Design of System for Multimedia Streaming Service

Chang-Soo Kim[1], Hag-Young Kim[1], Myung-Joon Kim[1], and Jae-Soo Yoo[2]

[1] Electronics and Telecommunications Research Institute,
161 Gajeong-dong, Yuseong-gu, Daejeon, 305-370, Korea
{cskim7, h0kim, joonkim}@etri.re.kr
[2] Division of Information and Communication Engineering, Chungbuk National University,
48 Gaesin-dong, Cheongju, 361-763, Korea
yjs@cbucc.chungbuk.ac.kr

Abstract. In the current life, Internet is playing a key role in transferring diverse information. As the performance of Internet has been growing, transferred information has the form of multimedia streams such as video and audio. However, in current computing and internet environment, multimedia streaming services can't be accomplished smoothly due to the various factors such as internet bandwidth, computing power and so on. In this paper, we are proposing an efficient system including hardware and software in terms of the design concept and system architecture for the multimedia streaming service. We also present the implementation result for the system.

1 Introduction

As computers and high speed networks have been popularized, Internet becomes a main means of transferring information over the world in our lives. As communication networks to the home will be upgraded to 10 ~ 100Mbps bandwidth level, high quality multimedia services such as internet broadcast, remote medical services and internet video services will be more generalized [1]. For example, a FTTH (Fibre To The Home) village is planned to be constructed in Korea. Those high quality multimedia services require a system that can handle multimedia streaming services over network fast.

In general, networking operation has high cost in computer system in that it requires several data copies from one place to another and processing steps whenever data passes from one protocol layer to another [2], [3]. In order to provide efficient multimedia streaming services over network, we need to minimize the cost of networking operation. Also, service providers should deliver multimedia contents to end users with stability and efficiency over network in real time [8], [9], [10], [11]. As service requests are increased, the traffic of a back-bone network increases, and as a result, the construction cost for the back-bone network increases. This problem should also be resolved.

In this paper, we are proposing an efficient system including hardware and software in terms of design concept and system architecture for the multimedia streaming

service. We also present the result of implementation and performance evaluation for the system.

2 Design Concepts and Goals

2.1 Design Concepts

The proposed system is a system specialized to internet-based multimedia streaming services with high-performance communication infrastructure. In order to provide high quality multimedia streaming services, high network bandwidth is required. Also, efficient management of infrastructure and distribution of service requests are required to reduce back-bone traffic and server loads respectively.

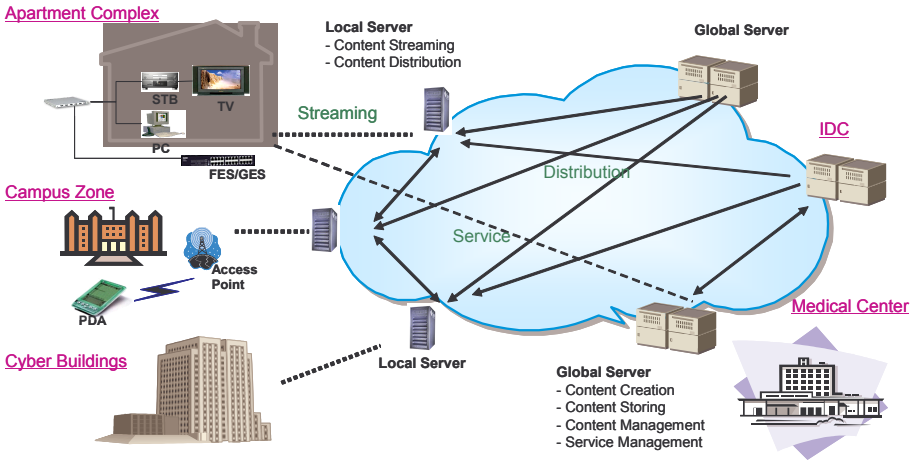

Fig. 1. Service Diagram of the System

Fig. 1 shows the service diagram of the system. Our system hierarchically distinguishes services into global and local services that are processed separately to use network bandwidth efficiently. A global server is a computer system that provides overall multimedia contents and located at Internet Data Center (IDC), e-government and so on. While a local server is a computer that provides multimedia streaming services for local communities such as apartments, companies, universities that are based on local networks. It transparently communicates with a global server. The global server should be designed to provide various functionalities for contents service and to guarantee high scalability as well as high availability. And, the local server should provide not only single-node server solution to achieve cost-effectiveness and high-performance but also multiple-node cluster server solution to build scalable and high available system.

2.2 Design Goals

As first phase, before overall system including the global server is designed, we are focusing at local server solution. Therefore, in this paper, we use the term "server" for a local server. In this case, the role of the global server as contents provider must be simulated. For this role, we will introduce CSN (Contents Server Node) node in our system infrastructure. In order to satisfy the requirements described at section 2.1, the server has following design goals.

- The server supports up to 200 clients and provides up to 4 Gbps of concurrent streams, which is equivalent to MPEG streaming service with the bit rate of 20 Mbps.
- The server provides not only clustering solution to build scalable and high available system but also efficient content placement mechanism for distributed local servers in a cluster without shared storage.
- The server provides an efficient content distribution mechanism for storing the right content at the right time.
- The server provides an optimized multimedia file system.

In order to satisfy these goals, the server should be designed to improve performance in storage including efficient multimedia file system and networking system. Also it should support efficient clustering and content distribution features. In storage and file systems, it is needed to support continuous disk access at high speed. In networking systems, stream data stored at disks should be delivered to clients through the network in a zero-copy fashion. In clustering solution, the server should provide single system image for the cluster and efficient load balancing features between participating local servers. And, in content distribution solution, the server should provide an efficient content distribution mechanism from CSN to local servers.

2.3 Design Issues

In general, normal computer has high level of CPU utilization for networking operations with gigabit level. The reason is that complicated communication protocols are handled in host CPUs. Therefore, using a separate processor for handling networking operations in offload fashion will be useful [2], [3]. Another reason of degradation of system performance for multimedia streaming services is several data transfers and copy operations in disk-to-network path. These several copy operations are the main performance bottlenecks [3]. Therefore, several data copy avoidance architectures such as integrated layer processing [4] and direct I/O to eliminate memory-to-memory data copy are suggested trying to minimize data transfers and copy operations. We will design a special PCI card that integrates SCSI controller, TOE engine and Memory module in order to reduce CPU utilization and copy operations.

In addition, because large volume of data storage is needed for multimedia streaming services, it is also required to use striping technique such as RAID (Redundant Array of Inexpensive Disks) for fast disk reading speed [3]. The size of a streaming data is relatively large and the access pattern has continuous form. So, handling

streaming data requires developing a multimedia file system that can consider the characteristics of continuous form and large size of streaming data.

According to design goals, a local server is designed to support up to 200 concurrent streams with 20 Mbps bit rate. The regional community where proposed system provides multimedia streaming services has several range of concurrent user size. From this, the server should support scalable solution. General scalable solution is clustering technique. There are several clustering related works [5], [6], [7]. The proposed server has two-level cluster architecture in that the cluster consists of several server nodes and each server node can have several NS cards which will be described at section 3.1. Also it is formed like shared nothing architecture between server nodes, and in turn between NS storages of each server node. The clustering solution should support these shared nothing and two level features.

Even though the server provides large capacity of storage, all of the multimedia contents to be serviced cannot be stored at the server. As described at section 2.1, the global server plays the role as contents provider. It is important to store the right content in the right place at the right time to satisfy client's requests. Also, when a requested content is not locally available, it should be obtained from the global server or the CSN node as fast and reliably as possible. In order to support these requirements, the server should provide an efficient content distribution mechanism.

3 System Architecture

3.1 Hardware

The server consists of a conventional mother board with 2 processors and one or more special purpose PCI cards called as NS (Network-Storage) Card. The number of NS cards is ranged from one to four. The NS card is implemented as a PCI card having a PCI-to-PCI bridge, disks, a PMEM (PCI Memory), a TOE (TCP/IP Offload Engine) and two Gigabit Ethernet ports.

Fig. 2 (a) illustrates the architecture of the NS card's hardware unit. The PCI Bridge connects the local PCI bus of the NS card to the host PCI bus and transmits data directly to the network interface without disturbing host PCI bus. The Memory unit supports up to 512MB capacity of PCI memory (PMEM). The PMEM is utilized as memory space between the TOE and disks in order to support the zero-copy mechanism. The TOE is TCP/IP Offload Engine that plays the key role in offloading the TCP/IP protocol processing from the host CPU.

Fig. 2 (b) shows the software unit for the NS card, which is required for correct operations of the hardware and making its capabilities to be available to applications. The Stream Disk Array Unit supports pipelined I/O and data split mechanism, which is similar to striping technique in RAID system, to provide fast I/O of disks.

The size of a media file used in multimedia streaming services is very long. In addition, the access pattern of the media is sequential. In this case, large size of I/O unit is very helpful to provide good I/O performance. With consideration of those situations, the proposed server takes very long block size for disks. The block size can be 128K, 256K, 512K, 1M or 2Mbytes. The Linux I/O subsystem for block devices

supports up to 4Kbytes block size in its kernel. This makes a conflict between the block size that NS card supports and Linux kernel can support. Because of this difference between block sizes, the server should utilize the NS character device for fast and efficient disk I/O. The SDA Block Device is also provided for compatibility of Linux. The PMEM can be used at user application directly through PMEM allocation library.

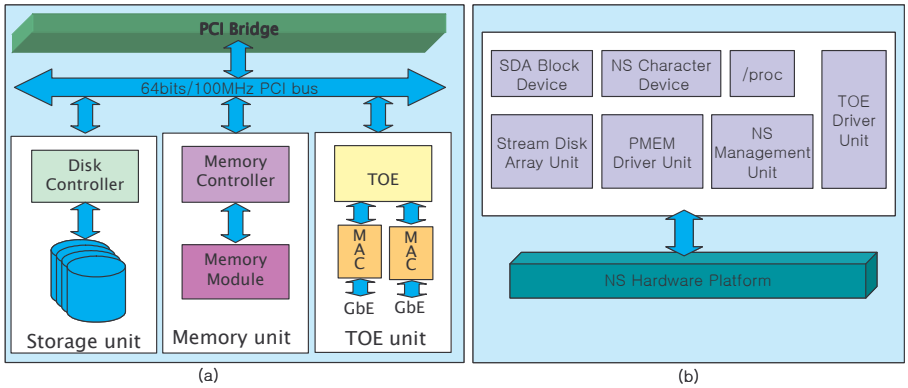

Fig. 2. Architecture of the NS Card's Hardware (a) and Software (b) unit

3.2 Multimedia File System

For high-performance multimedia streaming services, we need a multimedia file system that can utilize the facilities of the NS card and supports large volume of data, a file with large size and the access pattern of continuous form efficiently. The multimedia file system specialized to the proposed server has following features including large volume and large file size supports.

Optimized to the NS Card. The file system should handle large block sizes for high performance of disk I/O. In order to achieve this goal, the file system manipulates data through the NS character device and the PMEM.

Minimized Initial Access Time and Efficient File System Recovery. General file system maintains an unbalanced inode structure. This is optimized to small sized files. However, the size of streaming data is very large, and thus the unbalanced inode structure makes performance degradation in accessing multimedia streaming data having large size and cannot guarantee even initial access time. The proposed file system maintains the inode structure as a modified B+ tree structure. Also, the multimedia file system supports log-based fast recovery for maintaining consistent state of the file system at system failure.

3.3 Content Distribution

Even though the server provides large capacity of storage, all of the multimedia contents to be serviced cannot be stored at the server. Also, if services are processed globally, the traffic of the global network infrastructure increases. This increased traffic makes the service to be unreliable or unavailable. To resolve this problem, a service provider generally adopts a CDN (Content Delivery Network) technique that locates many local servers or cache servers at various specific regional areas. In a CDN infrastructure, multimedia streaming services are provided with end users through a nearest local server [8], [10], [11].

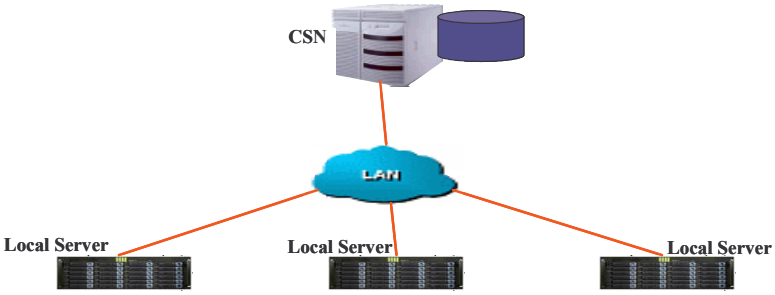

Fig. 3. Service Environment

As described at section 2.1 and 2.2, the global server plays the role as contents provider, and we introduce the CSN (Contents Server Node) node concept for replacing the global server. Fig. 3 shows the service environment for a specific regional community. With this environment, the CSN node is equipped with contents library having very large shared storage.

It is important to store the right content in the right place at the right time to satisfy client's requests. Also, when a requested content is not locally available, it should be obtained from the CSN node as fast and reliably as possible. In order to support these requirements, the server should provide an efficient content distribution mechanism. The more the server caches streaming media, the more the server can provide services to users. For this reason, our server system also adopts prefix caching mechanism for making more services to be available on the spot.

The content distribution software provides following features for versatile transfer and distribution methods, and content usage monitoring.

Content Transfer. The content distribution unit transfers contents from the CSN node to a local server. The content transfer job can be executed at promptly (On-Demand Transfer) or at delayed scheduled time (Scheduled Transfer).

Dynamic Loading. When content is requested to be serviced and if the content is not available completely, the content distribution unit transfers the content dynamically.

Preloading and Prefix Caching. The content distribution unit provides a mechanism to preload contents from the CSN prior to service requests by using either on-demand or scheduled transfer. For better chance of meeting user requests, it is desired to store as large number of contents as possible. This is a major reason of prefix caching. The content distribution unit also supports preloading with prefix size.

Content Purge. When a content transfer job is activated and if the storage has no space to store the content to be transferred, the transfer job will be failed. For preventing this situation, the content distribution unit prepares needed storage space in advance by purging some contents existed in the storage.

Content Storage Management. For determining the necessity of purge operation, the content distribution unit should know the status of storage.

Content Usage Monitoring. Selection of appropriate contents to cache at the server can affect the cache utilization and thus service performance. The unit should provide information about usage of contents.

3.4 Clustering Support

In order to response to user requests fast and correctly without creating unwanted delays, the server should be hosted on a cluster consisting of multiple streaming servers. Actually, the clustering technique is widely recognized to be a viable solution to build a scalable high-performance server system. The clustering unit should distribute all incoming service requests efficiently across the multiple streaming server nodes in the cluster environment. Also, it should place contents effectively across the multiple streaming server nodes for efficient and balanced resource consumption. The clustering software provides following features for efficient request dispatching and content placement.

Request Distribution for Virtual Single System Image. Even though the streaming service server is configured as a cluster, it is needed for the appearance of the streaming service server to be single system image. For single system image, the cluster has a representative virtual IP. Every client can connect to only the virtual IP, and then clustering software unit distributes all incoming service requests efficiently across the multiple streaming service nodes. In addition, each server node can have up to 4 NS cards. Since the NS card has its own IP address, pumps streaming data independently and selects shared nothing architecture, each NS card could be regarded as a distinct virtual sub-node. In other words, we should consider two-level clustering support as depicted in Fig. 4.

The request distribution called as dynamic scheduling is supported in fashion of 4-way redirection. Handling of a request in 4-way redirection fashion has following procedures as depicted in Fig. 5:

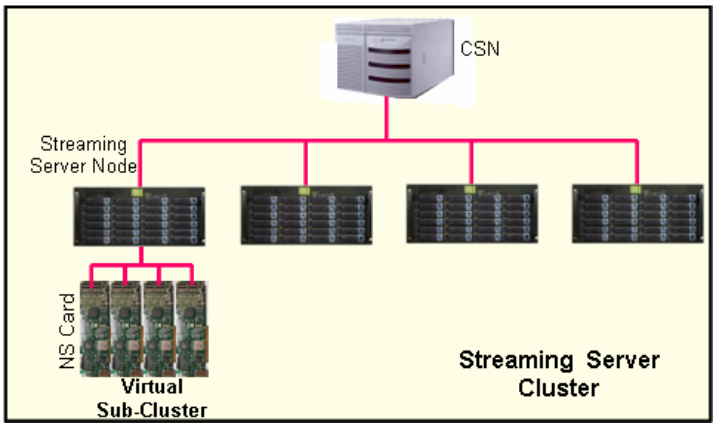

Fig. 4. Two-Level Clustering

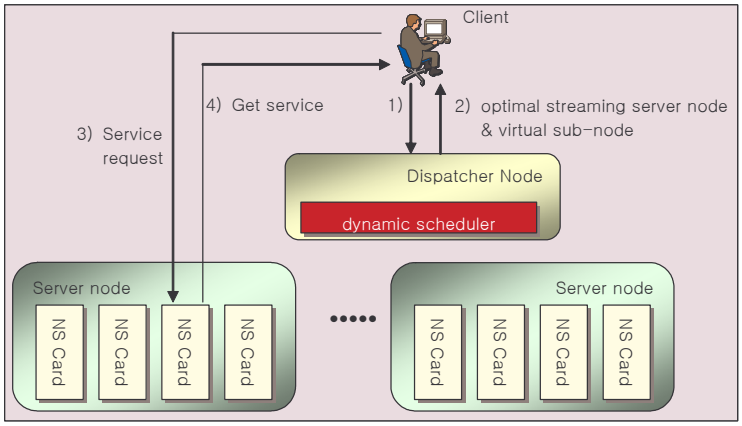

Fig. 5. Dynamic Scheduling Mechanism

1. A client requests a service at VIP
2. The dynamic scheduler at dispatcher node selects an optimal streaming server node and a NS card, and then directs the client to reconnect to the selected virtual sub-node (NS card)
3. The client reconnects to the selected virtual sub-node
4. The client is receiving the streaming service through the selected virtual sub-node within the selected streaming server node.

If a specific node or a NS card is failed, this should be removed from the request distribution target. The clustering software notifies its component's health status to prevent incorrect request distribution to failed component.

Content Placement for Efficient and Balanced Resource Consumption. In a cluster configuration, contents should be placed and managed effectively across multiple streaming server nodes and multiple NS cards in order to balance workloads. When a content is transferred from the CSN node, the clustering software should select an optimal streaming server node and a NS card to store the transferred content based on the status of each resource usage including storage. It is frequent that many service requests concentrate on the same content. In this case, if the content is located at only one NS card or one streaming server node, the NS card and the server node suffers from high overhead. The content placement mechanism should resolve this problem by replicating the content at other NS cards or streaming server nodes.

3.5 Software Architecture for Content Distribution and Clustering Support

The content distribution and clustering software architecture is shown in Fig. 6. The Job Manager which runs at a dispatcher node in the cluster, manages jobs submitted from user or other components of the system, and distributes the jobs into appropriate servers. Jobs are managed in on-demand or scheduled fashion. Also, the Job Manager handles content placement based on various information including server load, storage status, and so on.

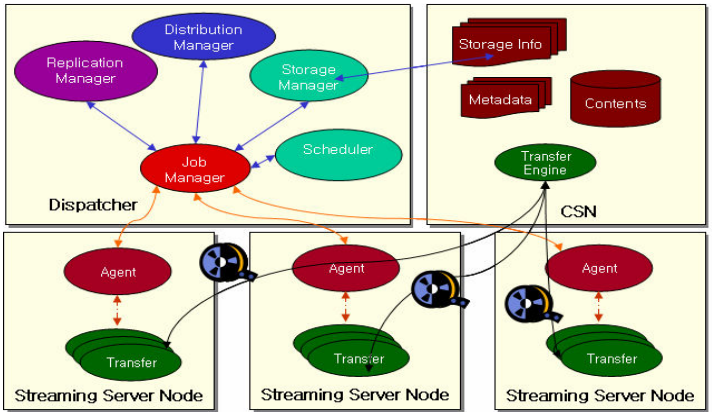

Fig. 6. SW Architecture for Content Distribution and Clustering Support

The Agent is run at every server node in the cluster, and executes jobs distributed from the Job Manager. Most of those jobs are content transfer requests. A transfer job is executed by invoking the Transfer client. The Transfer client connects to the Transfer Engine at the CSN node and pulls the requested content from the CSN. The Agent also notifies the status of streaming server node and NS cards in order to support efficient content placement and request dispatching. If the status of a component is not updated in given time period, the system considers it as failed component.

The Scheduler supports the single system image in cluster environment by dispatching all incoming service requests to optimal NS cards within streaming server

nodes. It also activates dynamic loading when a requested content is not available completely. Additionally, the scheduler manages content usage information.

The Distribution Manager maintains overall metadata required to manage the content distribution, and provides utilities that control the content distribution job having on-demand, scheduled and/or prefix mode. The Storage Manager keeps track of storage usage of each NS card and if needed, activates content purging. The Replication Manager checks the necessity of replication of some contents periodically, and if needed, registers replication jobs. A replication job is processed in similar way to a transfer job.

4 Performance Evaluation

4.1 Implementation Result

(a) (b) (c)

Fig. 7. Typical Streaming Server: Compact Model with 1 NS Card (a), Standard model with 3 NS Cards (b), and NS Card (c)

The streaming server consists of a conventional mother board and a media streaming accelerator called as NS (Network Storage) card. The NS card is implemented as a PCI card having a PCI-to-PCI bridge, disks, a PMEM (PCI Memory) and a TOE. Fig. 7 shows a typical streaming server with storage box and the NS card. The server has two Intel Xeon Processors with cache size 512KB, 1.28GB Memory, on-board two gigabit Ethernet ports and up to 4 NS cards.

4.2 Performance Evaluation

We measured the performance of the proposed server by changing the type of contents format (MPEG-2, MPEG-4 and H.264), bit rate of contents and I/O block size, and so on. Also we tested the effect of clustering solution and prefix caching. All tests were run at TTA(Telecommunication Technology Association).

As shown in Fig. 8, we evaluated the capacity of stream services with zero-loss by changing the number of NS cards and bit rate on MPEG-2 contents. From the results, we can know the fact that the proposed server supports 4.1Gbps for the 4Mbps content and 4.44 Gbps for the 10M and 20Mbps contents respectively. Other test results for single-node streaming server represent similar performance.

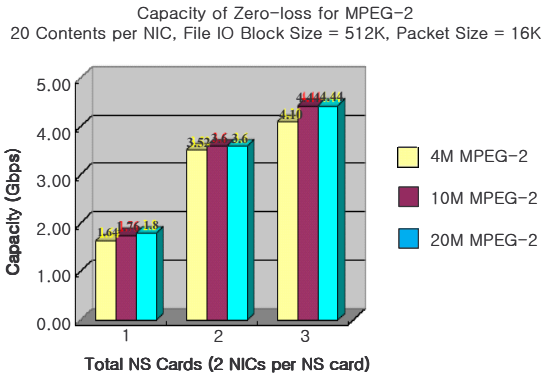

Fig. 8. Zero-loss Server Capacity for MPEG-2

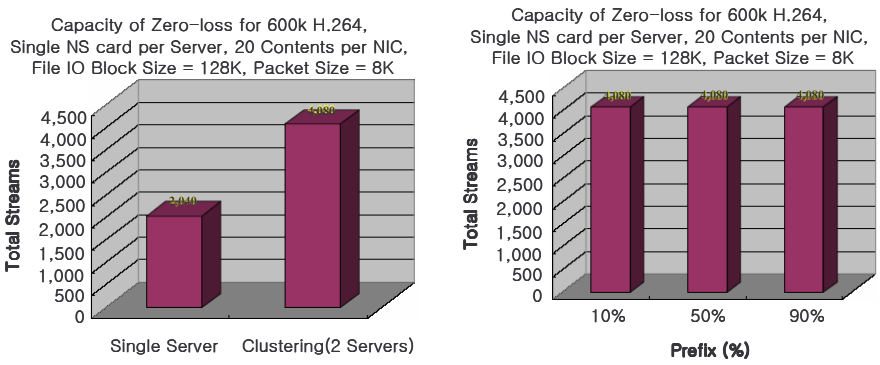

Fig. 9. Zero-loss Server Capacity for Clustering (left) and on Different Prefix Rates (right)

Fig. 9 shows the test results related to clustering and prefix caching features. The results tell us that clustering software distributes all incoming requests evenly to the participating streaming servers. Also, we can know that prefix caching doesn't affect the streaming performance, and from which we know the fact that prefix caching can help to give more chance to hit user requests.

5. Conclusions

In this paper, we are proposing an efficient system for multimedia streaming services in terms of the design concepts, the characteristics and the system architecture including hardware and software. When designing the system, we had to consider the design goals and several design issues. The design issues for the system are the CPU utilization, zero-copy, disk throughput, multimedia file system, clustering solution and content distribution. Therefore, we used special hardware, disk striping, special multimedia file system, clustering solution and efficient content distribution mechanism.

We measured the performance on various situations and got desirable results. But, more refinement on the system for multimedia streaming services is required to enhance the overall performance and stability of system. Specially, we implemented the multimedia file system as an user-level application. The file system needs to be joined with VFS(Virtual File System) I/O sub system in Linux kernel layer. Furthermore, in order to provide total solution for multimedia streaming services, we have to design and implement the global server.

References

1. Korea National Computerization Agency, Ministry of Information and Communication: 2002 Korea Internet White Paper (2002)
2. Adaptec Company: Advantages of a TCP/IP offload ASIC. White paper
3. Thomas Plagemann, Vera Goebel, Pal Halvorsen and Otto Anshus: Operating System Support for Multimedia Systems. The computer Communication Journal, Elsevier, Vol. 23, No. 3 (2000) 267-289
4. M.B. Abbott and L.L. Peterson: Increasing network throughput by integrating protocol layers. IEEE/ACM Transactions on Networking, Vol. 1, No. 5 (1993) 600-610
5. Gregory F. Pfister: In Search of Clusters, Prentice Hall PTR (1998)
6. T. Schroeder, S. Goddard, and B. Ramamurthy: Scalable Web Server Clustering Technologies. IEEE Network, 14(3) (2000) 38-45
7. Wensong Zhang: Linux Virtual Server for Scalable Network Services. In Proc. of the Ottawa Linux Symposium (2000)
8. Kasenna: Video Content Distribution: A Hybrid DCS/Caching Architecture for Broadband Video. White paper
9. Subhabrata Sen, Jennifer Rexford, and Donald F. Towsley: Proxy Prefix Caching for Multimedia Streams. In Proc. of INFOCOM (1999) 1310-1319
10. Stardust: Content Networking and Edge Services: Leveraging the Internet for Profit. White paper
11. HCL Technologies: Content Delivery Networks: An Architectural Overview. White paper