Wide-Area Multicasting based on Flexcast: Toward the Ubiquitous Network

Takeru INOUE[1], Seiichiro TANI[2], Katsuhiro ISHIMARU[3], Shinichi MINATO[1], and Toshiaki MIYAZAKI[1]

[1] NTT Network Innovation Laboratories
1-1 Hikari-no-oka, Yokosuka-shi, Kanagawa, 239-0847 Japan
{inoue.takeru, minato.shinichi, miyazaki.toshiaki}@lab.ntt.co.jp

[2] NTT Communication Science Laboratories
3-1 Morinosato-Wakamiya, Atsugi-shi, Kanagawa, 243-0198 Japan
tani@theory.brl.ntt.co.jp

[3] Makuhari Gigabit Research Center, TAO
1-9-1 Nakase, Mihama-ku, Chiba-shi, Chiba, 261-0023 Japan
ishm@makuhari.tao.go.jp

Abstract

The Internet is starting to provide a broadcasting service for broadband streaming media. It is expected that this service will become more popular and more available in the near future; users will be able to not only receive stream data but also send it from anywhere at any time, that is, the ubiquitous broadcasting service is coming. However, current Internet technology does not suit this goal.

Our solution is an autonomous wide-area multicast protocol called Flexcast. The Flexcast protocol works even in legacy IP networks where IP multicast cannot. The protocol is also highly scalable and maintains optimum delivery trees dynamically.

This paper proposes a translator node, called Flexcast gateway, which interworks the Flexcast protocol with IP multicast transparently. It enables IP multicast applications to work anywhere even across legacy IP networks, and will bring about ubiquitous broadcasting. Finally, we provide brief reports on two field experiments. In the experiments, we multicast prospective broadband media such as high-definition television with IP multicast applications passing through wide-area networks that do not support IP multicast themselves. The results confirm that our proposed system suits wide-area multicasting and that the prototype system offers excellent performance.

1. Introduction

Broadband Internet access services such as DSL (Digital Subscriber Line) and FTTH (Fiber To The Home) are spreading rapidly. In the near future, wireless LAN and W-CDMA technologies are also spreading and are expected to provide broadband access services anywhere at any time, namely, ubiquitously.

The service of broadband streaming media is starting to gain some adherents, and multicast technologies are expected to support the services. These technologies allow the server to issue a single stream which is then replicated in the network by routers or special nodes called splitters, for delivery to each recipient. IP multicast is one of the most popular multicast technologies, and requires special IP addresses, called IP multicast addresses, to specify the groups of recipients. However, current wide-area networks do not support routing protocols for IP multicast packets for technical and economical reasons. One alternative, IP unicast based multicast technologies, is gathering much attention as a broadcasting tool. Though they work in such legacy IP networks, they have several problems in terms of scalability and flexibility, as described later. Consequently, existing multicast techniques are being deployed only in private or closed networks. To realize ubiquitous broadcasting, we consider that multicast technologies must satisfy three following requirements:

- to support the use of legacy network resources, so as to broadcast from anywhere to anywhere,
- to provide high scalability, to handle lots servers and clients,
- and to maintain the delivery trees dynamically without prior knowledge of the network configuration, in order to begin broadcasting at any time.

Our proposal is an autonomous wide-area multicast protocol called Flexcast. The Flexcast protocol uses just unicast packets, so it works even in legacy IP networks. The protocol is also highly scalable and maintains tree-like optimum delivery paths automatically without prior configuration. Thus, the Flexcast protocol is well-suited to ubiquitous broadcasting. Unfortunately, we have few application programs[1] designed around the Flexcast protocol, and it may take time for such applications to appear. In contrast, IP multicast has supported many applications in its long history.

[1] We discuss just broadcast style (single-source) applications, not conference style (multi-source) applications in this paper.

In this paper, we propose a translator node called Flexcast gateway, which interworks the Flexcast protocol with IP multicast transparently. This feeds advantages of the Flexcast protocol to IP multicast applications, and realizes ubiquitous broadcasting. This paper is organized as follows. Section 2 describes existing multicast technologies and the Flexcast protocol, while Section 3 proposes the Flexcast gateway. Section 4 shows brief results of two field experiments, and we provide a short conclusion in Section 5.

2. Preliminary

2.1 Conventional Multicast Technologies

IP multicast [1] is the most popular mechanism for multicasting. Many protocols [2]-[4] for IP multicast have been proposed and most are being investigated for standardization at IETF. Before commencing an IP multicast session, we must obtain a globally unique IP multicast address. This complicates the use of IP multicast. Furthermore, all routers along the paths to the group members must be able to route IP multicast packets, which currently restricts IP multicast to just closed or experimental networks such as Mbone [5]. This is why IP multicast does not support wide-area multicasting.

In the source-specific multicast protocol [6], each multicast group is represented by a pair of its server (source) address and a multicast address. Since the server address is globally unique, the multicast address does not need to be globally unique. However, all routers still need to support IP multicast routing protocols.

Automatic tunneling protocols, which create tunnels to between multicast networks, are proposed in reference [7], [8]. However, the tunnels waste too much bandwidth and is not expected to be scalable, since the server-side end of each tunnel is forced to unicast a packet to the other end of each tunnel directly, not multicast.

IP unicast based multicast technologies are gathering much attention as ubiquitous broadcast tools. It is possible for legacy routers to be located on the paths between the clients and the server, since they use just unicast packets. This makes it easy to use the multicast mechanism in IP networks that include legacy routers. There are two types of IP-unicast-based multicast technologies: application layer multicast works in the application layer while the other works across the network and application layers.

In application layer multicasting [9], [10], replicator nodes are usually located at the edge of the network or the clients themselves. The stream is terminated by a replicator, which duplicates the stream and forwards them to the (other) clients. Since these processes are carried out in the application layer, they are called application layer multicasting. However, they require clients to retrieve the nearest replicator's address to receive the stream, which often needs pre-configuration phase. Also the delivery trees are sometimes constructed independently of the network layer, and this can bring about route redundancy and makes this approach not scalable.

There are several technologies that work across the network and application layers. The Flexcast protocol is one of them, and we explain it in the next section. The protocol proposed in [11] is so simple that it seems to be scalable. However, it is possible that the protocol reconstructs the whole delivery tree when the first client leaves, which makes delivery trees unstable. Recently, [12] proposed another approach for multicasting called Xcast. In Xcast, all the clients' addresses are listed in the packets instead of the multicast group. Each Xcast-aware router along the path replicates the packets if necessary. The protocol has scalability problem when there are many recipients, and so does not seem to support *broadcasting*.

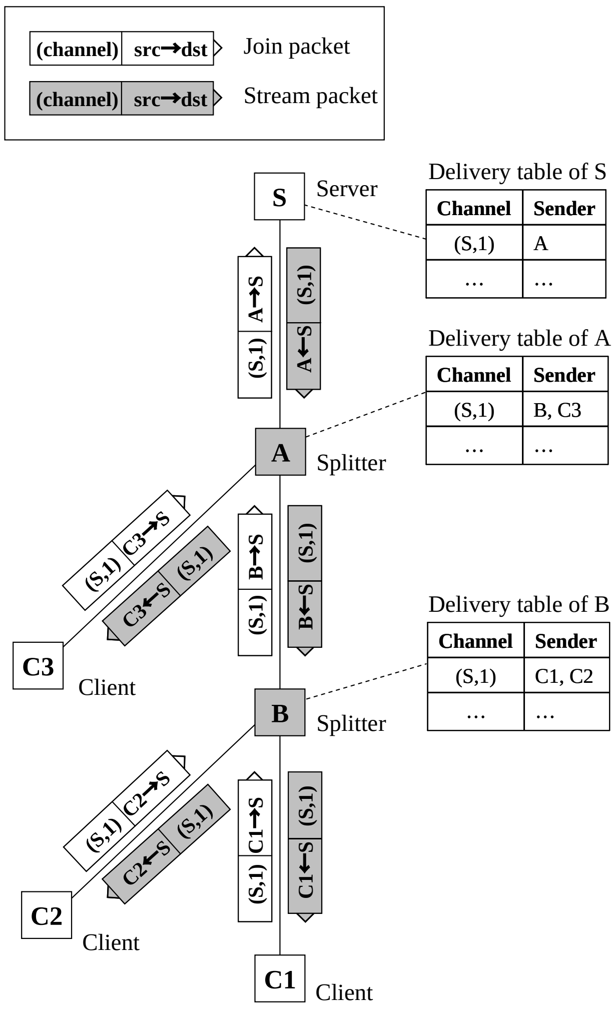

Figure 1: Basic operation of Flexcast protocol

2.2 Flexcast protocol

Reference [13] proposes a multicast technology called Flexcast. The Flexcast protocol realizes multicasting with unicast packets, and autonomously regenerates the optimal delivery tree when recipients emerge or disappear.

The Flexcast protocol is composed of clients, servers, and splitters. A splitter is a branch point of the delivery tree, and it replicates and forwards the stream.

Clients who want to receive a stream send *join* packets destined to the server periodically that contain the paired information of IP address of the server and the port. The port is used to identify the stream within the server, and the paired information specifies a delivery tree. The paired information is called a *channel*. When a join packet arrives at a Flexcast splitter located on the path between the client and the server, the splitter terminates the join packet, and registers the sender address in the routing table, called the delivery table. If the requested channel is a new one, the splitter creates a new record for the channel in the table, and enters the sender address into its delivery table. In either event, the splitter sends a join packet to the server periodically, the same as clients. This joining operation propagates from the client through intermediate splitters until the final join packet reaches the server, and constructs a delivery tree.

When the server receives a join packet, it sends the stream to the child[2] on the tree. The splitters on the tree receive the stream from their parent, and deliver it to their immediate children. Finally, the stream is delivered to each client.

The Flexcast protocol uses the keep-alive mechanism to maintain the delivery tree. The parent of a client relays the stream provided it receives a join packet from the client within some interval, such a client is called *active*. In other words, a client that stops sending join packets expires, and no stream is delivered to the client. The parent node also continues to send join packets to the server while it has at least one child that remains active. The same keep-alive mechanism works between the splitter and its parent.

We show an example of the above operation in Figure 1. Flexcast splitter A connects path splitter B and client C3 to server S; splitter B connects path clients C1 and C2 to server S. Clients C1, C2, and C3 periodically send join packets to server S. These packets are routed as ordinary unicast packets. Splitter B intercepts the packets from C1 and C2 and registers the sender addresses, clients C1 and C2, in its delivery table, and sends a join packet whose source address is splitter B. Similarly, splitter A picks up the join packets from splitter B and client C3, registers them in its table, and sends a join packet to server S. When server S receives the join packet, it sends the stream to splitter A. Splitter A copies the stream and sends them to splitter B and client C3 after referring to the delivery table. In the same way, splitter B sends the stream to clients C1 and C2.

From the example, it is clear that the delivery stream traces the reverse of the unicast transmission path from the clients to the server. The Flexcast protocol can work even if the reverse-paths differ from the unicast paths from the server to clients, say, the forward-paths. While, in general, forward-paths yield higher stream delivery quality than reverse-paths, forward-path-based tree construction often results in complicated or non-adaptable protocols [11]. It is reasonable to assume that there will be little difference in quality between forward-paths and reverse-paths, since IP networks are being optimized to support bidirectional communication. Thus, the Flexcast protocol adopts reverse-path-based tree construction which yields scalability in terms of the number of nodes and adaptability to IP routing changes.

The Flexcast protocol works even if legacy routers are located between clients and servers, since they can simply forward the packets to the next hop based on the unicast destination address. Also, the Flexcast protocol is so simple that it is extremely scalable [14]. As shown in Figure 1, the Flexcast protocol constructs delivery trees automatically and starts broadcasting without any pre-configuration phase. However, there are few application programs designed around the Flexcast protocol.

3. Wide-Area Multicasting using Flexcast

This section proposes the Flexcast gateway; it interworks the Flexcast protocol with IP multicast transparently and enables IP multicast applications to work anywhere even across legacy IP networks. We also discuss the address resolution process between Flexcast and IP multicast, which enables our proposed system to maintain delivery trees automatically.

3.1 Flexcast Gateway

First, we focus on IGMP [15]-[17], the client management protocol of IP multicast. IGMP must be implemented by all IP multicast routers and all IP multicast clients. IGMP informs IP multicast routers of whether IP multicast clients exist in the adjacent subnet. A multicast router sends IGMP queries periodically, and one or some of the clients in the subnet respond with IGMP membership reports. The multicast router then runs a multicast routing protocol [2]-[4] and forwards multicast stream packets while it has active clients.

IP multicast routers do not need to control IP multicast servers. They can forward multicast streams just when they receive them from the IP multicast server.

In this section, we introduce a protocol translator node called Flexcast gateway, which implements IGMP as well as the Flexcast protocol and translates them transparently. Flexcast gateway changes its behavior depending on the connected network. When a gateway is placed in the subnet to which IP multicast clients are connected, the gateway is called client-side gateway and controls IP multicast clients by using IGMP. Upon receiving IGMP membership reports, the client-side gateway starts sending Flexcast join packets like Flexcast clients. A server-side gateway, which is connected to the server's subnet, receives Flexcast join packets. The gateway then encapsulates IP multicast streams from the server and forwards them just as do Flexcast servers.

We show an example of gateway operation in Figure 2. Gateways G1 and G2 are connected to the subnet of an IP multicast server and IP multicast clients respectively. Client-side gateway G2 watches for IGMP membership reports sent by IP multicast clients C1 and C2, extracts the IP multicast address M, and starts to send join packets to server-side gateway G1. We discuss later a method of

[2] Following the usual terminology of a tree, for each node (including the server and clients) of a multicast tree, we refer to the server-side and client-side neighbor(s) as, respectively, the parent and the children of the node.

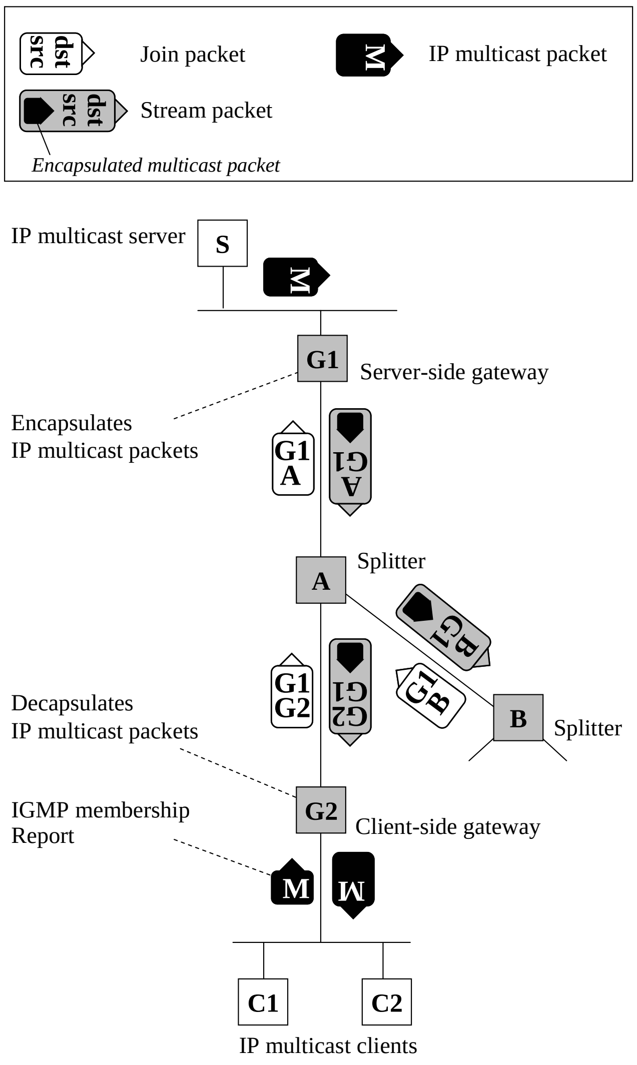

Figure 2: Basic operation of Flexcast gateway

resolving the address of the server-side gateway from the IP multicast address. The requested channel is identified by the pair of gateway G1 and multicast address M. Gateway G2 uses IGMP query to check if IP multicast clients C1 and C2 want to receive the stream and sends join packets only while they are active. When server-side gateway G1 receives the join packets, gateway G1 picks up the IP multicast packets of multicast address M that are passing through the local segment, encapsulates the packets, and forwards them as a Flexcast stream. Gateway G2 extracts the original IP multicast packets from the encapsulated stream packets, and releases the IP multicast packets to the local segment. Finally, IP multicast clients C1 and C2 get the IP multicast packets.

From a different perspective, Flexcast gateways create a tunnel through which IP multicast packets from server pass to clients. The tunnel is highly scalable, since the tunnel itself is a delivery tree of the Flexcast protocol and can be a tree-shaped. Our approach realizes remarkable efficiencies compared to the automatic tunneling protocols [7], [8], which link multicast capable networks directly.

The overhead of encapsulation is less than 4% if the original multicast IP packet occupies 1,400 bytes. We confirm the influence of encapsulation in Section 4.2.

3.2 Address Resolution Method for Client-Side Gateway

In this section, we discuss how the client-side gateway can resolve the address of the server-side gateway from the IP multicast address carried by the IGMP membership report.

In version 3 of IGMP [17], the IGMP membership report contains the server address field so the client-side gateway can use it to obtain the address of the server-side gateway. Versions under 3, however, require a separate resolution process. We propose the address mapping server, which maps IP multicast addresses to the server-side gateway addresses. In the proposed method, a broadcaster registers a pair of IP multicast address and the server-side gateway's address to the address mapping server prior to broadcasting. When a client-side gateway receives an IGMP membership report, it extracts the IP multicast address and asks for the address of corresponding server-side gateway.

Since the proposed method resolves the address of the server-side gateway dynamically, the Flexcast tunnels are automatically created and removed under the control of the actions of the IP multicast clients.

4. Experiments and Results

In this section, we briefly discuss the results of two field experiments. One was conducted across the Pacific Ocean to confirm that our proposed method is feasible in use, especially in wide-area networks that include legacy routers. The other involved the transmission of HDTV (High-Definition Television) class streams, whose average bandwidth exceeded 20 Mbps, to confirm the scalability and performance of our proposed system. We also confirm that Flexcast tunnels are automatically created under the control of IP multicast clients.

4.1 Inter-Pacific Experiments

To confirm that the proposed method works in wide-area networks including legacy routers, we conducted streaming experiments over NTT's experimental fiber-network connecting Japan and U.S.A., called GEMnet, and the networks of Internet2 [18], which is a consortium established to develop advanced network applications and technologies. The experiments were carried out as part of the Internet 2 Fall 2002 Member Meeting.

Figure 3 illustrates the network topology in the experiment. IP multicast servers and clients were located in NTT's Yokosuka R&D Center, the University of Southern California (USC), Los Angeles, and the University of Illinois (UIC), Chicago. Yokosuka R&D Center was connected to GEMnet; USC and UIC are connected to Internet2. Since GEMnet and Internet2 have a bidirectional access point in Sunnyvale, USC and UIC could communicate with Yokosuka via Internet2 and GEMnet. GEMnet is bottleneck of the whole network in

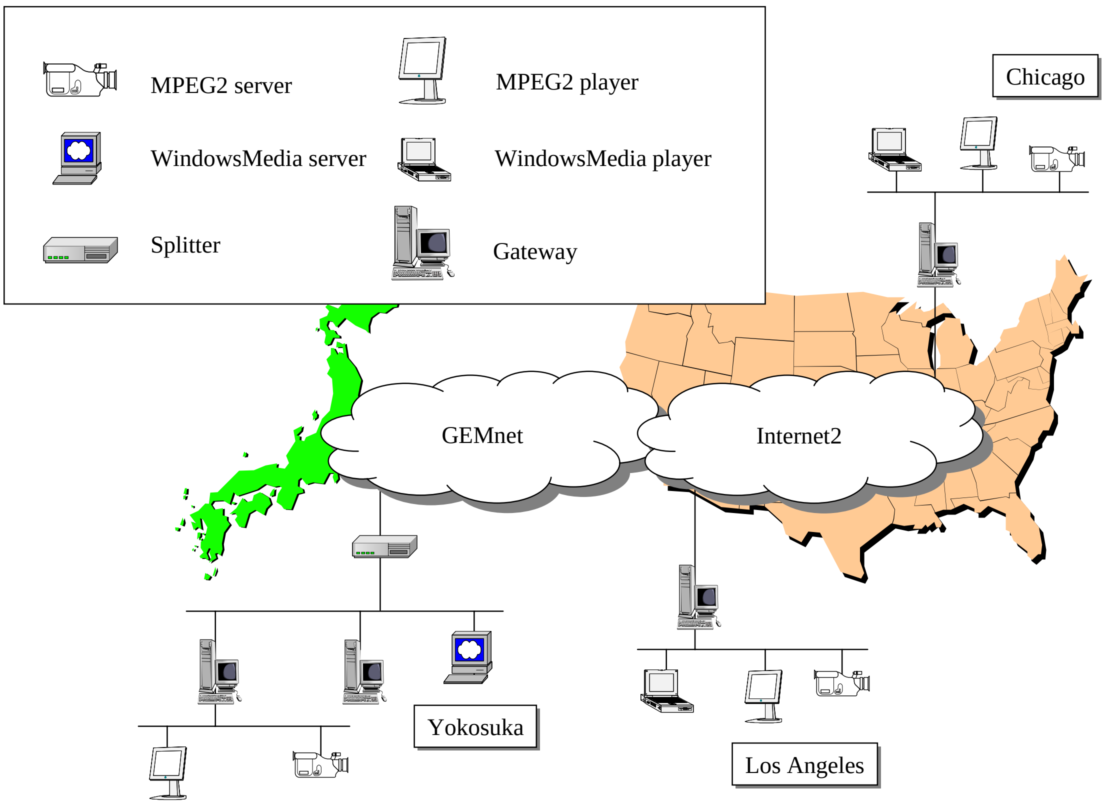

Figure 3: Schematic view of the Inter-Pacific experiments

the experiments, and it has a constant bitrate speed of 17 Mbps in each direction. Other links are more than 100Mbps. We did not let IP multicast packets be routed in GEMnet and Internet2, that is, both networks worked themselves as legacy IP networks in the experiments.

We implemented software-based Flexcast splitters and gateways using ordinary PCs (Pentium IV 2.0 GHz with 512 MB of main memory). The Flexcast protocol does not specify the underlying transport protocol. We implemented Flexcast over UDP and over TCP. The IGMP version we implemented is 2, and we installed an address mapping server in Yokosuka. We used two types of IP multicast applications; one was an MPEG2 server/player [19] and whose average bandwidth was 6 Mbps. The other was Windows Media 7 [20] whose average bandwidth was 500 kbps.

Each site had one or two gateways and several IP multicast servers and players. A splitter was placed at Yokosuka.

In the experiments, we could enjoy a smooth video chat across the legacy IP networks. Both Flexcast over UDP and over TCP worked well. Flexcast tunnels were automatically created and removed under the control of the actions of the IP multicast clients. The results showed that our proposed system allowed the IP multicast applications to work across the legacy IP networks and delivery trees were maintained dynamically.

4.2 High-Definition Television Experiments

The second series of experiments was conducted over JGN (Japan Gigabit Network) [21], which was designed for the research and development of very high-speed networking and high-performance application technologies. As Figure 4 depicts, we used three servers (Chiba, Tokyo, and Kochi), and fifty players[3] (Chiba, Tokyo, Kochi, and Miyagi). Chiba site offers 1 Gbps networks while other networks limit their bandwidth around 100 Mbps. An address mapping server was sited in Chiba. We broadcasted an HDTV class stream to fifty clients simultaneously to confirm the performance of our proposed system.

The average bandwidth of HDTV streams is large, 20 Mbps to 25 Mbps, so we installed Flexcast splitters in powerful server PCs; all had 64-bit bus, Xeon 2.0 GHz and 1024 MB of main memory. We adopted UDP for the transport layer, since it has smaller overhead than TCP.

During the experiments, all clients played error-free HDTV pictures. Flexcast splitter could create at most twenty five streams from one input stream. Flexcast tunnels were well controlled by IP multicast clients. The experiments confirmed the feasibility of broadcasting future broadband contents, and demonstrated that the proposed system has excellent scalability and

[3] In order to increase the number of players, we used forty three dummy players in the experiments, which do not decode the stream just receive it. In Figure 4, a dummy player is depicted as three players, but it was actually three to twenty dummy players running. After all, fifty players were receiving the stream simultaneously.

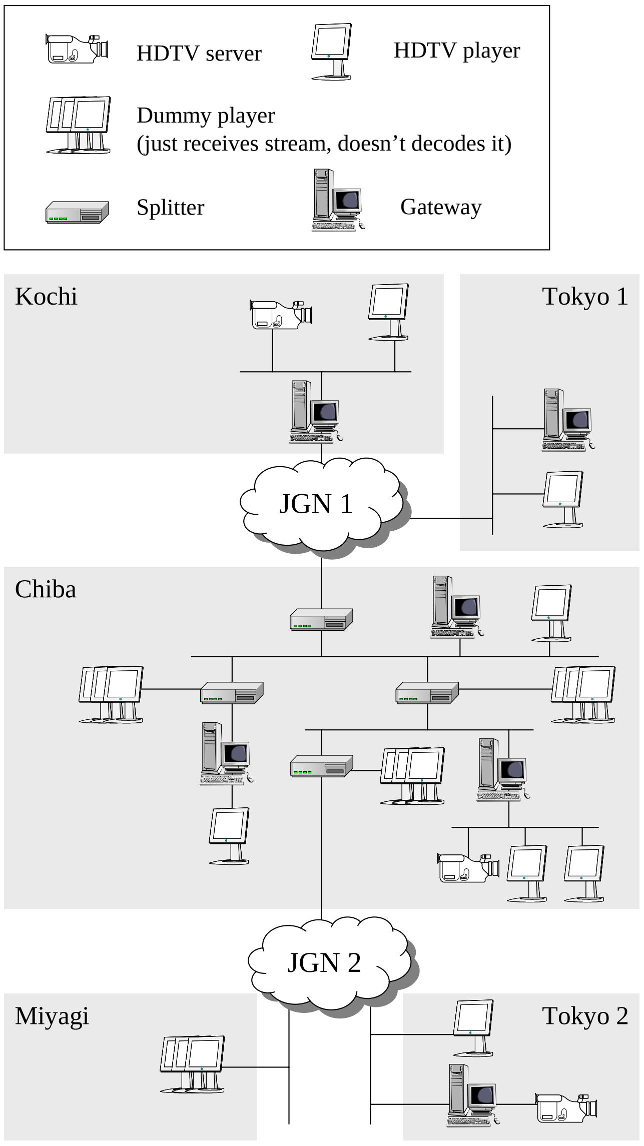

Figure 4: Schematic view of the high-definition television experiments

performance.

5. Conclusion

This paper proposed the Flexcast gateway, a network node that interworks the Flexcast protocol with IGMP transparently. The Flexcast protocol works even in legacy IP networks where IP multicast cannot. The protocol is also highly scalable and maintains the optimum delivery trees dynamically. By introducing Flexcast gateways, IP multicast applications can now be broadcast over legacy IP networks. We also briefly showed the results of two field experiments. They confirm that our proposed system works well in wide-area networks and can broadcast broadband media such as high-definition television. The combination of Flexcast and IP multicast raises the possibility of realizing ubiquitous broadcasting in the near future.

Acknowledgements

First, we would like to thank Prof. T. Aoyama, our advisor at the University of Tokyo, for letting us work on these exciting experiments. We would also like to thank H. Takahashi, S. Kotabe, T. Murooka, Dr. T. Shimizu, and Dr. A. Takahara, our colleagues at NTT, for their help in the inter-Pacific experiments. We are also grateful to T. Ogura, our colleague at TAO, K. Hayashi and K. Toyoshima, our colleagues at NTT, for their help in the high-definition television experiments.

References

[1] S. Deering, "Host Extensions for IP Multicasting", IETF RFC 1112, August 1989.

[2] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei, “Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification,” IETF RFC2117 June 1997.

[3] D. Waitzman, C. Partridge, and S.E. Deering, “Distance Vector Multicast Routing Protocol,” IETF RFC1075, November 1998.

[4] J. Moy, “Multicast Extensions of OSPF,” IETF RFC1584, March 1994.

[5] V. Kumar, “Mbone: Interactive multimedia on the Internet ’95,” New Riders Publishing, 1995.

[6] H. W. Holbrook and D. R. Cheriton, “IP Multicast Channels: Express Support for Large-scale Single-source Applications,” Proc. of ACM SIGCOMM1999, pp. 65-78, August 1999.

[7] K. Patel and R. Perlman, "Host Extensions to Protocol Independent Multicast," IETF Internet Draft, work in progress, July 2002.

[8] R. Finlayson, R. Perlman, and D. Rajwan, "Accelerating the Deployment of Multicast Using Automatic Tunneling," IETF Internet Draft, work in progress, February 2001.

[9] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang, “Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture,” Proc. of ACM SIGCOMM2001, pp. 55-67, August 2001.

[10] M. Kawada, H. Morikawa, and T. Aoyama: "Collaborative Multicast Architecture for Multi-Stream Applications," Proc. of APCC2001, pp. 626-629, September 2001.

[11] I. Stoica, T. S. Eugene, and H. Zhang., “REUNITE: A recursive unicast approach to multicast,” Proc. of INFOCOM2000, pp. 1644-1653, March 2000.

[12] R. Boivie, N. Feldman, Y. Imai, W. Livens, D. Ooms, and O. Paridaens, "Explicit Multicast (Xcast) Basic Specification", IETF Internet Draft, work in progress, January 2003

[13] S. Tani and T. Miyazaki, and N. Takahashi, "Adaptive Stream Multicast Based on IP Unicast and Dynamic Commercial Attachment Mechanism: An Active Network Implementation", Proc. of IWAN2001, September 2001.

[14] S. Ohta, S. Tani, T. Miyazaki, and N. Takahashi, "Multicast as a traffic variance smoother for IP streaming service,” Proc. of Networks2002, pp. 105-110, June 2002.

[15] B. Fenner, “IANA Considerations for IPv4 Internet Group Management Protocol (IGMP),” IETF RFC3228, February 2002.

[16] W. Fenner, “Internet Group Management Protocol, Version 2,” IETF RFC2236, November 1997.

[17] B. Cain, S. Deering, I. Kouvelas, B. Fenner, and A. Thyagarajan, “Internet Group Management Protocol, Version 3,” IETF RFC3376, October 2002.

[18] Internet2, http://www.internet2.edu/.

[19] Kubotek Corp., http://www.kubotek.com/enginfo/Ehome.html.

[20] WindowsMedia, http://windowsmedia.microsoft.com/.

[21] JGN, http://www.jgn.tao.go.jp/english/index_E.html