Extending an Object-Oriented Data Model for Representing Multimedia Database Applications: A Position Paper

Kingsley Nwosu
AT&T Bell Laboratories
67 Whippany Road
Whippany, NJ

Bhavani Thuraisingham
The MITRE Corporation
202 Burlington Road
Bedford, MA 01730

This position paper describes the issues involved in extending an object-oriented data model for Multimedia data management applications.

1. INTRODUCTION

A multimedia information system provides for the efficient storage and manipulation of data represented as text, images, voice, graphics and video. In addition, users can also periodically update the multimedia database so that information contained in it accurately reflects the real-world. Some multimedia systems are being extended to provide the capability of linking the various types of data in order to enable its users to have access to large amounts of related information within a short space of time either by browsing or querying the system. It has been found that such systems are useful for variety of applications including C4I, CAD/CAM, and air traffic control. Although multimedia information systems have received much attention recently, methodologies for designing applications which utilize such systems have received little attention. Some have proposed object-oriented database management systems (OODBMS) for storing and managing multimedia data as they have been found to be more suitable for handling large objects and multimedia data such as sound and video consume considerable storage space (see for example [WOEL86]). Although such systems show some promise, they are not sufficient to capture all of the requirements of multimedia applications. For example, in many cases, voice and video data which may be stored in objects have to be synchronized when displayed. The constraints for synchronization are not specified in the object models. In addition, the semantics of multimedia applications (e.g. content and context modeling) need to be captured.

Designing multimedia information systems applications involves (i) analyzing the requirements of the application, (ii) designing the database, the integrity constraints to be enforced, and the transactions, and (iii) designing the modules of the automated system (which automates the slice of the real world of interest). The automated system may utilize a database management system to manage the database. Since much of the focus to date has been on using OODBMSs for managing multimedia data without paying much attention to the unique requirements of multimedia applications, a tremendous burden will be placed on the multimedia application designer when utilizing such a system for the application. It is, therefore, imperative that tools be developed to aid the designer in analyzing the requirements, designing the database, the integrity constraints, the transactions, and designing the modules of the automated system.

An approach for the multimedia information system application design is as follows. The client, whose goal is to automate some slice of the real world of interest to him (such as a command and control application which involves multimedia data), defines the problem and subsequently the requirements engineer generates the requirements. This initial specification of the application is then given to the application designer who designs the application. The design process could consist of multiple stages. For example, the first stage could be to identify the entities of the application and the relationships between them, the next stage could be to represent the entities using some notation, the subsequent stages could include analyzing the representation for potential inconsistencies, designing the multimedia database, and designing the modules of the automated system. Throughout the design process the application designer, the requirements engineer, and the client may interact with each other until a satisfactory design is obtained.

In designing a multimedia information system, first of all, the entities and the relationships between them must be represented using an appropriate model unambiguously. It is also important to capture the semantics as accurately and completely as possible. The requirements must then be analyzed in order to detect possible inconsistencies. Therefore, a design tool should utilize the specification of the application, and subsequently conduct an analysis. The designer should be informed of any potential problems. In addition, a design tool must also consider the operational requirements. The tool must analyze the various operational scenarios and determine whether there could be potential inconsistencies. In order to successfully design such a tool, we believe that a powerful modeling and reasoning methodology is needed. This paper describes how the OMT (object modeling technique) methodology developed by Rumbaugh et al [RUMB91] could be utilized for this purpose.

We chose Rumbaugh et al's OMT Methodology because it was developed specifically for modeling and reasoning about complex applications for information systems. OMT is becoming one of the more popular methodologies for designing various database applications. It is a software engineering methodology and encompasses three viewpoints: the object model, the dynamic model, and the functional model. As stated in [RUMB91], the object model describes the static structure of the objects in an application and their relationships. The dynamic model describes the aspects of the application that change over time. It is used to specify and implement the control aspects. The functional model describes the data value transformation within an application. OMT consists of an analysis phase during which the application requirements are analyzed, a system design phase during which the database and the system processes are generated, and an object design phase during which the algorithms and the interfaces are generated. Note that it is not always necessary to construct all three models for a particular application. For example, if only the database is to be designed, then the object model would suffice. Furthermore, OMT separates the object, dynamic, and functional models. One could develop an approach where two or more of the models could be combined into one.

This paper focusses mainly on applying the analysis phase of OMT for multimedia applications. In particular, the application of the object model, the dynamic model, and the functional model of OMT are discussed. It also briefly discusses some of the essential points of the other phases of OMT. The organization of this paper is as follows. A brief overview of multimedia application requirements is given in section 2. Section 3 describes the application of the object model, dynamic model, and the functional model of OMT. Further considerations are discussed in section 4.

The paper concludes in section 5 with a discussion of future work. It is assumed that the reader is familiar with the essential points in object-oriented data models. For a discussion we refer to [RUMB91]. Some information on multimedia information systems and applications is given in [WOEL86]. A more detailed version of this position paper is given in [NWOS94].

2. MULTIMEDIA APPLICATION REQUIREMENTS

There are several requirements imposed on multimedia information systems applications. These requirements can be grouped into various categories including the data representation requirements and the data manipulation requirements. Data representation requirements may include support for generalization/specialization hierarchy, attribute specification, specification of operations, support for composite objects, multimedia data sharing, versioning, constraints for presentation (such as synchronization constraints), and support for extensible data types. Data manipulation requirements may include efficient data retrieval and update, concurrency control and recovery, efficient data storage and access (such as indexing different data types), support for the capture, storage, and presentation of various types of multimedia data, and schema and version management.

The requirements are specified in terms of the particular application. For example, a command and control application may include a description of the aircraft, commanders, missions, and administrative information such as policies and procedures, and the activities that pertain to a particular mission. Information on a mission may include the mission name, type of the mission, and possibly a video script describing the details of the mission. Information about a commander may include data such as his qualifications, number of years of experience, etc. The application will also include a description of the activities that could occur. These include a commander reserving an aircraft to execute a mission or a mission in progress.

Since the information to be represented in multimedia applications is complex, a powerful methodology is needed to capture the semantics. Furthermore, the model should also support reasoning capabilities, as much of the information may be derived information. Maintaining the integrity of the data as well as the operations is critical. For example, information about a mission must be accurate. Otherwise the consequences could be catastrophic.

Recently various extensions to object models have been proposed in the literature to represent multimedia applications. (See for example, [GIBB94], [IEEE93]). The intent of this position paper is not only to propose extensions to OMT's object model, but also focus on the complete design process of multimedia data applications.

3. APPLICATION OF OMT

In this section we discuss the use of the object model, dynamic model, and functional models of OMT for designing multimedia information systems applications.

3.1 THE OBJECT MODEL

The object model captures the static aspects of the application. That is, it represents all of the entities of the application, and the relationships between them at a single moment in time. The object-oriented data model of OMT has the inherent capability of modeling the real world of interest naturally. It also provides an

intuitive graphical representational scheme which an application designer could use to communicate with the client. We use the term entities to represent not only the objects of the object model, but also the classes, associations, relationships, links, events, and activities. The concepts of the object model include objects, classes, attributes, operations and methods, links and associations, composite objects, class hierarchy and inheritance, metadata and constraints. We only discuss the essential aspects in this paper. Note that there is not standards object model. Therefore, one has to take care of defining the various constructs like generalization and aggregation.

As in most object models, objects with similar properties are grouped into a class. The object instances of the class inherit these properties. For example, a group of aircraft could be represented by an AIRCRAFT class while a group of missions could be represented by a MISSION class. Missions PPP and QQQ could be instances of the MISSION class while aircraft AAA and BBB could be instances of the AIRCRAFT class.

Modeling classes and objects is less interesting if one cannot model the properties of objects. Properties are specified as part of the class definition. These properties are called attributes. Each class has ID as an attribute. This is the identifier of an instance of the class. For example, MISSION class may have attributes ID, name, mission-plan (whose value may be a video script) etc. An operation is a transformation that is applied to the objects of a class. The implementation of an operation for a class is via a method. In most object models, the objects interact with each other by exchanging messages which will result in executing appropriate methods.

(This paper will discuss operations and methods again as part of the functional model.) Note that there are some outstanding issues here. For example, should a method be associated only with a class or can a method be associated with a collection of classes (i.e. a collection)?

Links connect object instances such as an aircraft AAA takes part in mission PPP. Links may also be used to connect the multimedia data. For example, the textual description of a mission plan and the video description of the mission plan could be connected through a link. Associations describe groups of links such as an AIRCRAFT takes part in a MISSION. In this model, the associations are between classes while links are between instances. With this view, associations could have attributes. For example, the association between AIRCRAFT and MISSION could have attributes 'frequency' (for how often the aircraft takes part in a mission). Associations and links do not have to be limited to binary ones. They could be n-ary (n >= 3).

Generalization is the relationship between a class and refined versions of the class. These refined versions are called subclasses. The concept of generalization results in a class hierarchy also referred to as the IS-A hierarchy. A class may have multiple subclasses associated with it. If C1 is a subclass of C2, then C2 becomes a superclass of C1. In general, a class may also have multiple superclasses associated with it (although in some models it is limited to just one). For example, a subclass of MISSION could be FRIENDLY-MISSION. In addition to the attributes and methods of MISSION, FRIENDLY-MISSION might have additional ones. Inheritance is the mechanism by which subclasses take properties of their superclasses.

Constraints include those which specify exceptional conditions, application-independent integrity constraints, and application specific integrity constraints on the entities (such as objects, links, etc.). Application-independent integrity constraints include constraints such as an object must have a unique identifier. Application-dependent integrity constraints include constraints which synchro-

nize the display of different types of multimedia data. For example, video frame XXX should be displayed with voice frame YYY or video frame AAA must follow voice frame BBB.

3.2 DYNAMIC MODEL

The aspects of an application that deal with time and changes are represented by the dynamic model. That is, the changes to the objects and their relationships are captured by this model. The sequence of operations that must respond to events form the basis of this model. An event is an individual stimulus from one object to another. It results in a set of activities. An example of an event is an aircraft being hit by a bomb. Another use of the dynamic model is to analyze the activities and determine whether there could be any potential inconsistencies.

Since the dynamic model represents the events, the first question that must be answered is whether all objects can cause events to occur. Since some objects such as commanders and missions are active and other objects such as mission plans and aircraft are passive, OMT has grouped the objects into two categories: active objects (also called actors) and passive objects. The active objects may stimulate each other or stimulate passive objects causing some changes to occur. Passive objects do not stimulate any other objects. In developing a dynamic model, the first step is to determine which of the objects are active and which of the objects are passive. Determining whether an object is passive or active may not be straightforward. For example, a mission itself may be an active object while the mission-plan may be a passive object.

Once the various types of objects have been identified, then the interactions between objects need to be determined. The active objects in the object model may stimulate other objects (active or passive) and consequently may cause changes in the states of the objects being stimulated. Note that a state of an object is described by its attribute values and links. A stimulus from one object to another is an event. Events could occur concurrently or an event could follow another event. For example, the event

'commander John is watching the video script of mission-plan AAA' could occur at the same time as the event 'commander Paul is executing the mission BBB'. The event 'commander John is watching the video script of mission-plan AAA' must follow the event 'commander John is executing the mission AAA'. OMT uses objects and classes to represent events also. Once the various types of objects and events have been identified, an event analysis has to be carried out for each scenario. It is during the event analysis that potential inconsistencies are detected.

Other considerations with the dynamic model include concurrency of events and atomicity of events. Concurrent events are not uncommon. For example, a commander may view a video script while dictating notes. In certain cases, all of the events in a set of events should occur or none of the events should occur. That is, the atomicity property of the set of events must be preserved. For example, the events (i) commander John is watching the video script of mission-plan AAA and (ii) commander John is executing the mission AAA could be atomic events.

3.3 FUNCTIONAL MODEL

The functional model generates the methods (i.e. the functions) of a class. It does this by examining the object model, the dynamic model, and subsequently performing a dataflow analysis. A dataflow connects the output of an object or process to the input of another object or process. Dataflow diagrams represent the active

objects that drive the application, the datastores which are the passive objects, and the various arrows between the active objects, datastores, and requests. In generating the functions, issues such as when or how do the functions get executed are not taken into consideration. What is generated are the functions.

An example of dataflow analysis is as follows. Suppose in the application under consideration, various mission plans are documented in films and a film editor wishes to edit a film from a collection of films. The datastores could be EDITORS and FILMS. The editor retrieves a particular film from the collection and he then requests to edit a specific portion of the film. Then the index file maintained for the film may be updated appropriately (note that while issues on index files have been considered mainly for relational implementations, there is much research on developing appropriate index strategies and access methods for object-oriented multimedia data management systems). The active object in this example is the editor who initiates the operations. The methods that could result from this simple application are SELECT and UPDATE. The SELECT method would retrieve the specific film from FILMS. The UPDATE method would modify the 'index-file' attribute of the film (note that there are some outstanding issues such as precise specifications for dataflow which need further consideration).

4. THE NEXT STEP

As can be seen, the three models are related to one another. Each describes some aspect of the application utilized by another. The object model describes data structures. The dynamic model describes the control structures of the objects and specifies the activities. The functional model generates the functions to be executed on the objects. The three models of OMT will be used in the analysis phase of the application design. The output of this phase will be the objects, classes, and the various types of associations, a description of the events and the changes to the states of the objects, and the functions which would implement the activities. At this time, the designer will be alerted to the potential inconsistencies with the application which he would rectify or limit them possibly by consulting with the client. The analysis phase would be carried out independent of the implementation. For example, for a multimedia information system application, the analysis phase is not concerned whether the database is relational or object-oriented and also whether the system is general purpose or needs to be specially constructed. The question is, is it necessary to construct all three models for the design of a specific application? This would depend on what the client wants. If only the database and the constraints are to be designed then the object model would suffice. If the system is to be designed also, then the dynamic aspects of the application need to be analyzed. If in addition, the transactions are to be generated, then the functional model comes into play. Therefore, the level of detail depends on what the client expects from the application designer.

Once the analysis phase is completed, what is the next step? The subsequent phases are the system and object design phases. During system design, the focus on what needs to be done is shifted to how it is done. For example, in the case of a multimedia information system application, the steps in the system design phase would include:

(i) mapping the object model into a multimedia database,

(ii) determining the integrity constraints to be enforced,

(iii) mapping the dynamic and functional models into transactions,

(iv) designing the modules of the automated system.

While the analysis phase determines what the implementation must do, and the system design phase determines how it should be done, the details are determined by the object design phase. This will include algorithms and interface specifications for the implementation. It is during this phase, that the implementation details as to the specific DBMS to be selected and other packages to be used are determined. Rumbaugh et al. [RUMB91] have developed various tools (for example OMTool) for designing certain applications. Such tools need to be adapted for multimedia information systems applications.

5. SUMMARY

This paper has provided an overview of the design process of multimedia information system applications, discussed the issues in such applications, and described how OMT could be used for this purpose. The ideas presented in this paper are preliminary and much remains to be done on this topic. First, we need to identify the useful constructs of OMT and develop an approach to represent and reason about multimedia information systems applications. We also need to identify the various types of constraints that may be present in a realistic environment and investigate ways of enforcing them. Then the ideas have to be tested with a real-world application. Finally, the detailed design and implementation of the tool need to be carried out.

REFERENCES

[GIBB94] Gibbs, S., et al, "Data Modeling of Time-Based Media," Proceedings of the ACM SIGMOD Conference, Minneapolis, MN, May 1994.

[IEEE93] IEEE Transactions on Knowledge and Data Engineering, Special Issue in Multimedia Information Systems, August 1993 (Editors: B. Berra et al).

[NWOS94] Nwosu, K., and B. Thuraisingham, Applying OMT for Designing Multimedia Database Applications, Proceedings of the IEEE Dual Technology Conference, Utica, NY, 1994.

[RUMB91] Rumbaugh, J. et al, "Object-Oriented Modeling and Design," 1991, Prentice Hall, Englewood Cliffs, NJ.

[WOEL86] Woelk, D., and W. Kim, 1986, "An Object-Oriented Approach to Multimedia Database Systems," Proceedings of the ACM SIGMOD Conference, Washington, D.C.