO of the organization.

This paper recognizes the problem of a lack of concrete modeling methods for OMIS. Given the fact that the object-oriented systems development approach is superior in terms of its ability to describe a variety of entities and their relationships in the complex organizational memory space, the object-oriented approach should be considered in facilitating modeling OMIS.

The proposed object-oriented OMIS structure has many advantages. First, it employs a simple object-oriented modeling tool and facilitates communication in the course of OMIS development. Secondly, the technique integrates the six aspects (document, episode, goal, cognizance, mnemonic tools, and integration) of OMIS into a single object-oriented diagram. It provides measures to describe organizational knowledge in a static way, and describes the dynamic supporting properties of OMIS by defining message connections between the object classes. Thirdly, and more important, by the object-oriented diagram can be specified in great detail. These detailed modeling diagrams can be used for various purposes including system planning, system analysis and implementation, as well as user training and system evaluation. Thus, this would make it possible to integrate OMIS modeling with computer-aided systems engineering.

Acknowledgments

This research was supported by a grant from Social Sciences and Humanities Research Council of Canada. The comments of anonymous reviewers have contributed significantly to the revision of the paper.

References


Ariav, G. (1992). Information systems for managerial plan-


Related Content

A Lag Effect of IT Investment on Firm Performance
www.irma-international.org/article/lag-effect-investment-firm-performance/1285/

Cultural Effects on Trust Building in International Projects' Stakeholders
www.irma-international.org/article/cultural-effects-on-trust-building-in-international-projects-stakeholders/128208/

Borderline Fields of Information Architecture: Information Overload, the Literacies, and Personal Information Management
www.irma-international.org/chapter/borderline-fields-of-information-architecture/135764/

Challenges in Building a Culture-Centric Web Site
www.irma-international.org/chapter/challenges-building-culture-centric-web/23042/

A Conceptual Approach to Improve the Continuity and Integrity of Information in Project Management
www.irma-international.org/article/a-conceptual-approach-to-improve-the-continuity-and-integrity-of-information-in-project-management/154971/