Multimedia Delivery Service using Multicast Technique on the Head-End-Network

Taejune Hwang,[†] *and Iksoo Kim*[††]*,*

University of Incheon, Incheon, Korea

Summary
In this paper, we propose Multimedia Delivery Service (MDS) for providing fluent multimedia services using multicast technique on the Head-end-Network. The implemented system is not conventional server-based multicast system but provides client-based on-demand one that has been lacked in conventional multicast delivery. The scheduler generates a multicast group address and port number when a client requests a service, then it sends them to multimedia server and client who request service. And then multimedia server transmits requested streams with a multicast group address and the client joins the group automatically. The scheduler assigns the same multicast group address when other clients request an identical multimedia within the same scheduling duration. Otherwise, it assigns another multicast group address to them.

Key words:
Multimedia, Multicast, On-demand, VOD.

Introduction

In Multimedia, the problems for multimedia information service are excessive load of server and inefficient use of network resources. The services in multimedia have to support the on-demand service that can be provided to the consumer whenever or wherever he or she wants it[1].

To support the on-demand service for multimedia data must be developed high speed network, huge capacity of storage devices, compression technology and intelligent multimedia server which processes various multimedia streams including audio, video, images, animations and text. Two of the best ways to provides multimedia service on Internet fluently are multicast delivery technique and web caching strategy using proxy[2, 3, 4].

The proxy caching technique store the previously requested multimedia in proxy near clients and provides the service from proxy without accessing server through network when the stored multimedia is requested again. Although they can reduce the load a server and end-to-end delay, the main problems for traditional caching techniques are the imbalanced load of proxies and the duplicated copies. And the proxies storing popular multimedia are faced with the heavier traffic than other proxies. Thus, proxy web caching techniques for multimedia on-demand service have a load-balancing problem among proxies[4, 5, 6]. Also, the multicast and broadcast have been shown to be very effective in reducing the server bandwidth. The broadcast technique is better for very popular multimedia and multicast is more suitable for less popular ones. Those techniques maximize the efficiency of server and network resources. Multicast allows multimedia server to send the same multimedia streams to multiple clients who requested an identical multimedia without consuming extra resources [7, 8, 9].

The conventional multicast delivery systems announce service time for a specific multimedia in advance, and then clients access multimedia server at that time. And conventional Multimedia service at prescheduled service times [10]. Consequently, these systems do not support on-demand service and they are server-based multicast system. Thus they generate a multicast group address and port number in prior to clients' request, and all clients who want to view the whole multimedia have to access multimedia server at prescheduled service time. Or they establish publishing point for multicast delivery to some video items and then multimedia server starts transmission of multimedia streams when the first client requests service. Since all clients except the first client who requests multimedia service cannot view the whole multimedia, these systems do not support multicast but they rather perform broadcast on LAN[11]. Although conventional multimedia servers provide multicast delivery, they only are interesting in the point of view multimedia server. Thus for supporting on demand service with multicast, multimedia servers have to implement client-based multicast system.

In the paper, the implemented multimedia service using multicast delivery technique can support on-demand service. For providing on-demand service, scheduler on server side assigns a multicast group address and port number the instant that client requests a multimedia service and sends them to multimedia server and client. Thus this system has the characteristics of on-demand. The scheduler aggregates clients' request for an identical multimedia item within predefined scheduling duration (20 seconds) in order to maximize the efficiency of multimedia server and network resources. Therefore the clients who request an identical multimedia item within the same scheduling duration receive the same multicast group address and port number. We implement this system with JDK1.4 and JMF(java media framework).

Manuscript received February 25, 2006.
Manuscript revised February 28, 2006.

The rest of the paper is as follow: Section 2 describes the structure and operation of Multimedia Delivery Service(MDS) system using multicast technique in detail, and Section 3 addresses the implementation of MDS systems supporting multicast technique with JDK1.4 an JMF, Section 4 shows the result of improvement of proposed MDS system through simulation, finally Section 5 concludes the paper.

2. The Structure and Operation of Multimedia Delivery Service(MDS)

In the paper, The Multimedia Delivery Service(MDS) using multicast technique is composed of Multimedia Server, Scheduler, and a number of clients connected Head-end-Network as a VDSL/cable network, as shown in Fig.1[6]. Multimedia server transmit contents for multicast or unicast according to many request of clients. And Scheduler is a central control program that aggregates service requests from clients through HNET and generates multicast group addresses and port numbers through scheduling process.

2.1 Scheduler

Scheduler transmits multicast group addresses and port numbers to multimedia server and clients requesting service. The Multicast Scheduler groups a same multicast group when some clients request an identical multimedia during the grouping in order to increase channel efficiency and decrease the load of server. Thus, if any client requests MDS for a multimedia item, the scheduler investigates whether the thread for scheduling to an identical multimedia exist or not. The scheduler assigns the same multicast group address and port number to client when the thread for a multimedia requested exists. Otherwise, it generates another multicast group address and port number to clients and multimedia server. Therefore the scheduler creates different multicast address and port number to clients who request an identical item after the each grouping duration passed.

2.2 Multimedia Server

The multimedia server transmits multimedia streams as soon as receiving multicast addresses and port number from scheduler and the clients join a specific multicast group automatically after receiving them. The content server transmits advertisement contents during grouping time. There are two reasons for sending it; to remove the latency for service from multimedia server and to decrease the service charge. Although the clients join the same multicast group, the viewing time for the advertisement is different for each client. The first client who requests service views it for entire grouping duration, but other clients view it for less than grouping duration according to their request instant within a grouping duration.

2.3 HEN

The HEN manages clients' requests and multicast delivery. And it has a small buffer to store multimedia data for providing VCR operation in future study. HEN locates in between server and clients as a proxy, stores streams delivered from server, and serves them to clients. Also, it can access the others HENs for according stored video segments on them.

2.4 Clients

The Clients access scheduler in order to request multimedia service and then they receive multicast group address and port number. Next, the Media Player on client side joins multicast group in order to view the requested multimedia. Clients has buffer as much as N where N is the number of HENs. First, client is served by stored segments among HENs. When playback time exceeds the amount of stored segments among HENs, scheduler generates new multicast group and client join it.

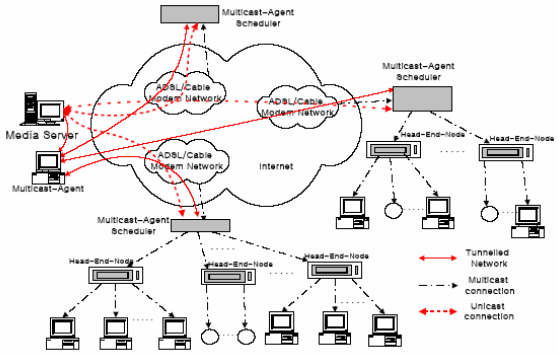

Fig. 1 The structure of distributive within the HNET.

3. The Implementation of MDS using Multicast Technique

The implementation of MDS using multicast technique uses JDK1.4 and JMF and it divides into three parts, client who requests service, scheduler as generation of multicast group address and port number according go clients' request, and multimedia server as transmitting stream and port number.

The scheduler divides into two parts, main scheduler and socket thread. The main scheduler using server socket receives clients' request, generates multicast groups

according to scheduling duration to each requested multimedia item and manages them. The socket thread maintains connection of client. The main scheduler generates *ServerSocket* and waits clients' request, and generates scheduler thread object from socket object which is delivered by *accept()* function of server object when client requests service. The *accept()* function is a member function that makes Socket object accepting clients' request by *ServerSocket* class. The *ServerSocket* of the main scheduler receives clients' request and generates a thread per client who request service, and then maintains each connection. It assigns multicast group address and port number and sends them to multimedia server and clients who request service.

The scheduler thread generates socket between scheduler and client in order to establish connection and gets ready for exchanging streams between them. It performs scheduling using *CommonData* class which includes client IP requesting service and video item number requested. *CommonData* class defines a generation function for multicast group address and port number. Also it defines arrays of multicast group addresses generated for scheduling multimedia item and of flags indicating whether generated address is usable or not. The scheduler thread disconnects socket connection when the scheduling is ended through socket object. Thus, the scheduler notifies multimedia server to send requested multimedia item to multicast group address as a destination and clients who request service to join a multicast group. Fig. 2 and 3 show overall service procedure of proposed system and scheduler procedure, respectively.

Step 1 : Client accesses scheduler for MDS system request : inputs requested multimedia item's ID
Step 2 : The scheduler investigates whether the scheduling session for requested multimedia ID exists or not. If it exists go to step 4.
Step 3 : The scheduler performs grouping and generates multicast group address and port number during grouping duration.
Step 4 : The scheduler sends multicast group address and port number to multimedia server and clients who request an identical multimedia item.
Step 5 : Server transmits requested multimedia to a received multicast group. The clients join the received multicast group and port number.
Step 6 : Java Media Player play outs video.
Step 7 : Complete video service.

Fig. 2 Overall MDS system and Join procedure

The scheduler is the heart of providing MDS system. It receives clients request and requested multimedia item, and generates immediately multicast group address and port number. Then it sends them to multimedia server and clients who request service. And it sends the same group address and port number to other clients who request an identical multimedia during a grouping duration. But it generates another group address and port number when clients request an identical multimedia item after a grouping duration passed. The steps 4 to 6 in scheduler procedure of Fig 3 perform these operations.

Step 1 : Initiate scheduler
Step 2 : The scheduler waits until the clients request MDS system.
Step 3 : Scheduler thread wakes up when the client requests service. It receive multimedia item ID from client.
Step 4 : The scheduler investigates whether the scheduling session for requested multimedia ID exists or not. If it exists, then go to step 6.
Step 5 : The scheduler performs grouping and generates multicast group address and port number during grouping duration.
Step 6 : The scheduler sends multicast group address and port address and port number to multimedia server and clients who request an identical multimedia item.
Step 7 : Completion scheduling.

Fig. 3 Scheduler on server side procedure

Fig. 4 and 5 show server and client procedure, respectively. The multimedia server generate Datagram socket and inserts the multicast group address and port number within the datagram packet, and transmits multimedia streams to clients. It divides into two parts, the one is Datagram socket as a maintaining indirect connection with clients and the other is data transmitting for sending requested multimedia item.

The multimedia server investigates whether the server resource are consumed or not, and the requested multimedia streams store in the disk or not. It does not need to know clients' ID who requests service, their IP and how many clients request an identical multimedia item. The scheduler performs such operations. The content performs only transmission of multimedia streams.

Step 1 : The scheduler calls multimedia server. : multimedia server starts.
Step 2 : multimedia server receives multicast group address and port number from scheduler.
Step 3 : multimedia server investigates compression format and the rate of transmission.

Step 4 : multimedia server transmits multimedia streams with received multicast address and port address
Step 5 : multimedia server complete transmission of multimedia streams. : completion server program.

Fig. 4 Multimedia server procedure

For requesting a specific multimedia item, client and scheduler generate sockets, and then client sends data to the main scheduler for service request. The client receives a multicast group address and a port number from the scheduler thread. Then, client generates multicast socket with received multicast group address and joins the group to receive transmitted multimedia streams from the multimedia server.

Step 1 : Client generates socket and prepares connection to scheduler.
Step 2 : Client requests multimedia items ID to scheduler after connection.
Step 3 : Client receive multicast group address and port number from scheduler.
Step 4 : Client closes socket for scheduling.
Step 5 : Client joins multicast group address in order to receive transmitted multimedia streams from multimedia server.
Step 6 : Java Media Player player plays out video streams.
Step 7 : Java Media Player stops play-out.

Fig. 5 Client procedure

Fig. 6 shows the result of execution of scheduler/server program. In Fig. 6, server side drives application program on command line, then it outputs client IP, multicast group IP, port number, video and audio data types prior to transmitting video streams. As shown in Fig. 7, the scheduler/server receives six clients' requests for three video items, and then scheduler generates three multicast group addresses and six port numbers, respectively. The reason of generation 3 multicast group address and 3 port numbers is that clients' request an identical video item within the same scheduling duration, and the transmission of video and audio needs to separate port. Thus the content server generates RTP session and starts transmission of video/audio files(bailey.mpg, practice.mpg and success.mpg) requested to multicast group addresses and port numbers : 230.0.161.86:4068, 230.0.75.238:4064 and 230.0.42.11:4040 that are sent from scheduler.

Also, the scheduler sends multicast group address and port number to client who requests service and closes connection requested socket. And scheduling finishes immediately scheduling thread for multimedia item #3 and starts new scheduling session when other clients request service after a grouping duration passed. Also, scheduler/server program shows the process of transmission completed. And it shows the detailed information of requested video items just like video and audio compression format, the size of frame and frame rate. But Fig. 6 shows only information of the first requested multimedia item(bailey.mpg) because the information of other two video items cannot show the window of command line within a 1-page.

Fig. 6 The execution of scheduler and server programs

Fig. 7 shows screen shot of scheduler frame. As shown in Fig. 7, the scheduler on server side indicates grouping requested multimedia items list on window of grouping multimedia item during a grouping duration. The grouping item list is added requested video item IDs when grouping item field within CommonData class becomes set. Also, the scheduler indicates generated multicast group addresses, port numbers, clients' IPs which request service and requested multimedia items on window of user.

Also, the scheduler sends multicast group address and port number to client who requests service and closes connection requested socket. And scheduling finishes immediately scheduling thread for multimedia item #3 and starts new scheduling session when other clients request service after a grouping duration passed. Also, scheduler/server program shows the process of transmission completed. And it shows the detailed information of requested video items just like video and audio compression format, the size of frame and frame

rate. But Fig. 6 shows only information of the first requested multimedia item(bailey.mpg) because the information of other two video items cannot show the window of command line within a 1-page.

Fig. 7 shows screen shot of scheduler frame. As shown in Fig. 7, the scheduler on server side indicates grouping requested multimedia items list on window of grouping multimedia item during a grouping duration. The grouping item list is added requested video item IDs when grouping item field within CommonData class becomes set. Also, the scheduler indicates generated multicast group addresses, port numbers, clients' IPs which request service and requested multimedia items on window of user.

Fig. 7 The scheduler application on server side

Fig. 8 and 9 show the result of execution client program and GUI of client application, respectively. As shown in Fig. 8, the execution of client program performs on command line through ClientFrame. The result of execution is the window of client GUI(Fig. 9). And then client writes server IP or DNS name of scheduler on ClientFrame(Fig. 9), well-known port number and video item number on corresponding window(Fig. 9). The port number 7777 is a port that the function of ServerSocket() uses. Then the result of scheduling by the execution of thread is transmitted to the client from scheduler. The transmitted multicast group address(230.0.161.86) and port number(4068) are written on the bottom of Fig. 8 and in the TextArea of Fig. 9. And then, the client automatically joins with 230.0.161.86 and 4068(port number) using MClient class and SimplePlayer class.

The MClient class generates socket to communicate with scheduler, and it outputs multicast group address and port number on client's window. Next, MClinet class closes socket and attempts to join a multicast group address and port number, and then generates SimplePlayer class.

The SimplyPlayer class drives Multicast Media Player using JMF(java media framework) and receives video streams transmitted from content server through RTP session with a multicast group address.

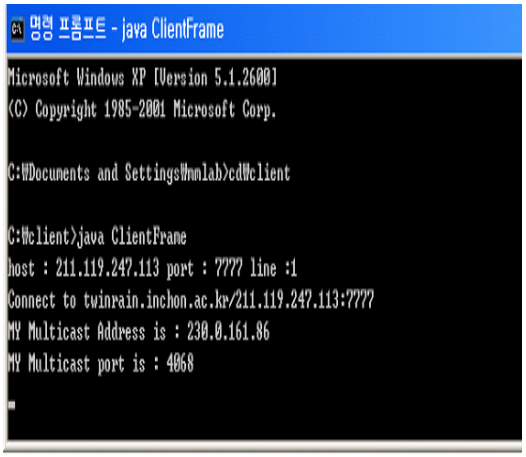

Fig. 8 The execution client program

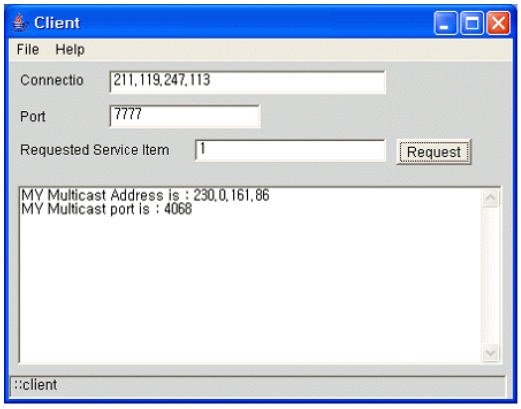

Fig. 9 GUI of client application

4. Simulation and Result

In this section, we show simulation results to demonstrate the benefit of proposed multicast network with multicast technique and analyzes on the results of performance using it. We assume that the system contains 1000videos, V_i=1000; all of them are 100 minute long, $|v|=100T_m$. The server is capable of supporting 10,000 channels and the total number of service request is limited 10,000 within 100 minutes. Let N be total number of videos in the server. Let $P_N(i)$ be the conditional probability that, given the arrival of a video request, the arriving request is made for video i.. Let all the videos be ranked in order of their popularity there video i is the i'th most popular video. We assume that $P_N(i)$, defined for i=1, 2, …, N, has "cut-ff" Zipf-like distribution given by

$$P(i) = \frac{\Omega}{i^Z}, \qquad \text{where } \Omega = \left(\sum_{i=0}^{N} \frac{1}{i^Z} \right)^{-1} \tag{1}$$

In the paper, we consider a broader class of distribution functions with exponents in skew factor z[12, 13]. A larger z corresponds to a more severs skew condition indicating that some videos are requested more frequently than the others. We set this value to 0.7. Using the Zipf-like distribution, if the overall clients' service request rate to the Content server is λ, the service requesting rate for i'th video is $\lambda_i = \lambda P_N(i)$. Its rate based on popularity is used to determine the traffic for each video. The most popular video(i=1) must have higher weighted value than the others because the request for the most popular video is more frequent than that fir unpopular ones. We use $P_N(i)$ as a weighting parameter for selection of videos in simulation and the service request rate λ follows Poisson distribution.

Consequently, we perform simulation such that the more popular videos with higher request probability. Fig. 10 show the mean number of transmission channels from multimedia server. In this case, the number of HENs was fixed at 10, server's bandwidth was 10,000, HEN bandwidth was 500 channels, and the mean request rate λ was 10 and 50 requests per minute. All of transmission channels from multimedia server are to transmit initial transmission and/or to transmit dynamic multicast after $n \bullet T_m$ of the video V_i. Those multicasts start playing after finishing the playback of all the segments cached among HENs. So, it can reduces the number of multicast to transmit video segments as many as $n \bullet T_m$ for each video.

In proposed multicast technique, the multicast grouping interval T_m is various from T_m to $n \bullet T_m$, where n is the number of HENs that cached the video V_i. We can reduce the number of multicast channels almost 59% for 100 videos and 15% for 1,000 videos at the mean arrival rate λ=10 in compare with conventional multicast. At λ=50, it can reduce 80% for 100 videos and 22% for 1,000 videos. If the number of video is 100 and 1,000, cache can store as many 1% and 0.1% of the total amount of video. So, as cache hit ratio of 1,000 videos is lower than 100 videos', server has to send more streams for clients to play uncached, called cache miss, video date. The number of multicast channels depends on cache hit ratio. Therefore, the number of transmission channels on server increases logarithmically because cache hit ratio grows logarithmically.

Fig. 11 shows simulation results of proposed multicast technique that the mean number of multicast channels in server as the function of the number of videos in various arrival rate λ 10, 20, 50, 100 and 200, and the others parameters were used the same ones in above simulations. This indicates that the number of channels in server grows depending on the number of videos logarithmically. It also grows depending on the arrival rate λ linearly.

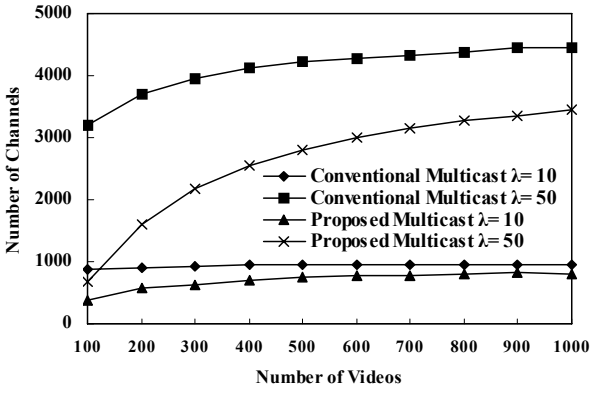

Fig. 10 The mean number of multicast channels for VOD server at arrival rate λ=10 and 50

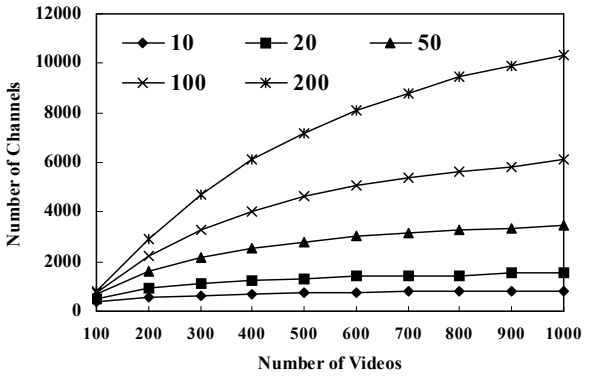

Fig. 11 The mean number of multicast channels in server as the function of the number of videos at

5. Conclusion

The proposed system implements Multimedia Delivery Service(MDS) system using multicast technique which applies client-based system not server-based one. The scheduler performs efficient scheduling that immediately generates multicast group address and port number according to clients' request. And it sends them to multimedia server and clients who request service. Then the multimedia server starts sending requested multimedia to group. And the clients automatically join that group and play out receiving multimedia streams. Thus, we expect that the proposed system can performs on-demand service on Internet.

This paper confirms that multimedia server reduce the load of server and can efficiently use network resources through client-based scheduling for sharing multicast group address and port number. Thus, this system can use commercial multimedia service on VDSL/Cable modem network. Currently we are trying to implement multicast tunneling mechanism for this system without the aid of Mbone. We believe that the range of application will be extended when the proposed system supports multicast tunneling on Internet and VCR operation for high quality service.

References

[1] D.Sitaram and A.Dan, "multimedia servers", Morgan Kaufmann Publishers 2000
[2] T.D.C. Little and D. Venkatesh, "Prospects for Interactive Video-On-Demand" In MCL Techical report 02-15-1994, 1994
[3] K.C.Almeroth and M.H.Ammar,"The use of Multicast Delivery to Provide a Scalable and Interactive Video-On-Demand Service," IEEE J. Selected Areas in Comm., Vol.14, No 6, Aug. 1996, pp.1110-1122
[4] R. Rajaie, M. Handley, H. Yu and D. Estrin, "Proxy cashing mechanism for multimedia playback streams in Internet," in Fourth International WWW Caching Workshop, Mar.1999.
[5] Carey Williamson, "On Filter Effects in Web Caching Hierarchies," ACM Transactions on Internet Technology, Vol. 2, No. 1, pp. 47-77, February 2002
[6] Iksoo kim, Taejun Hwang, Youngjune Kim & Yoseop Woo, "New Distributive web caching Technique for VOD Service," ITCOM2002 Conf. Boston U.S.A, 2002
[7] P.Basu, A.Narayanan, R.Krishnan and T.D.C. Little, "An Implementation of Dynamic Service Aggregation for Interactive Video Delivery", In MCL Technical Report 11-01-97,1997
[8] Yen-Jen Lee, David H.C.Du and Wei-Hsiu Ma, "SESAME: A Scalable and Extensible Architecture for Multimedia Entertainment", In IEEE, COMPSAC, 1996
[9] D.Eager, M.Vernon & J.Zahorjan, "Optimal and Efficient Merging Schedules for Video-on-demand Servers", Proc. ACM Multimedia 99, ACM 1999
[10] C.Aggarwal, J.Wolf, P.Yu and M.Epelman, "On Caching Polidies for web Objects," IBM Research Report RC20619 March 1997
[11] J.Pasquale, G.Polyzos and G.Xylomenos, "The multimedia Multicasting Problem," ACM Multimedia Systems, vol.6 no.1, 1998
[12] M.Arlitt and C.Williamson, "Trace-driven simulation of document caching strategies for Internet Web Server," Simulation Journal 68, pp.23-33 Jan 1997.
[13] Carey Williamson, "On Filter Effects in Web Caching Hierarchies," ACM Transactions on Internet Technology, Vol. 2, No. 1, pp. 47-77, February 2002.

TaeJune Hwang received the B.S. and M.S. degrees in Computer Science and Engineering from University of Incheon, Korea, in 1997 and 1999, respectively, and is currently a Ph.D. candidate in University of Incheon His research interests are VoD, Multicast Multimedia system, Mobile Ad-hoc Network and Database Security.

Iksoo Kim received the B.S., M.S., and Ph.D. degrees in Electronics Engineering from Dongkook University, Korea, in 1977, 1981, and 1985, respectively.. Since 1988, he joined in University of Incheon as a professor. He was a guest professor on North Carolina State University and California State University Sacramento in 1993 and 2004, respectively. His major interests are caching strategy, parallel & distributed processing, on-demand system, and multimedia system.