34

Multiware Database: A Distributed Object Database System for Multimedia Support

C. M. Tobar[a] and I. L. M. Ricarte[b]

[a]Institute of Informatics, Pontificial Catholic University of Campinas (PUCCAMP), Brazil. e-mail: tobar@dca.fee.unicamp.br

[b]Dept. of Computer Engineering and Industrial Automation, State University of Campinas (UNICAMP), Brazil. e-mail: ricarte@dca.fee.unicamp.br

This paper describes the Multiware Database, a database capable to store and manage complex multimedia documents using the features of both object database management systems and open distributed systems. This multimedia database is a component of the Multiware project[1], a platform supporting distributed multimedia cooperative applications. Representation of multimedia documents in this system is normalized, in the sense that distinct aspects of documents, such as structural and exhibition aspects, are stored as distinct objects in the database. This normalization enables concise representations and integration of distinct multimedia standards within the same framework.

Keyword Codes: H.5.1; H.2.8; C.2.4
Keywords: Multimedia Information Systems; Database Applications; Distributed Systems.

1. INTRODUCTION

Due to recent technological developments, the concept of a "computer-based document" has evolved sensibly since the days dominated by word processing. Computer documents incorporating sounds and images (still or animated) are already a reality, and uncountable applications can benefit with the utilization of this type of multimedia data. The manipulation of multimedia documents and their related activities, such as authoring and exhibition, involves large amounts of data, which should be persistent and efficiently retrieved. In order to support these requirements, some type of storage server should be provided. Ideally, a multimedia database management system would provide these capabilities, along with the standard features of database systems, such as data independency (thus supporting data sharing) and consistency control. However, there is no satisfactory implementation of such a system. Furthermore, access and manipulation requirements are even more severe when the application is intrinsically distributed.

This paper describes one approach to the implementation of such a distributed multimedia database, the Multiware Database, on top of an object-oriented database management system (ODBMS for short). One of the key points in the design of Multiware Database is that it should seamlessly be integrated with an open distributed platform, and

document representations should comply to existing or proposed multimedia standards.

The development of Multiware Database was motivated by the Multiware project [10]. Multiware is a platform to support multimedia cooperative applications incorporating ideas from the Reference Model for Open Distributed Processing (RM-ODP) and from existing industrial specifications such as the Common Object Request Broker Architecture (CORBA). Therefore, one of the major concerns on the design of this multimedia database system is to incorporate (not reimplement) services already provided by Multiware, whenever the platform service can improve efficiency of the final applications.

The Multiware platform project aims the development of a middleware level of software and hardware to allow the manipulation of multimedia information. Multiware's goal is to support the new generation of applications, which requires distributed heterogeneous computational environments, such as cooperative work (CSCW), decentralized artificial intelligence (DAI), and decision making support systems.

Multiware provides three functional layers, namely Basic Software and Hardware, Middleware, and Groupware (Figure 1). The strategy behind Multiware is to use, as much as possible, related available standards, specifications, and products, favoring the adoption of object-oriented technology. From a functional point of view, the Basic Software and Hardware layer provides the functions supported by operating systems and communication protocols, among others. Distributed processing facilities are provided by the Middleware layer, which aggregates ODP functionality to existing commercial distributed systems and supports functions specific to real-time multimedia processing, in special the multimedia database management system. The Groupware layer provides framework and components that support group work based on multimedia.

Implementation of multimedia information systems, and more specifically of multimedia databases, is a relatively new field. One important aspect is the integration of standards related to the many forms of representation of generic and multimedia data, since standards will be responsible to a more widespread use of multimedia support systems and object database systems.

Previous work on this field points toward the use of an ODBMS as the best way to support the basic requirements of storage of multimedia data and to describe and represent multimedia information [12]. Other related topics of interest to multimedia databases are user interaction, retrieval strategies, and standards, which are addressed in the following reviewed work.

Grosky [4] provides an overview on Multimedia Information Systems. He proposes a "generic architecture" for such systems, which would be logically composed by three interrelated repositories of data, namely a standard database (with non-multimedia data), a multimedia database (with uninterpreted multimedia objects), and a feature database (with information that could potentially help the retrieval of multimedia data). Although it is not required that these components are implemented as distinct databases, this architecture shows concern with the fact that current systems hardly can provide efficient manipulation of all types of data. However, the proposed architecture does not address aspects of distribution of such systems.

Bulterman [1] analyses multimedia document behavior during user interaction on a distributed multimedia application, and presents an infrastructure to support requests of real-time dynamic presentations of multimedia adaptable documents; that is, the appli-
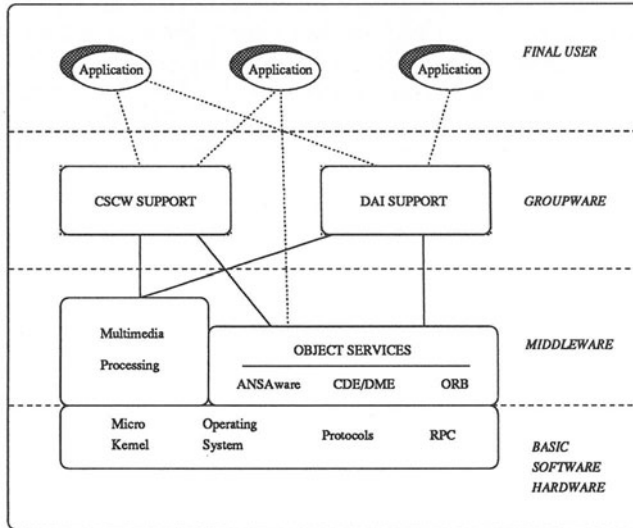
Figure 1: Multiware Platform.

cation can adapt to the available resources. Data transfer problems are not addressed by this infrastructure. The aspect of multimedia adaptation results in the reduction of cost related to authoring multimedia applications. The Multiware Database proposal presents a similar adaptation mechanism.

Koegel *et al.* [8] developed a distributed multimedia information system using a standard. This system constitutes one of the first efforts to validate the operational model of a multimedia standard, using an ODBMS to store document descriptions. Its three-layered architecture presents information of hypermedia content and structure, object instances, and application objects and their attributes. The authors intend to develop generic multimedia presentation services, avoiding reimplementation of the application layer for each new document type definition, as proposed for the Multiware Database.

Klas, Löhr, and Neuhold [7] present a tutorial on multimedia and its application requirements. Some multimedia standards are considered and integrated to a distributed and open extensible ODBMS system. They address some important features required by multimedia applications and not supported by object-oriented systems, such as time-dependent data support, user interaction mechanisms, and content-based query and retrieval techniques. Their multimedia database includes a synchronization manager, an interaction manager, and a continuous object manager. It is interesting to notice that in the Multiware Database design these functions are provided by isolated external servers.

The remaining of this paper presents the Multiware Database proposal, whose main goal is to provide multimedia storage and (subsequent) retrieval support for Multiware

distributed applications. Synchronization and user interaction (related to information exhibition), information preparation (for transport and exhibition), and information transport are necessary multimedia services, not directly supported by Multiware Database but for which there must be some correspondent data inside the multimedia database, in order to facilitate all related multimedia activities, from capture through management. Multiware Database provides this support in a distributed heterogeneous environment.

The organization of the paper follows. Section 2 describes the Multiware Database, addressing its constitution in terms of manipulation and storage units. Section 3 discusses object services supported (or expected to be supported) by open distributed systems, and introduces a set of multimedia services which are provided by the Multiware Database. Section 4 presents the technological structure of Multiware Database and discusses multimedia services localization. Section 5 presents the functional structure of Multiware Database, along with its implementation based on the object-oriented approach. Finally, Section 6 presents conclusions and future work.

2. THE MULTIWARE DATABASE COMPOSITION

The Multiware Database provides multimedia information storage and retrieval services to the applications, considering their need of timeliness and feasible manipulation mechanisms. The basic structures of the Multiware Database are based on the object-oriented approach, and they aim the achievement of this goal. The description of these structures is the subject of this Section.

2.1. Multiware Manipulation Unit

From the user or application point of view, the Multiware Database stores and retrieves multimedia documents (here denoted MMDoc), considered to be the unit of manipulation for multimedia information.

In the Multiware context, MMDocs can present different classifications according to their *structure* (simple or complex), *organization* (hypermedia, simple-media, or multiple-media), *behavior* (static or dynamic — interactive, when exhibited), *continuity* (with or without temporal synchronization requirements), or *influence* (with or without modification of information, respectively active or inactive, during or after exhibition). All these perspectives are orthogonal and are considered in the Multiware Database design. Thus, it is possible to store a multiple-media complex MMDoc, which during its exhibition will be dynamic and active and will need synchronization. User interaction can exist during active and inactive MMDoc exhibitions, requiring information updating in the database.

From the point of view of object representation, an MMDoc is composed by two parts, one *multimedia object* and its (possibly many) *presentation forms* (Figure 2).

A multimedia object contains all the user information, which is to be stored, retrieved and exhibited. This information can be represented by several data pieces of different media types, and requires structural and, optionally, continuity (temporal) meta-data in order to relate the user data pieces.

A presentation form describes exhibition features, which are related to available hardware, security, and user selectivity possibilities, described in details in Subsection 2.2.

The representation structure of an MMDoc reflects the fact that it has information content and description, which are independent of presentation features. On the other
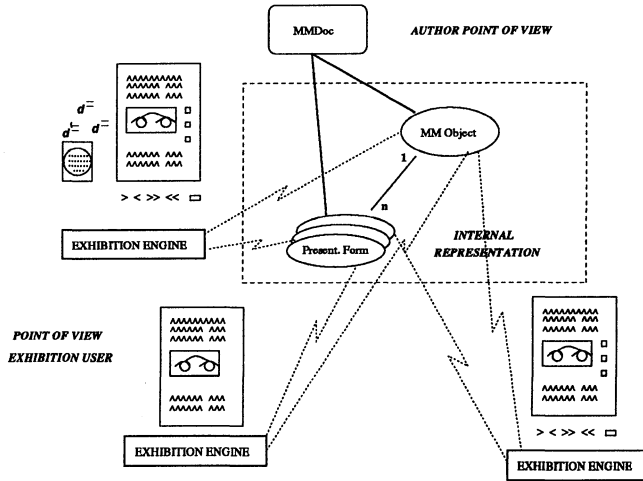
Figure 2: MMDoc Composition: it reflects the distinct multimedia capabilities of presentation terminals; in this example, interactive audio and video (top left), still images only (bottom left), and interactive video only (bottom right).

hand, presentation relates to exhibition aspects, which depend on available resources.

2.2. Multiware Storage Entities

The parts of an MMDoc described above yield the two storage entities of the Multiware Database, which are the multimedia object (denoted MMObj) and the presentation form.

An MMObj stores the (exhibition-independent) description of an MMDoc. It is composed by single media objects (such as an image, a text, or a sound), by hypermedia objects (such as hypertext, text linked with images or sound), and/or other composite multimedia objects. It also stores meta-information related to the multimedia document, such as author and creation date.

An MMObj should be able to represent the composition of single media objects and hypermedia objects. Thus, it can be recursively defined as follows:

> **a single media object** is an MMObj;
> **an hypermedia object or nested MMObj** is an MMObj; and
> **a composition of MMObj's** along with information describing the composition and relating the MMObj's, is also an MMObj.

A single MMObj is a piece of information created through one type of media representation and encoding format. For example, it could represent either an ASCII text or an MPEG item of video and audio.

One nested MMObj represents a kind of "hyper-MMObj", through which it is possible to navigate via links and to reach other simple or nested MMObj.

Each MMObj, besides the fact of being a component of other MMObj, has associated information, other than its structural information, related with intra-object synchronization properties, exhibition aspects of temporal and spatial synchronization, behavior, influence, identification and control. With this information it is possible to process the multimedia data, by an exhibition engine external to the Multiware Database, in order to *(i)* obtain proper synchronization between components, *(ii)* allow user interaction, and *(iii)* modify the information for later exhibitions.

The recursive nature of an MMObj, along with the disjoint information of its structure, provides two interesting forms of reuse for authoring purposes: reuse of description definition and reuse of multimedia pieces.

The information of one specific MMDoc is associated to a given description (that is, to one MMObj) and can be exhibited in multiple different forms. The reason for these different forms might be distinct hardware features, security-related aspects, or user selectivity. The characteristics of each possible exhibition format are stored and represented through a multimedia presentation form.

Presentation forms represent the information required to allow preparation, communication, and exhibition of an MMDoc on different distributed exhibition nodes. They include information about available hardware, location, quality (color, exhibition speed, resolution), completeness (partial or complete exhibition of each component of an MMDoc), fault resolution (delay, loss, acceleration), compression format, user selective choices, and so on. With this information and the description of the MMObj, it is possible to derive terminal-specific presentations of MMDocs, as further described in Section 5.

3. MULTIMEDIA DATABASE-RELATED SERVICES

Distributed processing comprises the class of information processing activities in which discrete components may be located at more than one location, and consequently requires explicit communication among components [6]. Heterogeneous distributed systems impose requirements of interoperability between components of different vendors, and if they present scalability characteristics, they are considered open systems.

One option to obtain distribution openess is through the adoption of a client-server-like architecture, where clients communicates to servers in order to obtain *services*.

The Object Management Group (OMG) effort has produced a terminology, used throughout this document, for object-oriented distributed services. The OMG aims the specification of standards to achieve interoperability, through a set of object services which provides database-like facilities for distributed objects. Database facilities usually provided by relational or object-oriented database systems have similar counterparts among OMG services, which cover most of the necessary services for multimedia manipulation.

Among OMG services there is a *brokering service* which is the key mechanism for the integration of other services and platforms, regardless the heterogeneities introduced by open distributed system architectures. The brokering server, known as Object Request Broker (ORB), is the main component of the OMG proposed architecture, and provides behavior invocation through a registration mechanism for location and naming.

This Section presents a brief description of services required by multimedia applications and not covered by current OMG services. The relationship between the complete set of services and the activities related to multimedia applications is then described.

3.1. Multimedia New Necessary Services

Multimedia manipulation requires specialized services not covered by the current OMG proposal, such as a specialized *retrieval service*, although the OMG query and archive services provide some retrieval functionality. Therefore, a specific retrieval service is considered from now on. *Wrap-up, transport, presentation*, and *performance monitoring* are four other necessary services for multimedia manipulation, not present in OMG specifications. The motivation for these multimedia services is described next.

The *Retrieval Service* is related (as far as possible) with content (semantic), structural, behavioral, and influence retrieval, along with whole or partial MMDoc description and content reuse.

The *Wrap-up Service* is responsible by the creation of multimedia objects in the necessary format for exhibition. Its main concerns are to generate adequate information for distributed transport and to filter unnecessary information related to presentation details, as discussed in Section 2.

The *Transport Service* provides efficient and reliable (as necessary) moving of multimedia objects. These objects, once in an exhibition node, are processed by presentation engines, which could demand further multimedia data transport, related to specific media types which composes the MMDoc to be exhibited.

The *Presentation Service* relates to aspects of processing of transported information in order to produce planned stimuli (upon creation) to a human being, *i.e.*, an exhibition. Synchronization, interaction, influence, security, and direct access are some of the issues related to this service.

The *Performance Monitoring Service* controls (fine tunes) and manages multimedia related aspects in the distributed environment through the generation of a log of performance events, such transmission initial time, final time, and interactive rate.

3.2. Multimedia Activities and Services

Among the activities related to the manipulation of multimedia information there are *capture, authoring, storage, communication, exhibition*, and *management*. Each of these activities requires support by a distinct set of services, as follows:

Capture: data interchange (conversion, compression), lifecycle (object instances), and archive.

Authoring: lifecycle, security, relationships, transactions, concurrency control, query, retrieval, archive and change management.

Storage: archive, lifecycle, security, relationships, transactions, interaction, concurrency control, change management, and replication.

Communication: retrieval, security, wrap-up (formatting, compression, etc.), and transport.

Exhibition: presentation (synchronization, interaction), security, and externalization.

Management: operational control, performance monitoring, startup, backup and restore, query, licensing, and security.

Trading, event notification, persistence, installation and activation, threads, time, and naming services belong to a set of basic support services necessary to all other services.

4. MULTIWARE DATABASE TECHNOLOGICAL STRUCTURE

The Multiware Database proposal, described in Section 2, presents an alternative architecture for multimedia database systems which does address distribution aspects. Therefore, it is important to analyze which services are provided by this platform in order to integrate seamlessly the database to Multiware. Several of the necessary services for multimedia manipulation are available in a ODBMS or can constitute autonomous specialized servers, available in an open architecture and accessible via an ORB.

In contrast to the ODBMS architecture, the ORB architecture enables each multimedia service to provide a separate piece of database technology. This allows the possibility of customized and expandable systems, with their components supplied by different vendors. A multimedia database system can adopt one or a combination of both architectures, if the combined approach is chosen, it will present several autonomous servers, interconnected through ORBs, some of which are distributed nodes of an ODBMS.

The Multiware Database has as framework one distributed ODBMS, several specialized media repository servers, and several autonomous multimedia object-related servers.

The ODBMS component of Multiware Database acts as a gateway to multimedia information, actually stored in specialized repository servers. The adoption of a distributed architectural organization for the Multiware Database permits the expansion of potential applications for multimedia services [9]. Two distinctive architectures are appropriate for this scenario, namely the master-slave and the federated organizations.

The ODBMS supporting the Multiware Database is a federated database server, that communicates via the brokering service (ORB). Each federated node presents a master-slave architecture between its object-oriented database manager and its related specialized media servers, as presented in Section 5.

In order to obtain a flexible and efficient access mechanism to MMDocs there is an Object Database Adaptor (ODA), through which the ODBMS is integrated to the ORB architecture [2]. ODA provides facilities to register with the ORB a set of stored objects identifiers and to allow access (direct access if desirable) to stored objects, as if they had been individually registered.

Depending upon the available functionalities of the integrated ODBMS, it is possible to locate each multimedia service either to the database system or to the platform specialized servers. There are three aspects to be considered in order to determine the location of each specific multimedia service:

> **broadness:** does the multimedia service apply only to the ODBMS or to several different clients?
> **actuation domain:** does the multimedia service act only upon objects stored in the ODBMS or upon other external objects?
> **efficiency:** which service does present better performance (CPU, I/O, and memory), the one in the ODBMS or the autonomous one?

Table 1 presents a cross-reference between the several multimedia services (previously referenced and identified in Subsection 3.1) and multimedia manipulation activities (introduced in Subsection 3.2) in order to present an analysis of the ideal service location, with the following notation: (A) capture; (B) authoring; (C) storage; (D) communication;

(E) exhibition; and (F) management. The *Relevance* column classifies each service taking into account the proposed aspects for allocating the service, and it uses the following notation: (1) server to multiple clients; (2) exclusive ODBMS server; (3) applicable to general objects; (4) applicable only to ODBMS objects; (5) critical efficiency; and (6) non critical efficiency.

OMG Service	(A)	(B)	(C)	(D)	(E)	(F)	Relevance
archive	X	X	X				(1) (3) (6)
backup/restore						X	(2) (4) (6)
change management		X	X				(2) (4) (6)
concurrency control		X	X				(2) (4) (6)
data interchange	X						(1) (3) (5)
externalization					X		(1) (3) (5)
licensing						X	(1) (3) (6)
lifecycle	X	X	X				(1) (3) (5)
operational control						X	(1) (3) (5)
performance monitoring						X	(1) (3) (5)
presentation			X	X			(1) (4) (5)
query		X				X	(2) (4) (5)
relationships		X	X				(2) (4) (5)
replication			X				(2) (4) (5)
retrieval		X		X			(2) (4) (5)
security		X	X	X	X	X	(1) (3) (6)
startup						X	(1) (3) (6)
transactions		X	X				(1) (3) (5)
transport				X			(1) (4) (5)
wrap-up				X			(1) (4) (5)

Table 1: Cross-reference between multimedia services and activities.

The results of the location analysis can be summarized as follows: on one hand, the services of archive, lifecycle, licensing, operational control, data interchange, externalization, security, startup and transaction services are well suited to be provided by specialized autonomous servers, without the interference of any ODBMS. On the other hand, backup and restore, change management, query, concurrency control, retrieval, relationships and replication services are better supported by the ODBMS.

Specialized servers can offer wrap-up, presentation, and transport services, though applied specifically to multimedia objects, because there is no restriction imposing that all multimedia objects should be stored in the ODBMS.

5. MULTIWARE DATABASE FUNCTIONAL STRUCTURE

Two distinct types of objects represent an MMDoc inside the Multiware Database, namely *information objects* and *presentation objects*, which are related respectively to the multimedia object and the presentation form introduced in Subsection 2.1.

A third type, an *exhibition object*, created dynamically during the retrieval of one MMDoc for exhibition, is the combination of data from one information object and one presentation object. It is the multimedia communication unit.

One information object can be associated to many presentation objects. It contains, besides content information, all the information concerning structure, synchronization, behavior, influence, identification, and control.

The structure of an information object can contain pointers to other information objects, simple media objects or hypermedia objects, allowing the definition of recursive multimedia data structures. Except by information objects, these objects are stored in distributed and specialized media repository servers, outside the scope of the ODBMS.

The Multiware Database provides basic object types to represent simple media objects and basic hypermedia objects, along with their associated basic operations. Through the ODBMS constructs, the user is able to define new types.

The way an MMDoc is represented produces a layered structural view of the Multiware Database, presented in Figure 3.
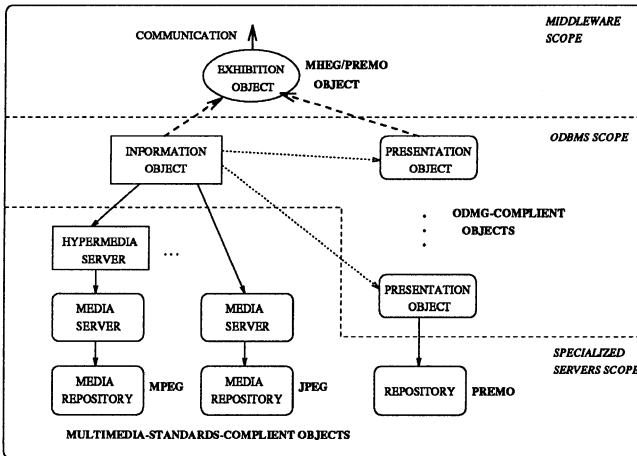

Figure 3: Multiware Database layered structure. The exhibition object is generated dynamically, not being permanently stored in the database scope. The distinct media repositories are also outside the scope of the object database.

One important aspect of multimedia information storage is that it should not be restricted to a central repository, since each type of media can be better supported by specialized servers. Thereafter, retrieval of a multimedia document involves retrieval operations from many individual distributed sources.

The open distributed architecture of Multiware presents potential heterogeneity problems between servers, platforms, and data formats. The adoption of standards mitigates the last problem, and also eases the integration with hardware and software products in order to support multimedia activities.

Some specialized media repository servers store hypermedia objects, which in turn have links to other simple or hypermedia objects. Each specialized media repository

can be implemented according to a specific related media representation encoding format (standard or standard to be), such as MPEG, JPEG, G711, and MIDI.

One exhibition object is composed by all the information necessary for its exhibition, *i.e.*, components localization, synchronization, and specific presentation information. It is self-contained, being able to be processed by a PREMO engine, which in turn could emulate an MHEG engine.

Document-oriented standards of interest are MHEG and HyTime, both related to multimedia and hypermedia documents, and PREMO, related to presentation of MMDocs.

MHEG (Multimedia and Hypermedia information coding Expert Group) [3] defines an interchange format for multimedia and hypermedia objects as well as a specification of how to process the data being interchanged. HyTime (Hypermedia/Time-Based Structuring Language) [11], on the other hand, is strong in mechanisms for the synchronization of media data, specifying the structure of an MMDoc through an extension of the Standard Generalized Markup Language (SGML). PREMO [5] is a standard concerned with presentation aspects of a multimedia document, specifying mechanisms to represent and control input and output interactions.

The adoption of object models by these standards eases the integration of the particular aspects addressed by each standard, and provides a natural framework for their representation in a multimedia database system.

6. CONCLUSIONS

In this work, the architecture of Multiware Database is described. Multiware Database is the multimedia storage server system of Multiware distributed processing platform. The motivation for this work is to provide a multimedia database to support the storage of multimedia information, which should become available to applications using Multiware.

Since Multiware adheres to RM-ODP and CORBA, questions were raised regarding the ideal allocation of services related to multimedia activities; some of these services could be supported by both a specialized component of the platform and by an object database system. A framework to define allocation strategies was presented in Section 4.

Besides the service allocation discussion, another important aspect of Multiware Database is the role of multimedia-related standards, mainly PREMO and MHEG. Internally, objects are structured following the OMG object database standard, ODMG-93 [2].

The current version of Multiware Database focuses on one node of the federated ODBMS, storage and communication activities aspects, inactive MMDoc's, presentation and exhibition objects adherents to PREMO and MHEG standards (to be), information objects adherents to ODMG-93 standard.

Future versions will integrate the distributed servers and the manipulation of active MMDoc's according to the following strategy. During the second phase, support for the remaining multimedia manipulation activities (capture, exhibition, management, and authoring) will be added. There are many expected design dificulties to afford the support of the authoring activity, such as long transactions, incomplete objects, and so on. The third phase will support the federated ODBMS and implement basic interaction functions for dynamic MMDoc's. The final and fourth phase will incorporate the full features and facilities described in this document.

References

[1] Dick C. A. Bulterman. Specification and support of adaptable networked multimedia. Published in Multimedia Systems, 1993. ftp: ftp.cwi.nl.

[2] R. G. G. Cattell, editor. *The Object Database Standard: ODMG-93.* Morgan Kauffman, 1994.

[3] Françoise Colaitis. Opening up multimedia object exchange with MHEG. *IEEE Multimedia*, 1(2):80–84, Summer 1994.

[4] William I. Grosky. Multimedia information systems. *IEEE Multimedia*, 1(1):12–24, Spring 1994.

[5] I. Herman, S. Carson, J. Davy, P. J. W. ten Hagen, D. A. Duce, W. T. Hewitt, K. Kansy, B. J. Lurvey, R. Puk, G. J. Reynolds, and H. Stenzel. PREMO: An ISO standard for a presentation environment for multimedia objects, October 1994. Submmited to ACM Multimedia'94 Conference.

[6] ISO/IEC JTC 1/SC 21. Basic reference model of ODP - part 2: Descriptive model. Working Draft N 7524, April 1993. 18 pages.

[7] Wolfgang Klas, Michael Löhr, and Erich J. Neuhold. Multimedia database systems. In *Proceedings of the Workshop on New Database Research Challenges*, pages 93–125, Rio de Janeiro, Brazil, September 1994. PUC-RJ/VLDB.

[8] J. Koegel, L. Rutledge, J. Rutledge, and C. Keskin. HyOctane: A HyTime engine for an MMIS. In *Proceedings ACM Multimedia*, August 1993.

[9] T. D. C. Little and A. Ghafoor. Spatio-temporal composition of distributed multimedia objects for value-added networks. *IEEE Computer*, pages 42–50, October 1991.

[10] W. P. D. C. Loyolla, E. R. M. Madeira, M. J. Mendes, E. Cardozo, and M. F. Magalhães. Multiware platform: an open distributed environment for multimedia cooperative applications. In *IEEE COMPSAC*, 1994.

[11] S. R. Newcomb, N. A. Kipp, and V. T. Newcomb. The HyTime, hypermedia/time-based document structuring language. *Communications of the ACM*, 34(11):67–83, November 1991.

[12] Darrell Woelk, Won Kim, and Willis Luther. An object-oriented approach to multimedia databases. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pages 311–325, 1986.