

# A Web Services Based Streaming Gateway for Heterogeneous A/V Collaboration

Hasan Bulut<sup>1</sup>, Wenjun Wu<sup>1</sup>, Geoffrey Fox<sup>1</sup>, Ahmet Uyar<sup>2</sup>, Shrideep Pallickara<sup>1</sup>, Harun Altay<sup>1</sup>  
<sup>1</sup>Community Grid Computing Laboratory, Indiana University  
Indiana Univ Research Park, 501 North Morton Street, Suite 222, Bloomington, IN 47404  
<sup>2</sup>Department of Electrical Engineering and Computer Science, Syracuse University

## Abstract

*One of the most widely used media delivery technology across Internet is streaming. It enables users to receive multimedia content and provides some control functionality over the media. Videoconferencing technologies such as H.323, SIP and AccessGrid, provide audio and visual interactive environment to their clients. We have developed Global multimedia Conferencing System (Global MMCS), which is an integrated videoconferencing system that allows heterogeneous multimedia clients, such as H.323, SIP and AccessGrid, to join the same videoconferencing session. In this paper we present a web services based streaming gateway that enables RealStream clients to join the real-time videoconferencing sessions and receive multimedia content generated by other clients in the system.*

*Keywords—RealStreaming, Videoconferencing, XGSP, Global-MMCS, Web Services*

## 1. Introduction

Videoconferencing systems provide a framework to send/receive audio and video streams. They consist of three parts: clients, session servers and communication channels. For example, in an H.323 [1] based system, a client refers to the H.323 endpoint that is capable of sending/receiving audio and video. H.323 client can only send/receive one audio and video stream. A session server refers to the Multipoint Controller that can generate multipoint session. A communication channel is the Multipoint Processor that can mix audio and video streams from different clients. In an AccessGrid [2] system, a client is based on the MBONE audio/video tools such as RAT for audio and VIC for video. Further there is a venues server in AccessGrid, which is responsible for scheduling meeting. Multicast RTP Channels are the communication infrastructure for AccessGrid.

AccessGrid clients VIC and RAT can receive and play multiple A/V streams in the session so that they can interact with each other.

Streaming provides a framework to deliver media stream across Internet. A streaming client connects to a streaming server, primarily using Real Time Streaming Protocol [3] (RTSP), to establish a session and receive the stream. Streaming is primarily used for Media-On-Demand, receiving media that resides on a streaming server whenever a client wants to play. It is also possible to make a live streaming broadcast. The connection between the client and the server is stateful. RTSP is a client-server multimedia presentation control protocol. It provides VCR-like control, so that clients can pause, fast forward, reverse, and absolute positioning etc. Streaming media aims to improve the quality of service of the stream, by using various streaming codec to improve stream quality and a buffering mechanism to reduce the jitter. This buffering mechanism causes the stream to be delayed for some time. Hence, streaming media does not provide a good interactive environment compared to the interactivity in a videoconferencing system, but the stream quality is improved.

Another feature of streaming that distinguishes it from conferencing is that streaming uses compression algorithms that cause streams to be compressed more so that less bandwidth is required in order to transmit the stream across the network. The side effect of this is that the compression/decompression requires much CPU because it involves too much computation. Also the buffer delay in streaming is much longer than the buffer delay in conferencing. While in conferencing it is around 200 msec, the delay in streaming buffer is around 2 – 15 sec.

We have developed XML based General Session Protocol [4-8] (XGSP), a web services based conference control framework, to integrate various videoconferencing systems: H.323-based systems, AccessGrid, and SIP [9]-based systems. We added

XGSP clients to this framework, so that any MBONE tool can join a XGSP session to send/receive A/V stream. In order to deliver A/V streams in a XGSP session to streaming media clients, we also need to extend this framework with defining and designing a novel Streaming Gateway to achieve this integration. In our works, we used RealNetwork [10] Streaming formats and Helix Streaming Server [11] and Helix Streaming Engines [12] to convert raw data into RealStreaming format.

The paper is organized in the following way: Section 2 describes the related work. Section 3 provides a brief description of XGSP and Global Multimedia Collaboration System [4-8] (Global-MMCS), which is a prototype implementation of XGSP. Section 4 explains streaming gateway design and messaging interface. We will explain the description of administrator and client interface for streaming gateway in section 5. Section 6 provides a brief performance discussion and we will present conclusion in section 7.

## 2. Related Work

Global-MMCS is a web services-based conference control system and supports multiple protocols such as H.323 and SIP. It is Web Services Reliable Messaging [13] (WS-RM) compatible. Streaming Gateway is also web services-based and provides the compatibility that a web service provides. Streaming Gateway, provided to Global-MMCS, is currently the only work that integrates videoconferencing environments to RealStreaming world. But there are other attempts to deliver RealStream in a webcasting style and other streaming media formats such as QuickTime [14], which is Apple [15] streaming technology.

The Infrastructure for Distributed Video and Audio [16-18] (INDIVA) is a middleware system that provides high-level abstractions for accessing equipment and managing the interface between production equipment and streaming media applications. It is a webcasting system where streams are converted into RealStream at the source.

Similar to Global-MMCS, Virtual Rooms Videoconferencing System [19] (VRVS) integrates MBONE and H.323 tools with AccessGrid Virtual Venues, but VRVS uses the reflector model to achieve this integration. Reflectors are brokers similar to NaradaBrokering. Using VRVS, users from disparate geographic locations can meet and participate in MBONE, H.323 or MPEG2 multipoint videoconferences. VRVS supports QuickTime to enable its clients receive audio/video using QuickTime player. But VRVS is not web services-based and it is not WS-RM compatible.

## 3. XGSP and Global-MMCS

XML based General Session Protocol (XGSP) conference control framework is a generic, easy to extend, reliable and scalable framework. XGSP conference control includes three components: user session management, application session management and floor control. User session management supports user sign-in, user create/terminate/join/leave/invite-into XGSP sessions. XGSP application session management provides the services to A/V and data application endpoints and communities, controlling multipoint A/V RTP and data channels. Since different A/V application endpoints have their own signaling procedures for joining and leaving A/V sessions, we defined a XGSP signaling protocol for H.225 [20], H.245[21] (H.323 signaling protocols) and SIP as well as AccessGrid. Floor control manages the access to shared collaboration resources.

Global Multimedia Collaboration System (Global-MMCS) is a prototype system to verify and refine our XGSP conference control framework. The prototype system is shown in Figure 1. Global-MMCS prototype is built based on the NaradaBrokering middleware. All the A/V processing components, including the video mixer, audio mixer as well as the image grabber servers are developed using Java Media Framework [22] (JMF). JMF package, which can capture, playback, stream and transcode multiple media formats, gives multimedia developers a powerful toolkit to develop scalable, cross-platform technology.

We use the protocol stacks from the open source projects, including OpenH323 [23] and NIST-SIP [24] project to implement the H.323 gateway and SIP gateway. The XGSP Session Server is built to manage real-time A/V sessions, receiving messages from gateways and the web server through different control topics on the NaradaBrokering. The XGSP Session Server translates the high-level commands from the XGSP Web Server into signaling messages of XGSP, and sends these signaling messages to the NaradaBrokering infrastructure to generate a publish/subscribe session over the Narada brokers. These Narada brokers take the tasks of routing and forwarding video/audio events to various communities and collaboration clients. The XGSP web server based on Apache Tomcat provides an easy-to-use web interface for users to make meeting schedules, join XGSP conferences and for administrators to perform the tasks of the system management.

The XGSP conference manager is implemented as an embedded server in the web container. It can create/destroy conferences, activate/deactivate A/V application sessions and generate the active conference directory to all the users. Users should log into Global-

MMCS through their web browsers and select active conferences. The NaradaBrokering infrastructure provides a scalable distributed messaging platform for RTP communications in these A/V collaboration applications. Whenever a new session is activated across Global-MMCS, the same *topic* will be generated inside NaradaBrokering system by XGSP Session Server. Any RTP client or server who wants to join this session, it can *subscribe* to this topic and *publish* its RTP messages through RTP Proxies in the NaradaBrokering system.

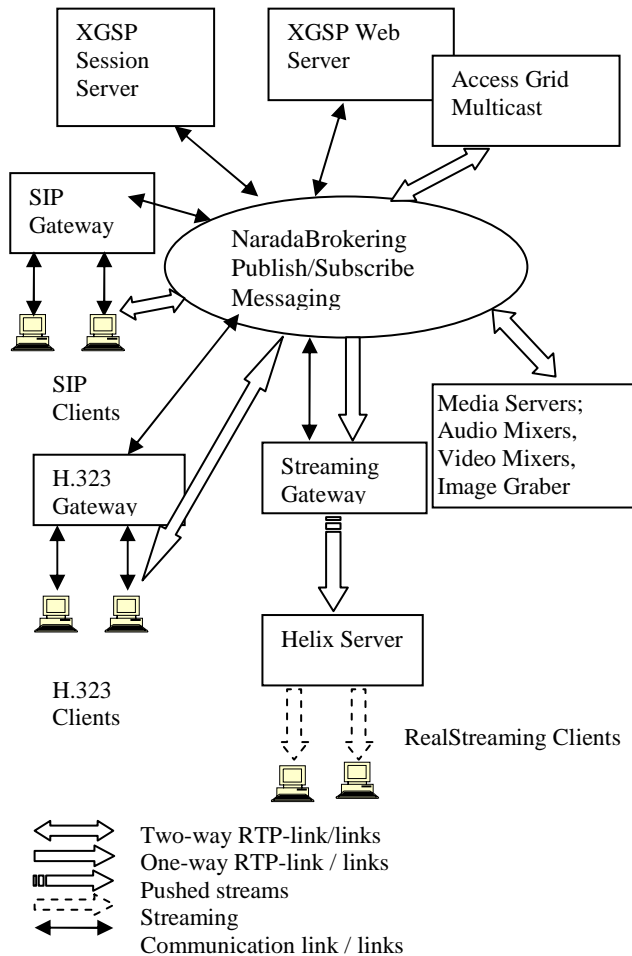


Figure 1: XGSP prototype systems

#### 4. XGSP Streaming Gateway

The implementation of XGSP Streaming Gateway is different from other gateways, i.e. H.323 and SIP, because the requirements of RealStreaming clients are different than the requirements of other clients in the system. H.323 and SIP gateways transform communication signal from one form to XGSP messages or from XGSP messages to another form.

Also they hide some communication details from XGSP Session Server. For example, H.323 is a complicated binary protocol because of the way it's implemented. The H.323 gateway transforms some H.323 binary messages into XGSP messages and from XGSP messages to H.323 binary messages and also hides the complexities from XGSP Session Server. Session Server only receives information that it requires from an H.323 client and sends information that will be needed by the client to join the session. Other signaling details are hidden from Session Server. SIP uses text messages but it requires different format than XML. Both protocols are signaling protocols and enable their clients to establish session and exchange streams. AccessGrid uses multicast, so AG clients just send/receive streams to/from a multicast address.

In RealStreaming case, clients require different stream format and use RTSP to receive streams. So, one of the jobs of Streaming Gateway is to convert the stream from formats like H.261 to RealStream format. This conversion process requires two transcoding stages: first step is to decode the received stream and the second stage is to transcode the output of the first stage into RealStreaming format. This conversion process causes delays in the generated stream. Other delays may be introduced from the system that Streaming Gateway is running and from the underlying network to transmit those streams to Helix Streaming Server. Due to these delays in the system and the network the generated A/V streams and other collaboration events should be synchronized. Streaming Gateway is also responsible for this synchronization.

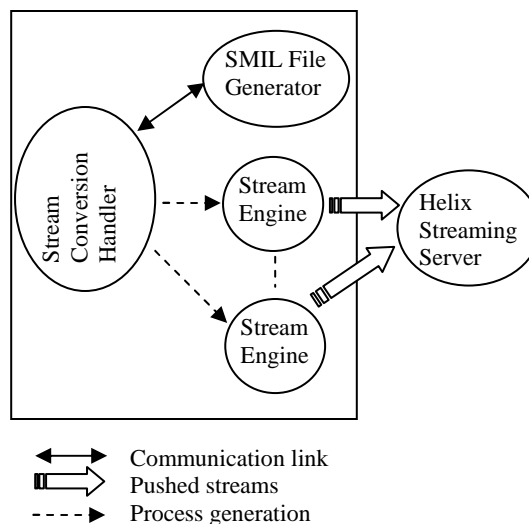


Figure 2: XGSP Streaming Gateway components

RealStreaming clients use RTSP to receive streams from RealStreaming Servers. We used Helix Streaming

Server, which is available under RealNetworks public and community source licenses. Streaming Gateway does not only provide a signaling mechanism but also provides some conversion mechanisms. This aspect of Streaming Gateway makes it different from other gateways. A RealStreaming client communicates with the server in order to establish a channel or channels to receive stream and to send/receive control information. Components of XGSP Streaming Gateway are shown in Figure 2.

## 4.1 XGSP Streaming Gateway Components and Design

Streaming Gateway is composed of several logical components: stream conversion handler, stream engine and SMIL file generator.

### 4.1.1. Stream Conversion Handler

Stream Conversion Handler handles the communication between Session Server and Streaming Gateway. It keeps an internal database for the streams being converted. This database is updated when the streaming jobs are started or deleted. In order to start a streaming job, it initiates a Stream Engine for the requested stream and passes required parameters such as conversion format, helix server address, stream name to the Stream Engine.

### 4.1.2. Stream Engine

This component can be considered as the most fundamental component of the Streaming Gateway. Stream Engine is responsible for converting the received audio or video streams into a specified RealStreaming format and pushes the converted stream to Helix Streaming Server. Streaming Engine is composed of two parts, JMF RTP Handler and HXTA Wrapper. HXTA is a conversion engine provided by Helix Community and converts raw audio and video data into RealStreaming formats.

JMF RTP Handler receives audio and video packets from a local port provided by Stream Conversion Handler. The purpose of this unit is to transform the received packets into a format that HXTA can understand and be able to make the conversion to RealStreaming format. Raw audio and video data can be passed to HXTA Wrapper. There are several color spaces for video representation. But two of the most common are RGB (red/green/blue) and YCrCb (luminance/red chrominance/blue chrominance). HXTA accepts different formats of RGB and YCrCb. As the first conversion step, JMF RTP Handler decodes the received video frames into YCrCb format. We prefer YCrCb, because RGB requires more memory to represent video images compared to

YCrCb. JMF RTP Handler passes these decoded frames to HXTA Wrapper over a buffer. Audio packets are decoded into raw audio format, for our case WAV format, before passing it to HXTA Wrapper. Another responsibility of JMF RTP Handler is to make sure that packets are processed in an order. If a packet arrives later than a packet that carries bigger timestamp, it is dropped.

JMF RTP Handler gets the media type of the stream from the input provided by Stream Conversion Handler. Based on that information it decodes received packets into raw video or audio format. Each stream, whether it is audio or video, is transcoded into streaming format independently from each other.

### 4.1.3. SMIL File Generator

Streaming Engine receives only one stream, whether it is audio or video, and produces one stream. In order to enable streaming clients to receive audio and video together, there is need for a SMIL file, which resides on the Helix Streaming Server side. Because of this, Stream Conversion Handler provides RTSP links of audio and video streams to SMIL file generator and SMIL file generator produces a SMIL file that includes those RTSP links. In our current implementation we have only one audio stream per session, which is mixed of all available audio streams in that session. So when a video stream is to be converted into streaming format, the RTSP link for the mixed audio stream is included to the generated SMIL file.

## 4.2. Streaming Gateway Interface

This part explains the XML message formats added to XGSP in order to enable the communication between Streaming Gateway and Global-MMCS. In order to define the required parameters for a streaming job, we defined an xml structure and named as *Input*. It provides the necessary information, such as broadcast address, stream name, stream server IP address and port number, media type, required to start a stream conversion job. XML message formats added to XGSP are as follows:

*InitializeRealGateway*: This message causes Streaming Gateway to delete all current jobs in the session specified.

*JoinStream*: When Streaming Gateway receives this message, it starts a job for the stream specified. Besides Input information, the message also includes session ID and other stream specific information, such as username, SSRC.

*JoinStreamReply*: This is the reply message for JoinStream. If somehow the job cannot be started, it returns FAIL, otherwise it returns OK to indicate the job is successfully started.

*LeaveStream*: Streaming Gateway stops the stream specified.

*LeaveStreamReply*: This is the reply message for *LeaveStream*. If somehow the stream is stopped successfully, it returns OK, otherwise it returns FAIL to indicate that any error occurred during the stop operation.

*RealStreamEvent*: It has two modes, *NewRealStream* and *ByeRealStream*. This event is generated by Session Server. Streaming lists are updated when this message is received. The stream is added to the list if the mode is *NewRealStream*, it is removed if the mode is *ByeRealStream*.

*RealStreams*: When a client/administrator joins to the session it requests the list of the *RealStreams* available in the session from Session Server by sending *RealStreams* message.

*RealStreamsReply*: This is reply message for *RealStreams* message. *RealStreams* are expressed in *RealStreamEvent* format and the list is included to *RealStreamsReply* message.

### 4.3 Services Added to Session Server to Enable Communication

In order to support Streaming Gateway, SessionServer provides some services. These services are *RealStream Join Service*, *RealStream Leave Service*, *RealStream List Service* and *RealStream Gateway Service*. Session Server also keeps a database for the streams being converted. The description of these services is as follows:

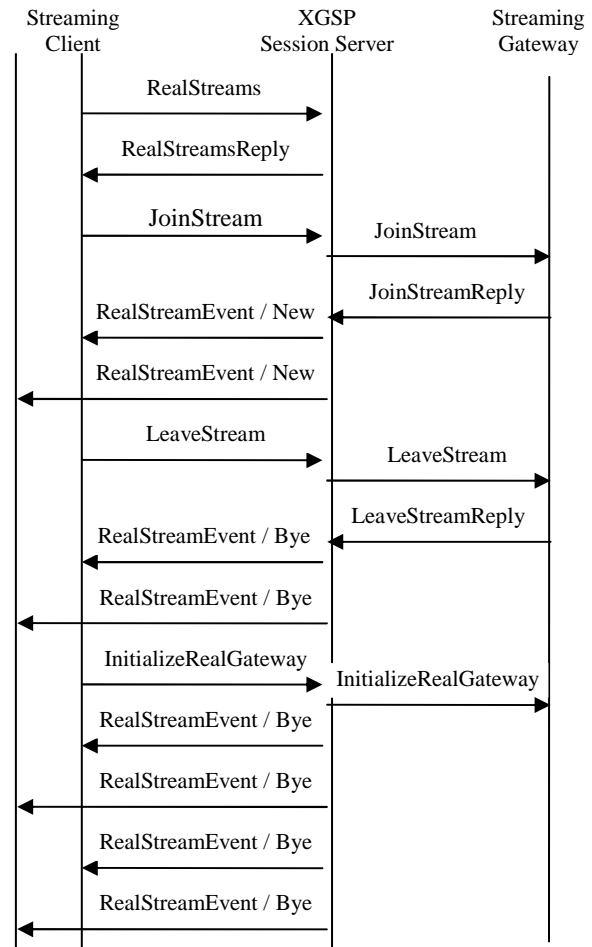
*RealStream Join Service*: This service handles *JoinStream* messages. Session Server includes additional information to *JoinStream* message and sends it to Streaming Gateway. If Session Server receives OK reply from Streaming Gateway, it sends *RealStreamEvent* message with *NewRealStream* mode to administrator and clients in that session.

*RealStream Leave Service*: This service handles *LeaveStream* messages. SessionServer forwards the message to Streaming Gateway. Streaming Gateway sends *LeaveStreamReply* message to SessionServer after processing the request. Then SessionServer generates a *RealStreamEvent* with *ByeRealStream* mode.

*RealStream List Service*: When the streaming client/administrator first joins to the session, it sends a *RealStreams* message to request a list of the available *RealStreams* on the Streaming Gateway. Session Server replies back with *RealStreamsReply* listing all the available *RealStreams* in the session.

*RealStream Gateway Service*: Streaming Admin can send *InitializeRealGateway* message to initialize the session for Streaming Gateway. Session Server forwards the message to Streaming Gateway and generates a *RealStreamEvent* with *ByeRealStream* mode for each *RealStream* in that session.

### 4.4 Signaling Between Session Server and Streaming Gateway



**Figure 3:** A sample signaling case among Session Server, Streaming Gateway and Streaming Client/Administrator

Figure 3 shows a signaling scenario among streaming administrator, streaming client, XGSP Session Server and Streaming Gateway. When streaming administrator first connects to the Session Server, it requests a list of the available streams and *RealStream* streams in the session. Streaming administrator sends *RequestStreamList*, to request all of the available audio/video streams and *RealStreams*, to request all of the available *RealStream* streams consecutively. In the reply, Session Server replies with *RequestAllStreamsReply* and *RealStreamsReply*. Streaming client only sends *RealStreams* message to receive available *RealStream* streams list. Next streaming administrator sends *JoinStream* for the chosen stream. Session Server adds some other fields to the same message and forwards it to Streaming Gateway. Streaming Gateway replies with *JoinStreamReply* and as a result Session Server generates *RealStreamEvent* messages with *NewRealStream* mode and sends them to

streaming administrator and streaming clients. LeaveStream has a similar scenario. Only instead of sending RealStreamEvent with NewRealStream mode, Session Server sends RealStreamEvent with ByeRealStream mode. In InitializeRealGateway case, Session Server forwards the message to Streaming Gateway and also sends RealStreamEvent with ByeRealStream mode for each of the streams removed.

## 5. Streaming Interface

Two types of interfaces are developed for Streaming Gateway. Streaming Admin and Streaming Client interfaces. As can be deduced from their names, Streaming Admin is implemented for administrative purposes and Streaming Client is implemented for accessing streams from an interface. Some part of the Streaming Admin interface is adapted from Streaming Client interface to enable administrator with client capabilities.

### 5.1 Administrator Interface

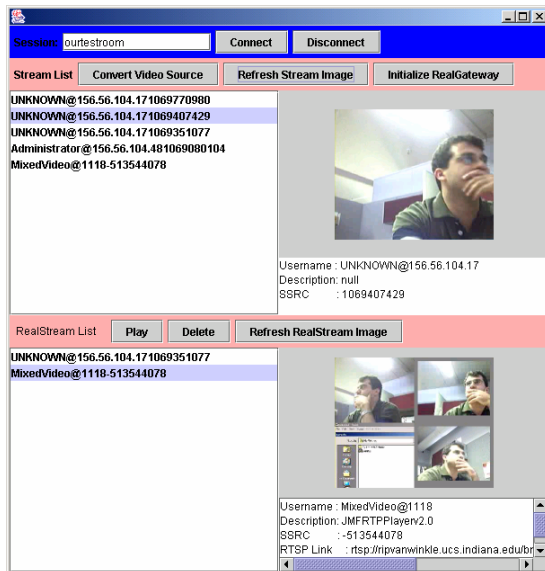


Figure 4: A screenshot of Streaming Admin interface

Streaming Admin is designed and implemented to enable only the administrator to choose stream to be converted to RealStream. This conversion, especially for video streams, takes noticeable CPU percentage. Because of this and other administrative purposes, regular clients cannot be allowed to start and stop streaming jobs. The administrator can see the available streams in XGSP sessions. This enables the administrator to select streams to be converted.

As shown in Figure 4, the streams are listed by their unique IDs. ID for each stream is constructed by appending the SSRC number to their usernames. The

administrator can visualize the information for that stream by selecting it. Available image and other information; username, SSRC and description, regarding to a stream selected are also provided. After selecting the video source, administrator can start the streaming job by clicking *Convert Video Source*. The south of the panel is adapted from Streaming Client interface. In addition to playing streams, administrator can stop a RealStream by selecting it and clicking *Delete* button.

### 5.2 Client Interface

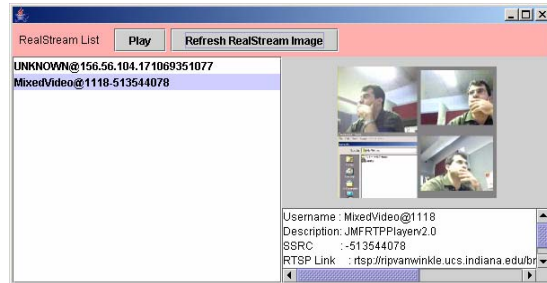


Figure 5: A screenshot of Streaming Client interface

Streaming Client allows users to see the stream information and to play a stream from RealPlayer window. This interface does not allow clients to stop any stream. A screenshot of the Streaming Client Interface is shown in Figure 5. The west of the panel shows the available streams, the east of the panel shows the image and information regarding to the stream chosen. When user selects a stream and clicks *Play* button, a RealPlayer windows pops up and plays the stream.



Figure 6: A screenshot of a RealStream played in RealPlayer window.

The stream lists on interfaces are dynamically updated when a stream leaves/joins the session. Figure 6 shows a stream played in a RealPlayer window.

## 6. Performance Discussion

Stream conversion is a CPU intensive application. In order to see the CPU and memory usage of this conversion we also observed the effect of number of

streams converted on CPU and memory usage. Streaming Gateway is running on the machine specified in Table 1.

**Table 1:** Streaming Gateway machine specifications

OS	Windows XP
CPU	Intel® Pentium® 4 CPU 2.26 GHz
Memory	512 MB
JVM version	1.4.2_02

**Table 2:** Approximate CPU and memory usage with respect to number of streams

Number of Streams	Frame rates of streams involved	CPU Usage Range	Memory Usage Range
1	23 fps	%10 - %25	29 MB
2	23 fps, 25 fps	%22- %40	34 MB
3	23 fps, 25 fps, 26 fps	%50 - %75	43 MB
4	23 fps, 25 fps, 26 fps, 16 fps	%65- %95	53 MB

Table 2 provides approximate CPU and memory usage of the streams. As the number of streams converted increases the CPU usage and memory usage also increases. In this specific machine we could successfully convert 4 streams without causing quality of services decrease. If we increase the number of streams, other streams are also affected and some time later the conversion performance reduces much. In this test we kept the frame rate high for most of the streams, which is normal in an Access Grid session.

## 7. Conclusion and Future Work

In this paper, we have described a web-services based Streaming Gateway for integration of real-time videoconferencing and streaming media. Streaming Gateway is part of Global-MMCS and uses XGSP in order to achieve this integration. XGSP framework and Global-MMCS enable multiple communities to collaborate with each other. Streaming Gateway enriches this heterogeneous system by integrating streaming media world into Global-MMCS. In order to increase the number of streams converted, we also would like to build a streaming job scheduler, which will schedule and coordinate streaming jobs in a distributed environment.

## 8. References

[1] International Telecommunication Union, "Packet based multimedia communication systems", Recommendation H.323, Geneva, Switzerland, Feb. 1998.  
 [2] Access Grid, <http://www.accessgrid.org>  
 [3] Real Time Streaming Protocol (RTSP), <http://www.ietf.org/rfc/rfc2326.txt>

[4] Wenjun Wu, Geoffrey Fox, Hasan Bulut, Ahmet Uyar, Harun Altay "Design and Implementation of a Collaboration Web-services System", to be published in special issue on Grid computing in Journal of Neural Parallel and Scientific Computations (NPSC)  
 [5] Wenjun Wu, Hasan Bulut, Ahmet Uyar, Geoffrey C. Fox, "A Web-Services Based Conference Control Framework For Heterogenous A/V Collaboration", 7th IASTED International Conference on Internet and Multimedia Systems and Applications ~IMSA 2003~ August 13-15, 2003 Honolulu, Hawaii, USA  
 [6] Geoffrey Fox, Wenjun Wu, Ahmet Uyar, and Hasan Bulut, "A Web Services Framework for Collaboration and Audio/Videoconferencing", The 2002 International MultiConference in Computer Science and Computer Engineering, Internet Computing(IC'02), June 2002, Las Vegas  
 [7] Wenjun Wu, Ahmet Uyar, Hasan Bulut, Geoffrey Fox, Integration of SIP VoIP and Messaging Systems with AccessGrid and H.323, the proceedings of The 2003 International Conference on Web Services (ICWS'03), June 2003, Las Vegas, ND, USA.  
 [8] Geoffrey Fox, Wenjun Wu, Ahmet Uyar, Hasan Bulut, Shrideep Pallickara, "Global Multimedia Collaboration System", 1st International Workshop on Middleware for Grid Computing, June 2003, Rio de Janeiro, Brazil.  
 [9] Session Initiation Protocol (SIP), RFC 2543, <http://www.ietf.org/rfc/rfc2543.txt>.  
 [10] RealNetworks, <http://www.realnetworks.com/>  
 [11] Helix DNA Server <https://helix-server.helixcommunity.org/>  
 [12] Helix DNA Producer, <https://helix-producer.helixcommunity.org/>  
 [13] Web Services Reliable Messaging Protocol (WS-ReliableMessaging) <http://www-106.ibm.com/developerworks/library/ws-rm/>  
 [14] QuickTime, <http://www.apple.com/quicktime/>  
 [15] Apple, <http://www.apple.com/>  
 [16] L.A. Rowe, W.T. Ooi, and P. Pletcher, "INDIVA: Distributed Streaming Media and Equipment Control Middleware", Berkeley Multimedia Research Center, TR 2002-163, April 2002  
 [17] L.A. Rowe, "Streaming Media Middleware is more than Streaming Media", International Workshop on Multimedia Middleware, October 2001  
 [18] T.P. Yu, D. Wu, K. Meyer-Patel, and L.A. Rowe, "dc: A Live Webcast Control System", SPIE Multimedia Computing and Networking, January 2001  
 [19] Virtual Rooms Video Conferencing System (VRVS), <http://www.vrvs.org>  
 [20] International Telecommunication Union, Calling Signaling Protocols and Media Stream Packetization for Packet-based Multimedia Communication Systems", Recommendation H.225.0, Feb, 2000  
 [21] International Telecommunication Union, Control Protocols for Multimedia Communication", Recommendation H.245 Feb, 2000  
 [22] Sun Microsystems, Java Media Framework 2.1, (2001), <http://java.sun.com/products/javamedia/jmf/2.1.1/index.html>  
 [23] The OpenH323 Project, <http://www.openh323.org/>  
 [24] National Institute of Standards and Technology, <http://dns.antd.nist.gov/proj/iptel/>