Storage and Retrieval for Image and Video Databases II, IS&T/SPIE Symp. on Elec. Imaging Sci. & Tech., San Jose, CA, February 1994.

A Distributed Hierarchical Storage Manager for a Video-on-Demand System

Craig Federighi and Lawrence A. Rowe
Computer Science Division - EECS
University of California
Berkeley, CA 94720
(craigf@cs.berkeley.edu / rowe@cs.berkeley.edu)

Abstract

The design of a distributed video-on-demand system that is suitable for large video libraries is described. The system is designed to store 1000s of hours of video material on tertiary storage devices. A video that a user wants to view is loaded onto a video file server close to the users desktop from where it can be played. The system manages the distributed cache of videos on the file servers and schedules load requests to the tertiary storage devices. The system also includes a metadata database, described in a companion paper, that the user can query to locate video material of interest. This paper describes the software architecture, storage organization, application protocols for locating and loading videos, and distributed cache management algorithm used by the system.

1.0 Introduction

The high speed of the newest generation of workstation and network technology makes possible the integration of high-quality full-motion video as a standard data type in many user environments. One application of this new technology will allow networked users to view video material on their workstation screens on-demand.

Video-on-demand (VOD) applications will initially be divided into two major categories: video rental and video library. Video rental applications will offer users very simple interfaces to select from a small number of currently popular movies and television programming. Time Warner and Silicon Graphics, for instance, will be testing a system in Orlando Florida that provides cable TV viewers with on-demand access to 250 popular titles [19]. Video library applications, on the other hand, will support sophisticated queries against a large database of video material. A video library might include course lectures and seminars, corporate training material, product and project demonstrations, video material for hypermedia courseware, documentaries and programs broadcast on public and private TV channels, and feature films. We also expect this system will store personal material such as video email and personal notes.

Network users are already accustomed to being able to search large document databases and retrieve text for viewing on their workstation screens [2]. The goal of the Berkeley Distributed VOD System described in this paper is to provide a similar service for continuous media data. By continuous media we mean data types with dynamically changing real-time presentations such as audio, video, and animation. Our system has five central goals

1) Provide on-line access to a large library of video material (e.g., over 1000 hours).
2) Support both local and wide-area (Internet) access to the library.
3) Scale gracefully using the existing network infrastructure
4) Support ad hoc queries to allow users to find video material based on bibliographic, content information, and structural information.
5) Handle a wide variety of multimedia document types.

Providing networked access to a large video library presents several challenges. First, the size of digitized video and the high bandwidth requirements of video playback require both the capacity of tertiary storage (e.g., tape jukeboxes) and the speed of secondary storage (e.g., magnetic disks). Second, while the economies of scale in massive storage devices motivate the use large central repositories, limited internetwork bandwidth and the desire for low latency access suggest the use of distributed stores.

This paper describes the design and implementation of a distributed hierarchical storage manager for multimedia data called the VOD Storage Manager (VDSM). VDSM addresses the challenges of local and wide-area video-on-demand service with a hierarchical storage architecture in which archive servers (AS) use tertiary storage devices to act as central repositories for video data and metadata indexes, and video file servers (VFS) that are "close" to clients cache video on magnetic disk for real-time playback. The system scales by allowing many VFSs on a client LAN to support video playback and improves VFS cache effectiveness by taking advantage of user-supplied caching directives in making cache replacement decisions and scheduling movie prefetching. Finally, the system supports a compound object model that allows the storage manager to support a wide range of multimedia formats.

The remainder of this paper is organized as follows. Section 2 motivates the distributed hierarchical storage architecture employed by VDSM. Section 3 describes the architecture of the Berkeley Distributed VOD System, and section 4 describes our initial implementation.

2.0 Distributed Hierarchical Storage Management

At its barest level, video-on-demand can be viewed as a distributed file system problem. As in a conventional network file system, users of a VOD system have two basic needs: locate relevant material and retrieve it for viewing on their local workstations. However, providing widely distributed clients access to very large video databases poses several challenges that are inadequately addressed by conventional network file systems. First, video data is both difficult to store and difficult to deliver. Delivery for video playback requires stringent real-time scheduling and significant network bandwidth (e.g., 0.5 to 4 Mb/sec for VHS quality video), while conventional wide area networks offer neither. Second, cost effective storage requires the use of tertiary storage devices, which provide access times too slow for interactive response. Typical devices also support few concurrent users (e.g., a typical jukebox might have only four readers). Although recent work has been done to develop real-time file systems for delivery of video over high speed local area networks, very little has been done to address the need of cost effectively storing large video libraries or on delivering that media over heterogeneous wide area networks.

2.1 Real-time Playback

A fundamental challenge in delivering on-demand video is meeting the real-time data delivery requirements of video playback. Unlike conventional computer applications, continuous media playback requires that data be delivered at a steady rate without significant variations [1]. High-quality, full-motion video, for instance, requires that data be delivered at approximately 2 Mb/sec. If data is not delivered on time then audio output may be disrupted or video frames may be dropped, resulting in an unpleasant jerkiness in the presentation.

Conventional network file systems, such as NFS [16] and AFS [6], are not designed to accommodate real-time file service. Consequently, they may fail to deliver data on time, which will result in unacceptable playback quality. Real-time file servers, on the other hand, address the problem of real-time delivery over local area networks by carefully scheduling I/O operations to meet the consumption constraints of their clients [14, 25, 22, 5, 24, 10].

Unfortunately, real-time delivery over wide-area networks presents a new set of problems. In a wide-area network, packets of data sent by a real-time file server to remote clients must cross through network bridges, routers, and hubs, any of which could drop or delay packets, resulting in poor playback quality. Although work has been done on protocols for real-time delivery over WANs [3, 18], these protocols require that all networks along the path run special networking software. Many years are likely to pass before all nodes in the Internet are updated to run such software. Some observers predict that there will be a large number of notes that are public and that will implement first-come, first-serve protocols for a long time [7]. Thus, systems developed to deliver video-on-demand to widely distributed clients cannot expect real-time guarantees from the network.

2.2 High Bandwidth Requirements

Another problem in delivering video-on-demand is the high bandwidth required when several videos are played simultaneously. A conventional ethernet LAN has a peak bandwidth of 10 Mbs, with observed performance typically being 6-8 Mbs. Thus, a LAN could support at most 4-5 viewers before reaching saturation.

Newer network technologies, such as FDDI and ATM, hold greater promise. FDDI networks, for instance, offer a peak shared bandwidth of 100 Mbs, which can support 25 or more simultaneous users. ATM networks, on the other hand, offer between 25 and 250 Mbs on each link, with even higher aggregate bandwidth. Thus, a fast ATM switch could support simultaneous video transmission to every host on a local area network.

Wide area networks, however, present a far bleaker picture. The peak bandwidth out of a server to all of its clients cannot exceed the bandwidth of all intervening networks and hubs. Recent measurements of TCP bandwidth on the Internet between hosts in Berkeley and machines separated by between one and fifteen routers indicate an available bandwidth of only 10-100KB/sec [11]. However, performance is highly variable because we have played video stored at UCSD on clients at Berkeley using the Internet, and we have observed up to 2Mb/sec bandwidth.

Even with fast networks, playback can bottleneck on the I/O system of the file server. A single SCSI disk can support a sustained read transfer rate of about 2 MB/sec, or enough for 6-8 playbacks. Stripping data across a disk array can vastly improve the transfer rate but output is ultimately limited by the bandwidth of the file server backplane. For example, one commercial VFS with a disk array with four disks achieves a peak bandwidth of 25Mbs [23].

2.3 Video Databases

Probably the greatest challenge in supporting on-line access to video libraries results from the size of the objects themselves. A single hour of compressed VHS quality video consumes 1 gigabyte of storage, and real-world video libraries will require many hours of video. Take, for instance, an archive for course lectures and related material for a typical university department like the UC Berkeley Computer Science Division. Each semester the department offers 17 undergraduate courses and 13 graduate courses, each running for 15 weeks per semester with three hours of lectures each week. The total amount of video required for a year of lectures is substantial:

3 hours per week * 15 weeks/semester = 45 hours/course * 30 courses / semester = 1350 hours / semester

Thus, many video libraries will require terabytes of storage. In 1993 prices, magnetic disk storage costs approximately $1000 per gigabyte, or $1 million for one thousand hours of video. A more cost effective solution is offered by tertiary storage, which uses robotic arms to serve a large number of removable tapes or disks to a small number of reading devices. As illustrated in Table 1 below, tertiary storage offers a substantial reduction in price per megabyte, but at the expense of increased access times.

TABLE 1.

Media	Cost/GB	Throughput	Seek Time	Total Storage
Hard disk	$1000	2.0 MB/sec	10 ms	2 GB
Optical jukebox	$500	0.5 MB/sec	60 ms - 20 sec	100 GB
Tape jukebox	$100	1.2 MB/sec	30 sec - 1.5 min	10 TB

Because of these long seek and media swap times, tertiary storage devices (especially tape jukeboxes) are poor at performing random access within movies. Moreover, they can support only a single playback at a time on each reader. A jukebox typically has between one and four readers. Thus, tertiary storage devices are inappropriate for direct video playback.

2.4 Distributed Hierarchical Storage Management

The high-bandwidth and real-time delivery constraints of video playback are best addressed by LAN-based file servers located close to playback clients, while the cost effectiveness of large tertiary storage devices and the desire to share video widely motivate the use of a central repository. What is needed is a way to preserve the cost effectiveness of centralized storage while maintaining the high performance and scalability of distributed servers. The solution is to design a distributed hierarchical storage management system.

A hierarchical storage manager applies the concept of caching to tertiary storage, using fast magnetic disks to cache frequently accessed data from large tertiary storage devices [8, 9]. Distributed hierarchical storage management extends this idea by allowing multiple caches to be distributed across a network. If a high percentage of client accesses are to data stored in a local cache, the perceived performance is very high and WAN traffic is limited. If, however, user accesses are unpredictable or have poor reference locality, or if cache write conflicts are common, most accesses will require an access to the tertiary storage device and performance and scalability will suffer.

Fortunately, the access characteristics of video-on-demand applications make them especially good candidates for distributed hierarchical storage management. First, user accesses are likely to demonstrate a high locality of reference. A small set of movies, and television programs are likely to be popular at a given time, while older or more obscure programming will be seldom accessed. Furthermore, for a large class of applications, users or programming providers may know well ahead of time

what videos are likely to be accessed in the near future. For example, an instructor knows ahead of time that recent class lectures and other footage related to topics currently being covered in class are likely to be viewed by his or her students. Similarly, users may decide hours ahead of time that they want to watch a particular movie for evening entertainment. If this sort of predictive information can be fed to the storage manager, it can prefetch data into caches so that it will be there when users request it. Finally, distributed cache consistency problems are unlikely to occur in video-on-demand applications because they are essentially read-only.

3.0 Distributed VOD System Design

The Berkeley Distributed VOD System uses distributed hierarchical storage management to provide widely distributed clients on-demand access to a large continuous media library. Figure 1 depicts the architecture of the system. Archive servers (AS) act as central repositories for published objects, typically employing tertiary storage. Each AS has an accompanying catalog, or metadata database that stores information about the videos available on the archive. This information, which can range from simple bibliographical descriptions to sophisticated content indexes, can be queried by users using an ad hoc query interface, called a *video database browser* (VDB), to locate movies for playback. Real-time playback is provided by one or more video file servers (VFS) located on the client's LAN. When a client selects a video for viewing, the browser checks if it is already cached on one of the local VFSs. If so, no access to the archive is necessary, and playback can begin immediately. Otherwise, the client may request that the video be loaded from the archive.

Figure 2 depicts the basic communication for video playback. The components of a Distributed VOD System play comparable roles to those in a conventional movie rental scenario. The archive server plays the role of the video rental store, holding a large collection of movies for users to check out. The metadata database acts as the boxes on the rental store shelves, describing available movies. The video file server plays the role of the VCR, providing real-time playback to the user's screen. The primary difference between a VFS and a VCR is that the VFS can play video to many clients at once, and can eliminate costly trips to the video store by saving copies of previously watched material for repeat viewing. In fact, the VOD system extends this model by allowing the user to shop at many video stores (multiple archive sites), to have many VCR's (server arrays), and to avoid suffering through many trips to the video store by scheduling to the delivery of videos to his VFS before they are needed (i.e. prefetching).

The rest of this section describes the VOD storage manager, including the media object model, the naming and distribution policies of the archive server, the VFS intelligent cache management system, and support for VFS arrays.

3.1 Compound Media Object Model

Two goals of the Berkeley Distributed VOD System is to support sharing of a wide variety of multimedia document types in a variety of representations and encodings and to support sophisticated queries on those documents. The proliferation of multimedia formats and video compression standards means that the VOD system must allow users to publish movies in multiple formats (e.g., QuickTime, AVI, OMF, or our own CM Script [15]), in a variety of encodings (e.g., MPEG, DVI, Motion JPEG, etc.), and in a wide range of fidelities (i.e. various image sizes, frame rates, color depths, and sound resolutions). The system should also efficiently support publishing of media with added or altered streams (e.g., close captioned, sub-titles, sound tracks in different languages, etc.) and support the extraction of small segments of a large movie for playback or incorporation into other documents. The VOD system provides this flexibility through the Compound Media Object (CMO) abstraction, which provides a uniform, media independent access layer through which the VOD storage manager deals with all published material.

Each CMO has a globally unique object identifier (OID), a type name, one or more named bitstreams (files) and, optionally, a list of external references to other CMOs. Rather than defining ridged object semantics, the CMO provides a generic container facility upon which clients can define their own document structures and semantics. In this regard, the CMO abstraction resembles the Unix file system or the Bento file format [5]. The key difference is that the CMO exports a list of external object references to the storage manager, thereby allowing the construction of compound objects.

The ability to construct compound objects allows a movie to be composed of many subobjects, which may in turn be shared with other movies. For example, the audio and video streams of a movie may be segmented into several CMOs (clips) of types "AudioClip" and "VideoClip." Another CMO of type "Movie" may then reference these clips and describe how they are ordered for playback. The storage manager itself need not understand the structure or semantics of these typed objects. It need only know the object dependencies to support storage and migration. Decomposition of media into shared subobjects offers several advantages: 1) storage space is saved for multiple representations of a single movie or for movies that share scenes, 2)

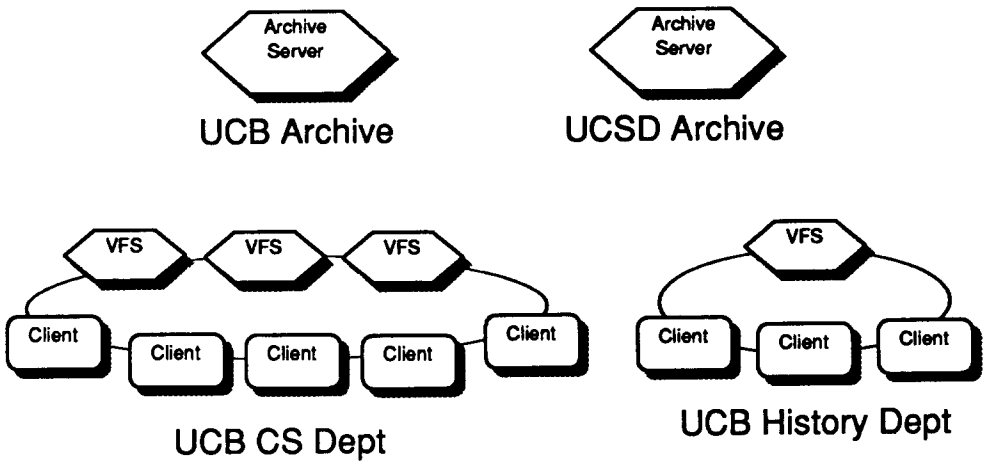

FIGURE 1. Example VDSM Hierarchy

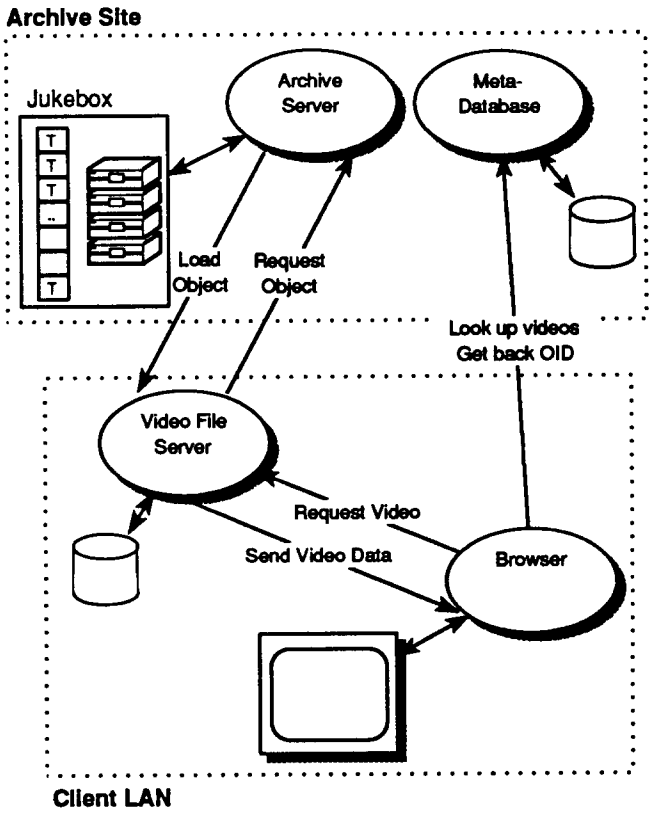

FIGURE 2. Basic Components of a Distributed VOD System

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.