Cable Networks and Distributed Video Repositories

Shahram Ghandeharizadeh
Matsushita Information Technology Laboratory
Panasonic Technologies Inc.
2 Research Way
Princeton, NJ 08540
shahram@research.panasonic.com

Abstract

Cable networks stretch to many homes in the United States. The available bandwidth of this network exceeds the amount of transmitted content (programming). One way to increase revenue is to employ the idle bandwidth to transmit on-demand programming for a nominal charge either on a pay per view or a monthly flat fee basis. This service would require deployment of a distributed video server across a geographically distributed region, in order to provide programming based on demographic characteristics of each region. This system could provide the necessary infrastructure for the cable companies to expand their home programming with Internet services. This paper provides a global perspective of such a system.

1 Introduction

The cable network with its extensive outreach to homes can conceptually be compared with either the Internet or the telephone network. Currently, the programming potential of the cable network far exceeds the current broadcasted programming it carries to homes. This is partially due to lack of content that is not appropriate for broadcasting. This study proposes extensions to the cable network to provide on-demand service to homes. To realize this objective, it proposes the use of a distributed continuous media server to segment the cable networks of a service provider into partitions. A partition for a specific region strives to service the customers located in that region with the appropriate content.

The envisioned continuous media server consists of a central server and a number of regional servers. The regional servers are a client of the central server. Moreover, they are geographically located such that the load imposed by the customers of each region on each regional server is approximately the same. Each server can support continuous display of audio and video clips. In a typical configuration, the central server would have a higher throughput than each of the regional servers. The central server is a library that contains a copy of all the programs available on the system. Each regional server caches a fraction of the programming available on the central server. This caching is done intelligently based on demographics and the programming expected to be of interest to majority of the residents in a geographical region.

To provide on-demand service, the startup latency observed by a customer should be minimized. *Startup latency* is defined as the amount of time elapsed from when a customer requests a video clip until its display starts. To minimize this latency, the system should connect a user with a regional server that is in close proximity. This would also reduce the number of participating cable links in

order to maximize the number of simultaneous displays supported by the cable network. Of course, the regional server should contain the programming desired by a client. Otherwise, it may pursue two alternative possibilities. First, it may forward the client to be serviced by either another regional server that contains the desired programming or the central server. Second, it may elect to cache the requested program from the central server in order to service the client. In both scenarios, the client will observe a higher startup latency. The first approach is appropriate if the requested program is unpopular (not frequently accessed) with that region. With the second scenario, the client would cause the regional server to contain a copy of the referenced program. This caching is appropriate if the requested program is expected to be frequently referenced by the other clients of this regional server. Note that the regional server might have insufficient space and be forced to delete some other program in favor of materializing the requested program.

This paper describes *scalable* servers that can participate as either the central or regional servers. In addition, it hints at *caching* techniques to minimize the amount of storage required at a regional server. A scalable server is desirable for several reasons. First, it separates the software from the physical parameters of its underlying hardware, e.g., the number of mass storage devices. Second, the software does not restrict the number of simultaneous displays that can be supported by the hardware. As the hardware grows, the software enables the system to support a higher number of displays (as long as the hardware platform does not hit a physical limitation). Third, it provides for an incremental investment approach where a service provider starts with a small hardware configuration and expands the system based on customer demand.

Intelligent caching techniques minimize the amount of storage required at a regional server. This reduces the number of storage devices that in turn reduces costs. In addition, it reduces the mean time to failure of mass storage devices in order to improve the availability of data. To illustrate, at the time of this writing, the mean time to failure of a single disk drive is once every 600,000 hours (or 70 years). With a system that consists of ten disks, the mean time to failure of *some* disk is once every 60,000 hours (or 7 years). The mean time to failure of some disk in a system consisting of one thousand disks is once every 600 hours (or 25 days). Of course, these are theoretical expectations with almost always a lower practical expectations.

2 Scalable Servers

Recent advances in computer processing and storage performance and in high speed communications has made it feasible to consider continuous media (e.g., audio and video) servers that scale to support thousands of concurrently active displays. The principle characteristics of continuous media is their sustained bit rate requirement. If a system delivers a clip at a rate lower than its pre-specified display rate without special precautions (e.g., pre-fetching), the user might observe frequent disruptions and delays with video and random noises with audio. These artifacts are collectively termed *hiccups*. For example, CD-quality audio (2 channels with 16 bit samples at 44 kHz) requires 1.4 Megabits per second (Mbps). NTSC quality video (640 $\times$ 480 pixels per frame, 24 bits per pixel) at 29.9 frames a second requires 210 Mbps. At the time of this writing, a variety of HDTV quality video are available with display bandwidth requirements ranging from hundreds of Megabits to Gigabits per second. These bandwidths can be reduced using compression techniques due to redundancy in data. Compression techniques are categorized into *lossy* and *lossless*. Lossy compression techniques encode data into a format that, when decompressed, yields something similar to the original. With lossless techniques, decompression yields the original. Lossy compression techniques are more effective in reduc-

ing both the size and bandwidth requirements of a clip. For example, with the MPEG standard, the bandwidth requirement of CD-quality audio can be reduced to 384 Kilobits per second. MPEG-1 reduces the bandwidth requirement of a video clip to 1.5 Mbps. With some compression techniques such as MPEG-2, one can control the compression ratio by specifying the final bandwidth of the encoded stream (ranging from 3 to 15 Mbps).

The size of a compressed video clip is quite large by most current standards. For example, a two hour MPEG-2 encoded video clip, requiring 3 Mbps, is 2.6 Gigabytes in size. (In this paper, we focus on video due to its significant size and bandwidth requirements that are higher than audio.) To reduce cost of storage, a typical architecture for a video server employs a hierarchical storage structure consisting of DRAM, magnetic disks, and one or more tertiary storage devices. As the different levels of the hierarchy are traversed starting with memory, the density of the medium and its latency increases while its cost per megabyte decreases. It is assumed that all video clips reside on the tertiary storage device. The disk space is used as a temporary staging area for the frequently accessed clips in order to minimize the number of references to tertiary. This enhances the performance of the system. Once the delivery of a video clip has been initiated, the system is responsible for delivering the data to the settop box of the client at the required rate so that there is no interruption in service. The settop box is assumed to have little memory so that it is incumbent on the server and network to deliver the data in a "just in time" fashion.

Due to a sequential display of a movie at a home, the data can be retrieved from the available disks one block at a time and delivered to the cable network for display at a customer's television. A server typically consists of a many magnetic disks. **The key to realizing a scalable server is to distribute the load imposed by a display evenly across the available disks.** This is attained by dispersing the blocks of a movie across all the disks using a pre-defined data layout function [BGMJ94]. The pre-deterministic nature of data layout brings about a regular schedule for activation of disks in order to enable the system to maximize the number of serviced customers. When additional disks are introduced into a system, the data layout must be modified to preserve the original pre-defined function. In [GK96], we describe a technique that re-organizes the data while the system is servicing customers.

Mitra [GZS+97] is a research prototype, designed and developed at the University of Southern California, that demonstrates the feasibility of a scalable continuous media server.

3 Pipelining to Minimize Startup Latency

With a geographically distributed server, when a request references an object that does not reside on the regional server, one approach might materialize the object on the regional server drives in its entirety before initiating its display. In this case, the startup latency is determined by the allocated bandwidth (by the network, the central server, and the regional server) to materialize the referenced object and its size. With continuous media (e.g., audio, video) that require a sequential retrieval to support its display, a better alternative is to use pipelining in order to minimize the latency time [GS94, GDS95]. Briefly, the pipelining mechanism groups the blocks of a referenced object X into s logical slices ($S_{X,1}$, $S_{X,2}$, $S_{X,3}$, ..., $S_{X,s}$) such that the display time of $S_{X,1}$ ($T_{Display}(S_{X,1})$) eclipses the time required to materialize $S_{X,2}$ from the central server ($T_{Materialize}(S_{X,2})$), $T_{Display}(S_{X,2})$ eclipses $T_{Materialize}(S_{X,3})$, etc. This ensure a continuous display while reducing the latency time because the regional server initiates the display of an object once $S_{X,1}$ becomes available.

In [GDS95], we studied the pipelining mecha-

nism for two possible scenarios: the bandwidth allocated to materialize object X is either (1) lower or (2) higher than the bandwidth required to display X. We refer the interested reader to [GDS95] for details.

4 Partial Caching

Assume that the number of customers supported by both the central server and the cable network (i.e., their aggregate throughput) is significantly larger than that of each regional server. Based on the discussions of Section 3, a regional server no longer must cache an entire copy of a presentation. Instead, it caches the first few sections of each object (say $S_{X,1}$ and $S_{X,2}$ for object X). With this approach, the regional server can provide the illusion of having a entire copy of X by initiating the display of $S_{X,1}$ and establishing a pipeline between the central server and the customer. The startup latency observed with this technique is equivalent to the scenario where object X is cached on the regional server in its entirety. When compared with a technique that caches a copy of X in its entirety, partial caching has the following advantages. First, given a fixed amount of storage, each regional server can now cache many more objects to provide its region with more content. Second, with this caching technique, the amount of storage required at each regional server is minimized, reducing cost. (As discussed in Section 1, this also enhances the fault tolerance characteristics of the regional server.) Third, it provides for a more centralized control of the entire system. Fourth, for those programs where a customer starts to display and then aborts in the middle of presentation, this paradigm saves storage space at the regional client (because the last slices are neither retrieved from the central server nor staged on the regional server).

However, partial caching suffers from a number of limitations. First, each display requires the participation of the main server. Second, pipelining utilizes the bandwidth of the cable network from the central server to a customer's home. If not managed properly, the bandwidth of either the main server or the cable network might be exhausted, reducing the number of customers that can be serviced simultaneously.

The tradeoff of this caching technique must be analyzed carefully before designing a system around it. Briefly, the results of [GS94] demonstrate that this caching technique is inappropriate when the peak overall system load exceeds the bandwidth of the central server because the central server becomes a bottleneck.

Ideally, the system should employ a hybrid caching solution: A regional server caches the frequently accessed objects in their entirety and employs partial caching for those objects whose popularity cannot be established. Partial caching can be designed to behave in this manner.

5 Conclusion

This paper proposes the deployment of scalable continuous media servers in the existing cable networks to provide on-demand service to homes. The basic skeleton provided here can be extended with a variety of services. One might envision intelligent agents that monitor the television programs viewed by each customer in order to build a profile. These profiles can be used to suggest to a customer on-demand programming that is similar to those they watch regularly.

While the technology has matured to a stage to make the ideas proposed in this short papers feasible, there are social and legal issues that require a careful study. As a simple illustration, the concept of monitoring the programming selected by users' in order to suggest similar programs is trivial. However, the users' right to privacy must also be considered where a customer might refuse the service because the system is collecting has his or her personal information. It is worthwhile to note that

computer users are starting to become less sensitive to such privacy issue (e.g., see "As the Web Expands, So Do Surveillance Tools," New York Times, Monday, January 6, 1997, Page D5). If the Internet users are employed to model the future cable network warriors, these legal issues might not pose a significant challenge.

References

[BGMJ94] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju. Staggered Striping in Multimedia Information Systems. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 79–89, 1994.

[GDS95] S. Ghandeharizadeh, A. Dashti, and C. Shahabi. A Pipelining Mechanism to Minimize the Latency Time in Hierarchical Multimedia Storage Managers. *Computer Communications*, March 1995.

[GK96] S. Ghandeharizadeh and D. Kim. On-line Reorganization of Data in Scalable Continuous Media Servers. In *Seventh International Conference on Database and Expert Systems Applications*, September 1996.

[GS94] S. Ghandeharizadeh and C. Shahabi. On Multimedia Repositories, Personal Ccomputers, and Hierarchical Storage Systems. In *Proceedings of ACM Multimedia Conference*, 1994.

[GZS+97] S. Ghandeharizadeh, R. Zimmermann, W. Shi, R. Rejaie, D. Ierardi, and T. Li. Mitra: A Scalable Continuous Media Server. *Kluwer Multimedia Tools and Applications*, January 1997.