Design of a video-server complex for interactive television

by T. Sanuki
Y. Asakawa

This paper describes the architecture and implementation of a scalable video server for interactive television services, which has been used in several video-on-demand projects in Japan. This server supports a large number of users with the delivery of high-quality video and provides many types of interactive multimedia applications. The server, using the IBM RISC System/6000® and RISC System/6000 Scalable POWERparallel Systems®, permits full interactive operation with clients. It also can deliver a large number of streams when it is expanded from a single-processor system to a multi-processor system, without any change of the server architecture. In this paper, we describe the end-to-end architecture, the design considerations regarding scalability of the delivery scheme for MPEG-2 transport streams, and the management mechanism for interactive applications. This paper also addresses the implementation of quality of service for digital services through a hybrid fiber-coax network infrastructure, and protocols for interactive television services.

Introduction

As a result of recent rapid progress in computer, networking, and broadcasting technologies, new services and new applications based on multimedia are increasing around the world. Interactive television, one of these new services, is a technology for delivering, on demand, television programs and multimedia applications to households and businesses through a broadcasting infrastructure. This scheme is applicable not only to "on-demand services" such as video-on-demand, but also to many kinds of information delivery, such as public-information services.

Scalable video servers play the most significant role in interactive television in a broadband-network infrastructure. Many companies and academic organizations are working in this area, engaging in trials and providing services [1]. There are, however, many technical problems in attaining the necessary quality of service from server to clients. For instance, providing a large number of continuous streams of digital video requires a high aggregate I/O bandwidth and the management of large-capacity storage devices. Moreover, quick response time for interactive operations by several hundreds of users requires special considerations for application protocols between clients and server.

0018-8646/98/$5.00 © 1998 IBM

Table 1 Abbreviations used in this paper.

CATV = cable television
CS = control server
DCE = Distributed Computing Environment
DM = device monitor
DMA = direct memory access
DSM-CC = Digital Storage Media Command and Control (part of the MPEG-2 standard)
FDMA = frequency division multiple access
GW = gateway
HFC = hybrid fiber-coax
HPS = high-performance switch
LCU = line control unit
MCA = microchannel architecture
MPEG-2 = Moving Picture Expert Group Phase-2
MSC = MPEG stream controller
NAR = network access routine
NTSC = National Television Systems Committee
ONC = Open Network Computing
PAT = program association table
PCR = program clock reference
PID = packet ID
PMT = program map table
PSI = program specific information
QAM = quadrature amplified modulation
QoS = quality of service
QPSK = quadrature phase shift keying
RPC = remote procedure call
RS/6000 = RISC System/6000
SP2 = RISC System/6000 Scalable POWERparallel Systems
SSA = Serial Storage Architecture
STB = set-top box
TDM = time division multiplexing
TMG = Tokyo Metropolitan Government
VCR = videocassette recorder
VSD = virtual shared disk

In collaboration with the IBM Research Division, we have developed a video-server complex for interactive multimedia services to handle several hundreds of users through a hybrid fiber-coax (HFC)[1] network [2]. The server program runs on an IBM RISC System/6000* (RS/6000*) and a Scalable POWERparallel Systems* (SP2*) platform, providing fully interactive digital video operations for the clients over the HFC network. In addition, this system provides many other interactive multimedia services to several hundred households. The framework of these services is based on the MPEG-2 (Moving Picture Expert Group Phase-2) information-delivery scheme and allows a wide variety of multimedia applications such as movies-on-demand, home shopping, news-on-demand, education-on-demand, karaoke-on-demand, and public-information access. The architecture has both scalability with regard to the number of users and flexibility with regard to system configurations.

First,[2] this paper describes the system requirements and the design principles used to attain the best performance and functionality for interactive multimedia services, in compliance with international and industry standards such as MPEG-2 Systems [3] and DSM-CC (Digital Storage Media Command and Control) [4].

Next, this paper discusses the design considerations for the end-to-end architecture of interactive television using an HFC infrastructure. This section illustrates the functional model of the applications and discusses the protocol between client and server in the asymmetric network environment of HFC. This section also discusses the architecture for scalability to deliver a large number of video streams and defines the components of the server complex.

In the following section, the paper describes actual implementation approaches for the server. This section focuses mainly on the design and implementation of concurrent transmission of MPEG-2 "transport streams" without any jitter or glitch. We describe a unique buffer-management algorithm that we introduced to keep the MPEG-2 transport streams consistent, and we describe the adapter for multiplexing the streams. Additionally, this section describes the control mechanism in the video-server complex for handling interactive operations with the client.

Finally, this paper introduces a measurement of streaming performance, analyzes the results of such measurements, and evaluates the scalability of streaming based on the measurement. The video-server complex described in this paper has been successfully deployed in several trial interactive television projects, such as the video-on-demand project of the Association of Tokyo Multimedia Systems of the Tokyo Metropolitan Government (TMG) [5]. Evaluation of the system in the actual end-to-end environment is also addressed.

Objectives and design principles

- *System requirements*

The system satisfies the following requirements:

1. Can deliver a different video stream or application for each user.
2. Can allow all users to access a single video stream or application.
3. Can deliver any requested item at any time to any user, without a significant delay.
4. Can deliver over one hundred digital video streams simultaneously.

[1] Abbreviations used in this paper are listed in **Table 1**.

[2] A complete table of contents is provided in **Table 2**.

5. Provides broadcast-level quality with industry-standard-format digital video (this implies 6Mb/s[3] MPEG-2 streams).
6. Stores over one hundred video streams, each of which has several hours of contents.
7. Delivers the contents, in digital format, through an HFC network.
8. Can be expanded, both with regard to throughput and contents, without significant changes.
9. Provides a variety of types of multimedia applications.
10. Operates in conjunction with a standard cable television (CATV) system; provides its digital services in addition to, and compatible with, the CATV service.

Requirements 1–3 imply that the system must be able to deliver the full contents of any video file at any time. Requirements 4 and 5 mean that the system must have an aggregate bandwidth of over 600 Mb/s. Requirements 5 and 6 mean that the system has over half a terabyte of storage. Requirements 5 and 7 imply that the contents must be encoded in 6Mb/s MPEG-2 streams with an MP@ML (Main Profile@Main Level) format defined in the standard for digital broadcasting [6] and must be delivered without any conversion, such as digital to analog, in the transmission stage. Requirement 9 means that in addition to providing file transfer from server to clients, the system must have application protocols for interactive operations between clients and server. Requirements 8 and 10 are obvious.

Design principles

The system should also meet the following criteria, in order to remain a state-of-the-art system in the future.

- *Comply with international and industry standards*
  The system should adhere to international and industry standards, in order to support equipment of different manufacturers and to be compatible with future digital services. Therefore, the system must comply with the standard specifications for each interface and for the format of the contents. Accordingly, we decided to follow the MPEG-2 standard in ISO/IEC [7] and DAVIC (Digital Audio-Visual Council) specifications [8], which define the overall framework for video-on-demand systems.
- *Use general-purpose servers*
  Some trial systems use special-purpose video servers optimized for high-bandwidth streaming. This approach, however, has the many risks of new-hardware development and the high cost of specialized servers. We decided to use general-purpose servers as video servers and to maximize streaming performance by means of software.
- *Maximize scalability*
  The server should work very well in a small configuration, for demonstration systems and testing. It should also work well when providing actual services to a large number of users, without any change of system architecture. Therefore, the server itself should have a scalable architecture with respect to capacity.

[3] Mb/s = mega*bits* per second.

Table 2 Table of contents.

- *Provide flexibility of configuration, and extendibility*
  The system should have a high degree of flexibility, in order to meet the requirements of many customers. In order to do this, the server components should be divided into major functional blocks, each component communicating with the others through a standardized interface and protocol. Furthermore, components should be able to be added later to meet capacity and functionality requirements.
- *Provide fully interactive applications with affordable performance*
  The system should provide good performance for fully interactive multimedia applications. Its performance should not be degraded when the number of users is increased.

Architecture

- *End-to-end framework*

Stream-delivery scheme

This section discusses the scheme for the delivery of video streams, from an end-to-end viewpoint. In order to simplify the discussions in this section, we assume a model in which video streams are stored in the server, and the client decodes the video streams from the server. In general, there are two schemes for transmitting data streams from server to client: *pull* and *push*. The former is based on packet transmission with appropriate flow control between client and server. For instance, the TCP/IP protocol is a pull scheme, which can transmit data with no loss of packets. This protocol requires processing power for dealing with protocol-stack handling. Many difficulties exist. For example, if a CPU is switched to another task, the transmission process is interrupted, and the stream is not delivered with continuity. In the worst case, the user sees a broken video image on the screen of the client, even though the packets of the stream are not lost. This situation will be even more serious for higher-bit-rate streams, since data of the stream in the decoder of the client are consumed at the bit rate specified during the encoding process, and the transmission of each packet of the stream must keep up with the decoder's rate of consumption. In addition, the server must support multiple users simultaneously. Thus, the overhead of protocol processing is higher at the server side, and it is difficult to maintain the data transfer at the required rate.

The push scheme is similar to a broadcasting approach. Once a stream file has been opened with a specified transfer rate, the server keeps transmitting the stream data at that data rate. Since there is no flow control between server and clients in this scheme, the processing power needed for the stream can be minimized at both sides. This means that the server can support the transmission of a higher-bit-rate stream for many users and can continuously supply streams of data to the decoders of clients. In order to meet the requirements described in the previous section, we selected the push scheme.

As for the data format of the stream, the MPEG-2 transport stream was chosen in order to minimize the overhead of the transmission while keeping the original quality, since the stream consists of fixed-size packets and its transmission is based on synchronization between the sender and the decoder of the recipient. Furthermore, this format allows multiple streams to be multiplexed naturally into a single stream [9]. Therefore, the format is suitable for the transmission of digital video in a broadcasting mode. On the other hand, transmitting an MPEG-2 "program stream," which is suitable for use with storage media like DVD (digital video disc), requires appropriate flow control, in order to avoid underflow in the decoder buffer. This is more difficult to manage for large-scale video delivery.

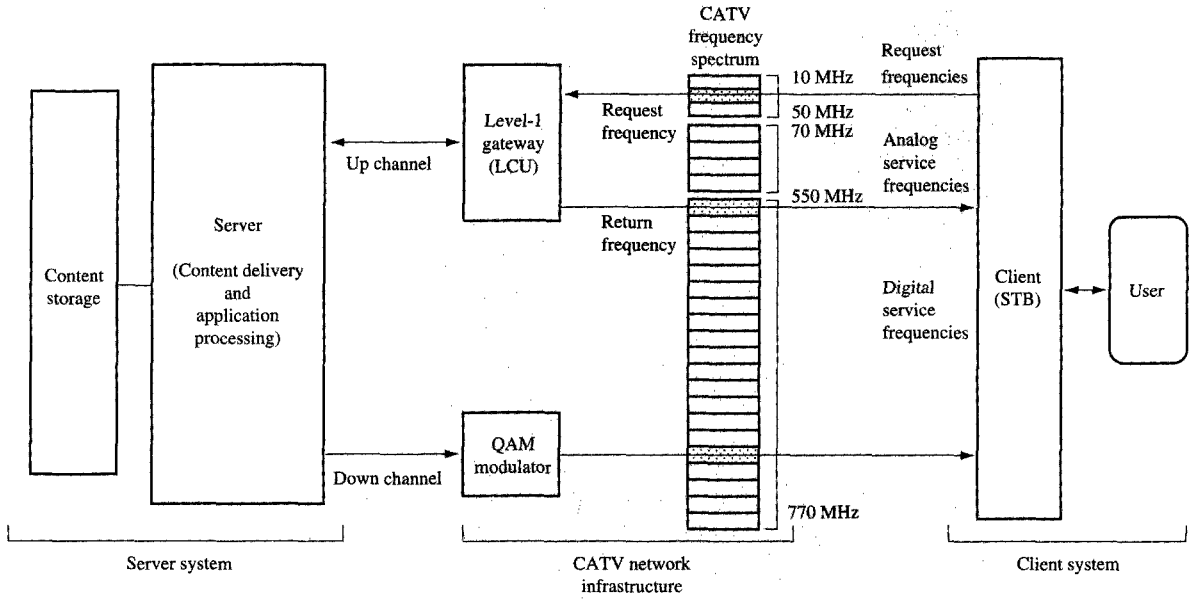
Control of streams

Interactive television requires mechanisms for controlling the stream in various ways, with VCR-like (videocassette recorder) operations at any time and no significant delays. Therefore, the control path between client and server should be full-duplex in order to maintain interactive operations. This path, called the "up channel" in this paper, enables communication between client and server through use of the TCP/IP protocol.

In a push scheme, a stream of video (MPEG-2 transport stream) is delivered continuously from server to client. This unidirectional communication path is called the "down channel" in this paper.

There are several means of handling requests from clients. The most frequently used are the queuing method and the RPC (remote procedure call) method. Though the former method is rather simple and makes it easy to isolate the client and the server, a bottleneck of requests can occur when the number of clients is large. If some user's request takes a long time for processing, the request queue for client requests can fill up, and the server may not be able to respond to the clients quickly. On the other hand, the RPC model establishes a path between the server process and each client process, and control is easily passed from the client process to the server process; thus, users' interactive operations can be dealt with without significant delay. We chose the RPC model.

As for controlling streams and application objects, we adopted the command set of DSM-CC, a part of the MPEG-2 standard. This command set supports not only stream-control operations and file-related controls, but also application-protocol management between client and server [10]. It covers the requirements of a variety of



Architectural framework of digital video-server complex sharing a CATV infrastructure.

multimedia applications in this interactive television system.

Network infrastructure considerations

The HFC distribution network shown in **Figure 1** provides several sorts of information delivery through channels allocated in the CATV frequency spectrum. For instance, requests from clients are mapped on several channels between 10 and 50 MHz, analog broadcasting signals are mapped on several channels between 70 and 550 MHz, and down channels for digital services are mapped on several channels between 550 and 770 MHz, in order to coexist with CATV analog broadcasting services. Other mappings are possible, of course. In a broadcasting infrastructure based on the NTSC (National Television Systems Committee) standard, each channel has a 6-MHz bandwidth.[4] In order to establish fully duplex communication between client and server, another part of the up channel, for the transmission of return signals from the server, is assigned a dedicated channel within the frequency band of digital services.

For the interface of the up channel with the server, as shown in Figure 1, we introduced a kind of Level-1 gateway [2] to the HFC network, called the line control unit (LCU), for isolating the broadcasting service from the interactive television services within the same network infrastructure. By using high-speed polling of the request signals, this unit detects which client has made a request to the server, and it assigns the client a communication slot, which is a bandwidth in the request frequency band, by using the FDMA (frequency division multiple access) method. After the assignment, the unit allocates the communication slot within the return-frequency band that corresponds to the slot in the request-frequency band, by using TDM (time division multiplexing). These communication slots provide a full-duplex TCP/IP session between client and server. The signals for the communication slots are modulated by the QPSK (quadrature phase shift keying) method, with 64Kb/s[5] bandwidth [11].

Since the down channels for digital services use frequencies from 550 to 770 MHz (which can be changed), the digital streams formatted in MPEG-2 can use only 36 6-MHz channels in spectrum. In order to transmit over one hundred 6Mb/s streams simultaneously in the limited number of channels, technologies for the multiplexing of streams and digital modulation of analog carriers are required. Several single-program transport streams (e.g., videos, audio, data) in MPEG-2 can be multiplexed into a single multi-program transport stream. For instance, four 6Mb/s transport streams can be

[4] In the case of the PAL (phase alternation line) TV standard, each channel has an 8-MHz bandwidth.

[5] Kb/s = kilo*bits* per second.

Table 3 Summary of the characteristics of up and down channels in an interactive television system.

	Up channel	*Down channel*
Frequency range	10–50 MHz	550–770 MHz (can be changed)
Bandwidth for each client	64 Kb/s	6 Mb/s
Modulation	QPSK	64 QAM
Frequency allocation	TDM/FDMA	Setting of QAM modulator
Network protocol	TCP/IP	MPEG-2 system (transport)
Application protocol	DSM-CC on RPC	MPEG-2 system (transport)
Error correction	TCP/IP	Reed–Solomon (204, 188)

multiplexed into a single transport stream of about 24 Mb/s. The multiplexed stream is modulated into a carrier signal of 6 MHz by using a 64-QAM (quadrature amplified modulation) modulator. This means that four digital streams can be transmitted through a 6-MHz channel, and more than one hundred digital streams in total are available in the CATV spectrum.

The down channel also employs error correction because of the high data rate and huge volume of data transmission. The QAM modulator adds a 16-byte correction code to each 188-byte packet (totaling 204 bytes) of the MPEG-2 transport stream, according to the Reed–Solomon method. By the introduction of error correction, even if a stream had a bit error rate of 10^{-4}, the error rate could be reduced to a level between 10^{-10} and 10^{-11} after the correction. This means that a one-hour stream (about 2.7 GB) with error rate 10^{-4} will have, after error correction, only one bit in error, in the worst case, which is practically negligible for the services provided. Actually, the fiber cable is installed between the broadcasting center and each client's building. The HFC infrastructure can minimize the amount of intermediate equipment from end to end, such as amplifiers, splitters, and switches. This approach improves the quality of the network while reducing the total cost. The characteristics of up and down channels in this HFC network are summarized in **Table 3**.

Application delivery scheme

In most interactive television systems, the client system is a set-top box (STB) that does not have any storage devices like hard disk drives; application programs must be delivered from service providers. Then, not only must the STB handle digital video programs, but it must also download application programs and data from the server.

There are two delivery paths: the up channel and the down channel. The former method may be slow; for instance, a 500KB (500-kilobyte) object takes more than one minute to download through the up channel. But this channel, which uses the TCP/IP protocol, is error-free. The latter path has a higher speed of transmission from server to STB. A 500KB object can be transmitted through the down channel in less than one second. This channel, however, does not have a flow-control mechanism between client and server; if errors occur during the transmission process, the client notifies the server of the errors through the up channel, and the server retransmits the data. With the error-correction mechanism in the QAM modulator mentioned in the previous section, the bit-error rate is estimated to be 10^{-10} at most. Thus, the error rate is negligible for the actual implementation when application programs and data objects of less than a few megabytes are downloaded. (Furthermore, error correction by retransmission of packets found to contain errors reduces the error rate for data and programs to insignificance.) One of the main objectives of the system is to provide high interactivity for digital service; therefore, we decided that the application programs and data objects would be transmitted through the down channel, with the up channel used for the feedback of the transmission status. Consequently, application programs are delivered in the format of private streams, as defined in the MPEG-2 System standard [3].

- *Application model*

In the applications of interactive television such as movie-on-demand, karaoke-on-demand, news-on-demand, and home shopping, the following items should be taken into consideration in the design of the application framework:

- The user can perform the operations with no significant delay.
- The server must support several types of applications in the same platform.
- The contents should be controlled by the applications through such processes as authentication.
- The work of application development should be reduced.

In order to meet these requirements, we introduced a conventional client/server model. The system uses a combination of an application program in the client (STB), which primarily manages the user's interaction and application logic, and an application program in the server, which deals with common services such as

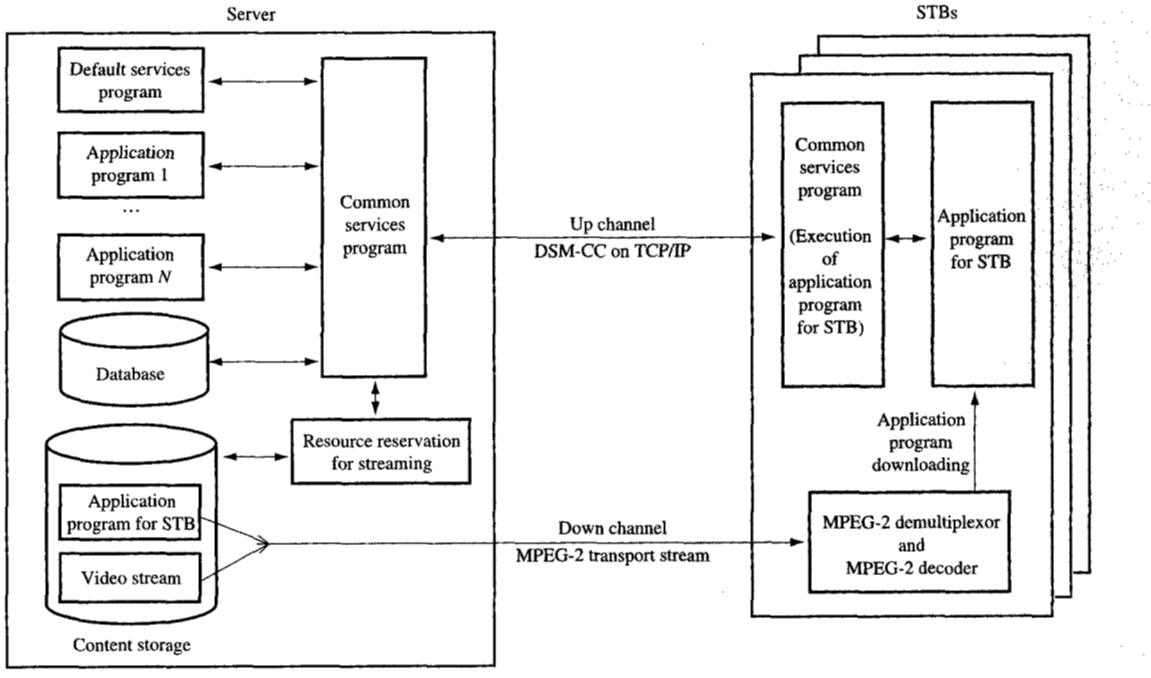


Figure 2

Application program model of interactive television.

database, content, and transaction management. Both application programs are divided into two layers: a common services program, which manages the handling of DSM-CC commands and other common services such as database management for applications, and the application program itself. The common services program in the STB executes the application program interpretively, translating it into a protocol between the STB and the server. The common services program in the server receives requests from the STB and forwards them to the appropriate application program modules in the server. In the case of a stream that includes the binary file of an application program being requested, the server transmits it through the down channel as an MPEG-2 transport stream. A diagram of the application model is given in **Figure 2**.

- *Design of video-server complex*

Functional definition

The video-server complex for interactive television is composed of several functional blocks. In order to attain some degree of flexibility for the server configuration, the video-server complex in a push scheme can be internally divided into several elements, as follows:

1. Request management for the up channel.
2. Application program processing.
3. Server resource management and control.
4. Content loading and management.
5. Data pump for continuous streaming.
6. Network interface for the HFC.
7. System administrator support.

These elements of the video-server complex are categorized into two major parts from a functional point of view: the non-real-time part (1, 2, 4, 7), which mainly manages the application processes, and the real-time part (3, 5, 6), which deals with MPEG-2 streams.

Designing for quality of service

Quality of service (QoS) is a system attribute that involves minimum jitter, minimum end-to-end delay, and error-free reception. System characteristics affecting QoS include network bandwidth, storage bandwidth and capacity, and CPU power. **Figure 3** is a structure designed to maintain QoS in the video-server complex. We mention here some of the design factors affecting QoS.

On the server side, there are two key factors for guaranteeing QoS: network resource reservation and real-time I/O operation. The former is the reservation of a

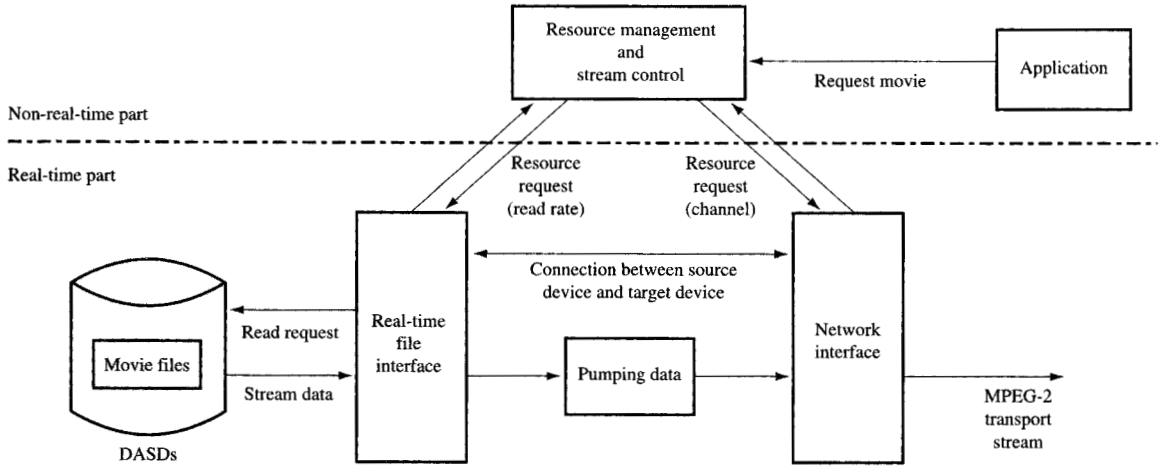


Figure 3

Structure to maintain QoS in the video-server complex.

communication path from one end of the network to the other. In the HFC network, the resource is defined as a channel. Once a channel is reserved for a stream, packets of the stream output from the video-server complex will be transmitted on the specified channel within the spectrum of the HFC network.

The latter factor, real-time I/O operation, should be taken into consideration in every stage of the data path in the server. The digitized video data is stored on the hard disk drives of the server as files. The file system interfaces of conventional operating systems, such as AIX*, do not guarantee the data rate of the read and write operations. Therefore, a real-time file interface with a guaranteed I/O rate must be introduced. On the other hand, MPEG-2 transport streams require precise transmission rates, and the interface to the HFC network must maintain the data transmission of the streams continuously. In order to deal with the continuous data transfer, the connection management between source device and target device, which defines the ports of both devices involved in the data transmission, and the MPEG-2 transport-stream-processing part, which identifies the transmission parameters for the network interface, must be introduced. On the other hand, stream control, which supports various access patterns to the streams such as play, jump, and pause, must be introduced in order to allow interactive operations by users.

The video-server complex must also provide these data-pumping capabilities to multiple users without the streams interfering with each other.

Consideration of scalability

The video-server complex must have scalability of stream delivery, which means that the following factors are maintained, even if the number of clients, concurrent streams, and volume of content grows: the architecture of the video-server complex; the quality of the streams; response time to users; and code and data of application programs. The video-server complex also must provide expandability for storage and network capacity without any architectural change to the server, and must provide an integrated interface to administrative operators for the management of the video-server complex elements.

In general, high I/O throughput is the key factor for a video server [9]. The transmission capability of the system is determined by the following factors, illustrated in **Figure 4**:

- CPU processing capability.
- Memory size and bandwidth.
- System-bus bandwidth.
- Disk bandwidth, the number of disks, and read-block size.
- Network-interface bandwidth.

Stream data stored in the DASD are transmitted through the following path during the DMA (direct memory access) process:

$$c \rightarrow b \rightarrow a \rightarrow b \rightarrow d.$$

This implies that the data cross the system bus (b) twice. Therefore, the transmission rate of a stream cannot

exceed (bandwidth of system bus)/2. Actually, because of the overhead for setting the bus I/O controller and the arbitration of the adapters plugged into the system bus, and other overhead for data handling, the actual maximum bandwidth for streaming is given by $R \times$ (bandwidth of system bus/2) − overheads, where R is between 0.4 and 0.5 for microchannel architecture.

In single-processor systems, there are other factors reducing system performance and capacity. The limitation on the number of system slots for the network adapters restricts the output bandwidth from the server. The limitation of DASD capacity attached to the server also limits the expandability of the system. Consequently, a single-processor system cannot be sufficiently expanded for video delivery to a large number of clients.

Another approach to expanding server capacity is to connect several single-processor servers with a LAN, with each server having the same contents in its attached DASDs. Although it is easy to expand the network bandwidth in this manner, efficiency of DASD usage is low, since the contents are duplicated. In still another approach, servers connected with LAN share the contents, which are placed in other servers connected through LAN. This approach experiences a performance bottleneck in the LAN communication.

In order to solve these issues, the RISC System/6000 Scalable POWERparallel Systems (SP2) was introduced as the video-server complex. The processors that manage the DASDs are called storage nodes, and the processors that transmit the streams to the network are called network nodes. Each processor in our video-server complex is connected to the full-duplex, high-performance crossbar switch (HPS) of the SP2. See **Figure 5**. Each network-node processor can access any of the storage-node processors by using virtual shared disk [12]. This data-server configuration, based on the SP2, has flexibility of bandwidth and of both storage and network capacity and can provide scalability for data pumping.

As the number of processors and DASDs increases, the probability of system failure grows because of the increase in the complexity of the server system. In order to minimize the impact of failures, the storage devices are shared by the processors.

Even if the configuration of the network nodes and storage nodes were changed to meet various requirements, the interface to the server for application programs should not be changed, in order to maintain the compatibility for application programs.

System components

Figure 6 shows the components of the video-server complex we developed. All components are described in the following subsections.

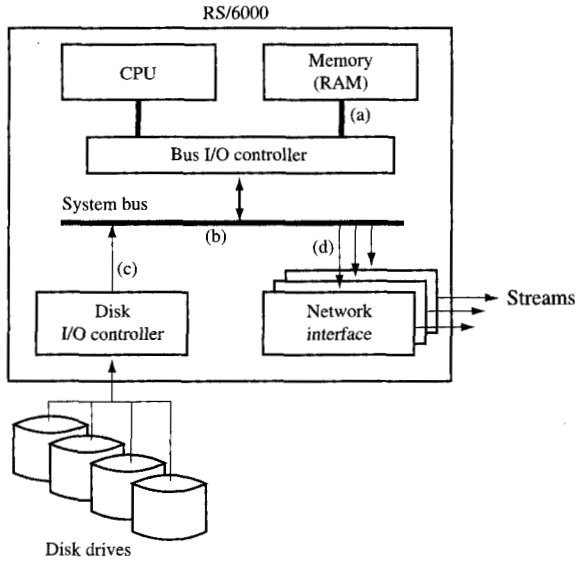


Figure 4

Single-processor-server configuration. An RS/6000 processor is used as both a storage node and a network node.

- *Data server*

This server component efficiently stores and retrieves data from the storage system and maintains integrity of the data. Each node of the data server contains the components shown in Figure 6. The data are delivered from the storage nodes to the network nodes and are output through the MPEG-2 stream-controller adapters plugged into the network nodes, under the control of the control server. In the case of 6Mb/s MPEG data, four data streams are multiplexed by the adapter and sent to a QAM modulator, so that one analog channel can send four concurrent streams to four separate subscribers. The data server can be configured for both single and multiple processors.

Tiger Shark file system

Tiger Shark [13, 14], a new file system for the AIX operating system, was developed at the IBM Almaden Research Center to provide optimized file I/O for real-time access in multimedia applications. The following are the unique features of Tiger Shark:

- File I/O with the same interface as the AIX virtual file system [15].
- I/O scheduling to maintain the specified read rate.
- Data placed over multiple disks (disk striping).
- Data placed over multiple SP2 nodes (wide striping).
- Reservation of disk I/O bandwidth.

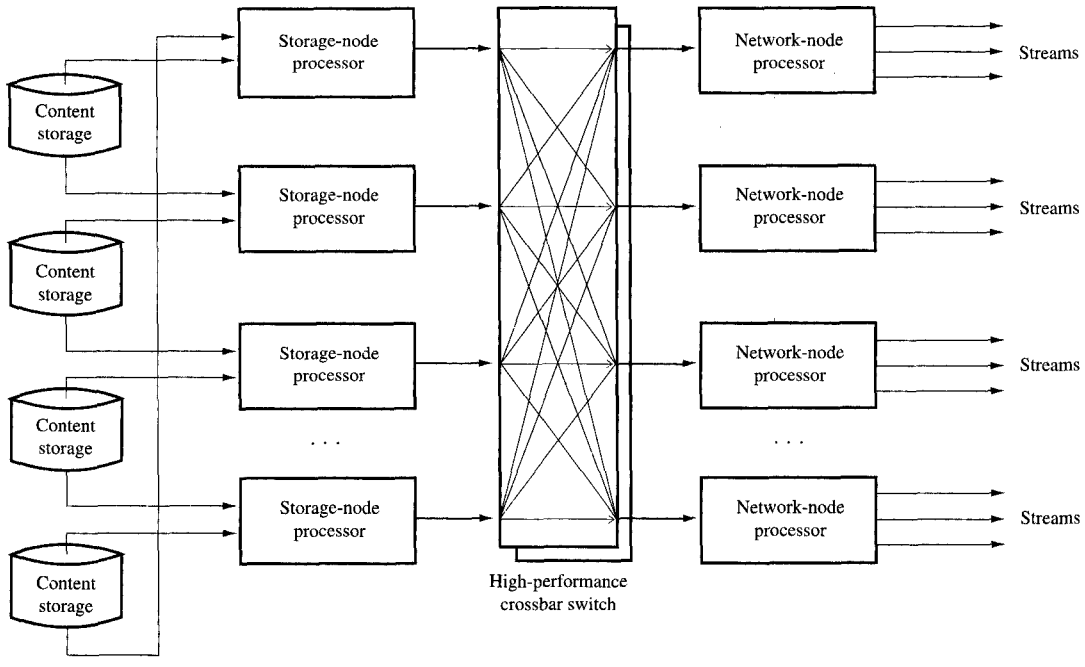


Figure 5

Configuration of multi-node data server in the video-server complex. All processors are RISC System/6000 scalable POWERparallel systems nodes.

- Large-block disk I/O.
- Use of SCSI or SSA [16, 17] disks.

In the case of multi-processor configurations, Tiger Shark runs on each network node and provides the same file-system image for each network node, with guaranteed stream delivery.

Calypso

Calypso [18], developed at the IBM Thomas J. Watson Research Center, provides catalog-management capability for realizing a single-system image for files and directories spread across multiple nodes of the SP2. Calypso is a high-performance, high-availability catalog server that uses extensive caching at clients and has a strong cache-consistency feature. This module is used only for multiple-node configurations of the SP2.

Real-time VSD

The real-time virtual shared disk (VSD) [19], developed at the IBM Thomas J. Watson Research Center, is the VSD product optimized for the transmission of real-time continuous data between nodes through the HPS of the SP2. In particular, this implements a deadline-management mechanism for disk I/O scheduling, permitting the read and write operations to storage to be processed within specified periods. This module is used only for multiple-node configurations of the SP2.

Data exporter

The data exporter, developed at the IBM Thomas J. Watson Research Center, performs the following operations, according to requests from the control server. The main objectives are the following:

- Open and close files on Tiger Shark.
- Set the read rate of files on Tiger Shark.
- Initialize network devices via the network access routine (NAR), described below.
- Read data from Tiger Shark.
- Write data to the network interface, such as the MSC (described in the following subsection), through the NAR.

MPEG stream controller (MSC)

The MPEG stream controller (MSC) is an adapter card, developed at the IBM Thomas J. Watson Research Center, that plugs into a microchannel-architecture-based

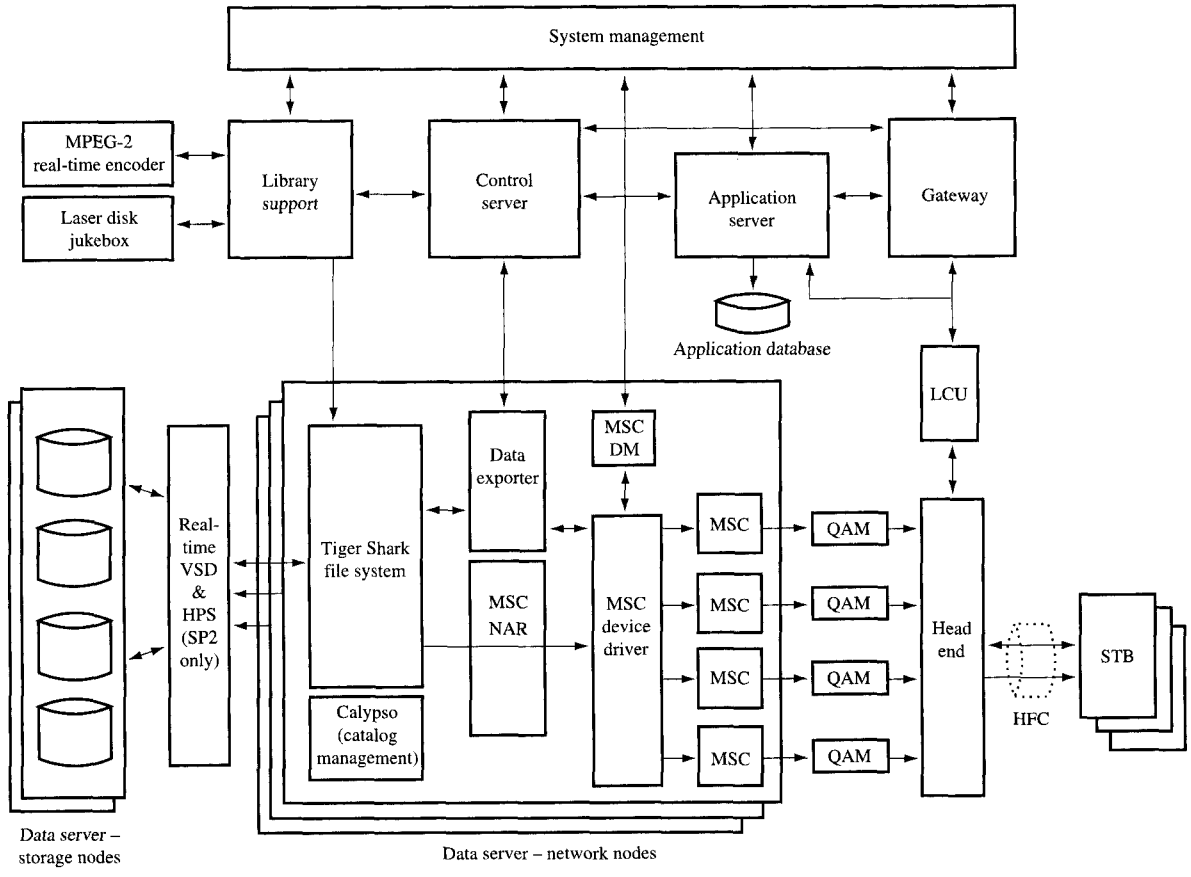


Figure 6

Elements of the video-server complex.

(MCA) RS/6000 system to multiplex MPEG-2 transport streams. It accepts multiple transport streams and outputs a multiplexed transport stream. The number of streams to be multiplexed, the bit rate of each input stream, and the PID[6] assignment in the output stream can be changed. To perform multiplexing, the MSC does the following:

- Schedules packet transmission.
- Swaps PIDs.
- Re-time-stamps PCRs.[6]
- Inserts new PSI[6] packets.

MSC network access routine

The data exporter has a generalized interface, called the network access routine (NAR), for accessing network devices. The MSC-NAR, a subclass of the generic NAR class, handles

- Management and generation of the PID swap table, which is used by the MSCs.
- Management and generation of PSI packets, which are periodically inserted into the multiplexed output stream by the MSCs.
- Initialization and control of the MSCs via MSC device driver IOCTL [20] calls.

MSC device driver

The MSC device driver for AIX

- Configures the MSC adapters.
- Sets up the MSCs for multiplexing streams.
- Sets up DMA for transferring data from memory to internal MSC buffers.

MSC device monitor

The device monitor gathers the information about status and errors that occur in the MSCs and network devices

[6] PID, PCR, and PSI are defined in the MPEG-2 standard and are described in the section on details of system design in this paper.

connected to MSCs and reports to the system management component (described below) by means of Simple Network Management Protocol (SNMP) [21] messages.

- *Control server*

The control server is the central control point of the system. It manages QoS for the delivery of streams across the data server nodes. This component manages system resources associated with MPEG-2 streams and controls data flow, upon request from the Application Server (described below), with no significant delay. Developed by the IBM Thomas J. Watson Research Center, the control server performs the following functions:

- Resource management such as allocation and release of disk I/O bandwidth and network I/O (HFC channel).
- Management of contents and associated meta-information.
- VCR-like play control.
- Provision of application programming interfaces accessible by means of the Distributed Computing Environment (DCE) [22] remote procedure call.

- *Gateway*

This is an interface component to the STBs through the LCU and the HFC network for the up channel. The gateway also manages LCU initialization. Communication between this gateway and the STB is performed by means of TCP/IP through the LCU. This component is capable of preliminary user authentication.

The gateway manages sessions (the period of interaction between client and server, e.g., the communication during the entire time a video is played) between the control server and STBs. Upon a logon request from an STB, the gateway performs authentication of the client, opens a session, and requests the application server to launch new services for the client.

- *Application server*

Any application processes other than content delivery (e.g., application-unique user authentication, billing, and database support) that cannot be processed on the STB are run on the application server.

The application server provides the following services based on DSM-CC according to ISO/IEC standards: stream (VCR functions), file (host file-system access), and view (database access). The application server receives a request from the STB with the Open Network Computing (ONC) RPC [23] protocol. When the request is one related to stream operations, it is redirected to the control server with the DCE RPC.

- *Library support*

Library support is the interface for importing encoded data from external devices to the data server and defining the imported data as a stream to the control server. External devices include MPEG-2 real-time encoders, digital laser-disk jukeboxes, and tape drives. Library support also extracts attributes of the stream to be imported, such as stream type, packet IDs, and bit rate. Attributes are stored as a part of meta-information maintained by the control server and used to set up the MSC to transfer the stream.

Unfortunately, AIX 4.1 has an upper limit on file size (2 GB). In the case of a long stream, such as a movie, library support splits the stream into several files, each of which is smaller than 2 GB, defines each file as a stream, and sets the sequence of split streams to the control server, which handles the sequence as a single stream. On AIX 4.2 and later versions, this is not necessary.

Library support also generates stream profiles used by the application server to support DSM-CC.

- *System management*

This component provides system control functions to the system operator. The functions include monitoring the status of each component, system initialization and termination, and enabling and disabling certain resources of the video-server complex. It provides a consistent, graphical user interface for operators to control the entire video-server system from a single RS/6000 system. This includes the following:

- Start and stop each component.
- Display component status.
- Display messages from the components.
- Display usage of channel resource and status of logged-on STBs.

- *Set-top box*

The STB issues requests on the up channel for required data from the server system and receives the data from the server system on either the up or down channel. The STB manipulates received data and outputs screen data to its NTSC output port for TV. The STB works with the server system to display contents the user wants to see and sends to the server system data input by the user (by means of an infrared remote control unit), such as personal identification number and selection of products for shopping. In order to meet the information-provider's requirements, such as easy user interaction and the ability to accommodate a large variety of applications on the STB, the STB provides a common service program for execution of the application program in the STB. This service program interprets the code of the application

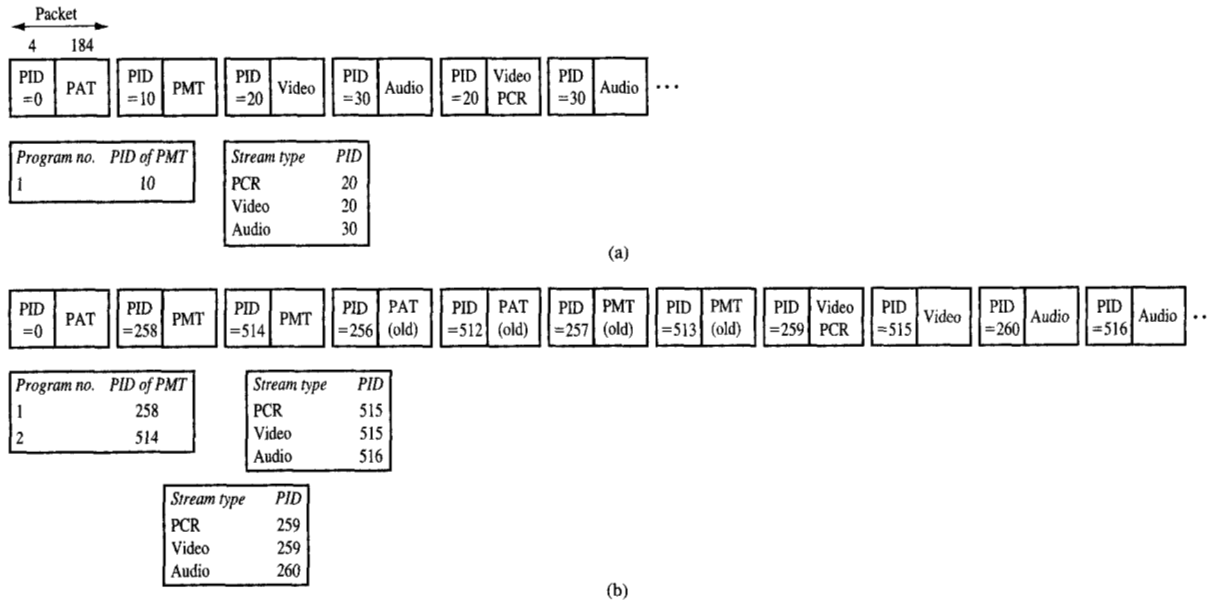


Figure 7

Multiplexing MPEG-2 transport streams: (a) Example of a single-program transport stream; (b) example of a multi-program transport stream created by multiplexing two single-program streams. In this example, the PCR is carried by the same packets that carry video streams.

program into DSM-CC commands. Details of this component are addressed in [11].

- *Other elements*

QAM modulators
By means of 64-value QAM, the modulators compress input signals with bandwidth up to 29.1 Mb/s into single 6Mb/s CATV channels.

Line control unit
Provides IP connection to STBs by using QPSK modulation, and acts as a level-1 gateway for the HFC network.

Head end
Combines multiple analog CATV channel inputs and generates an output signal to the HFC network. It also transfers up-channel signals from the HFC network to the LCU.

Encoder
Encodes audio/video source material and generates an MPEG-2 transport stream.

Details of system design

As described above, the system we designed complies with the MPEG-2 standard, which defines the syntax and semantics of a stream as well as the decoder model, but does not define the logic of encoding or multiplexing. Thus, we introduced a multiplexing scheme suited for interactive services; that is, it allows a stream to be added to or deleted from multiplexing during operation. In this section, we present some topics on the system design, focusing especially on multiplexing.

- *Multiplexing streams*

A stream that includes only video or audio is called an elementary stream. A set of elementary streams that share the same time base is called a program. In the MPEG-2 standard, two methods are defined for constructing a single program from elementary streams: program stream and transport stream. The transport stream method is used in the system we developed. **Figure 7(a)** shows the structure of a signal-program transport stream, which consists of 188-byte packets, each of which has a 4-byte header and a 184-byte payload. The header has a sync byte (47 in hexadecimal) and a 13-bit packet ID (PID). An elementary stream is split into packets, all of which have the same, unique PID. A program is defined in a transport stream by a table called the Program Map Table (PMT), which includes an entry for each elementary stream, each entry consisting of the type of the elementary stream and its PID. The PMT also includes the PID of packets that carry Program Clock References (PCRs). A PCR

represents a time stamp in the transport stream from which decoder timing is derived. A PCR can be carried in a packet by itself or in a packet that carries audio or video. In a transport stream, each PMT is transmitted in a packet that has a unique PID, different from all of the other PIDs in the stream.

In a multi-program transport stream, PMTs for different programs are transmitted in packets with different PIDs. The list of programs in a transport stream is given in a table called the Program Association Table (PAT), which includes a list of pairs, each consisting of the program number and the PID of the corresponding PMT. A PAT is always transmitted by a packet with a PID of 0. PATs and PMTs, called program specific information (PSI), are required to appear periodically in a transport stream, for example, every 0.5 second. **Figure 7(b)** is an example of the result of multiplexing two single-program transport streams of the sort shown in Figure 7(a) by the system we developed. This is done by the following steps:

1. Assign a new PID to each audio stream, video stream, and PCR, so that each new PID is unique in the multiplexed transport stream. If the PCR is carried with audio or video, the new PID for the PCR is the same as the new PID for the audio or video. [In the case of Figure 7(b), we assume that both input streams originally had the same PIDs (20 and 30). For the first input stream, 259 is assigned to video and PCR and 260 to audio; for the second input stream, 515 and 516 are assigned similarly.]
2. For each PMT, create a new PMT, which uses the PIDs assigned in step 1 instead of the original PIDs, and assign a new PID to the PMT. [In the case of Figure 7(b), we assume that PID 10 was originally used for both input-stream PMTs. For the first stream, 258 is assigned to the PMT, and 514 for the second. The new PMT for the first stream uses PIDs 259 and 260 instead of 20 and 30, and the PMT for the second uses PIDs 515 and 516.]
3. Create a new PAT that, instead of the original PIDs, includes the new PIDs assigned in step 2 for all programs (of course, the PID of the PAT is 0). [In the case of Figure 7(b), the new PAT uses PIDs 258 and 514 to specify the two programs.]
4. Assign PIDs not used in any program to the original PAT and PMTs, in order that they be ignored. The PAT and PMT packets included in the original streams are included in the output stream, in order to simplify the MSC process, but they are assigned PIDs that are not used in the new PAT and PMTs; thus, they are not accessed by any clients. [In the case of Figure 7(b), PIDs 256, 257, 512, and 513 are used in this way.]
5. During actual data transmission, replace all of the PIDs in the single-program input streams with the PIDs assigned in the preceding steps.
6. During actual data transmission, insert the new PAT and PMTs created in steps 2 and 3 at equal intervals, e.g., every 0.5 second.

The MSC-NAR performs steps 1 to 4 and updates a table called the PID swap table, which expresses the mapping between the PIDs found in the input streams and the new PIDs in the multi-program output stream. When a new program is added to a multiplexed stream, this table is updated according to the above steps and sent to the MSC, along with the new PMTs and PAT created by steps 2 and 3. The MSC swaps PIDs in the input streams with corresponding PIDs found in the PID swap table during actual data transmission. The MSC also inserts new PAT and PMT packets at equal intervals. When a stream is removed from a multiplexed stream, appropriate updating of the PAT, PIDs, and PID swap table takes place.

While the MSC is multiplexing streams, there might be a case in which a packet cannot be transmitted at the time it should be because a packet from another stream is being transmitted. In that case, the MSC schedules transmission of the packet so that it will be sent after completion of the current packet transmission. If the delayed packet includes the PCR, the MSC updates the PCR information with the new transmission time (this is called PCR re-time-stamp).

- *Decoding a stream*

When a client requests a new session from the server, the server searches for a channel in which the requested bandwidth is available and, if a channel is found, searches for a program number in the channel and returns the channel number/program number pair to the client. The channel number corresponds to a physical CATV channel, and the program number is the program number specified in the PAT of a transport stream. The channel number/program number pair is called a logical channel. When a client receives a logical channel, the client tunes the CATV analog tuner to the physical channel and sets its MPEG-2 transport demultiplexor to select the program specified by the program number. The client can request the play of more than one stream in a session, in which case the server transmits all of the streams requested by the client through the same logical channel. This means that the client does not need to adjust the CATV tuner once it is set in a session. Once transmission of a stream has begun, the PAT and PMTs are transmitted periodically (e.g., every 0.5 second), so that a client can start decoding at any time. Initially, the client finds a PAT packet (PID = 0) in the received transport stream and then selects the PID of the PMT corresponding to the program number. Before the server starts the actual data

transmission, the PMT packet is transmitted periodically, and the client finds a PMT packet and sets its decoder so that it decodes the audio, video, and PCR specified in the PMT.

• *Data transmission*

In a pull scheme, as in conventional file access, a client reads stream data sequentially from a server. In that case, the time for reading, the position in the file to be read, and the read rate are under the control of the client, but it is hard to maintain a high transfer rate. In push, which we adopted for our system, once the server receives a play request from a client, the server continues to transfer data at the specified transfer rate without interaction with the client (unless a new event, such as "end of stream," or a request from the client occurs). When a stream is opened, the read rate is sent to Tiger Shark; then the client issues a DSM-CC dsmStreamPlay request, and the control server calls the data exporter with meta-information associated with the stream. The data exporter creates two new threads, one for reading data from Tiger Shark and the other for writing data to the MSC via the MSC-NAR. Meta-information is sent to the MSC-NAR so that the MSC-NAR can update the PID swap table, create new PSI, and set up the MSC for data transmission, as described above in the subsection on multiplexing streams. Once data transfer starts, four processes work asynchronously: reading data in advance in Tiger Shark, reading data from Tiger Shark and writing to the MSC-NAR by the data exporter, data transfer to MSC internal buffers by DMA, and transferring data from the MSC to external network devices such as QAM. Each process tries to maintain a specified transfer rate, with the help of buffers of appropriate size. The transfer cycle continues until the end of the stream is reached, an immediate pause is requested from a client, or a specified pause position is reached (DSM-CC allows requests such as "pause at position X," "play to position X," and "play from A to X," which imply that the stream is paused when position X is reached).

• *Informing clients of stream status*

With the push method, a client cannot know about events happening asynchronously in the server, such as "end of stream" or "paused at specified position" without the server informing the client. There are two possible ways to inform the client of an event—using RPC callback from the application server to the client or sending a packet denoting the event, in an MPEG-2 transport packet. Using RPC callback assumes that clients have the capability of becoming RPC servers, but most client systems have few hardware resources for such an implementation. In our system, a transport packet is used to transmit event information to clients. The MPEG-2 standard defines the framework called "private section," to allow the introduction of new kinds of tables that can be carried in a transport packet. In our system, a new table is introduced to represent one of the following stream statuses: playing, paused, and end-of-stream. When the stream status is changed, the MSC-NAR updates the table so that it includes the new stream status. The private section, including the table, is converted to a packet whose PID is the same as that of the PMT, and the packet is passed to the MSC as part of the PSI. The MSC sends the private section periodically, just like PSI (for example, every 0.5 second); therefore, a client can pick the packet from a received transport stream and determine the current stream status.

• *Meta-information*

We call information about the original program, which is required for setting up the MSC to transmit the program, meta-information. When a program is defined in the control server, library support analyzes the program, creates meta-information, and defines the program with its meta-information. Meta-information includes

- Original PMT and its size (PMT size varies).
- PIDs of the packets that include the PMT and PCR.
- Number of elementary streams in the program.
- Stream type and PID of each elementary stream.
- Bit rate of the program.
- Private data.

Private data consist of

- Program status, as described in the previous subsection.
- NPT_reference, as defined in DSM-CC (start time and its PCR).
- Program ID, assigned by the control server.
- Copyright information (This may be also included in a PMT of the original stream.).

These are sent in a private section, which is transmitted by a packet that has the same PID as the PMT. Meta-information stored in the control server is passed to the data exporter and then to the MSC-NAR without modification when playback of the program is started.

• *Resource management by control server*

The control server provides two strategies to manage network resources (logical channels, in the case of HFC): dynamic allocation and static allocation. In static allocation, a logical channel is allocated at session-open time, and the requested bandwidth is reserved for the logical channel until the session is closed. All of the stream play in the session is performed by using the logical channel. In dynamic allocation, a logical channel is

allocated at each stream-open time instead of session-open time, and the bandwidth required for the stream is reserved until the stream is closed. In the case in which the up channels allow clients more than the number of down channels connected to the server, dynamic allocation will maximize the usage of down channels. For example, consider the case in which up channels have enough bandwidth for 200 clients, but down channels have bandwidth for only 100. In static allocation, the first 100 clients to request a session can get down channels, and the remaining 100 clients cannot get a down channel, even though some clients who received down channels are not playing streams. In dynamic allocation, the first 100 clients who request a stream open can get a down channel, but the channels become free when the streams are closed. In the Tokyo Metropolitan Government (TMG) video-on-demand system, the numbers of up channels and down channels are the same; thus, only static allocation is used, for simplicity. Please note that in dynamic allocation, it is not necessary that all contents (streams) defined in the server can be transmitted via any logical channel at all, because the server can select the logical channel used to transfer the stream. In static allocation, all contents should be able to be transmitted via any logical channel. This means that if the server system consists of multiple network nodes, content replication over network nodes or storage sharing is required. In the SP2 configuration of our system, VSD is used for sharing storage, and all contents are accessible from any network node.

When allocating a logical channel, the control server also considers the load balance among network nodes, so that all network nodes have the same load level as nearly as possible [24].

- *PID assignment*

PID re-assignment in multiplex streams is basically dynamic. That is, the new PID for each added elementary stream is not fixed. In some applications, such as digital broadcasting, however, it is necessary that certain kinds of elementary streams should always be mapped to the same PID. We implemented the PID-assignment logic so that the system can be applied to a wide range of applications, not only for interactive multimedia but also for digital broadcasting or near-video-on-demand.

- *Application flexibility*

To make it possible to build flexible applications on our system, we introduced two techniques: downloading programs and data by private stream, and object-oriented application server (service) definition. In the TMG system, a private stream format is defined and used in downloading programs and data, within the framework of the MPEG-2 standard. The private stream is transmitted in the same way that audio and video streams are. Unfortunately, at this point, the private stream is specific to some client systems or STBs, but the technique of downloading by private stream is important because it provides a way to download programs and data rapidly.

The application server, defined by an object-oriented approach adopted in the DSM-CC standard, provides a generalized service called DefaultService. The service provider can create new services by making new classes that inherit the DefaultService. For example, DefaultService does not provide an accounting function, but a provider can create a pay-per-view service by adding an accounting process at stream-open or play time.

- *Service launching and unlaunching*

Figure 8 shows how end-to-end service is established in the system, especially focusing on the application server and the gateway. Initially, a client knows only the IP address of the gateway host, and the RPC program number/version number of the gateway, which are predefined in the system. When a client requests a new service, it issues a DSC-CC ServiceGatewayAttach call to the gateway by RPC, using the gateway's IP address, RPC program number, and version number. The gateway has a table that specifies clients that can access the server; for each client, the table lists the accessible services, the default service (the first application service invoked when a client is connected), and the subnet to which the client belongs. When the gateway receives the request, it authenticates the client according to the table and then requests a session with fixed bandwidth (6.1 Mb/s[7] in the TMG system) from the control server. If the bandwidth is available, the control server reserves a logical channel and the requested bandwidth, and returns the logical channel identification to the gateway. Next, the gateway calls the application server to get an object reference[8] of the default service associated with the client and returns it to the client. The object reference is actually implemented as a structure that consists of the IP address of the machine where the service is running and the RPC program number and version number of the service. After receiving the object reference, the client sends all DSM-CC requests except the ServiceGatewayDetach directly to the service object, by IP address and RPC program number extracted from the object reference. At the first request from the client, the service object forks a child process, which performs all the work requested from the client.
In order to terminate the session, the client issues ServiceGatewayDetach to the gateway, which calls the application server to unlaunch the currently running service. The application server sends an AIX SIGTERM

[7] All streams are encoded at 6 Mb/s; however, in some cases, the rate may exceed 6 Mb/s by a small amount. To reduce the workload of retrying the encoding, every session is opened with a bandwidth of 6.1 Mb/s.
[8] DSM-CC is based on Common Object Request Broker Architecture [25] and treats services, streams, files, etc. as objects.

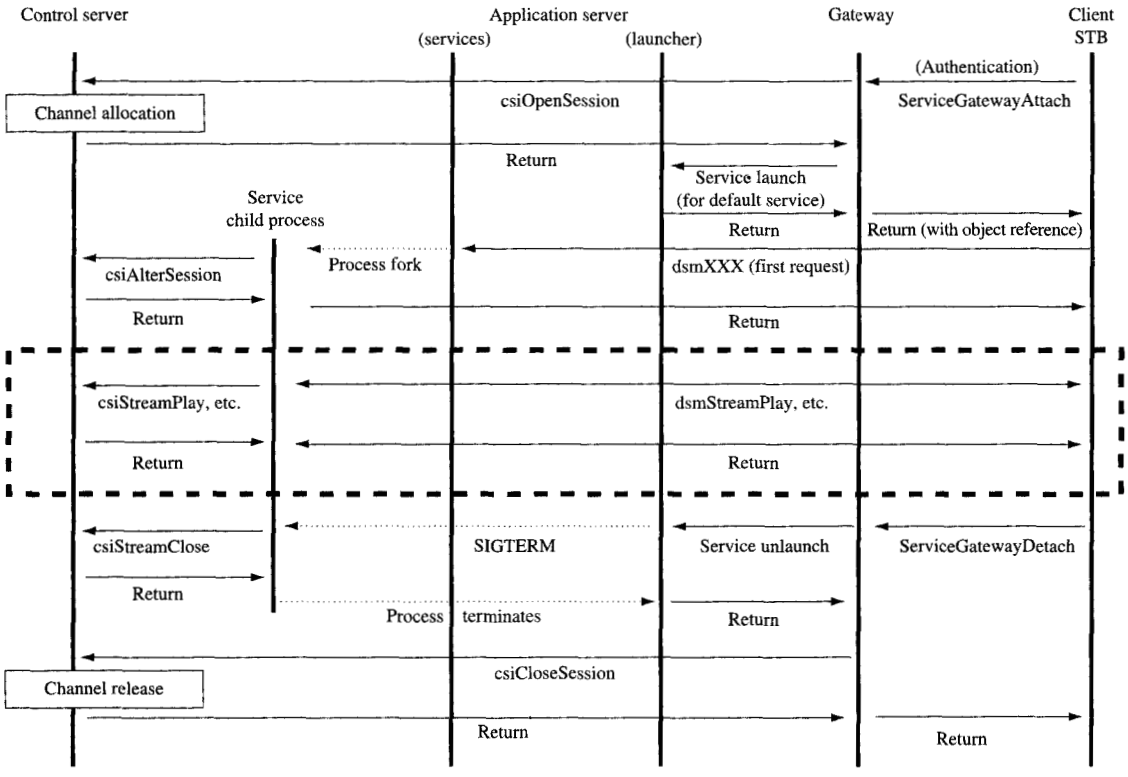


Figure 8

Control flow for service launching and unlaunching.

signal to the corresponding child process in order to "kill" the child process. If a stream is still open, the child process closes the stream before termination. Then, the gateway calls the control server to close the session, thus releasing the logical channel and bandwidth.

Please note that communication among the gateway, the application server, and the control server is performed by DCE RPC, but communication between a client and servers is performed by ONC RPC. Also note that it is not required that all of the application services run on the same machine. We can configure the application server so that services run on multiple machines, thus providing good performance even though the application server supports many clients.

Evaluation of the system

Performance measurements were made of single- and multi-node systems. The evaluated systems comprised storage nodes, network nodes, and other RS/6000 computers that executed other software components such as the control server, the application server, and the gateway.

• *Performance of single-node system*

Figure 9 shows CPU utilization measured on a single-node system, an RS/6000 Model 58H workstation. This single node performed both storage and network functions, including the Tiger Shark file system and the data exporter. One stripe group was created over sixteen 4.5GB Serial Storage Architecture (SSA) disks connected in a single SSA loop. (The Tiger Shark installation procedure performs a simple continuous-read performance test from the stripe group; the result was about 35 MB/s[9] in the test configuration.) For the measurement, up to sixteen 6Mb/s streams, all using the same data file, were transmitted by four MSC adapter cards (four streams per adapter card). Most of the "user" portion of the CPU time was due to the data exporter, and most of the "system" portion was due to Tiger Shark and the MSC device drivers. When sixteen 6Mb/s streams are multiplexed, the bus traffic is at least 24 MB/s, which is 30% of the 80MB/s MCA bus peak bandwidth [26]. (In some RS/6000 models, such as SP2, the peak bandwidth is 160 MB/s.) Under this condition,

[9] MB/s = mega*bytes* per second.

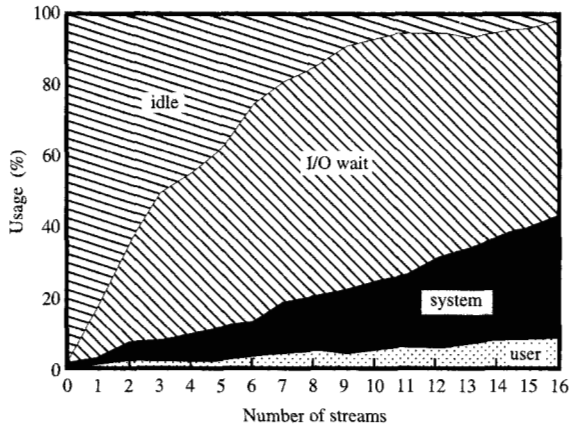


Figure 9

Measured CPU utilization of a single-node system (RS/6000 Model 58H, 55.5 MHz, 128MB memory, disks: SSA 4.5GB × 16 in an SSA loop, MSC × 4).

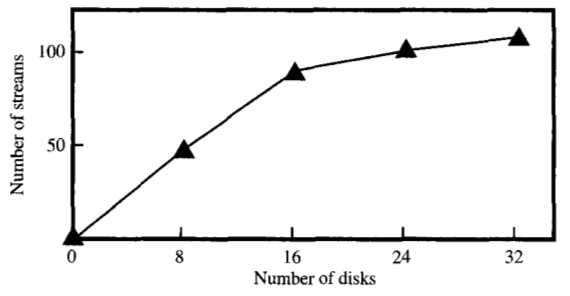


Figure 10

Measured maximum number of streams that can be read simultaneously on system with four storage nodes (SP2 nodes, 66 MHz, 128MB memory).

45% of the CPU was used, and 55% of the CPU time was spent in the I/O wait state. During the measurement, streams transmitted from the MSCs were successfully decoded by a client system; that is, no buffer underflow or overflow occurred.

- *Performance of multi-node system*

For the evaluation of a multi-node system, we also configured an SP2-based system, consisting of seven network nodes and four storage nodes. Each network node had four MSC adapter cards and performed transmission of up to sixteen 6Mb/s streams. Each storage node had two SSA adapters, each connected to a different SSA loop containing sixteen 4.5GB SSA disks and performing disk I/O for half of the disks in each loop (i.e., each storage node is connected to 32 disks and performs I/O for 16 of them). One stripe group was created over the 32 disks of the four storage nodes.

1. *Storage-node CPU utilization*
   The average CPU utilization of each storage node when the system was reading 100 6Mb/s streams from the network nodes was

user	system	idle + I/O wait
0.1%	30.5%	69.4%

   This shows that the storage nodes had enough CPU capacity for the transmission of 100 6Mb/s streams.
2. *Storage-node capacity*
   Figure 10 shows the relation between the number of SSA disks in the stripe group and the number of 6Mb/s streams that can be read from the network nodes before the actual read rate becomes degraded. It shows that the system with four storage nodes allows more than 100 streams to be read simultaneously, but it cannot support a significantly greater number of streams by adding disks to the stripe group. That is, one storage node of the SP2 can support more than 25 6Mb/s streams.
3. *Peak read performance over switch*
   When data from a stripe group were read from the network as rapidly as possible, we measured 70MB/s peak read performance, and when data were read from local SSA disks on the RS/6000 Model 58H workstation, the performance was limited to 35 MB/s. Thus, it can be concluded that the SP2 nodes and switch are fast enough to transfer streams instead of reading data directly from SSA storage on an RS/6000 such as the Model 58H.
4. *Network-node performance*
   The CPU utilization on the network nodes, which transfer sixteen 6Mb/s streams by four MSC adapter cards (i.e., four streams per adapter), was as follows:

user	system	idle + I/O wait
6.0	30.6%	63.4%

   In the case of the measurement of the single-node system, the system portion was about 35% of the CPU usage (see Figure 9). In the cases of the storage node and the network node, the system portion has almost the same value as for the single-node system. Thus, the overhead of the SP2 switch is very small.

- *Flexibility and scalability*

We subsequently built four systems, which are in actual use. They have the following different requirements and configurations:

1. Eight streams delivered by a single RS/6000 system with SSA disks.
2. 100 streams delivered by five storage nodes and seven network nodes on an SP2 system with SSA disks.
3. Ten streams delivered by two storage nodes and two network nodes on an SP2 system with SCSI-2 disks.
4. 32 3Mb/s streams delivered by a single RS/6000 system with SSA disks; the control server runs on the same RS/6000.

These facts demonstrate the flexibility and scalability of the system.

Summary

The scalable video server provides the capability of delivering a large number of MPEG-2 streams with no significant change in the architecture from small to large system. This delivery mechanism is useful not only for interactive multimedia services but also for digital broadcasting services.

This video-server complex has already been deployed in several interactive television projects and is providing many multimedia applications through an HFC network. In particular, the video-server complex used in the TMG project is currently providing a wide variety of interactive applications, such as movies-on-demand, karaoke-on-demand, news-on-demand, home shopping, public information services, and language education, to 500 households. This gives one confidence in the validity of the design and the implementation of the video-server complex.

Acknowledgments

The system design, implementation, and deployment of the video server have been a complex and lengthy effort involving many talented people in several organizations of the IBM Corporation worldwide. Many individuals from the Research Division, product divisions, and marketing organizations have contributed to the substantial technical effort of offering the system to real customers. The authors express their appreciation to Ahmed N. Tantawy and Frank A. Schaffa for the MSC, Roger L. Haskin for the Tiger Shark file system, Martin G. Kienzle and Marc Eshel for the control server, Kalman Meth for the data exporter, Daniel Dias and Rajat Mukeerjee for the real-time VSD, Muthy Deverakonda for Calypso, Hisa Yamasaki and Rikiyo Imai for the MSC device management, Tsukasa Takemura and Kohji Hashimoto for the application server, Hikaru Tanabe for the gateway, Akira Ogasawara for the system management, Shingo Ohmori and Emiko Kohyama for the library support, Peter Schirling for the MPEG-2 implementation standard, Jeffrey S. Lucash for the SP2 architecture, and Hiroshi Maruyama for the design work.

References

1. Tekla S. Perry, "The Trials and Travails of INTERACTIVE TV," *IEEE SPECTRUM*, pp. 22–28 (April 1996).
2. Daniel Minoli, *Video Dialtone Technology—Digital Video over ADSL, HFC, FTTC, and ATM*, McGraw-Hill Book Co., Inc., New York, 1995, pp. 229–269.
3. "Information Technology—Generic Coding of Moving Pictures and Associated Audio Information—Part 1: Systems," *ISO/IEC 13818-1*, International Standard, 1994.
4. "Information Technology—Generic Coding of Moving Pictures and Associated Audio Information—Part 6: Digital Storage Media Control Command," *ISO/IEC 13818-6*, Draft for International Standard, 1995.
5. Basic Plan for the Multimedia Experiments at the Tokyo Metropolitan Waterfront Subcenter Multimedia Experiments, *http://www.tokyo-teleport.co.jp/english/atms/atmsj201.htm* .
6. Ministry of Post and Telecommunication, "Technical Requirements with Regard to Digital Broadcasting Systems," *Committee Report No. 74*, Ministry of Post and Telecommunication, Tokyo, Japan, 1995.
7. P. Venkat Rangan, Srihari Sampath Kumar, and Sreerang Rajan, "Continuity and Synchronization in MPEG," *IEEE J. Selected Areas in Commun.* **14,** No. 1, 52–60 (January 1996).
8. Digital Audio-Visual Council, "DAVIC 1.0 Specification Part02, System Reference Models and Scenarios," Technical Report of DAVIC, Geneva, Switzerland, 1996.
9. A. Mourad and Y. T. Wang, "Architecture and Performance Issues of Media Server and Content Delivery," *Proc. SPIE* **2615,** 182–194 (1996).
10. Robert C. Hutchinson, "Architectural Framework for Standardizing Multimedia Services on Fiber-Coax," *Proceedings of the 1st International Workshop on Community Networking*, IEEE, 1994, pp. 213–217.
11. T. Ohki, H. Kinoh, H. Ohue, I. Kanno, and M. Kitao, "Full Interactive Digital CATV System," *National Tech. Rep.* **42,** No. 5, Osaka, Japan, 1996, pp. 8–15.
12. T. Agerwala, J. L. Martin, J. H. Mirza, D. C. Sadler, D. M. Dias, and M. Snir, "SP2 System Architecture," *IBM Syst. J.* **34,** No. 2, 152–184 (1995).
13. R. L. Haskin, "Tiger Shark—A Scalable File System for Multimedia," *IBM J. Res. Develop.* **42,** 185–197 (1998, this issue).
14. R. L. Haskin and F. B. Schmuck, "Tiger Shark File System," *Proceedings of the IEEE COMPCON Conference*, 1996, pp. 226–231.
15. International Business Machines Corporation, *IBM RISC System/6000 AIX Version 3, Files Reference*, 1991, Order No. GC23-2200; available through IBM branch offices.
16. International Business Machines Corporation, *7133 SSA Serial Storage Architecture (SSA) Disk Subsystem*, 1996; Order No. G225-6641; available through IBM branch offices.
17. SSA Industry Association, *Serial Storage Architecture, A Technical Overview*, 1995; *http://www.ssaia.org/docs/docs/white_paper.pdf* .
18. Ajay Mohindra and Murthy Deverakonda, "Distributed Token Management in the Calypso File System," *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, New York, 1994, pp. 290–297.
19. Renu Tewari, Rajat Mukherjee, Dan Dias, and Christos Polyzois, "Read-Time Issues for Clustered Multimedia Servers," *Research Report RC-19819*, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, 1994.

20. International Business Machines Corporation, *AIX V4.1 Kernel Extensions and Device Support Programming Concepts*, 1996; Order No. SC23-2611; available through IBM branch offices.
21. William Stallings, *SNMP, SNMPv2, and CMIP: The Practical Guide to Network Management Standards*, Addison-Wesley Publishing Co., Inc., Reading, MA, 1993.
22. John Shirley, Wei Hu, and David Magid, *Guide to Writing DCE Applications*, 2nd edition, O'Reilly & Associates, Inc., Sebastopol, CA, 1994.
23. John Bloomer, *Power Programming with RPC*, O'Reilly & Associates, Inc., Sebastopol, CA, 1991.
24. A. Dan, P. Shahabuddin, D. Sitaram, and D. Towsley, "Channel Allocation User Batching and VCR Control in Video-On-Demand Systems," *J. Parallel & Distr. Computing* **30,** 168–179 (1995).
25. Object Management Group, "The Common Object Request Broker: Architecture and Specification, Revision 2.0," Object Management Group, 492 Old Connecticut Path, Framingham, MA 01701. E-mail: *info@omg.org* .
26. Dan M. Neal and James O. Nicholson, "Micro Channel Features," *PowerPC and POWER2 Technical Aspects of the New IBM RISC System/6000*, Order No. SA23-2737, IBM Corporation, Austin, TX, 1994, pp. 200–204.

Received December 19, 1996; accepted for publication January 20, 1997

Toshiyuki Sanuki *IBM Japan Ltd., Network Computing, 1623-14, Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (sanuki@jp.ibm.com).* Mr. Sanuki is a Senior Architect for Video Systems in Network Computing in Japan. He is currently involved with the Department of Digital Technology Development for establishing broadband network solutions. He received his M.S. degree in medical science in 1983 from the graduate school of Tsukuba University, Japan. He joined IBM in 1983 as a researcher at the Japan Science Institute (formerly the Tokyo Research Laboratory). Since then, he has participated in the design of a programming language and the development of its applications. Since 1992, he has led projects on multimedia product development at Multimedia Operations and has been a product manager of multimedia platforms, for Asia Pacific. Since 1994, Mr. Sanuki has been the technical leader of several video-on-demand projects in Japan, and has been responsible for system design of broadband network solutions. He is the co-author of several books in the computer science area. Mr. Sanuki is a member of the IEEE and the Information Processing Society of Japan. He is also an editorial committee member of the Annual Multimedia Whitepaper of the Ministry of International Trade and Industry, Japan.

Yasuo Asakawa *IBM Japan Ltd., Network Computing, 1623-14, Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (jl05032@yamato.ibm.com).* Mr. Asakawa is an Advisory I/T Specialist of Client/Server. He received his B.E. and M.E. degrees in computer science from the Tokyo Institute of Technology in 1982 and 1984, respectively. Upon graduation, he joined the IBM Tokyo Research Laboratory, where he worked on Prolog compiler projects and a workflow management project. From 1991 to 1994, he worked in multimedia development at the Yamato Laboratory. Since 1994, he has been working on the Open Client/Server, which is currently called Network Computing, at Tokyo, especially on development of video-on-demand systems. Mr. Asakawa's current interests are in multimedia, network computing, and information infrastructures. He is a member of the ACM, IEEE, Information Processing Society of Japan, and the Japan Society for Computer Science and Technology.