13

MidField: An Adaptive Middleware System for Multipoint Digital Video Communication

Koji Hashimoto and Yoshitaka Shibata
Iwate Prefectural University
Japan

1. Introduction

As broadband networks become more common, using high quality video streams such as DV (Digital Video) or HDV (High-Definition Video) has become increasingly popular in multipoint communications. Today's video communication systems which support high quality digital video formats on broadband networks are used for a variety of purposes (Bak, 2006; Gharai, 2006; Bodecek & Novotny, 2007). Although those systems have many useful functions for multipoint communications, it requires much bandwidth for video streams. A unidirectional DV stream requires about 28.8 Mbps. HDV 720p and HDV 1080i require about 19 Mbps and 25 Mbps respectively. Therefore, when we use high quality digital video communication systems, in many cases we should consider whether each location has enough communication environments to process the video streams or not. Of course, we can't always use effective format transcoding functions in multipoint communications. As the result, most video formats that are used by current multipoint communication systems are low resolution formats than DV and HDV even if we can use DV/HDV cameras at each end point.

If suitable format transcoding functions as defined by RTP (Schulzrinne, 2004) or stream integration functions for large scale video communications (Gharai, 2002) are available on our communication environments, it may be able to transcode the stream format into another suitable stream format or integrate several streams to a mixed stream. If the required communication environments are permanent and the total number of participants is always limited, we may be able to prepare the required transcoding and integration functions into suitable intermediate nodes in advance. However, we can't always use enough bandwidth and CPU power, and our communication environments aren't always permanent, therefore we should consider on demand transcoding functions with relocatable mechanisms.

In order to design and implement relocatable transcoding functions for multipoint digital video communications, employing an effective middleware (Ferreira, 2003; Jameela, 2003) on programmable networks (Campbell, 1999) is an effective way. As one of the methods for executable program modules to migrate to anywhere, many mobile agent systems (Guedes, 1997; Ohsuga, 1997; Dong & Seung, 2001; Antonio, 2002; Guo, 2004) also have been proposed. We consider that mobile agents which perform given tasks on suitable nodes are one of important base techniques for constructing flexible and interactive multipoint digital video communication environments.

Source: Digital Video, Book edited by: Floriano De Rango,
ISBN 978-953-7619-70-1, pp. 500, February 2010, INTECH, Croatia, downloaded from SCIYO.COM

To perform various streaming functions in suitable intermediate nodes according to each user's communication environment, the executable and relocatable program modules should perform the required functions for audio-video streams in realtime. We are developing a new middleware system (Hashimoto & Shibata, 2008) for multipoint audio-video communications which can use suitable formats that include high quality digital video formats such as DV and HDV. The system runs on a mobile agent subsystem and relocatable transcoding program modules as the mobile agent process audio-video streams. The system has also resource monitoring/management functions and video session management functions.
The remainder of this chapter is organized as follows. The next Section presents MidField System architecture and describes flexible and interactive communication sessions which are constructed by MidField System entities. Section 3 presents a communication protocol and packet transmission mechanisms for streaming modules, and Section 4 illustrates inside of the streaming module. Then the current implementation and use cases are described in Section 5 and 6. Finally, Section 7 concludes this chapter.

2. MidField system

Figure 1 shows the system architecture, which is between the application and the transport layer. MidField system consists of three layers and four vertical planes.

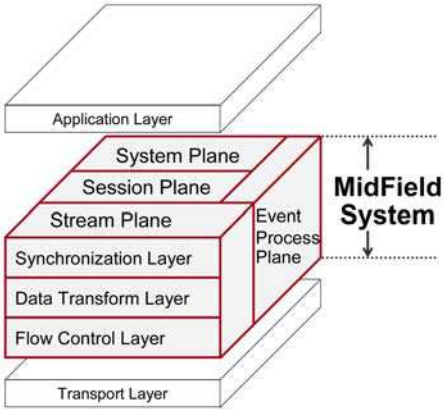


Fig. 1. MidField System Architecture

The stream plane consists of three layers, which are synchronization, data transform and media flow control layers. This plane performs inter and intra media synchronization, transcodes audio-video formats as necessary. In the flow control layer, packetization and depacketization of continuous audio-video stream data are performed and audio-video packets are sent and received among other MidField Systems.
On multipoint communications, the session management is an important function. The session plane manages multipoint session information (e.g. the number of users, available audio-video formats in the session and available computer/network resources, etc.). We defined MidField Session as a multipoint communication session in MidField System. By using several IP multicast sessions in a MidField Session, MidField System can connect remote users according to users' communication environments.
Local system resources such as CPU and required bandwidth are managed in the system plane. This plane monitors incoming and outgoing network traffic and CPU utilization rate

in a local node, and performs admission tests to check whether the required stream processing is possible on the node.
The system plane includes a simple mobile agent platform. Agents that are executable software components in MidField System can use an inter-agent message protocol and mobile facilities. Because each functional entity is based on the agent basically, each entity can exchange messages easily. Of course, although not all entities require mobile facilities, mobility of agents is able to connect users' various communication environments effectively and dynamically.
MidField System is based on an event driven model. Each function of these planes and layers is performed by various events, which are associated with media processing tasks and inter-agent messages. The event process plane not only handles various events on several threads but also manages message connections with remote MidField Systems.

2.1 MidField session

If our computer network environment for interactive audio-video communications has enough resources to use DV or HDV streams, of course we can communicate with remote users by DV or HDV streams. However we can't always use required enough resources for high quality digital video streams. MidField System can establish a communication session according to each user's communication environment by using IP multicast sessions.
Figure 2 shows an example of MidField Session, which includes three users and uses three video streams at each end point. A MidField Session consists of at least one multicast session and provides peer-to-peer communications to users.

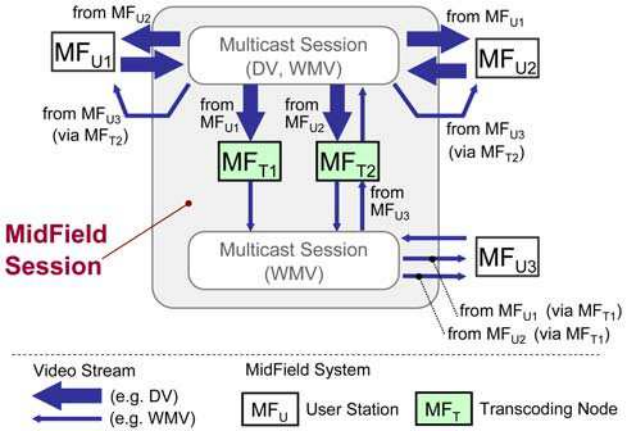


Fig. 2. MidField Session

In Figure 2, the user station MF_{U1}, MF_{U2} and MF_{U3} join a MidField Session. Now, MF_{U1} and MF_{U2} have enough communication resources to process DV streams that are required about 28.8 Mbps per stream. On the other hand, the MF_{U3} does not have enough communication resources to process DV streams because of lack of enough bandwidth or computing power to process DV streams.
In this case, to connect these users MidField System tries to use IP multicast sessions and transcoding nodes. If the MF_{U3} is able to process another video stream format such as Windows Media Video (WMV), the system will prepare transcoding nodes for transcoding DV streams into WMV streams.

In Figure 2, there are two transcoding nodes (MF_{T1} and MF_{T2}). At first, MF_{U1} and MF_{U2} join the DV multicast session, then MF_{U3} joins the WMV multicast session. Then, for connecting three users, MF_{U1} and MF_{U2} send program modules that are implemented as mobile agents into MF_{T1} and MF_{T2}. On these transcoding nodes, the program modules will start to transcode or relay streams.
As the result, three users will be able to communicate with each other by using suitable video stream formats according to each user's communication environment.

2.2 MidField stream

To establish a multipoint communication session, the system must have adaptive streaming modules, which need to support several audio-video stream formats, to be able to transcode stream formats and to perform on suitable node anywhere.
We have considered which formats should be supported in our system. In order to realize high quality digital video communications, DV and HDV formats should be supported. We can use DV and HDV cameras as input devices for our PCs commonly. However, not all users can use high specification PCs and enough broadband networks always, therefore, for connecting various communication environments the other stream formats also should be supported. At the beginning of the system design, we had employed MPEG4 video format for users who don't have enough communication environments to process DV/HDV streams. Although MPEG4 was one of suitable formats for multipoint communications, but it is one of choice. Now our system mainly supports DV, HDV and ASF (Advanced Stream Format) that includes WMV (Windows Media Video) and WMA (Windows Media Audio) formats.
Figure 3 illustrates the outline of a streaming module, which processes audio-video streams in the stream plane. A Stream Agent is a part of an end-to-end audio/video stream.

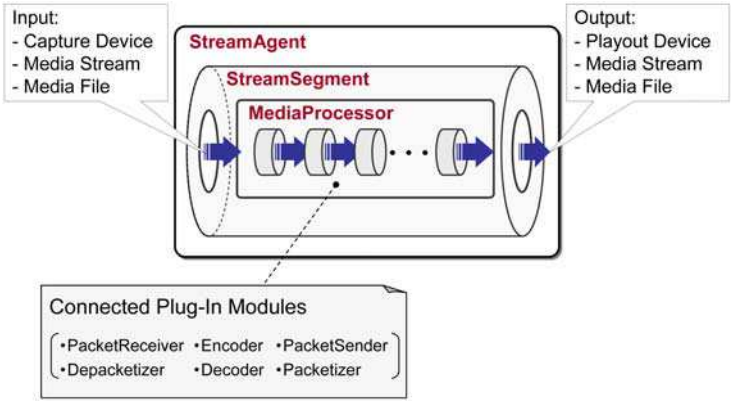


Fig. 3. Stream Agent

Stream Agent consists of mainly three parts and the implementation is based on mobile agent technique, therefore, it can exchange control messages with the other Stream Agent(s). In addition, Stream Agent can migrate from a local MidField System to another MidField System running on a remote node. By implementing Stream Agent as a mobile agent, our system is able to process the required streaming on any remote nodes where a MidField System is running.

The implementation of a simple mobile agent platform in MidField System is written in Java programming language. Stream Agent also is implemented in Java. At the beginning of the system implementation, we had been trying to use only Java for the whole system, because we considered that Java would become one of the most effective programming languages to implement the mobile agent platform.
Although we had implemented Stream Agent by using Java Media Framework API (JMF), unfortunately the implementation could not get enough performance for practical multipoint communications. However, because a transcoding module with mobility is required for connecting IP multicast sessions in a MidField Session, we had replaced a part of streaming modules with native program modules. As native program modules for audio-video streaming, currently, we employ DirectShow of Microsoft. Of course, these native modules don't have mobility. Therefore, the native modules don't migrate even if a Stream Agent migrates. However, a Stream Agent can migrate with I/O information for a stream segment and then on the destination node the Stream Agent can handle stream data by using native resources of the destination node. For the above reason, Stream Segment and Media Processor in Figure 3 are implemented in C++ programming language.
Stream Segment is mediator for Stream Agent and Media Processor. Because Stream Agent is implemented in Java, in order to issue stream control commands to native program modules, Stream Segment uses Java Native Interface (JNI). In fact, there are two instances of Stream Segment in Java and C++ context respectively.
Media Processor processes audio-video data by using some suitable plug-in modules. A Media Processor has at least one input source and creates at least one output source. When a Media Processor gets a set of input(s) and output(s), the Media Processor tries to connect suitable plug-in modules according to the input and output requirements and the audio-video data move from one plug-in module to the connected next plug-in module. This mechanism is able to be implemented in JMF, DirectShow, and the other existing libraries. Besides, each plug-in module has the own interface for controlling data process from application programs.

2.3 Software module configuration of MidField system

MidField System is a middleware system for flexible and interactive communication environment, which provides various kinds of audio-video communication functions as API to application programs. The software module configuration is shown in Figure 4. The system has three interfaces for Stream, Session and System Plane. Stream Agent, Session Agent and System Agent are agent-based software modules for each plane.
As written above, Stream Agent performs audio-video streaming and Session Agent manages audio-video session information. System Agent monitors local resources and performs admission tests when users join MidField Session.
Here, Agent Place manages all active agents in a local system and performs creation, migration and termination of agents. Agent Place uses Agent Loader(s) and Agent Server for migration functions. Agent Place also is a kind of agent.
These agents can exchange messages with other agents that are active in local or remote systems. Connection Manager in the event process plane manages Connected Socket objects for inter-agent message delivery functions. System Event Manager has mechanisms to process various events that are created by each agent. If an agent creates some events, the agent posts the events to System Event Manager and the events are stored into Event Queue. Each stored event is processed by one of Event Processors that have an idle thread.

2.4 Adaptive communication session

One of the important functions of MidField System is to support adaptive communication sessions. Software modules in Figure 4 cooperate to establish a MidField Session. Figure 5 shows an example of MidField Session.

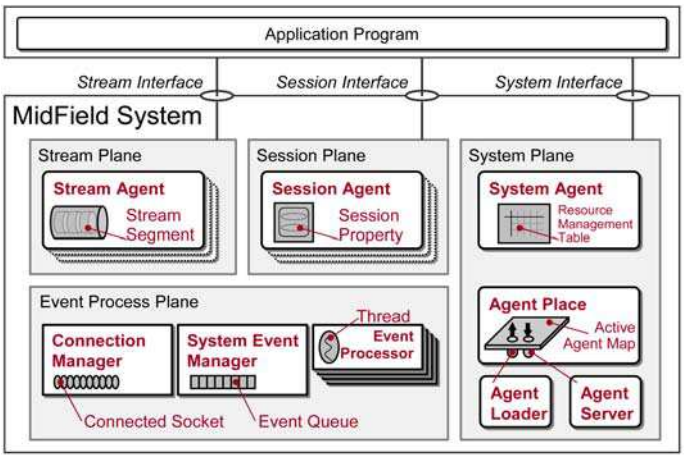


Fig. 4. Software Module Configuration of MidField System

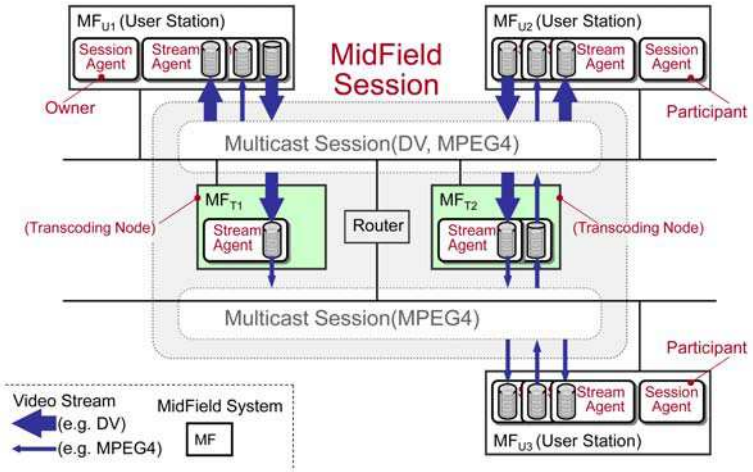


Fig. 5. An Example of MidField Session

In Figure 5, five MidField Systems are connected to computer networks. Three of them are User Stations (MF_{U1}-MF_{U3}), and remain two are Transcoding Nodes (MF_{T1}, MF_{T2}). At first, a Session Agent in MF_{U1} opens a new MidField Session as session owner. This MidField Session consists of two IP multicast sessions that are for DV streams and for MPEG4 streams respectively.

Now, a Session Agent in MF_{U2} joins the MidField Session as a participant, and begins to communicate with MF_{U1} on the multicast session for DV streams. MF_{U1} and MF_{U2} have

enough bandwidth and CPU power to process DV multicast streams. Then, MF_{U3} tries to join the MidField Session but unfortunately MF_{U3} doesn't have enough resources for DV streams. MF_{U3} requests MF_{U1}'s Session Agent to use the multicast session for MPEG4 streams.

In this case, the Stream Agents in MF_{U1} and MF_{U2} try to extend their DV streams to the MPEG4 multicast session. Selecting suitable transcoding nodes, after that Stream Agents for transcoding migrate to the selected transcoding node. Then each Stream Agent in transcoding node receives a DV stream and transcodes it into a MPEG4 stream. MF_{U3} is able to receive the streams from MF_{U1} and MF_{U2} as MPEG4 video streams.

Finally the Stream Agent in MF_{U3} starts to send a MPEG4 stream to the MPEG4 multicast session, after that the Stream Agent in MF_{T2} will receive the MPEG4 stream and relay it to the multicast session for DV streams. Thus MF_{U1}, MF_{U2} and MF_{U3} are able to communicate with each other by using a suitable IP multicast session.

We had developed a version of MidField System that includes an application that enables multipoint audio-video communications and experimented with 8 PCs in 2004. Figure 6 shows an image of the experimental communication. In this image, the upper 4 PCs use a multicast session for DV streams and the under left 2 PCs use another multicast session for MPEG4 streams. The under right 2 PCs are running as transcoding node.

Fig. 6. Experimental Communication (6 User Stations and 2 Transcoding Nodes)

MidField System had realized a dynamic session configuration method according to users' communication environments. It is one of results that the dynamic session configuration by integrating IP multicast sessions was introduced to multipoint communications. However, in many cases our communication environments are restricted to use IP multicast functions. As the result, unfortunately when we communicate with remote users on computer networks, we almost can't use IP multicast.

Currently, we are trying to design and develop a new adaptive session configuration method, which is based on the dynamic session configuration.

On the other hand, of course, in order to adapt to various communication environments, it is required for the system to have adaptive and flexible streaming functions. The following sections illustrate the design and implementation of Stream Agent, which has audio-video streaming functions for supporting adaptive communications.

3. Inter stream agent communication

In MidField System, an end-to-end audio/video stream consists of a chain of Stream Agents. Since a Stream Agent can handle several inputs and outputs, it is easy to relay audio-video streams from each source node. Figure 7 shows the outline of Stream Agents with direction of media streams (audio/video streams). In this figure, the relay stream agent stm(n) receives three input media streams from the upper Stream Agents $stm(n-1)_0$ - $stm(n-1)_2$, and sends three output media streams to the lower Stream Agents $stm(n+1)_0$ - $stm(n+1)_2$. If a Stream Agent has several inputs, the Stream Agent performs mixer functions for these inputs, the mixed stream is sent to lower Stream Agents.

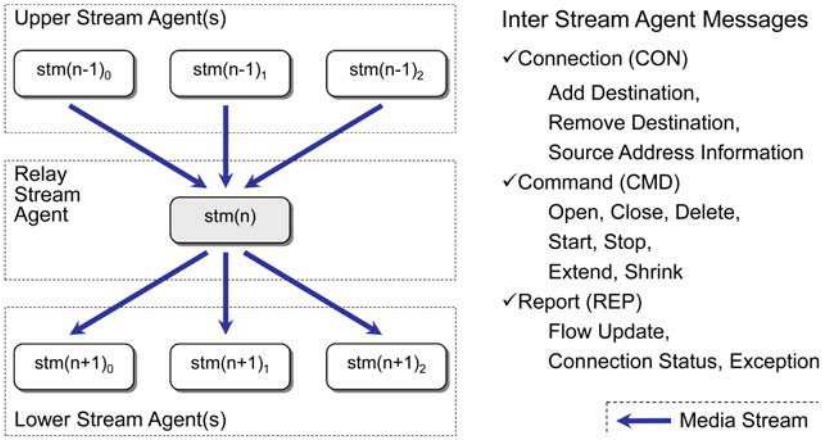


Fig. 7. Stream Agent and Messages

In order to realize adaptive connections, Stream Agents exchange messages shown in Figure 7. The connection (CON) messages are used for media stream connections. Once a media stream connection is established, it is controlled by the command (CMD) messages. In addition, to report flow update, connection status and exception to uppers, the report (REP) messages are exchanged.

3.1 Message flow of inter stream agent

Figure 8 (a) shows message flow for addition and removal of a media stream. At first, a NEW event is created in a lower node to receive a media stream from an upper Stream Agent stm(n-1), and stm(n) in the lower node gets the NEW event. Then stm(n) sends a Flow Update message to stm(n-1). In this figure, since stm(n-1) is connected to an upper Stream Agent, stm(n-1) also sends the Flow Update message to stm(n-2). After the Stream Agents receive Flow Update messages from lower Stream Agents, each upper Stream Agent updates internal data structures for media stream flow.

Next, stm(n) gets an OPEN event in the lower node and sends an Add Destination message to stm(n-1). After that, stm(n-1) replies to stm(n). The response message includes a source address and a port number of the media stream. Establishing media stream connection between stm(n-1) and stm(n), stm(n-1) starts to send media packets to stm(n).

A Remove Destination message for a CLOSE event is used to remove the media stream connection. Finally, after a DELETE event in the lower node, stm(n) sends a Flow Update message to the upper Stream Agent and uppers update the flow's data structure.

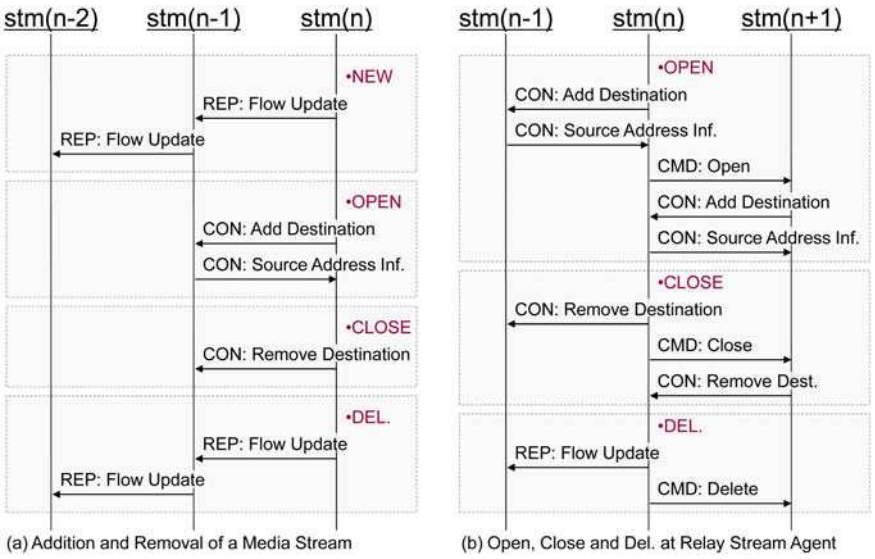


Fig. 8. Inter Stream Agent Message Flow #1

If stm(n) is already connected to lower Stream Agents, the lowers receive command messages. Figure 8 (b) shows message flow in the case of generating OPEN, CLOSE and DELETE events in a relay node. An Open message which is sent from stm(n) to stm(n+1) is handled on stm(n+1) as the same as a new created OPEN event. CLOSE and DELETE events also are handled as the same as OPEN event. Thus, MidField System is able to change flows of media streams adaptively.

As Figure 9 (a) shows, all command messages such as START and STOP are sent from upper Stream Agents to lower Stream Agents, and all report messages are sent from lower Stream Agents to upper Stream Agents.

Figure 9 (b) and (c) show a stream extension process briefly. As previously stated, Stream Agent has mobility in the capacity of mobile agent, which is used to extend a media stream. A Stream Agent starts to perform extension process when the Stream Agent gets an EXTEND event. In Figure 9 (b), stm(n) has already I/O parameters, and after getting an EXTEND event, stm(n) creates a clone of itself and the clone migrates to the other node. The clone stm(n+1) is able to get NEW and OPEN events in the node in which the clone has immigrated, and start to receive the media stream from stm(n). Of course stm(n+1) can relay the receiving media stream to other Stream Agents.

This extension process can be also started from an upper Stream Agent. In Figure 9 (c), stm(n) sends an Extend message to stm(n+1) and the stm(n+1) performs extension process.

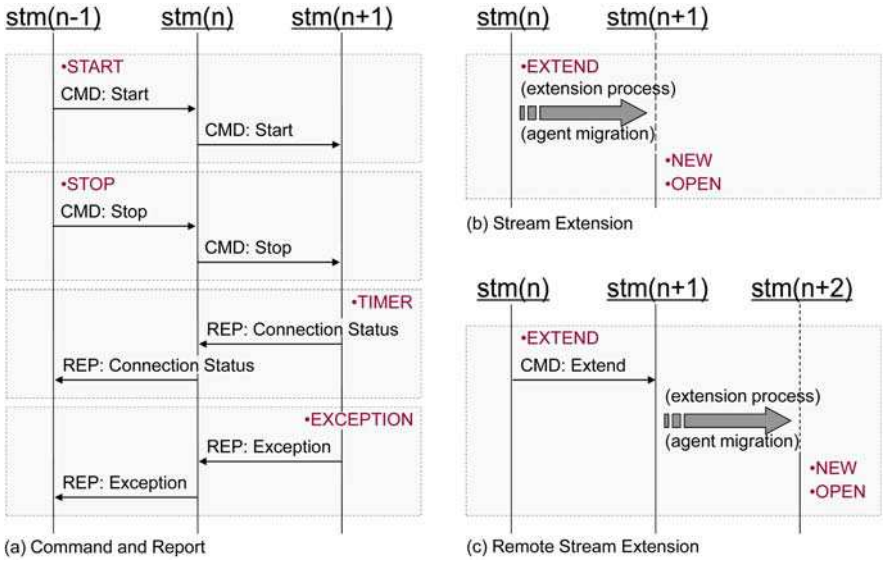


Fig. 9. Inter Stream Agent Message Flow #2

3.2 Stream extension process

In order to integrate various audio-video communication environments, to relay media streams adaptively is the one of important functions in audio-video communication systems. MidField System performs adaptive relay processes of media streams by mobility of Stream Agent. If it is clear which node is a suitable node to perform required relay functions, of course MidField System will employ the node as a relay or transcoding node. However, in case that there are several nodes which are able to employ as a relay or transcoding node, the system needs a node selection function.

MidField System's stream extension process realizes node selection and load balancing functions. Figure 10 shows the stream extension process.

At the beginning of a stream extension process, the system sends resource status requests to other MidField Systems which are running on other remote nodes. After receiving the request, those remote systems reply with a report of current resource status that includes available bandwidth (incoming/outgoing) and CPU utilization rate.

Replied resource status reports are added to a temporary remote resource status list, and after a timeout that has specified in advance, an available relay node list is created. Each element of the list includes resource status of relay nodes which have enough network bandwidth and CPU power in order to relay or transcode the required media stream. Then, the relay stream agent tries to migrate to a relay node which is included in the list.

If the number of timeouts is less than the constant value N and there are no available relay nodes, the system sends resource status requests to other MidField Systems again. If the number of timeouts is over the N and there are no available relay nodes, then the whole system for this communication environment can't allocate the required resources for the stream extension and this extension process is failure.

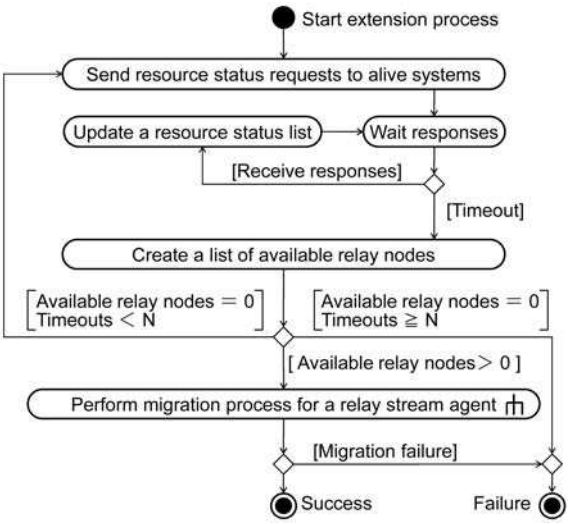


Fig. 10. Extension of Media Stream

The migration process for a relay stream agent to migrate to another node is shown in Figure 11. At first, a migration request is sent to the relay node. The request includes parameters for input and output associated with this relay stream. In the relay node, admission tests are performed for specified input and output parameters. If the required resources to relay or transcode the media stream are able to allocate, the resources are allocated in the relay node. Finally, the relay stream agent migrates to the relay node and starts to relay or transcode the media stream.

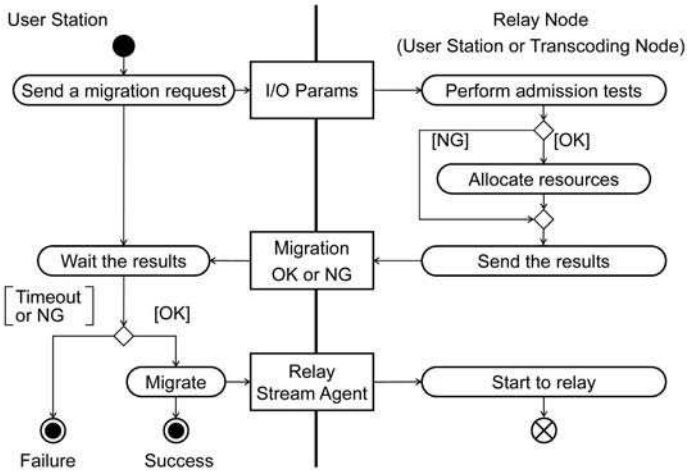


Fig. 11. Migration of Relay Stream Agent

As shown in Figure 9 (b) and (c), these processes for a stream extension can be performed in not only a local node but also another remote node. By considering an end-to-end media

stream as an adaptive object, MidField System has realized that as one of functions to establish a flexible and interactive communication environment.

3.3 Multithreaded stream packet transmission

As each element in an end-to-end media stream of MidField System, Stream Agent is required to handle several inputs and outputs dynamically. As stated in the previous section, all stream data in MidField System are processed by DirectShow, but DirectShow doesn't provide any packet transmission filters basically. For realising multipoint communications, we have newly designed and implemented packet transmission mechanisms, which work as filters of DirectShow. In order to adapt with the modular architecture of DirectShow, a part of the modules for packet transmission is implemented as filters of DirectShow.

Here, Packetizer is a filter that fragments each audio-video sample data which comes from an upper filter into transmittable data packets, and Depacketizer is a filter that assembles each data packet which receives from the source node into a sample data.

(1) For sending packets

Figure 12 illustrates the inside of a Media Processor in sender side, which includes a Packetizer that is connected to several Packet Senders for each destination. A pair of Packetizer and Packet Sender is associated with a packet queue, and at the both sides of packet queue, packet data are processed on respective threads.

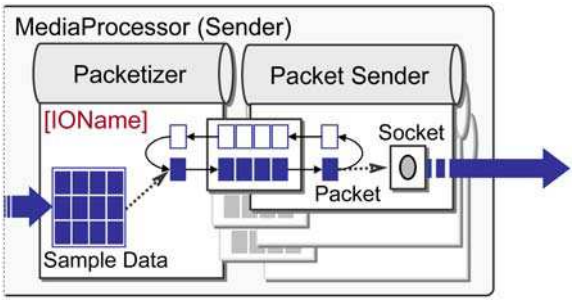


Fig. 12. Packetizer and Packet Sender(s)

In multipoint audio-video communications, to use IP multicast is effective clearly. However, on actual computer networks such as just the Internet, IP multicast is almost not available, therefore, audio-video communication systems should be able to adapt with the network environments. In MidField System, Packet Sender supports both IP unicast (MFSP/TCP and MFSP/UDP) and multicast (MFSP/UDP), in addition, a Packetizer is able to add and remove Packet Sender(s) dynamically depending on the number of destinations. Thus, Packet Sender supports not only IP multicast but also application level multicast (ALM) functions.

Since MidField System supposes a source node has several different source streams simultaneously, each stream must have a unique ID. The IOName in Figure 12 shows the unique ID, and on the destination nodes, each receiver uses the same IOName that is created in the source node.

(2) For receiving packets

Figure 13 illustrates the inside of a Media Processor in receiver side, which includes a Packet Receiver that is connected to several Depacketizer(s) for each lower filter. Like the sender side, a pair of Depacketizer and Packet Receiver is associated with a packet queue and packet data are processed on respective threads at the both sides of the packet queue.

As stated above, a stream has unique ID between a pair of sender and receiver node. By using this unique ID in a receiver, receiving redundant streams can be eliminated. For example, suppose that a receiver node is receiving a DV stream from a sender node and relaying the DV stream to another node, then next the receiver node starts to transcode the DV stream into a WMV+PCM stream newly. In this case, if the receiver node is receiving a DV stream of the target IOName, the receiver doesn't receive the same DV stream newly. The receiver node can create a new Depacketizer in a Media Processor and connect to the Packet Receiver that is receiving the target DV stream. Thus, the receiver node is able to start transcoding the DV stream newly without receiving a redundant DV stream.

The packet transmission mechanism in MidField System adapts media streams for various requirements. The mechanism is able to distribute streams dynamically and effectively. The combination between DirectShow's filters and this transmission mechanism will support a wide variety of multipoint audio-video communications.

Our current implementation has supported IPv4 and IPv6. In addition if users can use a relay node which has a global IP address, MidField System can communicate by multipoint peer-to-peer media streams even if each end user's node has a private IP address.

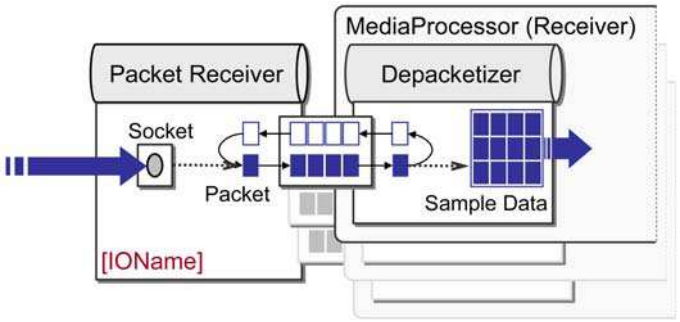


Fig. 13. Packet Receiver and Depacketizer(s)

4. Inside of Media Processor

Configurations of Media Processor are concerned with adaptive connectivity. In order to process a variety of inputs and outputs effectively, Media Processor's implementation employs DirectShow which has a modular architecture that can connect several software components called filters.

DirectShow provides several components that can be used to build filter graphs which are a set of connected filters. Suitable filters for a required streaming process are connected automatically by using the intelligent connection mechanism of DirectShow.

However, since each streaming process for DV, HDV, WMV, etc., has various special characteristics, not all filters are connected well automatically always.

4.1 Supported patterns of filter graphs

To satisfy the requirements for implementing adaptive streaming modules, we have designed eight patterns of filter graphs including packet transmission filters, which are shown in Figure 14.

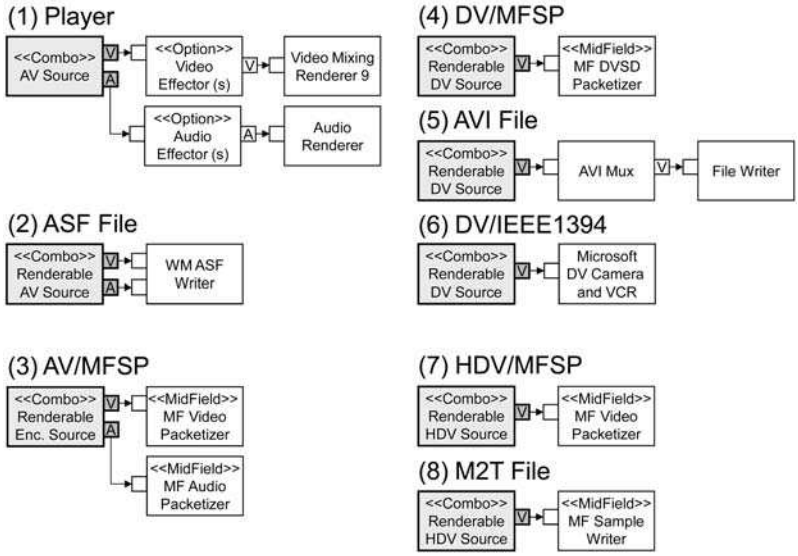


Fig. 14. Supported Filter Graphs in Media Processor

As the diagrams show, each filter is connected to one or more other filters, and the connection points are called pins. All filters use pins to move audio-video data from one filter to the next. The arrows in the diagrams show the direction in which the data travels.

Here, in the diagrams, <<Combo>> means the filter includes more several connected filters. For example, AV Source filter in Figure 14 is one of connected filter patterns shown in Figure 15. <<Option>> filter is inserted as necessary. In Figure 14 (1), Video Effector(s) and Audio Effector(s) are optional filters. Then <<MidField>> means the filter is implemented newly in MidField System for realising Stream Agent. There are several <<MidField>> filters, which appear also in following diagrams.

In Figure 14, the pattern (1) is a player that renders various AV sources shown in Figure 15. The pattern (2) creates an advanced systems format (ASF) file from a renderable AV source. Here, ASF is a container format for Windows Media Audio (WMA) and WMV contents. Then, the pattern (3) transmits outputs of Renderable Encoded Source filter shown in Figure 16 to destination(s). In order to send and receive media packets, we have designed MidField Streaming Protocol (MFSP), which is a simple stream transmission protocol that designed by referring to RTP. The MFSP is able to be used on TCP or UDP.

Next, in the pattern (4), outputs of Renderable DV Source are transmitted to destination(s) by using MFSP. The pattern (5) creates an AVI file from Renderable DV Source and the pattern (6) transmits a DV stream to an IEEE1394 device.

In order to process HDV stream, the pattern (7) transmits packets for a HDV stream to destination(s) and the pattern (8) creates a M2T (MPEG2-TS format) file from Renderable HDV Source.

4.2 Source filters and renderable sources

In Figure 15, there are six patterns of AV source filters, and each pattern creates outputs of audio or video sample data for a connected lower filter continuously. Basically, the output audio format is PCM and the output video format is RGB.

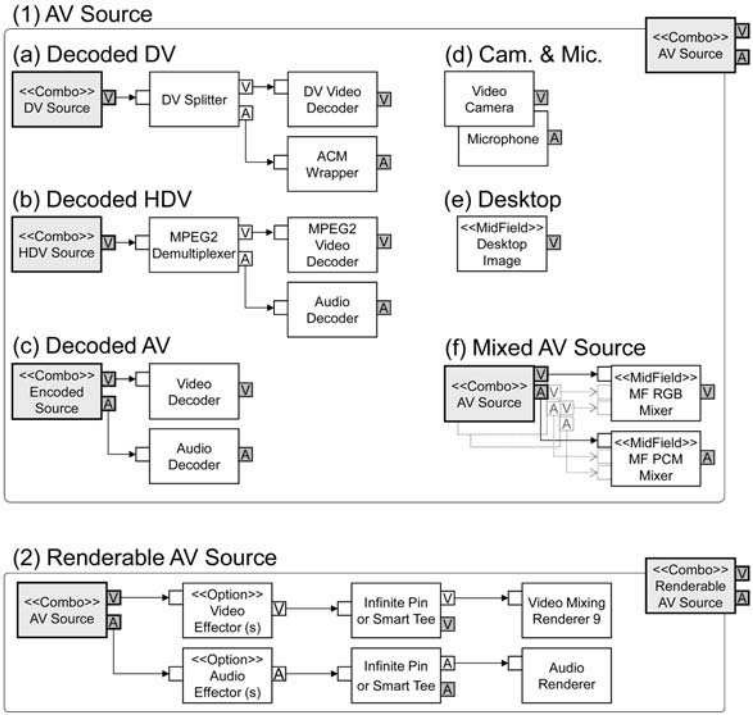


Fig. 15. AV Source Filters

Combo filters appear again in Figure 15 (a), (b) and (c). These combo filters are DV Source, HDV Source and Encoded Source. Decoded audio-video samples in these filters are available to use as an AV Source filter. In case of using microphones and video cameras such as USB video camera, the pattern (d) is applied to process these input devices. The pattern (e) captures and outputs continuously desktop images in RGB video format.

In addition, we have implemented audio-video mixer filters shown as the pattern (f). These filters take several input streams and mix the inputs. MF PCM Mixer adds each input PCM sample simply. MF RGB Mixer mixes each input RGB sample in spatial.

Although these AV Sources create output samples continuously only, if rendering audiovideo samples is required, Renderable AV Source filter is available. Renderable AV Source includes an AV Source, and can insert Audio/Video Effector(s) as necessary. The effected audio/video samples are shared in two directions for rendering by the infinite pin filter or the Smart Tee filter that are included in DirectShow. One of the outputs is connected to a renderer filter and another is just an output of Renderable AV Source.

Next, Figure 16 shows connected filters for encoded sources. MF Video Depacketizer and MF Audio Depacketizer filters receive media packets continuously. File Reader is used to read encoded media files stored in a local system. The same as AV Source in Figure 15, if

rendering encoded media samples is required, Renderable Encoded Source filter is available. Furthermore, in Renderable Encoded Source, type (b) supports re-encoding, where the source filter is a Renderable AV Source. Of course, Renderable AV Source is including the all patterns of Figure 15.

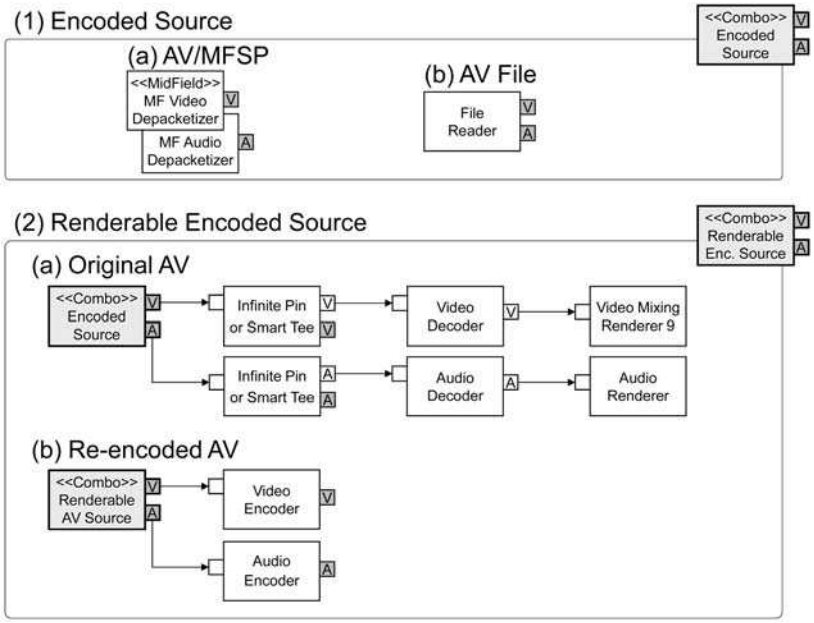


Fig. 16. Encoded Source Filters

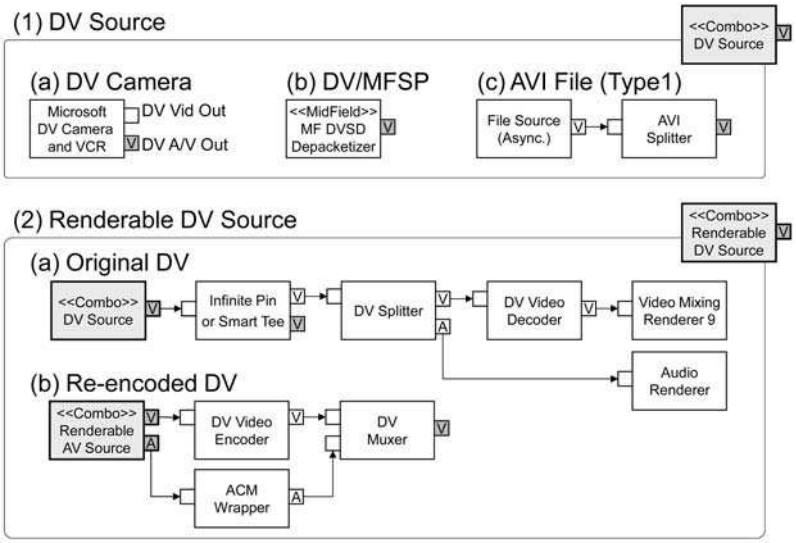


Fig. 17. DV Source Filters

Then, Figure 17 shows DV Source filters. As DV Source, our implementation supports IEEE1394 DV Camera, DV/MFSP and AVI File. The same as Encoded Source filter, both rendering and encoding of DV samples also are supported. In our system, supported filters for HDV are shown in Figure 18. The same as DV Source, our implementation supports IEEE1394 HDV camera, HDV/MFSP and MPEG2-TS File. Although rendering is available, unfortunately HDV encoding is not available.
Media Processor can select and connect these filters according to a set of input and output requirements. Then, a complete filter graph is created for handling internal media samples.

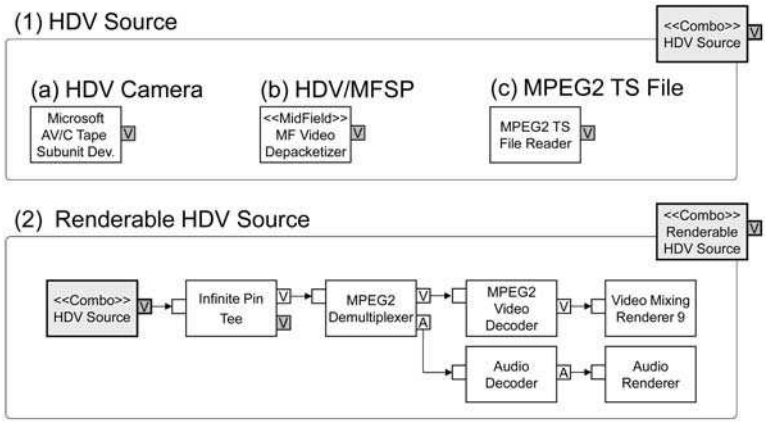
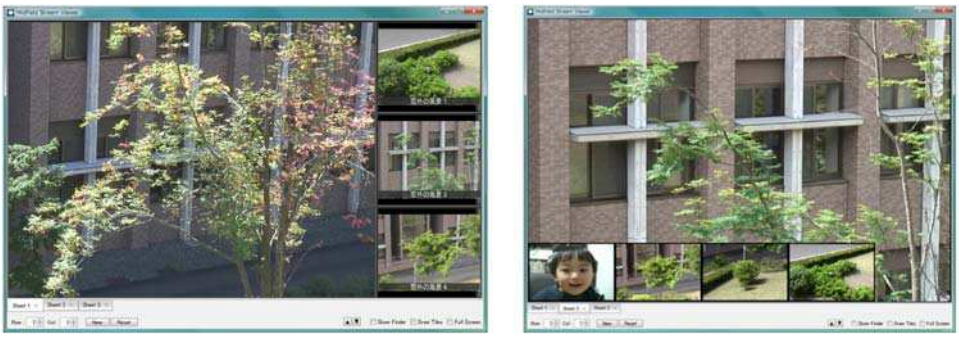


Fig. 18. HDV Source Filters

5. Current implementation

MidField System as a middleware provides both Java and C++ API for application programs. In the current implementation, Java API provides full functions that include a mobile agent platform of MidField System, and C++ API provides stream functions only.
Although MidField System is a middleware system, we have implemented not only the middleware functions but also an application program for audio-video communications. The application program is just a multipoint audio-video communication system.



(a) Tiled View (b) Picture-in-Picture

Fig. 19. Example Desktop Images of Stream Viewer

The application has a stream viewer that can customise for multipoint communication. Figure 19 (a) is a sample image of its tiled view, in this image the application is sending one video stream and receiving three video streams. Those streams are rendered in panel panes of the stream viewer respectively. Figure 19 (b) is another sample image. The application uses the video mixing function of MidField stream [Figure 15 (1) (f)] and each video stream is rendered by picture-in-picture in a panel pane.
In order to improve the performance of rendering video streams, all video images are drawn on a Java component's native window handler directly from a DirectShow's video rendering filter. Therefore, even if an application program uses the Java API of MidField System, the application is able to get high performance for rendering video images.

Video Input	
Capture Device	
Interface	IEEE1394, USB, VIDEO
Format	DV, HDV, RGB, YUV
Incoming Media Stream	
Protocol	MFSP/UDP, MFSP/TCP
Format	DV, HDV, WMV, etc.
Media File	
Format	DV(AVI), HDV(MPEG2-TS), WMV, etc.
Video Output	
Playout	
MidField Stream Viewer	
Outgoing Media Stream	
Protocol	MFSP/UDP, MFSP/TCP
Format	DV, HDV, WMV, etc.
Media File	
Format	DV(AVI), HDV(MPEG2-TS), WMV

Audio Input	
Capture Device	
Interface	Mic., Line, IEEE1394
Format	PCM
Incoming Media Stream	
Protocol	MFSP/UDP, MFSP/TCP
Format	PCM, etc.
Media File	
Format	WMA, etc.
Audio Output	
Playout	
MidField Stream Viewer	
Outgoing Media Stream	
Protocol	MFSP/UDP, MFSP/TCP
Format	PCM, etc.
Media File	
Format	WMA

Table 1. Supported I/O in Current Implementation

Table 1 shows audio and video I/O that current implementation of MidField System supports. Although a DV stream consumes about 28.8Mbps and a HDV stream consumes about 25Mbps network bandwidth, since we have supported WMV (Windows Media Video) and WMA (Windows Media Audio) as main codecs except for DV and HDV, MidField System can use WMV streams according to users' communication environments. Of course, MidField System can also employ external codecs.
Supporting these formats, by realizing adaptive connection mechanisms and relocatable transcoding functions, MidField System supports flexible and interactive communications according to users' resource environments.

6. Use cases

Since 2003, we have been using MidField System for multipoint communications. So far, the system has supported over thirty various remote communications.
The left of Figure 20 is a photograph of a symposium on disaster prevention (15 March 2005). The symposium was held on JGN2 (http://www.jgn.nict.go.jp/english/) that is an advanced testbed network for R&D. In the symposium, MidField System connected five remote locations in Japan. Each location sent a DV stream to a centre location and the five

DV streams were mixed for distribution, then the mixed DV stream were distributed to each location from the centre location.
We empirically know audio and video streams should be handled as separate streams in practical communication session. Although a DV is an interleaved audio-video stream, each location used not only a DV stream but also another audio stream. In this symposium, audio streams of each location also were mixed at the centre location. An audio stream that was received at each location contained audio data of the other locations only.
MidField System had supported one of distance learning projects from 2004 to 2008. The right of Figure 21 is a photograph of the distance learning class held on two locations (30 August 2007). In this distance learning, a large hydrogen ion accelerator at Tohoku University was controlled interactively from high school students in a remote location.
The location where the large hydrogen ion accelerator was in is different from the place of a lecturer, and therefore, several DV streams and WMV + PCM streams were used at the same time according to communication environments.

Fig. 20. Symposium on Disaster Prevention and Distance Learning

Since 2005, HDV streams also have been used for live video distribution. By using HDV streams we can get very high quality video images, but unfortunately HDV that employs MPEG2 codec has a delay than DV, therefore the HDV streams are not suitable for communications which require interactivity such as the above distance learning.
We are mainly using HDV streams as unidirectional multipoint live streams. The left of Figure 21 is an example of HDV distribution, which relayed Morioka Sansa Dance Festival in Morioka City of Iwate Prefecture, Japan.

Fig. 21. HDV and DV Distribution

The right of Figure 21 is a photograph of an international symposium that was held in University of Oulu, Finland (13 December 2007). MidField System was used as a unidirectional live video distribution system and a live DV stream was relayed from University of Oulu to Tohoku University.

The current implementation of MidField System can handle MIDI (Musical Instrument Digital Interface) messages. Captured MIDI messages from a musical instrument such as piano and electone are packetized and transmitted to remote musical instruments, and MidField System can make a sound on the remote instruments.
Combination of audio-video streams and MIDI streams may realize a new method of remote lesson. As shown in Figure 22, a remote lesson project is being carried out now, and if the special piano (right of Figure 22) is available the local piano's keys move up and down just like the remote piano.

Fig. 22. Remote Lesson of Piano

Figure 23 is an example of using external filters in Stream Agent. The left of photographs is a state of experimental communications for a remote mental health care project. In this project, MidField System uses an omni-directional camera as shown in the right of photographs and captures a view of 360 degree angle of a round table. The captured ring shaped video images are converted to panorama images in realtime (Ookuzu, 2008). The system realizes this panorama communication by Stream Agent which inserted conversion filters as Video Effector in Figure 15 (2).
Using DirectShow's filter architecture in implementation of Stream Agent has enabled easy functional extensions by external software modules.

Fig. 23. Using External Filters on MidField System

Moreover, by using MidField System as a middleware, a new surveillance system has been developed that name is Telegnosis System (Sato, 2009). The system has a combination of omni-directional camera and networked Pan-Tilt-Zoom (PTZ) camera, which can detect moving objects in wide area with 360 degree by the omni-directional camera image, and extract the position of the moving object. Then the relative pan and tilt angle for the PTZ camera are calculated from the current position of the extracted moving object, and the PTZ camera can focus on the object automatically.

The left photograph of Figure 24 shows the omni-directional camera and the networked PTZ camera, which are installed under the ceiling. In the right photograph, a moving object is detected by the omni-directional camera image and the object is focused by the PTZ camera.

Fig. 24. Telegnosis System

7. Conclusions

MidField System is a **Mid**dleware for **F**lexible and **i**nteractive **e**nvironment in **l**ong **d**istance. We can communicate with each other on computer networks by using the system. Supported DV and HDV video formats enable us to get high quality digital video images for interactive communications, and the system is able to construct audio-video communication sessions according to users' communication environments.

We are always improving the system through many practical audio-video communications. If we can use a HDMI (High-Definition Multimedia Interface) capture card that is able to use as a DirectShow's source filter, the latest implementation enables us to communicate each other by full HD video streams. In our ordinary communication, to use high quality digital video streams has become easier than before.

However to use the video streams in multipoint communications is not always easy. Even if we have communication environments enough to use high quality digital video streams, the preparations for audio-video communications are often hard works. To modulate levels of each location's microphones and speakers takes much time always. Trouble shootings for burst packet loss cause often us great distress. For settings of communication nodes on each location, we often need cell phones. Through practical communications, we have many problems to be solved in multipoint digital video communications.

In this chapter, the idea for integrating IP multicast sessions and the implementation have been described. The current implementation of MidField System has many useful functions for audio-video streaming as a multipoint communication system or a middleware, and enables a wide variety of audio-video communications such as high quality digital video distributions, distance educations, remote lessons and customized communication systems.

On the other hand, MidField System had realized a dynamic session configuration method, but unfortunately in many cases communication systems are restricted to use IP multicast. Now, we are trying to design and develop a new adaptive multipoint session configuration method as a future work, which will be able to improve our communication environments.

8. References

Antonio, L.; George, P. & Graham, K. (2002). Exploiting agent mobility for large-scale network monitoring, IEEE Network, No.3, pp.7-15.

Bak, Y.; Kim, Kapdong., Lee,Y., Sok, S., Kim, C., Kim, H., Kim, M., Kim, Kyongsok. (2006). Design and Implementation of High Performance SMART for HD Video On Demand, IEEE Int. Conf. on Hybrid Information Technology.

Bodecek, K. & Novotny, V. (2007). From standard definition to high definition migration in current digital video broadcasting, IEEE Int. Multi-Conference on Computing in the Global Information Technology.

Campbell, A.T.; De Meer, H., Kounavis, M.E., Miki, K., Vicente, J. & Villela, D.A. (1999). A survey of programmable networks, ACM Computer Communications Review.

Dong, H.N. & Seung, K.P. (2001). Adaptive multimedia stream service with intelligent proxy, IEEE Int. Conf. on Information Networking, pp.291-296.

Ferreira, P.; Veiga, L. & Ribeiro, C. (2003). OBIWAN: design and implementation of a middleware platform, IEEE Trans. Parallel and Distributed Systems, Vol.14, No.11, pp.1086-1099.

Gharai, L.; Perkins, C., Riley, R. & Mankin, A. (2002). Large scale video conferencing: a digital amphitheater, IEEE Int. Conf. on Distributed Multimedia System.

Gharai, L.; Lehman, T., Saurin, A. & Perkins, C. (2006). Experiences with High Definition Interactive Video Conferencing, IEEE Int. Conf. on Multimedia & Expo.

Guedes, L.A.; Oliveira, P.C., Faina, L.F. & Cardozo, E. (1997). QoS agency: an agent-based architecture for supporting quality of service in distributed multimedia systems, IEEE Int. Conf. on Protocols for Multimedia Systems - Multimedia Networking, pp.204-212.

Guo, L.; Chen, S., Ren, S., Chen, X. & Jiang, S. (2004). PROP: a scalable and reliable P2P assisted proxy streaming system, IEEE Int. Conf. on Distributed Computing Systems, pp.778-786.

Hashimoto, K. & Shibata, Y. (2008). Design and Implementation of Adaptive Streaming Modules for Multipoint Video Communication, IEEE Int. Symposium on Frontiers in Networking with Applications.

Jameela, A.J.; Nader, M., Hong J. & David S. (2003). Middleware Infrastructure for Parallel and Distributed Programming Models in Heterogeneous Systems, IEEE Trans. Parallel and Distributed Systems, Vol.14, No.11, pp.1100-1111.

Ohsuga, A.; Nagai, Y., Irie, Y., Hattori, M. & Honiden, S. (1997). PLANGENT: an approach to making mobile agents intelligent, IEEE Internet Computing, Vol.1, No.4, pp.50-57.

Ookuzu, H.; Sato, Y., Hashimoto, K. & Shibata, Y. (2008). A New Teleconference System by GigaEther Omni-directional Video Transmission for Healthcare Applications, International Computer Symposium, pp.431-436.

Sato, Y.; Hashimoto, K. & Shibata, Y. (2009). A Wide Area Surveillance Video System by Combination of Omni-directional and Network Controlled Cameras, Int. Conf. on Complex Intelligent and Software Intensive System.

Schulzrinne, H.; Casner, S., Frederick, R. & Jacobson, V. (2003). RTP: a transport protocol for real-time applications, RFC3550.

Digital Video

Edited by Floriano De Rango

ISBN 978-953-7619-70-1
Hard cover, 500 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

This book tries to address different aspects and issues related to video and multimedia distribution over the heterogeneous environment considering broadband satellite networks and general wireless systems where wireless communications and conditions can pose serious problems to the efficient and reliable delivery of content. Specific chapters of the book relate to different research topics covering the architectural aspects of the most famous DVB standard (DVB-T, DVB-S/S2, DVB-H etc.), the protocol aspects and the transmission techniques making use of MIMO, hierarchical modulation and lossy compression. In addition, research issues related to the application layer and to the content semantic, organization and research on the web have also been addressed in order to give a complete view of the problems. The network technologies used in the book are mainly broadband wireless and satellite networks. The book can be read by intermediate students, researchers, engineers or people with some knowledge or specialization in network topics.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821