Multimedia Data Storage and Representation Issues on Tertiary Storage Subsystems : An Overview

Athena Vakali and Evimaria Terzi
Department of Informatics
Aristotle University
54006 Thessaloniki, Greece
{avakali, evimaria}@csd.auth.gr

Abstract

Storage of multimedia data is a critical issue for the overall system's performance and functionality. Multimedia applications must be able to support a variety of data defined as *multimedia objects.* Multimedia data consist of various objects such as text, image, video and graphics, which need to be synchronized and meet certain timing requirements. The development and evolution of these new applications characterized by high storage needs resulted in strengthening the role and importance of Tertiary Storage Systems. Recent technological advances have resulted in wide availability of commercial products offering near-line, robot-based, tertiary storage libraries. Therefore, such libraries have become an important component of modern large-scale storage servers. The issue of optimal data placement strategies in tertiary storage is considered below. Different configurations of tape libraries and magnetic tape technologies as well as the special attributes of stored data have an impact on the optimal placement.

1 Introduction - Research Review

Multimedia objects physical storage is a demanding and challenging problem due to multimedia objects two principal constraints, namely size and timing. Representation models for multimedia data have been devised in order to represent the temporal relations among objects and specify the times at which discrete events occur. In [1, 2] a classification of the representation models, based on the notion of time is presented. Every multimedia object is consisted of a number of physical stored objects with static and dynamic properties. The multimedia object representation model (and all of its components) depends on several issues such as the specific research approach, the implementation product and the available means and experiences. A number of different approaches have been also introduced in [1, 2, 8]. These, can be classified into three main categories, *Graph Models*, *Petri-Net models* and *Object-Oriented Models.*

The multimedia applications are quite demanding in terms of not only processor speeds but storage availability as well. Since new modern applications require enormous amounts of information, they demand storage systems with high capacity and service abilities. According to current technological trends, the amount of storage sold has been almost doubling each year. The rapid acceleration and growth of storage requirements and the higher cost of secondary memory have leaded systems designers to re-consider and re-estimate the adaptation of tertiary storage systems into the overall storage hierarchy, as long as they are an inexpensive and practical solution to the storage need problem. Although tertiary storage storage systems are inexpensive, they are slow, and the improvement of their performance is a major research issue. The interaction between Secondary and Tertiary storage levels proves to be quite useful towards the improvement of the system's functionality. Utilization of Tertiary Storage towards augmenting the system's performance is studied in [19] where continuous data are elevated from their permanent place in tertiary to the secondary storage.

In [15] the current state of the art in tertiary storage systems is discussed, tertiary system types are classified and performance metrics and figures are provided. Research reported in [6, 7] evaluate and discuss storage hierarchies and the usefulness of current tertiary storage systems for several new types of applications. More specifically, in [6] the design and performance of specific tertiary storage products are analyzed and evaluated under specific workloads. The physical structure of storage subsystems is described in [10] and the flow of data through the system is traced in order to provide guidelines for storage placement within the hierarchy. It is interesting to note that tertiary systems have different and diverse performance factors not applicable to all technologies. For example some tertiary storage technologies have seek time as the dominating factor, whereas on other tertiary systems switch time dominates the overall system performance, as indicated in [15, 16]. In [11] a serpentine tape drive is studied under model-driven simulation in order to estimate the locate time accurately.

Different scheduling algorithms have been applied on various tape libraries configurations in order to suggest improvement in the performance of tertiary storage media. Research efforts on Tertiary Storage I/O scheduling have been related to specific technologies.More specifically scheduling is discussed in [12] where efficient random I/O scheduling for serpentine track layout, show that suggested scheduling schemes provide a significant improvement in the latency of random access to the tape. The problem of scheduling I/O requests for tertiary storage libraries is studied in [16].

Also, information positioning across tertiary media areas affects the overall device performance. In secondary memory devices several policies have been proposed regarding positioning the most often requested data around the central of the disk's surface (organ-pipe) or around the middle of the two intervals derived by the central disk cylinder (camel). The data placement problem has been also studied for Tertiary Storage Systems as well. In [7] different data placement policies on Tape Libraries consisting of different technology tapes implemented, while in [17] optimal arrangement of cartridges and file-partitioning schemes are examined in Carousel type Mass Storage Systems where organ-pipe arrangement is applied an cartridges and it is shown to be optimal for both anticipatory and non-anticipatory schemes. Finally, in [20] information placement schemes are considered in three different models corresponding to three mid-range magnetic tape systems.

2 Multimedia Object Represenation

2.1 An Overview

Since timing is the most important factor in these models are classified in ([1, 2]) into three board categories : *timeline models* , *interval-based models* and *constrained-based models*.

Timeline models are instant-based models, based on the notion of *timeline*. In these models temporal requirements of objects are defined by specifying the starting and the ending time of each object composing the presentation. Both times are expressed in absolute time.Figure 1 depicts a timeline model.

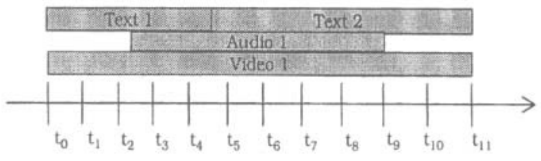
Text 1
Text 2
Audio 1
Video 1
t_0 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11}

Figure 1: A timeline model.

Interval-based models use *time interval* as primitive. An interval is a sequence of consecutive time instants usually represented as $[t_s, t_e]$ where $t_s \leq t_e$. In this model each object has an associated interval, representing the time required for its presentation. Synchronizatiion requirements are specified by relating the duration of the objects in an appropriate way. Allen's 13 (9 parallel and 4 sequential) temporal relations shown in Figure 2 are thought to be the basic interval relations.

In **constraint-based models** synchronization requirements are expressed as a set of *constraints*,in the form $event_1 = event_2 + delay$, defining the temporal relations of the objects during presentation. The above constraint denotes that $event_1$ must occur *delay* time units after $event_2$

A different classification of multimedia data representation models based on the conceptual view of the whole application is presented below. In this case there are three main categories too:

- **Graph Models :** In Graph models, the main idea is to represent a temporal

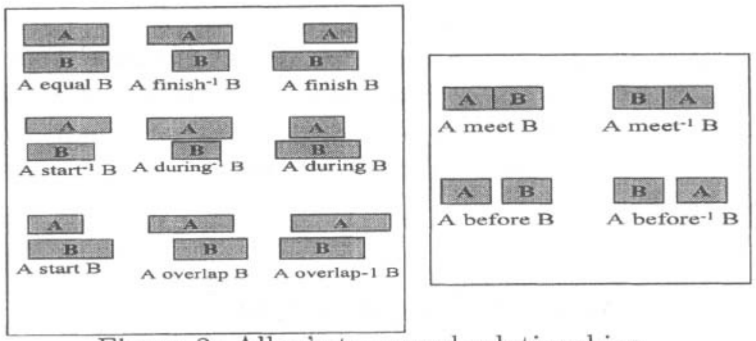

Figure 2: Allen's temporal relationships

scenario as a graph whose nodes denote the events/objects composing a multimedia scenario and the edges capture the temporal relationships among them.

- **Petri-Net models :** In Petri-Net models, the basic idea is to represent physical objects as places in the net and their temporal relations as transitions. Timed and augmented Petri-Nets have *traditionally* been used to analyze systems with temporal requirements (e.g. real-time systems) and are characterized by desirable attributes of representation of concurrent and asynchronous events.

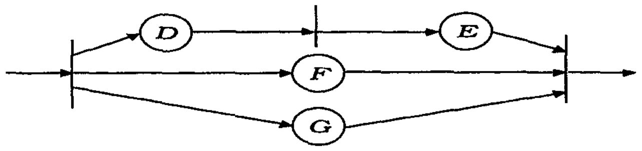

Figure 3: A Petri-Net Example

The Petri-Net is defined as a bipartite, directed graph $N = (T, P, A)$ [14] where :

$$T = \{t_1, \ldots, t_n\}$$

$$P = (p_1, \ldots, p_n)$$

$$A : \{T \times P\} \cup \{P \times T\} \rightarrow I,$$

$$I = \{1, 2, \cdots\},$$

where T, P, A represent a set of *transitions* (bars), *places* (circles) and a set of directed arcs, respectively. An example is given in Figure 3

Object Oriented Petri Net (OCPN) augments the conventional Petri net model with values of time, as durations, and resource utilization on the places in the net; therefore, an OCPN is defined as $C_{OCPN} = \{T, P, A, D, R, M\}$ where additionally,

$$D : P \rightarrow R$$

$$R : P \rightarrow \{t_1, \ldots t_n\}$$

where D and R are mappings from the set of places to the real numbers (*durations*), and from the set of places to the set of *resources* respectively.

- **Object-Oriented models :** Under an Object-Oriented approach, a multimedia object is modeled as a set of objects related to each other in several ways. Temporal information for synchronizing multimedia and physical objects are modeled by means of object attributes. Moreover, the notion of inheritance can be used, to define a class containing general methods for the sunchronization of objects. The different forms of synchronization required to model a given multimedia scenario or object are obtained by specializing this class. Furthermore, composite objects are a powerful tool to model synchronization requirements of a temporal scenario. A temporal scenario or a multimedia object can be modeled as a composite object, whose components are the physical objects composing the scenario. Attributes and methods of the object are used to express synchronization requirements.

3 Tertiary Storage Systems : An Overview

On top of the computer storage hierarchy is primary storage i.e RAM used for caches and main memory. Below RAM is solid state memory and then magnetic disk devices, commonly identified as secondary storage. At the bottom of the hierarchy are tertiary storage devices. Figure 4 displays a typical storage hierarchy of a computer system comprising of all three levels of storage systems. Tertiary storage includes magnetic tapes, optical disk devices and some ore recent technologies like optical tapes and holographic storage. From top to bottom of this hierarchy average access times increase, while the cost per megabyte of storage is dramatically decreased. Therefore, tertiary storage is inexpenseive but slow.

However, recent advances in tertiary storage technology have made it of increasing interest to computer system designers. First, increases in bits-per-inch and tracks-per-inch densities have increased tape capability. Second, a variety of inexpensive tape drives have become available. Next the low cost of magnetic tape media, compared to that of magnetic disks, makes it economical to built massive tertiary storage systems. Moreover, a large number of robotic devices for handling magnetic tapes, allow access to tertiary storage without human intervention, making response times more predictable.

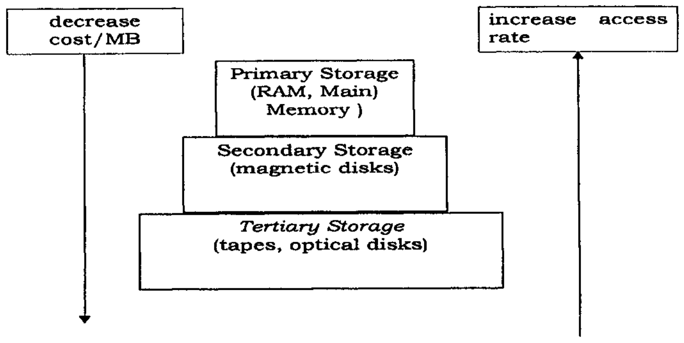
decrease cost/MB
increase access rate
Primary Storage (RAM, Main) Memory)
Secondary Storage (magnetic disks)
Tertiary Storage (tapes, optical disks)

Figure 4: The Storage Levels Hierarchy

Finally, increases in computer processing speed have enabled a growing number of applications, ranging from climate modelling and geographic information to digital libraries and multimedia servers, that would benefit from fast access to a massive storage system. However tertiary remains 3 to 4 order of magnitude slower than disks.

3.1 Tertiary Storage Topologies

The most common tertiary storage subsystems include *Magnetic Tapes* and *Tape Libraries*. Since these technologies are most widely adopted they are studied in the present paper and supported by the respective models. The most important characteristics of these technologies are:

- **Magnetic Tapes :** Magnetic tapes have been the most dominant type of tertiary storage media. Magnetic tape drives store data as small magnetized regions on a tape composed of magnetic material deposited on a thin, flexible substrate such as plastic. The thickness of the substrate determines tape's reliability and capacity. Most tapes are append-only. Update-in-place magnetic tape drives do exist, but these are not very popular as they succeed lower storage capacity.

  Magnetic tape technology includes mainly two classes of tape types:the Serpentine and Helical-scan technology. A serpentine tape drive records a track (or group of tracks) down the length of the tape, continuing back and forth. Serpentine tapes survive thousands of hours of intensive use. The Helical-scan drive uses rotary head technology, like that of home video recorder, writing diagonal tracks during a single slow pass over the tape. Their advantages are high data density, high speed transport mode, random I/O easily scheduled while their main disadvantage is that tape wears out under intensive random I/O.

- **Tape Libraries :** Several tertiary storage manufacturers have built automated library systems in order to provide higher bandwidth and capacity. These libraries hold large number of cartridges that can be loaded by robot arms into a collection of magnetic tape drives. Robot access times vary from a few seconds to about 20 seconds for most libraries. The most common classification of robotic devices include: *Large Libraries* which contain many cartridges, several drives and one or two robot arms handling cartridges which are often arranged in rectangular arrays. These libraries are usually expensive. *Carousel Devices* contain up to 100 cartridges, one or two drives and one(carousel) robot arm. The carousel rotates to position the cartridge over a drive and a robot arm loads the cartridge into the drive. These devices are moderately priced. *Stacker Devices* contain approximately 10 cartridges, one drive and one robot arm or magazine. A rack of tapes is moved vertically or horizontally to bring the desired tape onto a single drive, ot he stacker may have a robot arm that moves across the magazine to pick a cartridge. This type of devices is the least expensive of all the above robotic technologies.

3.2 Tape Models

There are two main models proposed for magnetic tapes. The first one considers serpentine tapes only. A serpentine tape drive records a track (or group of tracks) down the length of the tape, then reverses direction and shifts the heads sideways a small distance to record another track (or group) up the length of the tape, continuing back and forth until tens of track groups have been written. Each track contains a certain number of sections. The smallest accessible part of the tape is the *segment*, which is a fixed-sized chunk (32 KB is a acceptable size). This model have been used in [11, 18].

On the other hand, in [20, 7] the tape is thought to be a linear area divided into a specific number of fixed sized partitions. Moreover the tapes are divided into two main categories :

1. *Tapes that Rewind to the PBOT:* These are tapes that always rewind to their beginning before being ejected.

2. *Tapes that Rewind to the nearest Zone:* These tapes enable tape rewinding to the nearest zone as opposed to those require to be rewound to their physical beginning before being ejected.

Our approach is a combination of the two approaches described above. Tapes are thought to be linear areas. Each linear area is divided into a certain number of fixed-size *segments*, which are the smallest accessible parts of the tape. *Sections* consist of a number of consequent segments, while *tracks* consist of a number of consequent sections. The number of segments that will be allocated to a data object mainly depends on the object's and the segment's size. More specifically the number of segments s devoted to a single object is given by the equation

$$s = \left\lceil \frac{size\ of\ object}{size\ of\ a\ segment} \right\rceil$$

If the object's size is not divided by the segment's size, then the last segment allocated to an object is not full, and therefore some space remains empty. However, the amount of this space is relatively small when compared to the capacity of the tape and the total size of objects that can be stored on it.

3.3 Tertiary Library Models

Tertiary Storage Libraries come in many different sizes and configurations. Despite their significant differences, they all contain the following components:

- Drives
- Robot Arms
- Tapes or Disks

Consequently, they can all be modeled very similarly. The library is thought to have one or more robot arms, which are assumed to be identical. Each arm is assumed to be capable of moving between any tape stored in the library. There are three operations the robot arms make: **Pick, Put** and **Move**. A pick operation refers to the robot picking up a tape from its storage space in the library and taking it into the drive. A move operation refers to the robot moving between different locations on the shelf. Although there is variation in the times required for these operations, they are generally modeled as constant times. This is acceptable because the variation is much smaller than the total time required to exchange tapes or disks on drives. Consequently:

$$Robot\_Arm\_Service\_Time = Pick\_Time + Move\_Time + Put\_Time$$

The drives, which are also assumed to be identical, perform the following operations: seek, rewind, read, write, load and eject. The seek and rewind operations for tapes are modeled as constant startup times followed by a constant transfer rate. Therefore the tape access time is defined as:

$$\begin{aligned} Drive\_service\_time &= Rewind\_Time + Eject\_Time \\ &+ Load\_Time + Seek\_Time + Transfer\_Time \end{aligned}$$

Therefore the total access time for a tape operation which includes a tape switch operation is defined as follows:

$$Total\_Service\_Time = Robot\_Arm\_Service\_Time + Drive\_service\_time$$

A typical example of a tape library with four drives and 110 tapes is presented in Figure 5.

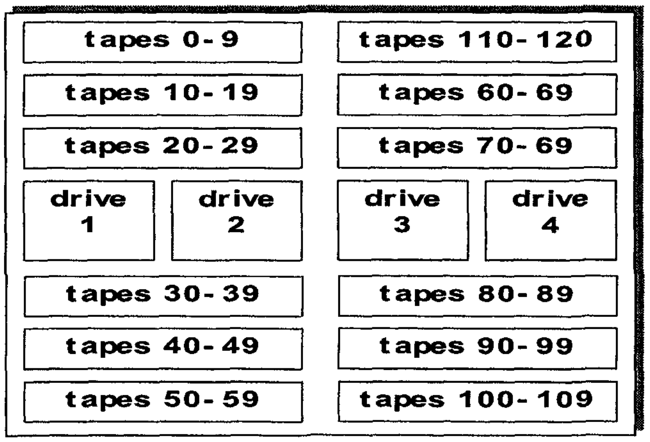

Figure 5: A Tape Library Model

4 Multimedia Data Placement Policies on Tertiary Storage Subsystems

The information placement policy is an important factor influencing the actual's system performance. The choice of the placement policy depends on the special nature of data that are going to be placed and on the attributes of the media these data are going to be placed on. We have considered Multimedia Data stored on magnetic tapes organized in a tape library.

4.1 Multimedia Data Constraints

As it has already been mentioned, multimedia applications are different because they must be able to support a set of multimedia objects. Multimedia data consist of various objects such as text, image, video and graphics, which need to be synchronized and meet certain timing requirements. Multimedia objects are composed of physical objects (i.e. stored images, text, audio etc), characterized by timing and synchronization relations need to be maintained during their playout. In our case of study, we have a pool of physical objects that should be placed in the right way so as to guarantee high quality of multimedia objects' presentation. Given the relative frequency of access of each multimedia node we must specify the relative frequency of access of each physical object of the pool independently. The problem that arises is that a physical object may be part of both popular and unpopular nodes. Therefore, a pattern should be found in order to specify which physical objects are popular and which are not. Furthermore, in a second level, we must take into consideration the synchronization conststraints between objects that should be played together within a multimedia node, or in other words we should specify which objects are found together in the existing multimedia nodes, and how many times this co-existence occurs. The combination of these two criteria can determine which physical objects are "hotter" than others and thus, imply the way they should be stored.

In our case the popularity of each physical object is determined as follows: Given the relative frequency of access s_i of each multimedia node n_i a measure of each physical object's frequency of access in a multimedia application consisting of N nodes can be :

$$p = \sum_{i=0}^{N-1} s_i \times (number\ of\ objects\ playouts\ in\ n_i)$$

It is obvious that the higher the frequency of access of a physical object the more popular this object is.

Sorting the physical objects of the pool according to this parameter and then store them implementing placement algorithms based on "popularities" is expected to perform well. This can be explained as follows: Parameter p is higher when an object is part of a popular node and when it should be played for several times within this node. Therefore all objects that are contained in popular nodes have high relative frequency of access. Furthermore, these objects are highly possible to co-exist in these nodes. Thus, this parameter is enough to determine the storage policy that should be followed as both "popularity" of physical objects is estimated and timing constraints between them are satisfied.

However, one more criterion can optionally been adopted. After the physical objects being sorted according to their frequency of access as described above, we can take the K most popular of them. The rest should be sorted according to the number of times each of them co-exists with the first K, in the whole multimedia application. The implementation of this criterion can guarantee that objects more closely related with the first K popular ones, will be placed in the right position on the available magnetic tapes. An issue that is inevitably arisen in the latter case, is the value of parameter K. As long as our storage system is a tape library, this parameter can be related to the number of the drives of the library. Small libraries, known as *stackers* have only one drive, while larger tape libraries have more drives which can be used for parallel reading of data from different tapes.

4.2 Tertiary Storage Devices Constraints

A description of the commercially available automatic tape and disk libraries or jukeboxes that provide automated access to large amount of tertiary storage seems to be necessary. These libraries can hold hundreds or thousands of media. Typically they consist of a small number of drives or readers and a few robot arms that load and unload media to and from these drives. Upon receiving a request, the robots will load the desired medium onto one of the drives (if there is already a medium loaded on the drive-it may need to be rewound in the case of tapes, ejected and replaced onto the shelf). The drive then loads the new medium, seeks to the starting location of requested data and then transfers them. The time required to switch a medium on a drive is usually of the order of few seconds, and the time required to seek an entire medium may be as high as a few minutes. When more than one drives are present they operate independently of each other, except that they share the robot arms to pick and put the media [15, 16].

When data are placed on magnetic disks it is sufficient to accumulate as big a probability as possible onto the first disk, then onto the second and so on. This is because the cost of seeking within a disk is less than that of switching from one disk to another. However, in the robotic library case, seek and switching costs are two contradictory factors and therefore both placement and retrieval of objects must be studied in detail in order to achieve as good performance as possible.

4.3 The Placement Algorithms

Having taken into consideration the special nature of Multimedia Objects and Tertiary Storage Devices the problem that remains to be solved has to do with the placement of O objects onto T tapes. Given the results of [7] the placement algorithms used for objects' storage on tapes that rewind to the PBOT and those that rewind to the nearest Zone, modified in order to satisfy the particular problem's constraints, are these described below. Note that our objects have both variable access probability-popularities and variable size.

Placement Algorithm (1) for Tapes that Rewind to the PBOT

```
Sort the Objects in decreasing popularity order i=1

while there are still Unallocated Objects and free Tape Space
begin
    place the i-th object in (i mod T) tape starting at
                   the first available segment
    if (the object is placed)
        i++
    else if (the space is not enough)
        try to place it on the first tape with adequate free space
        if (the object is placed)
            i++
        else  {there is no tape with enough space available;
            i-th object placement failed
end
```

Placement Algorithm (2) for Tapes that Rewind to nearest Zone

```
Sort the Objects in decreasing popularity order i=1

while there are still Unallocated Objects and free Tape Space
begin
    place the i-th object in (i mod T) tape starting at
               its middle segment and continue as determined by
               the organ-pipe policy.
    if (the object is placed)
        i++
    else if (the space is not enough)
        try to place it on the first tape with adequate free space
        if (the object is placed)
            i++
         else  {there is no tape with enough space available}
            i-th object placement failed
```

Except for the above placement policies, random placement policies can also be obtained if we do not sort the objects with concern to their "popularity", as it has been already defined.

4.4 Multimedia Data Timeline Object-Oriented Model : An example

The model consists of four main classes :

- **Class PhysicalObject** refers to the physical objects that compose multimedia objects. Each physical object of the application is an instance of this class. In our case we have a "pool" of physical objects, each of which may belong to more than one multimedia object.

- **Class MM-node** refers to independent to Multimedia Objects. An independent multimedia object is considered irrespectively to its relations with the other multimedia objects of the application. These objects may contain a number of Physical Objects with certain temporal relations among them. Every object of the MM-node class has its own time system. The starting and ending times of the playout

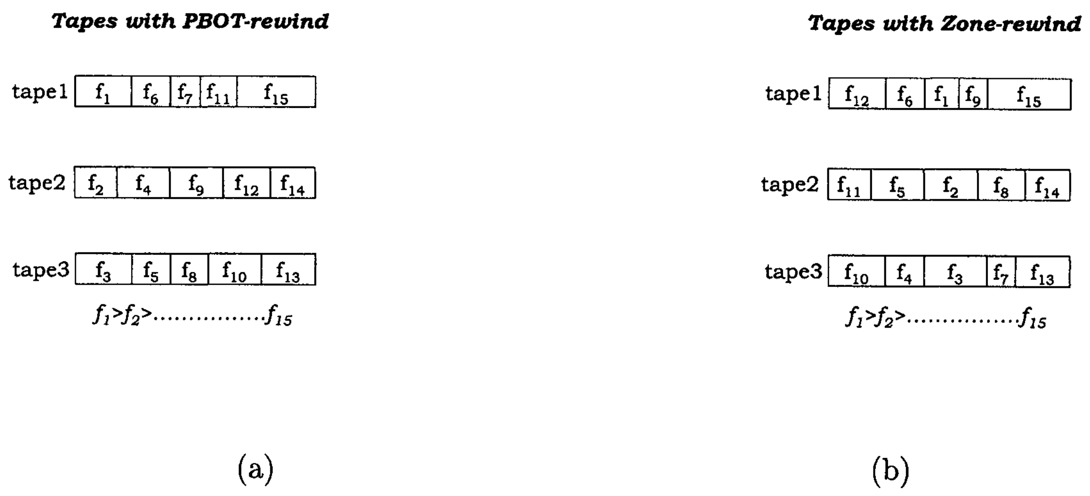

Figure 6: Optimal Placement Schemes:(a) Tapes with Zone-rewind. (b) Tapes with PBOT-rewind .

of the physical objects of the node are determined with respect to the latter's absolute time.

- **Class graphNode** refers to Multimedia Objects too, but it treats them as a part of the whole multimedia application. The nodes of the browsing graph described above, could be instances of that class.

- **Class node** refers to multimedia objects of the application. As long as multimedia objects are not completely independent, the need for a new class that inherits the attributes and methods of both MM-node and graphNode classes is arisen. Therefore, the multimedia objects are instances of neither MM-node class nor graphNode class, but they are instances of the composed node class. Considering a multimedia application with more than one multimedia object, the classes MM-node and graphNode will not be instantiated.

The hierarchy of the classes described is presented in Figure 7 . Class *node* inherits all the attributes and methods of both graphNode and MM-node classes. The $\triangle$ symbol between the *MM-node* and *physicalObject* classes denotes that the multimedia node may contain a number of physical objects.

The main factor of the model is time, and in particular the absolute time of a specific Multimedia Object. Once such an object should be displayed its time mechanism gets into action. As time passes the physical objects that should be played are fired for a time interval equal to their duration and then they are stopped until their next starting

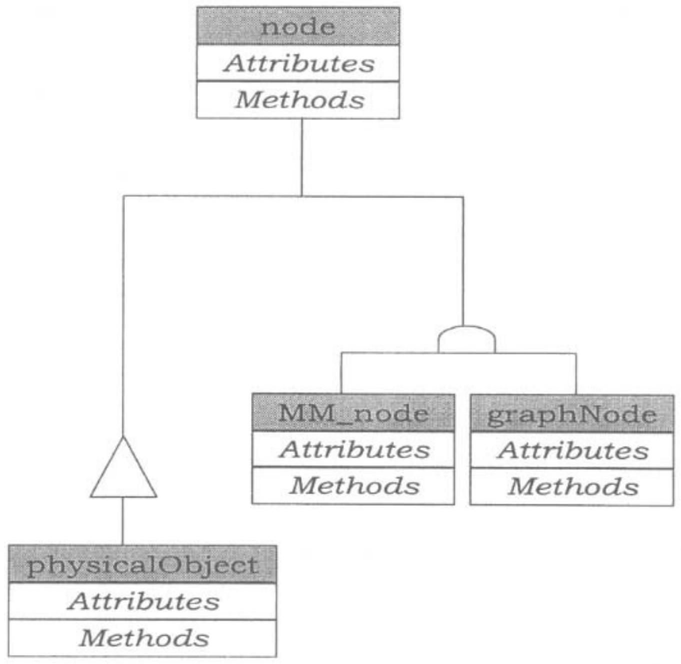

Figure 7: Classes' Hierarchy of the Timeline Object-Oriented Model

time, if they are displayed more than once within a single node. No time dependencies between physical objects belonging in the same multimedia objects are specified.

The example of Figure 8 will be used as a reference to view on how the timeline object-oriented model described above represents a multimedia object and the temporal constraints among the physical objects it contains. We elaborate on the micro level of a multimedia object, i.e. we explain the support of each node of the browsing graph. The example refers to an independent multimedia object which includes objects $0, 1, \ldots, 4$ with the temporal relations given in Figure 8. The objects identity and duration are given in Table 1. The starting times of the above objects are stored in dynamic arrays. Every object has its own array, named *playTimes*, which is dynamically created. The number of elements of an object's array equals to the number of the latter's discrete playouts. Therefore, if the physical objects 0, 1, 2, 3, and 4 are played 3, 3, 2, 1, 1 times respectively, during the display of a multimedia node, and their corresponding playTimes arrays will be those shown in figure 9.

The presence of a physical object at a certain time of the multimedia object is identified by the parameter *status.* For example, the state of the objects at time 25 (with respect to the absolute multimedia object's time) is that of Table 2. Therefore, at each time we know of which objects participate in the multimedia presentation.

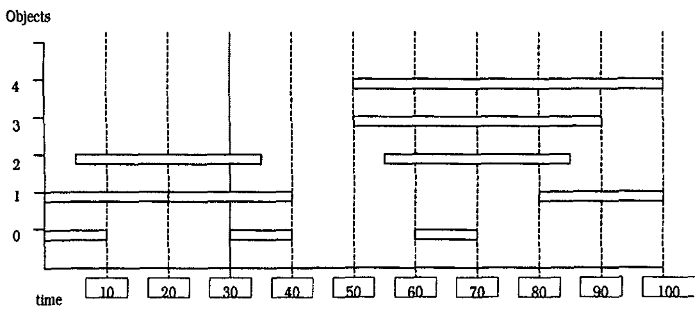

Figure 8: Timeline Object-Oriented Representation

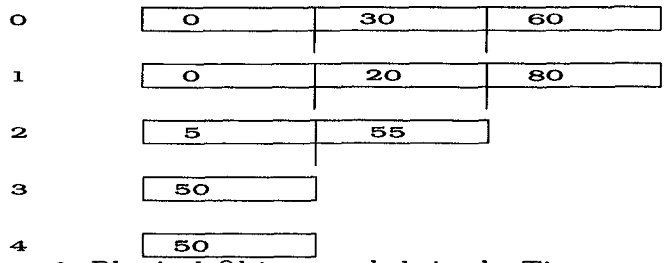

Figure 9: Physical Objects and their playTimes arrays

5 Conclusion

The multimedia data placement is a major research topic due to the emerging current applications which impose strong capacity and synchronization requirements. Several multimedia data representation models are discussed in relation to effective data placement policies. An indicative example of a timeline object-oriented model is further analyzed and discussed. The models described are based on the approach which favors the most frequently requested multimedia data and their related objects with respect to timing constraints. Most frequently accessed objects and their related objects are proposed to be stored effectively on the considered Tertiary Storage subsystem.

References

[1] Elisa Bertino and Elena Ferrari : Temporal Synchronization Models for Multimedia Data, *IEEE Transactions on Knowledge and Data Engineering*, Vol.10, No.4, 1998.

Object id	Duration
0	10
1	20
2	30
3	40
4	50

Table 1: The physical objects of a single node.

Object id	status
1	0
1	1
2	1
3	0
4	0

Table 2: Node's state at time 25

[2] Yong-Moo Kwon,Elena Ferrari, Elisa Bertino : Modeling spatio-temporal constraints for multimedia objects, *Knowledge and Data Engineering 30*, pp.217-238, 1999.

[3] Milind Buddhikot, and Gurudatta, Parulkar :Load Balance Properties of Distributed Data Layouts for Clustered Multimedia Storage Servers, Department of Computer Science *Technical Report, WUCS-95-32*, Washington University in St. Louis, 1995.

[4] Milind Buddhikot, et al. : Storage Hierarchies and Video Servers,*Handbook of Multimedia Information Management*, Prentice Hall, pp. 279-233, 1997.

[5] Yie-Tarng Chen : Physical storage model for interactive multimedia information systems, PhD Thesis, Department of Electrical Engineering, Purdue University, 1993.

[6] A.L. Chervenak: Tertiary Storage-An Elevation of New Applications, PhD Dissertation, University of California at Berkley, 1994.

[7] Stavros Christodoulakis, Peter Triantafillou and Fenia Zioga : Principles of Optimally Placing Data in Tertiary Storage Libraries, *23rd International Conference of Very Large Databases (VLB) Athens 1997*, pp.236-245.

[8] Soon M. Chung : *Multimedia Information Storage and Management*, Kluwer Academic Publishers.

[9] A.A. Ford and S. Christodoulakis : Optimal Placement of High probability randomly retrieved blocks on CLV Optical Disks, *ACM Transactions on Information Systems*, Vol. 9, No.1, pp.1-30, 1991.

[10] J. Gemmel and S. Christodoulakis : Principles of Delay-Sensitive Multimedia Data Storage and Retrieval, *ACM Transactions on Information Systems*, Vol. 10, No.1, pp.51-90, 1992.

[11] B.K.Hillyer and A. Silberschatz: On the Modeling and Performance Characteristics of a Serpentine Tape, *Proceedings ACM SIGMOD'96 Conference*, pp.170-179, 1996.

[12] B.K.Hillyer and A. Silberschatz: Random I/O Scheduling in Online Tertiary Storage Systems *Proceedings ACM SIGMOD'96 Conference*, pp.195-204, 1996.

[13] S. H. Kim, S. Ghandeharizadeh:Design of Multi-user Editing Servers for Continuous Media,*Proceedings of the RIDE' 98 Conference* , 1998.

[14] T.D.C. Little and A. Ghafoor :Synchronization and Storage Models for Multimedia Object, *IEEE Journal on Selected Areas in Communication*, Vol. 8, No.3, pp.413-427, 1990.

[15] Sunil Prabhakar, Divyakant Agrawal, Amr El Abbadi, Ambuj Singh: Tertiary Storage: Current Status and Future Trends, Computer Science Department, University of California, Santa Barbara, August 1996.

[16] Sunil Prabhakar, Divyakant Agrawal, Amr El Abbadi: Impact of Media Exchanges in Robotic Libraries,Computer Science Department, University of California, Santa Barbara, June 1997.

[17] S.Sesardi, D. Rotem and A. Segev : Optimal Arrangements of Cartridges in Carousel Type Mass Storage Systems, *The Computer Journal*, Vol.37, No.10, pp.873-887, 1994.

[18] B.K.Hillyer and A. Silberschatz: On the Modeling and Performance Characteristics of a Serpentine Tape *Proceedings SIGNMETRICS'96 Conference*, pp.170-179, 1996.

[19] P. Triantafillou, T. Papadakis : On Demand Data Elevation in a Hierarchical Multimedia Storage Server,*Proceedings of the Very Large Databases VLDB'97 Conference* , 1997.

[20] A.Vakali and Y.Manolopoulos: Information Placement Policies in Tertiary Storage Systems, Storage Models for Multimedia Object, *Proceedings of the Hellenic Conference of New Information technologies*, pp.205-214, Oct 1998.