Requirements for MMDBMS

— A Report —

Paul Pazandak [*]
Jaideep Srivastava
{pazandak|srivasta@cs.umn.edu}
Distributed Multimedia Center
University Of Minnesota

25 March 1996

ABSTRACT

This report describes metrics to be used to determine the suitability of object database management systems to support multimedia data storage and use. Support at all levels of hardware and software are discussed, as even the most ideal database software cannot operate independent of operating systems, networks, and hardware. A review of the multimedia support provided by current object database management systems is also included.

1 Introduction

In the past, general database management systems (DBMS) have typically managed simple data types such as strings and integers. Simple record structures were sufficient to represent the data being managed. More complex data requiring user-defined data types such as engineering designs, and software configuration management have been better addressed by object database management systems (ODBMS), which are based upon the object-oriented data model. One of the current trends is the use of DBMS for the management of multimedia data, particularly as software, networks, and computers are better able to handle audio and video data requirements. Certainly, multimedia data has been stored in DBMS since the 1980's, but with severely limited support. ODBMS have generally become the database management system of choice for multimedia data (vs. relational RDBMS),

[*]This work was funded by the National Institute of Standards and Technology (NIST).

since the mechanisms provided by ODBMS better support multimedia.[38] However, at this time, flat file storage is probably the most prevalent (reasons for this will be discussed later).

This paper describes metrics to be used to understand the capabilities that will be required of ODBMS that will provide true multimedia data management features (these metrics are summarized in the appendix). Other papers have described support requirements for multimedia databases, e.g.,[25,38] and although detailed in many respects, they describe a relaxed view of future systems. Conversely,[21] provides an excellent discussion of requirements for visual information management systems, agreeing more closely with this report regarding the dramatic changes that will be necessary to adequately support multimedia data.

In the future we will see distinctions between those ODBMS which provide simple multimedia support for multimedia data repositories, and a next generation of ODBMS that provide support for sophisticated distributed multi-user interactive and collaborative multimedia environments – called multimedia database management systems (MMDBMS). A number of projects have focused on multimedia data issues over the past fifteen years, a few of these include.[7,8,10,12,35,39,43,49]

In the next three sections we describe multimedia data types, uses of multimedia data, and examples of multimedia applications to set the stage for understanding how these uses will affect the design of current object database management systems. We then discuss general ramifications, followed by specific ramifications of managing multimedia data. These ramifications are far reaching, and not restricted to software. The support of multimedia data will require changes to ODBMS software, as well as operating systems, computer hardware and networks. So, although we focus on software metrics, we must also discuss hurdles of other system components to satisfy the demands placed on them through the use of multimedia data.

2 Multimedia Data Types

To understand the requirements that will need to be satisfied by ODBMS, we need to know the types of multimedia data that may be stored and managed. The following is a list of the most common multimedia (or media) data types:

- **Text.** Large amounts of structured text in the form of books, for example, which are composed of parts, chapters, sections, subsections, and paragraphs.

- **Graphics.** Graphics include examples like drawings and illustrations that are encoded using high-level descriptions (e.g., CGM, PICT, Postscript). This kind of data can be stored in a structured way within a database, and the content which exists as meta-data such as line, circle, and arc, can easily be queried (e.g., find me all graphics that contain a circle). Of course, finding shapes that are not composed of simple types (lines, circles, etc.), e.g. a chair, is not so easily accomplished.

- **Images.** Images include examples such as pictures and photographs. Their encoding is defined by standard formats such as BITMAP, JPEG, and MPEG. The storage representation of images is basically a direct translation of the image, pixel by pixel, so there is no concept of a line, arc or circle. Some formats then compress the representation to reduce the size of the resulting data (e.g. JPEG, MPEG). Since images cannot be described by basic components such as lines, finding such objects or more complex objects within an image is non-trivial.

- **Animation.** An animation is a temporal sequence of image or graphic data. It specifies an order in which the set of graphics or images should be rendered. The images or graphics are independently constructed, and then organized. Unlike simple image data which can be retrieved and viewed for any length of time, animation has a temporal viewing constraint requiring each image or graphic to be displayed and then subsequently replaced by the next image/graphic. The constraint may vary by animation (e.g., it may be two images per second or thirty images per second).

- **Video.** Video is a set of temporally sequenced photographic data. The data used represents a recording of a real-life event produced by a device such as a digital video recorder. The data is divided into units called frames. Each frame contains a single photographic image. In most cases, video is recorded at 24 to 30 frames per second (fps). Temporal viewing constraints are generally dictated by the recorded frame rates for optimal viewing (for human consumption).

- **Structured Audio.** Like animation, this data represents a sequence of independent components, having a temporal requirement. Each component is representable using a description, such as note, tone, and duration. The temporal listening constraints can vary and are generally defined at creation time, or inherent to the component descriptions (e.g. 1/8th notes).

- **Audio.** Audio data is a set of sequenced data which is generated from an aural recording. Audio data has a temporal listening constraint dictated by the sampling rate of the recording device for optimal playback for human consumption. The basic units of audio data are called samples.

- **Composite Types.** Composite multimedia data is created by combining data in the form of basic multimedia data types, and other composite multimedia data. Types can be physically mixed together to form a new type, or logically mixed. A physical mix results in a new storage format, where the data such as audio and video, is intermixed. A logical mix defines a new data type while retaining individual data types and storage formats. For example, a new type AV or audio-video would be composed of two distinct parts; however, when played the executing methods would have to deliver the data in a synchronized fashion making it appear as though the data is a composition. Composite data may also contain additional control information describing how the information should be rendered at the client.

- **Presentations.** Presentations are complex composite objects that also describe orchestrations of multimedia data for the purpose of data modification and presentation. The orchestrations may describe a simple temporal ordering, such as playing video v1, then video v2, and so on; or, they may be much more complex, specifying how user, system, and application interaction will determine the resulting presentation.[37]

3 Types of Applications

There are several uses for an object database management system that manages multimedia data. The requirements of the application (and the kind of multimedia data it will use) can be used to determine which features need to be supported within ODBMS. In fact, for some applications, current ODBMS can be used with little or no modification. For other applications, no ODBMS exist today (or for that matter operating systems, or computer hardware) that provide the necessary features.

Here we provide some typical applications that may use an ODBMS to manage multimedia data.

3.1 Data Repositories

The purpose of data repositories is to provide simple database management support such as security and data backup. There is no need to understand the data formats being stored as the repositories do not operate on the data. Transaction support is possible, however updates require replacement of the entire object. Since the objects are stored as BLObs – or binary large objects – they exist as single "simple" objects. Queries may be formulated involving meta-data, and other data in the repository, but generally not against the multimedia data. In addition, any temporal constraints inherent to the data such as video, are not understood by the repositories. The data is simply sent to the clients which have applications to handle the data appropriately. Here are a few examples:

- **Pseudo repository.** A pseudo-repository contains multimedia meta-data, such as the names, lengths, encodings used, descriptions and keywords of videos. Surrogate values stored within the database describe the pathname to the multimedia objects, which are stored as simple files within a local or network-accessible file system. The ODBMS has limited control of the files since they reside outside of the repository.
- **Simple repository.** A repository can provide restricted access to data, and also a central location from which data can be backed up. Some applications may simply want a central managed storage facility to store and retrieve multimedia data. The data is managed by the DBMS, and may be stored on local disks, or on tertiary storage devices such as optical disks. Applications retrieve the multimedia data, use it locally, and then return it to storage.
- **Electronic mail.** Electronic mail may include sending of multimedia data. The mail system may use a repository to store the multimedia data, or the data may simply originate from a repository. In either case, the repository acts as a server, merely sending the mail to the client when requested. To read the mail, the

client must have an application that understands the format of the multimedia data (e.g. MIME).

- **Engineering Designs.** Again, for the purpose of security and perhaps configuration management, engineering drawings and solid models may be stored within a repository. Any operations, such as modification, are performed on the client using software that understands the data.

- **Healthcare Information Systems.** For archival purposes, patient data such as x-rays, and doctors' annotations may be stored within repositories.

3.2 Intelligent Data Management

Rather than simply using an ODBMS as a data repository, an ODBMS also has the potential to understand the data it manages as it has done for years with simple data types. This includes being able to query multimedia object ***content***, and not simply meta-data. Here are some examples of applications that use an ODBMS for intelligent data management.

- **Working environments.** Traditional database management systems have offered basic creation, update and query capabilities to standard data types. Extensions to these systems may then also enable the same support for multimedia data types. Examples include:
  - **Multimedia editing.** Like other kinds of data, multimedia data is also expected to change. Users may add new data, as well as modify it within ODBMS. Since ODBMS understand the data formats, the users can request portions of videos, for example, for update. The ODBMS support data-specific operations such as cut, paste, crop, etc.
  - **Engineering design workflow.** In an engineering application, complex drawings are designed. These drawings need to be validated by mechanisms within the system whenever changes are made. These changes are validated against other parts of the design and existing design constraints, and update notifications are sent to affected components as well as the affected engineers. Design changes may also influence documentation, causing new diagrams to be generated and inserted into appropriate sections

of design and product documents.

  - **Intelligent Healthcare Networks.** These systems allow doctors to collaborate by including media-related patient data within their interactive communications. In addition, features such as intelligent data routing may be supported (media data may be analyzed when stored, and then routed to the appropriate healthcare specialists).

- **Presentation environments.** A further extension of support that ODBMS may provide addresses the delivery of multimedia data that have temporal constraints, such as audio and video. In these applications, the data is consumed as it is delivered, unlike electronic mail which is for consumption at a later time. The evolution here is that ODBMS are well aware of the time sensitivity of data delivery.

  - **Simple multimedia viewing.** Users retrieve multimedia data of interest and tell the system that they wish to view the data. As the data is retrieved from storage it is immediately delivered frame by frame to the user. Having selected a video for example, the user may have an interface similar to a VCR, allowing him or her to stop it, fast-forward, play the video in reverse, or jump to random points within the video. The delivered data satisfies the temporal viewing constraints.

  - **Complex Multimedia Presentations.** Users retrieve composite multimedia for viewing, which is delivered frame by frame (or sample by sample, etc.) by the ODBMS. Orchestration directions, stored as meta data within the composite multimedia, dictate the retrieval order of each component, whether in series or in parallel.

  - **Interactive Multimedia Environments.** These environments enable sophisticated database interactions, including real-time editing, analysis or annotating of video and audio, interactive multi-user collaborations and presentations that can be driven by user, application and system interaction, and advanced query capabilities.

So, the basic uses of a multimedia database can be summarized as follows:

- **Read.** Retrieve and view data, presentations, etc.
- **Update.** This includes creating new multimedia data, and the modification of data.

- **Compose.** Create compositions and presentations using basic multimedia data.
- **Query.** Search a multimedia database. Either indirectly querying multimedia meta-data, or directly querying the actual data itself.
- **Interaction.** This includes user and ODBMS interaction with multimedia data. The data is no longer static but made dynamic by support for user (and multi-user), application and system interaction. In addition, data can interact and affect other data (e.g. engineering designs). The ability to specify the expected behavior of these interactive systems must also exist.

However, the metrics one must use to select a multimedia database management system will depend upon the application, and it's requirements. As we have seen by the few examples above, there are a number of ways in which ODBMS can be used, and certainly most ODBMS existing today can act as pseudo- or simple multimedia data repositories.

4 General Implications on Software and Hardware

We first examine the general effects of handling multimedia data on software and hardware. Again, depending upon the actual application requirements, any or all of these may be a concern.

1. **Data Types.** Multimedia data can be treated as BLObs (binary large objects), as in simple data repositories which do not recognize/support multimedia data formats. Or, the data format can be understood. In the latter case, there are numerous format standards that exist for images, graphics, audio, animation, and video. Within object- oriented software these data types will exist as class definitions – so, there may be a large number of class definitions to support all of these types. For each class definition, associated methods must be defined to support operations on the specified data types. As some of these formats are based upon compression algorithms, some common operations may not exist in a given format (e.g. crop in an MPEG2 P frame). In addition, if the users define physical composite data types, there may be an eventual proliferation of classes.
2. **Data Size.** The size of multimedia data can be substantial. Even compressed movie formats result in four to five gigabytes of data for a two hour movie. This fact alone can have substantial effects on the design of

hardware and software as is illustrated later.

3. **Viewing.** In general, viewing a multimedia object requires retrieving it, and then rendering it on a screen (or playing it on a speaker). For data that doesn't have temporal constraints, this is straightforward. For data such as audio and video, this requires the proper bandwidth at several points to allow the temporal constraints to be satisfied. In many configurations, and for other reasons, the constraints may not be met. Therefore, the user may indicate a level of quality of service (QoS) that will satisfy his or her current needs. However, the environment may or may not be able to meet this level of service at a given time, allowing the user to try again later, or request a lower level of service.

4. **Querying.** A very common activity in any DBMS is querying. Queries involving typical data types such as integers and strings is generally straightforward. Multimedia data requires it's content to be interpreted to be queried. This can require sophisticated indexing schemes and image and audio analysis algorithms to generate content descriptions. Users may want to query for images that "look like this," or involve specific actions such as running. Mechanisms to generate these indices, interfaces and languages in which to pose the queries, and underlying components to optimize the queries may be required.

5. **Throughput.** To meet the requirements for playing audio and video data, the software and hardware must be optimized to ensure that it can satisfy the temporal constraints. The throughput of the system is the primary reason these constraints are not met. Specific issues are addressed below.

6. **Resource Scheduling.** A user may request multiple audio and video streams to be retrieved in parallel from disk. In addition, multiple users may be simultaneously requesting different data from the same disk. Finally, devices used to play out multimedia data, such as monitors and speakers, and devices used to record multimedia data, must also be scheduled without conflict.

7. **Memory, Bus, CPU.** To handle multimedia data, such as rotating high quality images, a computer must have sufficient main memory to load the images. Buffer strategies, and the speed of the data bus and processor can also have a significant affect on the throughput of the system.

8. **Special Chip sets and cards.** Because of throughput requirements for audio and video, specialized chip sets and boards have been manufactured. These hardware products are faster than the same operations

that may be performed using software, but of course they usually cost more, and are inflexible.

Hardware has been designed for data capture, presentation, data conversion, compression and decompression, and multimedia operations such as crop, rotate, etc. In addition, companies such as Sun Microsystems have developed CPUs that have specialized instruction sets for handling multimedia data. All of these solutions increase the throughput of the system, generally also providing added functionality.

9. **Storage.** Due to the size of the data, only two to three movies, for example, can be kept on the largest hard disk drives in mass production, which are about nine gigabytes. This means that repositories for multimedia data will have to include larger storage arrays, such as disk arrays and CD jukeboxes for tertiary storage. Due to throughput requirements for videos, the secondary storage devices will need to be fast enough to handle multiple requests. Therefore, parallel disk systems may be required.

10. **Networks.** Finally, if data is being delivered across a network, there are added concerns about throughput and reliability. Current widely used protocols and hardware are not sufficient for transporting high quality video. Users will generally tolerate a lower quality picture over speed degradation (e.g. jitter).[42]

5 ODBMS Specific Implications

As described above, the implications related to the introduction and use of multimedia are far-reaching, going well beyond software re-design. It will be difficult to isolate those requirements uniquely necessary for ODBMS to satisfy the demands of multimedia applications, since simply solving them will not be enough. Modifications to the operating systems, networks and computer hardware will also be required. In fact, MMDBMS may be built upon or integrated into optimized multimedia-aware operating systems, running on specialized hardware (in[22] they looked at kernel support for video and audio).

However, as previously stated, the requirements ODBMS must satisfy will depend upon the application. So, we should anticipate that each of the ODBMS products can be placed somewhere on a number of axes describing its degree of functionality with respect to the multimedia features it supports. In this section we describe the effects that multimedia will have on ODBMS, and based upon these generate a list of requirements.

5.1 Data Types (and Data Location)

Data type support can range from no support to extensive support. Here we describe five levels of basic type support that can be offered by an ODBMS.

To minimally support multimedia, ODBMS may simply act as data repositories. At the simplest level of support, the database will contain the filenames of the multimedia data with the latter stored directly within the file system. (The file system could be located on a dedicated and specialized continuous media server.) Access to the data will generally be faster, as the ODBMS storage manager will not be involved. However, the security of the data is left to the operating system, so the ODBMS cannot insure data integrity or validity of the file references.

At the next level, the data may be stored within the database. Providing limited support, the data is stored as BLObs, so that the ODBMS do not understand the data types being stored. They merely store and retrieve the data as requested. The data is secure, it can be updated using the transaction system, and it can be shared by multiple users. The ODBMS may also support random access to the data, allowing updates to portions of the BLObs,[8] but the user must define and maintain the pointer references.

In the next level of support, the data is stored within the database, and the ODBMS understand the structure of the data. The ODBMS will provide the class definitions for several standard ***static*** multimedia data formats (e.g. JPEG, GIF, MPEG, etc.) for non-continuous data. For each format, there must be associated methods that can operate on the data. The ODBMS may provide support for a single (perhaps proprietary) data format for each data type, requiring all data not in that format to be converted upon import.

The following level provides direct class support for temporal (or "continuous") data types, including audio and video. However, many current commercial ODBMS do not provide such support. In a few cases, this may be due to the amount of disk space required to support the storage of such data (e.g., some ODBMS databases are limited to the size of a UNIX partition).

Finally, the multimedia data types (particularly temporal data) are supported as basic data types within the ODBMS. Better performance gains would be achieved if the data types were supported by the operating system. Internalized support for multimedia data types allows the ODBMS to provide optimized handling of it.

In describing the data type support, we have also mentioned possible locations for the multimedia data. One other option not yet described is a specialized storage manager which would be part of the ODBMS (such as the continuous object manager described in[25]).

Levels of support (data types):

- File pointers (surrogates)
- BLObs support
- Class support for static data types
- Class support for temporal (and spatial, etc.,) data types
- System support for multimedia data types

Levels of support (data location and management):

- File System
- Non-specialized Database
- Continuous Media File Server
- Continuous Object Manager

5.2 Data Size

Current ODBMS may not provide the support necessary for large amounts of video data. There may be a limitation on the size of the database (a constraint enforced by the file system used), or simply a design restriction of the ODBMS. This may be acceptable if the amount of data will be limited, or if it acts as a simple data repository that only hold meta-data and filenames, and not actual multimedia data. Otherwise, it's conceivable that the ODBMS should be capable of storing and managing several gigabytes for small multimedia objects such as images, and several hundred terabytes or more if the database will expect to hold significant amounts of video or animation data. This will generally be accomplished by supporting tertiary storage, such as CD jukeboxes.

For a point of comparison, an uncompressed image 1024 x 1024 pixels at 24 bits/pixel would require about 3 MB of storage. Using compression, we perhaps get a 10:1 compression so that the image can be stored in 300 KB. For a video of a 5 minute duration at 30 frames per second (and 3 MB per frame), uncompressed storage would consume 90 MB per second, 5400 MB per minute, and 27000 MB (27 GB) in 5 minutes. With a better compression rate of about perhaps 100:1 that can be achieved (sometimes), we could store 5 minutes of high resolution video in 270 MB of storage.

Now, for the support defined below, we will arbitrarily say that a small database should be able to hold 100 images or videos; a medium database should hold 1000, and a large database 10,000 videos or images. Of course, the idea here is to provide some enlightenment as to the amount of disk space required to store multimedia data. However, remember that 10,000 videos of only 5 minutes each is just 833 hours of video (which may be considered quite small for some applications).

Levels of support (db size for non-continuous data):

- Small (30 MB)
- Medium (300 MB)
- Large (3000 MB — 3 GB)

Levels of support (db size for continuous data):

- Small (27000 MB — 27 GB)
- Medium (270 GB)
- Large (2700 GB — 2.7 TB)

5.3 Data Model

The richness of the data model plays a key role in usability issues. Certainly, multimedia data types must be supported, but that only provides an underlying foundation upon which additional features may be built. In this section we describe the support that can be provided by a data model.

5.3.1 Frameworks

In most cases, multimedia data types will be implemented as classes, having attributes and methods to support each data type. They may be part of a large class hierarchy, or framework, composed of similar data types (e.g. all sound-based data types, all graphic data types, etc.). A framework generally goes beyond simple type definition to more complete support for the data types including a wide range of methods and additional supporting classes (e.g., to import, view and export). Some companies develop frameworks to sell that can be imported into ODBMS.

The handling of multimedia data beyond storage and retrieval requires vast amounts of code, even for simple operations such as cut, paste, crop, rotate, and invert. In data repositories, these operations are not required since the multimedia objects are stored as BLObs. In several applications, such as a real estate database, the only operations that may be required are insert and delete, as all image editing is performed outside of the ODBMS. So, simple frameworks providing limited operations would be acceptable. Simple frameworks that support audio and video may include operations such as import, convert (to another format), export, and play (to send the data to a speaker or monitor). In simple frameworks, the "play" operation may just copy the multimedia data to the file system, and invoke a local multimedia application (so the code required to play the data is not defined within the framework).

However, for other applications, more extensive capabilities will be required. These capabilities may be provided in three forms. First, extensive support for multimedia data may be provided within multimedia frameworks (this may or may not include video, due to it's complexity). Second, to limit the size of a framework, specialized frameworks may be available that have been created to satisfy specific application requirements. Finally, and perhaps most likely, little support will be directly provided by the ODBMS itself. Rather, due to the complexities, size, and proprietary nature of providing an extensive framework, application hooks may be provided so that the user can select and use specialized multimedia applications that meet their requirements (e.g. MediaDB [1]). Alternatively, the specialized multimedia application developers may provide libraries (for a fee) that can be compiled with the database to provide the support required. It may also be possible that for very specialized multimedia data handling, service providers may be available on the internet. Data would be sent to them and returned sometime later, depending upon the quality of service requested. Of course, sophisticated MMDBMS will provide an extensive array of data types and support.

Level of support (frameworks):

- No support (users must rely solely upon Applications, Internet)
- External Support (Hooks, Libraries)
- Simple (e.g., insert/delete/play)
- Specialized
- Extensive (with video support)

[1] MediaDB is a trademark of MediaWay, Inc.

5.3.2 Support for multimedia relationships

There are several types of relationships that can be expressed between multimedia data for composition and presentation. The relationship semantics do not exist in current ODBMS. However, they can be expressed by augmenting standard relationships with additional methods and classes.

Composition The first of these relationships enables composition of multimedia data, to define more complex multimedia data types. Composite relationships supported by most ODBMS have no additional semantics other than "is logically part of", or "is physically part of." Alternatively, composite relationships could be extended to support multimedia semantics and constraints for composition. Composite relationships would not generally be applied to BLObs.

Level of support (composition):

- None
- Basic Composite Relationships
- Multimedia-aware Composite Relationships

Spatial and Temporal Structures Multimedia data can be viewed in some random fashion, or be organized into multimedia presentations. Unlike simple consumption of a single image, presentations will have a spatial and/or temporal dimension. Therefore, to support them, relationships are defined between the parts of the presentation to provide spatial and temporal structure. Spatial structures support the definition of books and papers – the layout of information which has no temporal constraints; while temporal structures allow the temporal dynamics to be specified – when data should be played. In contrast, simple composition describes associations without temporal or spatial structure.

These relationships define new layers upon basic multimedia data. The layers of any multimedia data model will vary depending upon use. For image analysis, models such as those used in[13] may be practical, while for presentations, temporal and spatial layers are required.[1,29,40] It is these layers that extract or generate a wealth of information, and the power to organize that information, so that it may be more easily and intelligently consumed.

We discuss temporal and spatial relationships because they will be commonly used within multimedia ODBMS.

Spatial relationships organize the visual layout of data on a virtual page or medium. The virtual medium may exist across multiple machines. Within a spatial presentation, users can move around within the boundaries defined by the virtual medium. Users may be allowed to move and restructure the data within the medium. Spatial structures may include three-dimensional environments, or virtual realities. Spatial constraints may be used to control three dimensional object movement, and inter-object spatial relationships. Tools may be available to define spatial relationships using graphical user interfaces.[4,16] Instead of providing semantically rich spatial relationships, some systems may support spatial grammars[48] which are closer to a scripting language. In this case, the spatial relationships are not directly represented within the database, and therefore cannot easily be queried or re-used.

Temporal structures dictate the temporal layout, orchestrating the presentation of the data. A simple example might be: "Play video v1 and audio a1, and, when finished, play music m1 until slideShow s1 is done." Basic temporal structures produce serial and parallel (hierarchic model[28]) presentations of data. Presentations can also be defined by simply associating a presentation time and duration (timeline model[10]) with each multimedia object, thereby not requiring the use of temporal structures; however, the model is severely limited since it can only be used to define static presentations. Other approaches for temporal specification include scripting languages, specification languages, and extended programming languages. While specification languages may be mapped to temporal relationships within ODBMS, scripts and programs cannot. Scripts and programs can be used to control the presentation of multimedia data, but with no representable temporal structure they cannot easily be queried, re-used, or supported by the underlying components of the ODBMS (e.g. the storage manager).

More advanced temporal models, such as,[4,37,47] are based upon the event-driven model, enabling user, application and system events (interaction) to affect the run-time presentation. In fact, with complex user interaction supported, ODBMS can become the infrastructure for a distributed interactive multimedia environment supporting multiple users, guided instruction, collaboration, and interactive virtual worlds. Of course, this will heavily depend upon the existence and inclusion of software and hardware to support these environments.

In addition to temporal relationships, temporal and spatial constraints and control structures are required for describing the control of the presentation. For example, fine-grain synchronization constraints, delivery constraints and presentation constraints. Fine-grain synchronization is used to control the temporal presentation of two or more multimedia objects, e.g. maintaining lip synchronization between a soundtrack and a movie. Delivery constraints specify quality of service parameters, and alternative presentations depending upon resource availability. Presentation controls enable a high level of control over a presentation, for example the application of special effects.

Level of support (spatial structures):

- None
- Spatial Grammars / Language support
- Spatial Structures

Level of support (temporal structures):

- None
- Timelines, Scripts and Programs
- Temporal Structures
- Simple Interaction
- Advanced Multi-user Interaction

Level of support (constraints):

- None
- Presentation Constraints
- Interaction Constraints
- Spatial Constraints
- Temporal Constraints

5.3.3 Device Hierarchies

The capture and presentation of multimedia data may involve hardware devices such as cameras, microphones, computer cards, speakers, monitors and other equipment. By modeling these devices as classes, the underlying communication details can be abstracted away from the user, and allow the entire process (capture, storage and presentation) to be represented within the data model. By providing this support, it will be easy to extend the device hierarchy to add additional devices as required. Since the devices exist as abstractions, some devices may actually be implemented completely in software. Finally, if devices are simply objects within the database, then they can be used without regard to location (location transparency).

It may also be practical to consider low-level device type support, similar to low-level multimedia type support.

If the ODBMS define low-level device types (or hooks), then this may provide better handling of device invocations.

Level of support (device hierarchy):

- None
- Device Hierarchy Support
- System-defined Base Device Types

5.4 Use implications

Beyond the data model and device support, an ODBMS can offer features related to user interaction, data manipulation, and data query. If the ODBMS acts as a simple repository, then these features may be completely provided by the client application.

5.4.1 User Interaction

User interaction support is critical for multimedia applications. In many cases a few of these features may be supported by application environments rather than the ODBMS itself. The first is sophisticated graphical user interfaces. With spatial and temporal rendering required, the interface design capabilities must be feature-rich. One example is the ability to control the presentation, and the devices which are rendering the presentation. This may include simple VCR-type control panels enabling the user to play, fast-forward, pause or rewind a presentation, or more advanced controls including interactive data filtering, querying and visual perspective controls. In addition, the ODBMS should support multiple view definitions of the data (as relational systems have done for years), as there are many ways in which the data can be viewed. The user should also be able to control the quality of service parameters to adjust for resource limitations, cost of delivery, and personal visual and aural preferences.

Level of support (presentation control):

- None
- Sophisticated UIs / Multiple Views / QoS controls

A second part of user interaction is the support of user events. The temporal data model may implement the event model, but there must be underlying support within the ODBMS for it. This support can be provided by features similar to the change notification features found in active ODBMS today, such as ITASCA,[20] and exemplified by the event manager in DAMSEL,[37] or in the interaction manager in the research MMDBMS Vodak.[44] Basic event handling may be supported by some of today's ODBMS (the response delay of the system to events may not be acceptable to some users). To support advanced multi-user collaboration and video interaction, the systems must provide soft real-time multi-threaded functionality. As the user may be interacting with several different windows and data types, the system should support different threads of control[25] within each user process.

Level of support (underlying event support):

- None
- Basic Event Support
- Real-time Multi-threaded Event Support

5.4.2 Data Manipulation

One of the purposes of a DBMS is to provide the ability to modify data stored within it. This is a trivial exercise for simple data types, and quite a non-trivial one for multimedia data. For simple data repositories, support for data modification isn't required. Of course, the ODBMS would support the editing of simple-typed meta-data associated with multimedia data. It's true that a well-developed multimedia framework, external applications, or hooks to applications can provide the editing capabilities required (see section 5.3.1). However, if modifications will be performed within the ODBMS, they can provide integrated editing environments through tool and language support. These tools will be practical for interactive editing, but may not be useful for modification integrated into applications. Therefore, multimedia data manipulation constructs or languages (DML) will also be required (e.g., SQL has a DML component). In ODMG-93, the DML is the programming language. However, programming languages do not have support for handling temporal data. Therefore, a multimedia DML or constructs should be provided (e.g.,[37]). Multimedia DMLs support data flow, indicating sources, sinks, and operations in between to analyze and modify the data. Then again, it is not enough to provide simple cut or crop operations; these operations must adhere to the real-time delivery constraints. For example, a user may decide to crop and rotate a video that is currently being viewed. To maintain the temporal constraints, these operations must be able to provide predictable and worst case information about the operation overhead so the system can determine if quality of service guarantees can be met when the operation is inserted within the data stream between the source and the sink (the viewer).

Once the data is modified, it can be stored again in the ODBMS. It may replace the old version, or exist as another version of the data. If users constantly create new versions of data, by constructing various presentations for example, the database will grow large rapidly. To reduce the need to store new versions of data, the ODBMS may support storing the operations to generate the new data, rather than storing the new data itself – called "derived data" – implemented, for example, in.[41] When derived data is accessed, the operation(s) is applied to the current state of the data to produce the new version.

Level of support (data manipulation):

- None
- Integrated Tool Support
- MM Data Manipulation Language / Constructs
- Real-time Aware Operations

Level of support (derived data):

- None
- Derived Data

5.4.3 Data Model Manipulation

In addition to supporting data manipulation, an ODBMS may provide run-time support to modify the data model (schema), called dynamic schema modification. If an application will be constructed and delivered to users as an end product, dynamic schema modification may not be required. However, if the application is expected to evolve over time, like most things in this world, then support for dynamic schema modification will be important. Currently, C++-based ODBMS require a complete re-compilation and re-distribution of code when the data model is modified (some very limited modifications to the data model are supported). Multimedia applications such as design engineering may be adding new kinds of parts, methods, and constraints at anytime, and cannot simply shut the system down each time a modification is made to the schema. Schema evolution can currently be accomplished by products developed using interpreted languages such as LISP, CLOS, Smalltalk and Java.

Level of support (dynamic schema modification):

- None
- Limited
- Complete

5.4.4 Querying

Once data has been stored within a database, retrieval is usually performed by applying queries against it. The queries contain predicates which must be satisfied by any data that is retrieved. The predicates usually involve partial or exact matches such as ***find all employees whose last-name is "DOE"***, and value ranges such as ***find all employees whose salary is between $2000 and $2500***. But how can multimedia data be queried?

Similar to data manipulation an ODBMS could support a multimedia query language. But, even at this point such issues as far from being solved. When a multimedia query language is provided, optimization techniques may be able to be defined to optimize multimedia-based queries.[5,15,17,18]

For simple multimedia applications and ODBMS products, the most straight- forward means to query multimedia data is to define keywords (meta-data) associated with the data, also known as manual indexing. Generally when the data is entered, the keywords are also entered. The use of a standardized keyword dictionary[23] is important so that all data is classified using the same terminology. When a user wants to find a ***white house with a bay window in front***, the query examines the keywords of all house images stored in the database. The images themselves are not queried. Some problems with a keyword approach do exist. First, keyword classification is subjective since it is performed by a human being. Second, there will always be exceptions, and some data may be wrongly classified. Lastly, keywording is usually limited to a well-defined abstraction of the data (e.g., for each house image, a specific set of features is classified). This means that if the abstraction is altered, perhaps by adding additional attributes, all of the data will have to be reviewed again, adding new keywords as required. With 5000 to 1,000,000 images this would be a very expensive task. On the positive side, keywording enables very fast retrieval of data, and standard indexing approaches can be used since the keywords (strings) are a data type supported by every DBMS. For specialized applications, such as real estate image databases this is probably all that is required. However, the classification of videos may generally be overwhelming. Considering that a video is simply a sequence of (related) images, classification could require indexing ***each*** scene – and, there may be hundreds (or more) within one video.

Keywording is a human-intensive, error-prone task. It also limits the kinds of queries that can be posed to the abstraction supported and the keyword dictionary that was used. A second approach that may be supported in MMDBMS (when very intelligent image and sound analysis is available) is content-based retrieval / querying (CBR/CBQ). CBR provides the ability to base queries on the content of multimedia data. However, analysis of the data must still be performed, but in this case it is done by audio and image analysis algorithms. Therefore, as better algorithms become available, the data may have to be re-processed – however, not by hand. Analysis of data may generally take place when the data is stored to the database. The results of the analysis may be keywords, or multi-dimensional indexing structures which describe the data.[13,33,51] Queries against this data then still takes place on meta-data, however this meta-data attempts to minimize the data abstraction by attempting to describe the data as completely as possible. Generated data could include attributes such as lines, shapes, colors, and textures from which objects could be determined at some later time. As these algorithms become more sophisticated, the amount of human-generated indexing will be reduced, but it's unlikely that it will ever be eliminated.

However, the area is still quite immature, yet there is a lot of research going on in this area. But, issues dealing with each of the various multimedia data types are non-trivial. For example, audio data can be queried for sounds, patterns, intonations, and words. It would also be useful if the ODBMS supported queries for audio-based interactions such as dialogues, arguments, musical duets, etc.

For queries involving images we have already mentioned querying on shapes, textures and objects. Video is simply a sequence of images, so image query techniques may also apply. However, video also has the temporal dimension, so therefore we may want to find specific scenes, scene changes, commercials, people eating, running, fighting, shooting, kissing, or riding bicycles for example. To distinguish CBR on temporal data such as video and audio, from that on images and graphics, we will call CBR on video "dynamic content-based retrieval", or DCBR; and, image-based CBR we will call "static content-based retrieval", or SCBR). The notion of analyzing video and detecting actions is far from a reality at this point in time. However, as the importance of multimedia databases grow, so will the demand for the ability to intelligently query them.

In addition to standard query models that work fine for simple data types, new query models will be needed (such as CBR). However, with CBR, providing queries with exact matches will generally not be practical (or possible). Therefore, query languages will need to be augmented with predicates such as "like" to allow approximate matches to be found. These techniques, called fuzzy predicates, are currently being research[6,9,50] (and to a limited extent are supported within some current DBMS for simple data types). In fact,[50] provides a very good description of retrieval mechanisms and multi-layered indexing techniques.

Querying on multimedia data will require advanced indexing techniques. These indices will provide more efficient query execution, and faster results. So, MMDBMS will include them.

Query by example Like the query interfaces of current DBMS, new query interfaces will be required to support CBR. These interfaces will enable the user to specify the interesting attributes by providing examples to match – such as a drawing, photograph, action, or sound.[7,11] The query manager can then use that example to find other examples in the database like it. This approach is called query-by-example (QBE), and some projects have already implemented static QBE in research systems for image retrieval. In static QBE, the users draw an example of the image they want to retrieve, using shape, color or texture for example. In dynamic QBE, a user would be able to illustrate an action perhaps from a palette, to describe an example of the video or audio sequences that should be returned. Current technology is far from supporting dynamic QBE.

Querying different data types Another area of concern for data analysis and querying is imposed by the wide variety of multimedia data formats for each data type. It means either that analysis code exists for each format, or that all data of a given type (e.g. image) is translated into one standard format. It would be useful to keep the data in this common format if the data is analyzed again at a later time, or if the ODBMS supports run-time querying of the database. A problem with data formats, however, is that formats for analysis may require more space than compressed formats. For example, MPEG enables a 100:1 compression on video, but no algorithms currently exist that can properly analyze the data in this format. In addition, many compression algorithms are not lossless,[39] so the repeated compression and decompression of media data using them will deteriorate data quality.

Of course, MMDBMS will include static and dynamic CBR support, efficient indexing techniques as well as layered data models to support them. However, due to the processing demands and the necessary sophistication of the algorithms, most DBMS for some time to come will be limited to manual keyword indexing.

Level of support (index creation):

- None
- Manual Entry
- Automatic Index Creation
- Assisted Interactive Index Creation

Level of support (index structures):

- None
- Basic Index Structures
- Multi-level Index Structures

Level of support (keyword query support):

- Basic Keyword Search Predicates
- Fuzzy Search Predicates

Level of support (content-based [fuzzy] query support):

- None
- Static CBQ
- Dynamic CBQ

6 Underlying Issues

6.1 Throughput

Given that multimedia data is generally large, throughput for most applications is critical. For static data repositories it may not be important because the user may be willing to tolerate some delays. In addition, if data activity doesn't involve a user (e.g., off-line data analysis), then the throughput may not be important. However, when human-consumption of temporal data is involved, throughput becomes critical. For single user applications, the effort necessary to support the throughput requirements will be lessened. In most cases, many applications will exist that must support multiple users accessing multiple audio and video objects simultaneously.

Temporal data could be delivered all at once, requiring the users to wait for all of the data to arrive – the approach taken with a data repository. However, this means that the user may have to wait perhaps a few hours or more to start watching a video. In more advanced applications, the users will want to start watching/listening to the videos and high fidelity audio immediately, and without jitter. This means that the ODBMS will have to regulate the delivery of the data, frame by frame, to the user. Current ODBMS do not provide this level of service.

Unlike simple data types, and even long text that may be several megabytes in size, for real-time consumption temporal data has delivery requirements that can only be satisfied by regulated delivery of data. This forces the ODBMS to manage access to the disk, negotiate delivery schedules, and set user-process priority levels to ensure that quality of service requests are met. To understand the delivery constraints, the storage manager which retrieves the data, must be well-integrated with the multimedia frameworks which understand the high-level requirements of each media type and the QoS parameters set by the application or the user. Problems that occur during delivery will probably not be handled by the storage manager, but rather by the software or methods that are controlling the overall delivery of the information. Integrated solutions such as the research discussed in[19] focus on providing multi-layered specification and distributed resource management for continuous data. Playout management as part of the AMOS project which integrates a playout manager within the DBMS is described in.[45]

In ODBMS, data are generally placed on pages wherever space is available. Some systems also support data clustering, so that the storage manager will (or attempts to) co-locate data on the same page to minimize page faults during retrieval. Due to the temporal constraints of continuous data, more rigorous placement algorithms will be required.[27] Since each multimedia data object consumes many pages, these algorithms are not concerned with minimizing page faults, but rather data placement on disk (or disk arrays) to optimize retrieval. Placement may take into account the expected retrieval pattern, and is further complicated when synchronization of multiple streams is required. Other retrieval algorithms may also focus on replication to satisfy multi-user demand.[14] These issues (described in greater detail in[26]) are important and will directly affect the system's throughput.

To complicate things, compression algorithms (such as MPEG) may generate compressed video frames of unequal size. This is a problem for placement and retrieval algorithms, since the compressed frame sizes are not predictable (however, with a 100:1 compression factor the space saved is considerable). Therefore, when retrieving the data, the disk, buffer, and network resources required will vary dramatically. Compressed streams of this kind are called variable bit rate streams. Other kinds of data, including audio, compress predictably, and therefore produce constant bit rate streams which are much easier to handle.

When defining the throughput support requirements, the execution environment must be taken into account. If the ODBMS application and the user(s) will work directly on the same machine, then it reduces some of the complexity of data delivery. Otherwise, if the user (client) is on a different machine from the ODBMS (the server), then the network must be taken into account. Let's look at options under each configuration:

6.1.1 Single machine

With the user sitting at the server, there will be no network concerns with respect to data retrieval, if all of the data is locally stored. However, if the data must be retrieved from devices across the network, then the problems are the same as those in the client-server configuration. Considering the size of future multimedia databases, it is very likely that tertiary storage will be used. This will add some delays to accessing the data. In a single-user configuration our basic choices are:

- Dump the data to a file. When finished retrieving the data, the user application can access it.
- Allow the user application to directly access the data within the database by supplying a block pointer. To allow this, the application will need to understand how the database is configured, and the data is formatted.
- Allow the user application to access the multimedia data in shared memory.

6.1.2 Client-Server

In a client-server configuration, the data must be delivered to the client machine so the network may become a bottleneck. Current network speeds available through the internet and local ethernet for example, are not sufficient at this time for high quality video and fidelity audio data. In addition, the protocols cannot support the time-constrained delivery of data, nor do they support QoS parameters – so new protocols will be needed.[8,32,34] In this configuration, device distribution, device scheduling and data delivery will all be concerns. Devices such as capture boards, cameras, VCRs, and special monitors available on the server or the network will be shared by multiple users, so the ODBMS will have to provide device transparency and device sharing.[25] To meet the bandwidth requirements for video, high speed networks that can understand QoS parameters will be necessary. For applications involving images, graphics and audio, this won't be as serious a problem. The ODBMS may have no control over the network used, however the network will play a major role in their ability to deliver video and audio data within acceptable QoS levels. Advanced MMDBMS will be connected to clients using high speed communications to ensure that QoS requests can be met. ODBMS may provide varying support for audio and video streams – e.g., supporting a maximum of 2 simultaneous streams (perhaps acceptable for a single user), to 1000+ simultaneous streams provided by more advanced systems and suitable for many users. Note that systems should support capture of continuous data as well as it's presentation.

Level of support (throughput):

- Batch (acceptable for images and graphics)
- Single-user video, multi-user voice (# of simultaneous streams)
- Multiple-user video, multi-user high fidelity audio (# of simultaneous streams)

6.2 Operating Systems

Due to the time-critical nature of temporal multimedia, ODBMS that will provide real-time access to video data will require multimedia-specific operating systems. Until these are available, soft real-time operating systems will likely be used. It is important to understand that many of the metrics only provide a portion of the solution. In that regard, real-time operating systems can only be useful when other components, such as the ODBMS software and network protocols, can work in an integrated fashion to satisfy all of the various constraints associated with multimedia capture and presentation. Even as computers and storage devices become faster, the number of users accessing a database will grow. In a single-user environment with a relatively fast system real-time operating systems may not be required. In multi-user systems, the ODBMS will need to be able to accurately control the delivery of data, and prioritize threads of execution, ensuring that QoS requirements are met. To aid in meeting these delivery constraints, new buffer management strategies[24,30] will be necessary within operating systems. In addition, to satisfy quality of service constraints, systems will have to define admission control policies based upon resource availability and other factors, such as priorities. As multimedia applications become wider spread, new operating systems will be introduced or extended address their needs. It is likely that operating systems specialized for multimedia and multimedia DBMS will be developed in conjunction with specialized hardware.

Level of support (operating system):

- Non-real-time OS
- Real-time OS
- Multimedia-aware or dedicated OS

7 Related Issues

7.1 Internet Access

While more and more people are accessing the internet, it makes it an attractive place to offer access to databases. And, in fact, many databases and repositories are currently accessible via the worldwide web - - image and graphic sites are quite prevalent. But, until the internet can offer sustained transfer rates suitable for multimedia video, users will not have access to video and high quality audio data in real-time. Of course, a user can download an entire video or audio file and then view it. One example research project reports offering 10 frames per second throughput over the internet,[7] which may be useful for limited applications.

7.2 Standards Impact

Currently, the object database vendors' standards group (ODMG) has not addressed issues related to support for multimedia data. However, the frameworks and internal architectures required for MMDBMS can still be supported while maintaining compatibility with the ODMG standards. Other standards efforts will have more of an impact on multimedia database systems. One such standard is ISO/IEC 14478 - ***Presentation Environments for Multimedia Objects (PREMO).*** It focuses on many issues related to multimedia data type support, synchronization, and programming. Standards compliance is important for portability of data between products, such as the ISO SGML/HyTime multimedia interchange standard.[31] As MMDBMS become a reality, standards like PREMO and SGML will play an integral role. Of course it's possible that a company will define it's own proprietary "standard," which may actually serve little or no purpose.

Level of support (standards support):

- None
- Proprietary
- OMG/ISO Compliant

7.3 Other Issues

There are, of course, other issues pertaining to ODBMS that are of interest. For example, what distribution capabilities will be supported. Particularly with the plausible advent of widely distributed database systems, what architectures will be used (e.g. CORBA). Of course, within distributed systems, multiple languages will likely be used to interface to a database. So, unlike several current ODBMS, these systems should support language-independent storage of multimedia data. Another related issue is that of execution location – client or server – where does the processing take place? (Within pseudo-repositories, processing is handled by the client).

What level of authorizations will be supported within each ODBMS? Portions of audio or video may be classified, or an area within a video or image may be classified. When the video is played back, the restricted portions are hidden (before sending the images to the client). It's also possible that videos may be viewed separately, but not simultaneously. Or, that parts of conversations must be blocked out. History-based authorization which restricts future accesses based upon previous ones may also be used. Certainly simple authorization could be provided at the object level, but undoubtedly some systems will eventually provide more sophisticated authorizations like those described above.

There are also issues related to system extensibility. As software and hardware evolve rapidly in the area of multimedia, systems that can provide some level of plug-and-play capabilities will enable a faster evolution and customizability of their software.

Finally, as multimedia objects consume several pages, the battle between object and page servers will no longer be a factor (until, and if, page sizes become so large that several objects will again fit on one page!). However, as has been described within this paper, there are a number of other issues to consider.

8 Current Efforts

The current state of ODBMS that support multimedia is shown in the table below. In most of these systems, the necessary software is not supplied limiting the databases to primarily be data repositories. Certainly, basic class support is provided by each system to store images, while few have class support for audio and video. The class support provided by most systems enables the data to be stored within the database in a recognizable format. However, they are severely limited in the operations that are defined within the class definitions. In addition, they are more likely to be simple class structures rather than complex multimedia frameworks with device support. A table is included below that shows the state of several common ODBMS products [2] , and their support for multimedia. The table has been limited to those features which are currently supplied by these vendors.

[2] Not all vendors responded to our requests, so some data may be incorrect or out of date. Most vendors can be found from the ODMG web page (www.odmg.org), or likely from www.{company name}.com.

Company	Product	BLObs	Image Classes	Video Classes	Multimedia Tools	Real-time Delivery	Extensible Storage Mgr	CBQ
Gemstone	Gemstone	$\surd$	$\surd$					
Informix	Illustra	$\surd$	$\surd$	$\surd$	$\surd$			$\surd$
IBEX Computing	Itasca	$\surd$	$\surd$	$\surd$-	$\surd$-			
ADB	Matisse	$\surd$	$\surd$	$\surd$				
MediaWay	MediaDB	$\surd$	$\surd$	$\surd$	$\surd$			
ObjectStore	Object Design	$\surd$	$\surd$		$\surd$+			
Objectivity	Objectivity	$\surd$	$\surd$					
ONTOS	ONTOS	$\surd$	$\surd$				$\surd$	
O2 Technology	O2	$\surd$	$\surd$	$\surd$	$\surd$-		$\surd$	
Persistence	IDB	$\surd$						
Poet	Poet	$\surd$						
Sybase	Sybase	$\surd$						
Unidata	Unidata	$\surd$						
UniSQL	UniSQL	$\surd$	$\surd$-	$\surd$-				
Versant	Versant	$\surd$	$\surd$					
GMD Research	Vodak	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$+	

Current Status of Several ODBMS (and O-RDBMS) and Multimedia DBMS Products

Image data should be able to be stored within most of these systems, at least as BLObs; and, since video data is so large, most systems cannot handle it, or at least handle it easily. In addition, since the video must be retrieved it would be better stored within a file system. If the system supports extensible storage managers, then dedicated and specialized continuous storage managers can be built and integrated within the system. These managers could be designed to provide the throughput necessary to satisfy the temporal constraints.

Out of all of these, the features of the GMD research database project Vodak is the furthest along. It is the only one of these to provide limited real-time delivery of multimedia data. *****This table is NOT accurate in this version of the paper.***

9 Summary

This report has described several metrics for multimedia object database management systems. It has outlined the breadth of requirements which will require massive changes to every component of current systems to enable the most advanced MMDBMS to evolve. There are, however, many forms that a MMDBMS may take from pseudo-repositories to very advanced intelligent multimedia data management systems. We will undoubtedly see products throughout this range to satisfy the requirements of many different systems. It is unlikely for several years to come that every DBMS product will evolve to the high end of the spectrum. However, as multimedia data is integrated into our everyday lives, most DBMS products will also evolve to become sophisticated multimedia systems with all of the features (and more) that have been described within this report.

10 REFERENCES

[1] Aberer, K., Klas, W., "Supporting Temporal Multimedia Operations in Object-Oriented Database Systems", GMD-IPSI Technical Report. ***aberer@darmstadt.gmd.de***

[2] Al-Salqan, Y.Y.; Chang, C.K., "MediaWare: a distributed multimedia environment with interoperability" ***Proceedings of the Fourth Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises. WET ICE '95***, p. 128-37, Berkeley Springs, WV, USA; 20-22 April 1995. ***alsalqan@cerc.wvu.edu***

[3] Bancilhon, F., "Object database morphology" ***19th International Conference on Very Large Data Bases, Proceedings***, p. 693, Dublin, Ireland; 24-27 Aug. 1993.

[4] Buchanan, M.C. and P.T. Zellweger. "Automatic Temporal Layout Mechanisms," ***ACM Multimedia 93.*** 1993.

[5] Campbell, S.T., and Chung, S.M., "The Role of Database Systems in the Management of Multimedia Database Mgmt Systems," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[6] Cardenas, Alfonso F. Ieong, Ion Tim. Taira, Ricky K. Barker, Roger. Breant, Claudine M., "Knowledge-based object-oriented PICQUERY plus language" ***IEEE Transactions on Knowledge and Data Engineering*** v 5 n 4 Aug 1993. p. 644-657.

[7] Chen, C.Y.R.; Meliksetian, D.S.; Cheng-Sheng Chang, M.; Liu, L.J., "Design of a multimedia object-oriented DBMS" ***Multimedia Systems***, vol.3, no.5-6, p. 217-27. ***rchen@cat.syr.edu***

[8] Chen, Te-Chih. Lin, Wei-Po. Wu, Chin-An. Shen, Chih- Shen., "Client-server database environment for supporting multimedia applications" ***Proceedings - IEEE Computer Society's International Computer Software and Applications Conference 1994***, p. 215-220. IEEE, Los Alamitos, CA, USA.

[9] Chom, L.H., "Developing a text and image-based database production system and search engine" ***16th National Online Meeting Proceedings - 1995***, p.53-5, New York, NY, USA; 2-4 May 1995

[10] Drapeau, G.D. and H. Greenfield. "MAEstro - A Distributed Multimedia Authoring Environment," in ***USENIX.*** 1991.

[11] El-Medani, G., ***A Visual Query Facility for Multimedia Databases***, Technical Report TR 95-18, Department of Computing, University of Alberta.

[12] Fujikawa, K., et al. "Multimedia Presentation System "Harmony" with Temporal and Active Media," ***USENIX.*** 1991.

[13] Gupta, A., Weymouth, T.E., Jain, R., "An Extended Object- Oriented Data Model For Large Image Databases", ***Second Symposium SSD 1991***, Zurich, Switzerland, 1991.

[14] Ghandeharizadeh, Shahram. Ramos, Luis., "Continuous retrieval of multimedia data using parallelism" ***IEEE Transactions on Knowledge and Data Engineering*** v 5 n 4 Aug 1993. p. 658-669.

[15] Ghandi, M., Robertson, E., Gucht, D.V., "Modeling and Querying Primitives for Digital Media," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[16] Hardman, l., Rossum, G., Bulterman, D. "Structured Multimedia Authoring," ***ACM Multimedia 93.*** 1993.

[17] Hibino, S., Rundensteiner, E.A., "A Graphical Query Language for Identifying Temporal Trends in Video Data," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[18] Hirzalla, N., Karmouch, A., "A Multimedia Query Specification Language," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[19] Huang, J., M. Agrawal, J. Richardson, S. Prabhakar, "Integrated System Support for Continuous Multimedia Applications," ***Int'l Conference on Distributed Multimedia Systems and Applications***, Hawaii, August 1994. ***huang@htc.honeywell.com***

[20] IBEX Computing, SA. ***The ITASCA DODMS User Manual***, 1994. ***ibexcom@iprolink.ch***

[21] Jain, R, Editor, ***NSF Workshop on Visual Information Management Systems***, Redwood, CA. 1992.

[22] Jeffay, K., Stone, D., Smith, F., "Kernel support for Live Digital Audio and Video", ***Computer Communications*** v15 n6, 1992.

[23] Kackenhoff, R.; Merten, D.; Meyer-Wegener, K., "MOSS as a multimedia-object server" ***Multimedia: Advanced Teleservices and High- Speed Communication Architectures. Second International Workshop, IWACA '94***. Proceedings, p. 413-25, Heidelberg, Germany; 26-28 Sept. 1994. ***kaeckenhoff@informatik.uni-erlangen.de***

[24] Kamath, M., Ramamritham, K., Towsley, D., "Continuous Media Sharing in Multimedia Database Systems" ***Fourth International Conference on Database Systems for Advanced Applications (DASFAA'95) Proceedings***. Singapore, April 10-13, 1995. ***kamath@cs.umass.edu***

[25] Klas, W., Aberer, K. "Multimedia Applications and Their Implications on Database Architectures," GMD Technical Report 95-20. ***klas@darmstadt.gmd.de***

[26] Kunii, T.L., Shinagawa, Y., Paul, R.M., Khan, M.F., Khokhar, A.A., "Issues in storage and retrieval of multimedia data" ***Multimedia Systems Journal***, vol. 3 Oct/Nov 1995. ***ashfaq@eecis.udel.edu***

[27] Kwon, T.G., Lee, S., "Data Placement for Continuous Media in MMDBMS," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[28] Little, T.D.C. and A. Ghafoor, "Interval-based Conceptual Models for Time Dependent Multimedia Data," ***IEEE Trans. on Knowledge and Data Engineering***, 1993. 5(4): p. 551-563.

[29] Marcus, S.,***Multimedia Database Systems***, Technical Report, Mathematical Sciences Institute, Cornell University.
marcus@msicedar.cit.cornell.edu

[30] Moser, F., Kraiß, A., Klas, W., "L/MRP. A Buffer Management Strategy for Interactive Continuous Data Flows in a Multimedia DBMS" ***21st International Conference on Very Large Databases*** (VLDB). Zurich, Switzerland, 1995. ***moser@darmstadt.gmd.de***

[31] Newcomb, S.R., "Multimedia interchange using SGML/HyTime" ***IEEE Multimedia***, vol.2, no.3, p. 60-4. ***srn@techno.com***

[32] Nicolaou, C., "An architecture for real-time multimedia communication systems," ***IEEE Journal on Selected Areas of Communications***, vol.8, no.3, 1990.

[33] Niu, Y., Őzsu, M.T., Li, X., ***A Study of Indexing Techniques for Multimedia Database Systems***, Technical Report TR 95-19, Department of Computing, University of Alberta. ***niu@cs.ualberta.ca***

[34] Orji, C., Xu, Y., Rishe, N., "An Architecture for Operating System Support of Distributed Multimedia Systems," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[35] Őzsu, M.T.; Szafron, D.; El-Medani, G.; Vittal, C., "An object-oriented multimedia database system for a news-on-demand application" ***Multimedia Systems***, vol.3, no.5-6, p. 182-203. ***ozsu@cs.ualberta.ca***

[36] Pronios, N.B., Bozios, T., "Multimedia and Hypermedia Synchronization: A Unified Framework" ***npro@intranet.gr***

[37] Pazandak, P., J. Srivastava and J. Carlis, "The Temporal Component of DAMSEL," ***Second Workshop on Protocols for Multimedia Systems (PROMS '95)***, Salzburg, Austria, 1995. ***pazandak@cs.umn.edu***

[38] Rakow, T.C.; Neuhold, E.J.; Lőhr, M., "Multimedia database systems-the notions and the issues" ***Database Systems in Office, Technology and Knowledge), p. 1-29***, Dresden, Germany; 22-24 March 1995. ***rakow@darmstadt.gmd.de***

[39] Rakow, T.C., Lőhr, M., "Audio Support for an Object- Oriented Database Management System", accepted to ***Multimedia Systems Journal***, vol. 3 Oct/Nov 1995. ***rakow@darmstadt.gmd.de***

[40] Schloss, G. and M. Wynblatt, "Building Temporal Structures in a Layered Multimedia Data Model," ***ACM Multimedia 94.***

[41] Speegle, G., "Views of Media Objects in Multimedia Databases," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[42] Steinmetz, R., Engler, C. "Human Perception of Media Synchronization," IBM European Networking Center, 1993.

[43] Sudhakar, G N M. Karmouch, A. Georganas, N D., "Design and performance evaluation considerations of a multimedia medical database" ***IEEE Transactions on Knowledge and Data Engineering*** v 5 n 5 Oct 1993. p. 888-894.

[44] Thimm, H., Rakow, T.C., "Upgrading Multimedia Data Handling Services of a Database Management System by an Interaction Manager", GMD-IPSI Technical Report. ***thimm@darmstadt.gmd.de***

[45] Thimm, H., Klas, W., "Playout Management - An Integrated Service of a Multimedia Database Management System," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[46] Vittal, C., Őzsu, M.T., Szafron, D., El-Medani, G., ***The Logical Design of a Multimedia Database for a News-on-Demand Application***, Technical Report TR 94-16, Department of Computing, University of Alberta.

[47] Vazirgiannis, M., Hatzopoulos, M., "Integrated Multimedia Object and Application Modelling Based on Events and Scenarios," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.

[48] Weitzman, L. and K. Wittenburg, "Automatic Presentation of Multimedia Documents Using Relational Grammars," in ***ACM Multimedia 94.***

[49] Woelk, D.L., Kim, W., Luther, W., "An object-oriented approach to multimedia databases," ***SIGMOD Record 1986***, 1986.

[50] Wu, J.K., et. al., "CORE: a content-based retrieval engine for multimedia information systems," ***Multimedia Systems***, n3, 1995. ***jiankang@iss.nus.sg***

[51] Yang, Q., Vellaikal, A., Dao, S., "MB+-Tree: A New Index Structure for Multi-Media Databases," ***International Workshop on Multi-Media Database Management Systems (MMDBMS '95)***, Blue Mountain Lake, N.Y., 1995.