Metrics for Evaluating ODBMSs Functionality to Support MMDBMS

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

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. Most commercially available ODBMS today provide basic multimedia support, but significant component changes will be required to support the next-generation of multimedia applications.

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 (ODBMSs). While initially multimedia data was supported in RDBMSs as BLObs [11], of late ODBMSs have generally become the database management system of choice for multimedia data, since the mechanisms provided by ODBMSs better model multimedia [43]. 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 evaluate the capabilities required of ODBMSs to provide true multimedia data management. Other papers have described support requirements for multimedia databases, e.g. [28, 43], and although detailed in many respects, they describe a relaxed view of future systems. Conversely, [24] 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.

In the future we will see distinctions between those ODBMSs which provide simple multimedia support for multimedia data repositories, and a next generation of ODBMSs that provide support for sophisticated distributed multi-user interactive and collaborative multimedia environments – called multimedia database management systems (MMDBMSs). A number of projects have focused on multimedia data issues over the past fifteen years, e.g. [8, 9, 12, 14, 40, 44, 48, 54].

To understand the requirements that will need to be satisfied by ODBMSs, we need to know the types of multimedia data that may be stored and managed, and the kinds of applications that may be developed. For brevity we simply list these data types in Table 2.

There are several uses for an ODBMS that manages multimedia data. Again, for brevity, we provide a list of some of these applications in Table 1. The requirements of the application can be used to determine which features need to be supported within ODBMSs. In fact, for some applications like pseudo repositories, current ODBMSs can be used with little or no modification. For other applications, such as complex or interactive presentation environments, no ODBMSs exists today that provide the necessary features.

The basic uses of a multimedia database include:

- **Read.** Retrieve and view data, presentations, etc.

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

Data Repositories
- Pseudo repository
- Simple repository
- Electronic mail
- Engineering Designs
- Healthcare Information Systems

Intelligent Data Management
- Working environments
  - Multimedia editing
  - Engineering design workflow
  - Intelligent Healthcare Networks
- Presentation Environments
  - Simple multimedia viewing
  - Complex Multimedia Presentations
  - Interactive Multimedia Environments

Table 1. Examples of Multimedia Applications

- **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.

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

A multimedia application may cover a broad range of requirements, and may demand significant changes to hardware and software. In Table 3, we list the potential areas that could impact the features necessary to support these applications.

2 ODBMSs 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 ODBMSs 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, MMDBMSs may be built upon or integrated into optimized multimedia-aware operating systems, running on specialized hardware, e.g. [25] which provides kernel support for video and audio.

Data Type	Example Application
Text	Books
Graphics	Drawings, Illustrations
Images	Pictures, Photographs
Animation	Simulations
Video	Movie
Structured Audio	Midi Recording
Audio	Movie Soundtrack
Composite Types	Multi-track Audio with Video
Presentations	MM-based Training Course

Table 2. Multimedia Data Types

Data Types Supported
Large Data Sizes
Viewing Multimedia Data
Querying Multimedia Data
Orchestrations / Presentations
Modification / Analysis
Throughput
Resource Scheduling
Memory, Bus, CPU
Storage
Network Access

Table 3. Potential Areas Of Impact

As previously stated, the requirements ODBMSs must satisfy depend upon the application. So, we anticipate that each ODBMSs product 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 ODBMSs, and based upon these generate a list of requirements.

2.1 Data Types

Data type support can range from minimal to extensive. Here we describe five levels of support.

To minimally support multimedia, ODBMSs may simply act as data repositories, i.e. the database contains the names of the files storing multimedia data.

As the ODBMS storage manager is not involved, the ODBMSs cannot insure data integrity or the validity of the file references.

At the next level, data may be stored within the database as BLObs, i.e. the ODBMSs do not understand the data types being stored. The data is secure, it can be updated using the transaction system, and shared by multiple users. The ODBMSs may also support random access to the data, allowing updates to portions of the BLObs [9], 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 ODBMSs understand the structure of the data. The ODBMSs will provide the class definitions for several standard ***static*** multimedia data formats (e.g. JPEG, GIF, MPEG, etc.) for non-continuous data.

The following level provides direct class support for temporal (or "continuous") data types, including audio and video. However, many current commercial ODBMSs 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 ODBMSs 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 ODBMSs. Better performance gains would be achieved if the data types were supported by the operating system. Internalized support for multimedia data types allows the ODBMSs to provide optimized handling of it.

2.2 Data Size

Current ODBMSs may not provide the support necessary for large amounts of video data. This may be acceptable if the amount of data will be limited, or if it acts as a simple data repository that only holds meta-data and filenames, and not actual multimedia data. However, in general, the ODBMSs 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.

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 5 minute duration at 30 frames per second (and 3 MB per frame), uncompressed storage would consume 27 GB. With a compression rate of about 100:1, we could store the video in 270 MB of storage.

2.3 Data Model

The richness of the data model plays a key role in its usability. Certainly multimedia data types must be supported, but they only provide 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.

2.3.1 Frameworks

In most cases, multimedia data types will be implemented as class frameworks, with attributes and operations to support each data type.

Handling multimedia data beyond storage and retrieval requires vast amounts of code, even for simple operations such as cut, paste, crop, rotate, and invert. 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 these 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 many applications, more extensive capabilities will be required. Extensive support for multimedia data may be provided within multimedia frameworks. And, perhaps most likely, little support will be directly provided by the ODBMSs 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]).

2.3.2 Support for multimedia relationships

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

Composition This enables composition of multimedia data, to define more complex multimedia data types. Composite relationships supported by most ODBMSs have no additional semantics other than "is logically part of", or "is physically part of." Alternatively, composite relationships could be extended to

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

support multimedia semantics and constraints for composition. Composite relationships would not generally be applied to BLObs.

Spatial and Temporal Structures Unlike simple consumption of a single image, most multimedia presentations will have spatial and temporal dimensions. To support them, relationships are defined between the parts of the presentation to provide spatial and temporal structure. Spatial structures support the definition layout of information; while temporal structures allow the temporal dynamics to be specified – when data should be presented.

These relationships define new layers upon basic multimedia data. For image analysis, models such as those used in [15] may be practical, while for presentations, temporal and spatial layers are required [2, 34, 45]. 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 ODBMSs.

Spatial relationships organize the visual layout of data on a virtual page or medium which may span multiple machines. 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 [5, 18]. Instead of providing semantically rich spatial relationships, some systems may support spatial grammars [53] which are closer to a scripting language.

Temporal structures dictate the temporal layout, orchestrating the presentation of the data. Basic temporal structures produce serial and parallel (hierarchic model [33]) presentations; although simple presentations can also be defined by specifying a start time and duration (timeline model [12]). These, and other models may be implemented using scripting languages, specification languages, and extended programming languages. While specification languages may be mapped to temporal relationships within ODBMSs, 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 ODBMSs.

More advanced temporal models, such as [5, 41, 52], 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, ODBMSs could become the infrastructure for a distributed interactive multimedia environment supporting multiple users, guided instruction, collaboration, and interactive virtual worlds.

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, quality-of-service delivery and presentation constraints.

2.3.3 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 details can be abstracted away from the user, and allow the entire process (capture, storage and presentation) to be represented within the data model.

2.4 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 ODBMSs acts as a simple repository, then these features may be provided completely by the client application.

2.4.1 User Interaction

User interaction support is critical for multimedia applications. The first is sophisticated graphical user interfaces. With spatial and temporal rendering required, the interface design capabilities must be feature-rich. An example is the ability to control the presentation, and the devices which render 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 ODBMSs should support multiple view definitions of the data, 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.

A second part of user interaction is the handling of user events. The temporal data model may implement the event model, but there must be underlying support within the ODBMSs for it. This support can be provided by features similar to the change notification features found in active ODBMSs today, e.g. ITASCA [23], exemplified by the event manager in DAMSEL

[41], or by interaction manager in Vodak [49]. To support advanced multi-user collaboration and video interaction, the systems must provide soft real-time multithreaded functionality. As the user may be interacting with several different windows and data types, the system should support different threads of control [28] within each user process.

2.4.2 Data Manipulation

One of the purposes of a DBMS is to provide the ability to modify data stored within it. For simple data repositories, support for data modification isn't required. In a well-developed multimedia framework, external applications, or hooks to applications can provide the editing capabilities required (see section 2.3.1). However, if modifications are performed within the ODBMSs, 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., [41]). Multimedia DMLs support data flow, indicating sources, sinks, and operations in between to analyze and modify the data. However, it is not enough to provide simple cut or crop operations; ideally these operations must adhere to real-time 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).

To reduce the need to store new versions of data, the ODBMSs may support storing the operations to generate the new data, rather than storing the new data itself – called "derived data" – implemented, for example, in [46]. When derived data is accessed, the operation(s) is applied to the current state of the data to produce the new version.

2.4.3 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 is constructed and delivered to users as an end product, dynamic schema modification may not be required. However, 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 database servers developed using interpreted languages such as LISP, CLOS, Smalltalk and Java.

2.4.4 Querying

Once data has been stored within a database, retrieval is usually performed by posing queries against it, containing predicates which must be satisfied by any data that is retrieved. The predicates usually involve partial or exact matches. But how can multimedia data be queried?

For simple multimedia applications and ODBMSs products, the most straight- forward means to query multimedia data is to define keywords (meta-data) associated with the data, also known as offline indexing. The use of a standardized keyword dictionary [26] 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 online. Some problems with a keyword approach do exist. First, keyword classification is subjective when 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. 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 5,000 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 used. A second approach that may be supported in MMDBMS is content-based retrieval or 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. The results of the analysis may be keywords, or multi-dimensional indexing structures which describe the data [15, 38, 57]. Queries are still processed on the meta-data; however, this meta-data attempts to maximize the answer quality 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.

Video is a sequence of images, so image query techniques may also apply. However, video also has the temporal dimension, so we may want to find specific scenes, scene changes, people eating, running, shooting, 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 as "static content-based retrieval", or SCBR. The notion of analyzing video and detecting actions is far from reality at this point in time.

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 researched [7, 10, 56], and to a limited extent are supported within some current DBMS for simple data types. A very good description of retrieval mechanisms and multi-layered indexing techniques can be found in [56].

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 [8, 13]. The query manager can then use the example to find objects in the database similar to it. This approach is called query-by-example (QBE), and some projects have already implemented static QBE in research systems for image retrieval (e.g. [13]). 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.

Querying heterogeneous 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. 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 [44], so the repeated compression and decompression of media data using them will deteriorate data quality.

3 Infrastructure Issues

3.1 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.

An important observation is that, to fulfill the needs of multimedia applications, an ODBMS must get the requisite help from the underlying infrastructure mechanisms; most notably the operating system.

Temporal data could be delivered all at once, requiring the users to wait for all the data to arrive – the approach taken with a data repository. 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 ODBMSs will have to regulate the delivery of the data, frame by frame, to the user. Current ODBMSs do not provide this 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 ODBMSs to manage access to the disk, negotiate delivery schedules, and set user-process priority levels to ensure that quality of service requirements 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. Integrated solutions such as the research discussed in [22] focus on providing multi-layered specification and distributed resource management for continuous data.

In ODBMSs, data are generally placed on pages wherever space is available. Due to the temporal constraints of continuous media, more rigorous placement algorithms will be required [32]. Placement should consider 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 [16]. These issues, described in greater detail in [31], are important and will directly affect the system's throughput.

To further 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 (VBR) streams. Techniques for their storgae and retrieval are presented in [29]. Other kinds of data, including audio, compress predictably, and therefore produce constant bit rate (CBR) streams which are much easier to handle [30].

3.1.1 Centralized Architecture

With the client and server colocated, there are no network concerns with respect to data retrieval. However, considering the size of multimedia databases, it is very likely that tertiary storage will be used. Given the sequential nature of most tertiary storage devices, new techniques must be developed to handle real-time retrieval from them. Further, an added level of storage hierarchy introduces new data transfer, buffering and caching problems [55].

3.1.2 Distributed Client-Server Architecture

In a distributed client-server configuration, since the data must be delivered to the client machine, the network may become a bottleneck. Current network speeds available through the internet and local ethernet for example, are not sufficient for high quality video and high fidelity audio. In addition, the protocols cannot support time-constrained delivery of data, nor do they support QoS parameters – so new protocols will be needed [9, 37, 39]. 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 ODBMSs will have to provide device transparency and device sharing [28]. To meet the bandwidth requirements for video, high speed networks that can understand QoS parameters are necessary. Advanced MMDBMS will be connected to clients using high speed communications to ensure that QoS requests can be met.

3.2 Operating Systems

ODBMSs that provide real-time access to video data will require multimedia-capable operating systems underneath. Many of the metrics only provide a portion of the solution; and hence, real-time operating systems can be useful only when other components, such as the ODBMSs software and network protocols, can work in an integrated fashion to satisfy all of the various constraints associated with multimedia capture and presentation. In a single-user environment with a relatively fast machine, a real-time operating system may not be required. In multi-user systems, the operating systems should 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 OS buffer management strategies [27, 35] will be necessary. 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 and criticality [22].

4 Related Issues

4.1 Internet Access

More people accessing the internet it makes it an attractive place to offer access to databases. 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 continuous data, 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 [8], which may be useful for some applications.

4.2 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)*** [19]. It focuses on many issues related to multimedia data type support, synchronization, and programming. Standards compliance is important for the portability of data between products, such as the ISO SGML/HyTime multimedia interchange standard [36]. As MMDBMS become a reality, standards like PREMO and SGML will play an integral role.

4.3 Other Issues

There are, of course, other issues pertaining to ODBMSs that are of interest. For example, what distribution capabilities will be supported. Particularly with the advent of widely distributed database systems, what architectures will be used (e.g. CORBA). What level of authorizations will be supported within each ODBMSs? 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, parts of conversations must be blocked out. History-based authorization, which restricts future accesses based upon previous ones, may also be used. Finally, as multimedia objects consume several pages, the competition between object and page servers may no longer be an issue.

5 Current Efforts

Based on the requirements described in this paper, we are currently in the process of evaluating a number of ODBMS for multimedia support. In most systems, the necessary software is not supplied, limiting the databases to primarily act as data repositories. 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 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.

Image data can be stored within most of these systems, at least as BLObs; but, since video data is so large, most systems cannot handle it even as BLObs. 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.

At this point our evaluation is ongoing, so we have chosen not to draw any firm conclusions about the suitability of ODBMSs for multimedia data. Detailed evaluations and conclusions will be presented in a later report. Due to space limitations our preliminary results cannot be published, but they have been made available on the internet [1].

6 Summary

This report has described several metrics for multimedia object database management systems. It has outlined a breadth of factors 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. The full paper with references is available at our WWW site [1].

References

[1] Pazandak, P., J. Srivastava and J. Carlis, "A Report on Metrics for Evaluating ODBMS Functionality for MMDBMS", Computer Science Department Technical Report, University of Minnesota 1996. Available at http://www.cs.umn.edu/~pazandak/mmdbms96-paper.html. ***pazandak@cs.umn.edu***