2006

Service Oriented Architecture for VoIP Conferencing

Wenjun Wu
Indiana University, Community Grids Computing Laboratory

Geoffrey C. Fox
Indiana University, Community Grids Computing Laboratory

Hasan Bulut
Indiana University

Ahmet Uyar
Syracuse University, Department of Electrical Engineering and Computer Science

Tao Huang
Indiana University, Community Grids Computing Laboratory

Service Oriented Architecture for VoIP conferencing

Wenjun Wu[1], Geoffrey Fox[1] , Hasan Bulut[1] , Ahmet Uyar[2], Tao Huang[1]

[1] *Community Grids Computing Laboratory, Indiana University, USA*
[2] *Department of Electrical Engineering and Computer Science, Syracuse University, USA*
{wewu, gcf, hbulut, auyar, taohuang}@indiana.edu
Indiana University Research Park, 501 N Morton St. 222, Bloomington, IN, 47408, USA

Abstract

Voice/Video over IP (VoIP) systems to date have been either highly centralized or dependent on the IP multicast in nature. Global Multimedia Collaboration System is a scalable, integrated and service-oriented VoIP conferencing system, based on the XGSP collaboration framework and NaradaBrokering messaging middleware. This system can provide media and session services to heterogeneous endpoints such as H.323, SIP, Access Grid, RealPlayer as well as cellular phone. In this paper, we address the challenges of scalability, interoperablity and heterogeneity in massive VoIP conferencing system. We believe that our approach opens up new opportunities for leveraging classic VoIP systems by using new technologies in service-oriented computing.

Keywords

Videoconference, Service-Oriented Architecture, Global-MMCS, XGSP, NaradaBrokering,

1. Introduction

VoIP and videoconferencing systems are increasingly becoming a very important and popular application on the Internet. There are various solutions to such multimedia communication applications, among which H.323 [1], SIP [2], and Access Grid [3] are well-known. It will bring substantial benefits to Internet users if we can build an integrated collaboration environment, which combines these systems into a single easy-to-use, intuitive environment. These emerging advanced distributed multimedia services demand a scalable, robust and adaptive service infrastructure. However, both the traditional telecommunication architecture based on client/server model and multicast based Internet model can't naturally offer such a platform. Currently, service-oriented architecture is regarded as a promising solution to Internet applications such as peer-to-peer computing, Grid and Web-Services computing. It suggests a powerful vision for the future VoIP and videoconferencing systems.

We propose a service-oriented VoIP conferencing framework which can interoperate with standards – based VoIP systems to forge an open software solution that will leverage existing conferencing resources. XGSP (XML based General Session Protocol) [4] is an interoperable framework based on Web services technology for creating and controlling audio and videoconferences. Just like the text messages in SIP, XML should be used to describe the XGSP protocol because it makes the protocol easier to be understood and to interact with other Web based components.

Based on this framework, Global-MMCS (Global Multimedia Collaboration System) [5] is developed to support scalable web-service based interoperable VoIP conferences and integrate various services including videoconferencing, instant messaging and streaming. Global-MMCS uses a unified, scalable, QoS aware "overlay" network – NaradaBrokering [6], to support audiovisual and data group communication over heterogeneous networking environments. Using XGSP schema, Global-MMCS specifies a distributed flexible conference management mechanism for integration of various VoIP conferencing services, and a common audiovisual signaling protocol for interactions between different conferencing endpoints. Instead of following the telecommunication monolithic service provision which usually mixes the packet delivery, call control functions and service logic intelligence in the central hardware boxes, it separates the MCU/Softswitch in VoIP into distinct services for media delivery, media processing and session management. These services can be distributed and if needed replicated for performance and fault tolerance. On top of the messaging and VoIP media services infrastructure, rich multimedia collaborations can be built to meet the demands in different application scenarios. Also several Gateway services are developed to enable multimedia clients to interact with the core services.

The paper is organized in the following way. Section 2 introduces related work and our design principles. System architecture is discussed in Section 3. Section 4 presents the implementation of the system and performance evaluation. Section 5 gives the conclusion and points out the future work.

2. Related Work

The VoIP conference has been existed over years. The well-known solutions include H.323, SIP and IETF MMUSIC [7]. Here we introduce them briefly and compare them with our design principles.

2.1 H.323 and SIP

H.323 is a communication standard produced by the ITU, initiated in late 1996, and aimed at the emerging area of multimedia communication over LANs. It is widely supported by many commercial vendors and used throughout the world in commercial and educational markets. H.323 is defined as an umbrella standard specifying the components to be used within an H.323-based environment. It provides conference management functionality for VoIP conferences using the call signaling functionality of H.225 [8], H.245 [9]. These protocols provide call set-up and call transfer of real-time connections to support small-scale multipoint conferences. The protocol H.243 [9] defines some commands between the MCU and H.323 terminals to implement audio mixing, video switch and cascading MCU. H.243 commands have been included in H.245. For the data part of a conference, the conference management of the T.120 recommendation [11] is used. This standard contains a series of communication and application protocols and services that provide support for real-time, multi-point data collaborative applications including desktop data conferencing, multi-user applications, and multi-player gaming.

An H.323 conference system for packet switched networks can include one or more of the following functional components: terminals, gatekeeper (GK), multipoint controller (MC), multipoint processor (MP) and multipoint control unit (MCU). The H.323 control messages and procedures define how these components communicate. The H.323 MC and each H.323 participant in the conference establish an H.245 control connection to negotiate media communication types. The MP provides media switching and mixing functionality, e.g. the MP decides which of the media streams generated by the clients will be forwarded or mixed as a single stream. The H.323 Gatekeeper provides services such as address translation, RAS control, call redirection and zone management to H.323 clients. The gatekeeper may also provide other optional functions such as call authorization and call accounting information.

The Session Initiation Protocol (SIP) defines how to establish, maintain and terminate Internet sessions including multimedia conferences. Initially SIP was designed to solve problems for IP telephony. To this end, SIP provides basic functions including: user location resolution, capability negotiation, and call management. All the capabilities are basically equivalent to the service H.225 and H.245 in H.323 protocol. The major difference is that SIP was designed in a text format and took request-response protocol style like HTTP. But H.225 and H.245 were defined in a binary format and kept a style of OSI (Open System Interconnection). Therefore SIP has some advantages of interaction with web protocols like HTTP in VoIP industry. More importantly, SIP doesn't define the conference control procedure like H.243 and T.120. Additional conference control mechanisms have to be implemented on the base of SIP to support the AV and data collaboration. Recently SIP research group begun to develop their framework and produced a few drafts. But SIP work is still in the beginning phase and has not been widely accepted.

A SIP system usually includes SIP clients, a SIP Proxy Server, a Registrar Server, a Location Server, and a Redirect Server as well as a SIP Multipoint Control Unit (MCU). The SIP Proxy Server primarily plays the role of routing, enforcing policy of call admission. The proxy interprets and if necessary, rewrites specific parts of a request message before forwarding it. The SIP registrar accepts REGISTER requests and places the received information in those requests into the location service for the domain it handles. In addition, the SIP Proxy provides instant messaging service, forwarding SIP Presence Event messages and SIP text messages to SIP clients. Both H.323 and SIP products are centralized conferencing systems which don't have good scalability because they usually rely upon a single conference server (MCU) for multi-party communication.

2.2 IETF MMUSIC and Access Grid

The IETF's Multi-Party Multimedia (MMUSIC) working group proposed its own solution SCCP (Simple Conference Control Protocol) [12], but gave up and removed conference control from the WG charter in 2000. The project Access Grid started from the MBONE tools: VIC and RAT, and is also trying to define its own conference control framework rather than SCCP.

Access Grid can support a large scale videoconference based on a multicast network. Access Grid provides the group-to-group collaborations among 150 nodes connected to Internet 2 world wide. Access Grid improved MBONE audiovisual tools VIC and RAT. Permanent virtual meeting rooms were also introduced in Access Grid as "virtual venue" for the purpose of collaboration services management. Users are allowed to establish their own venue server which hosts the information about the user registration and

venue addressing and offers rendezvous service to all the users. Users have to log into the venues server and start the multimedia clients in their nodes for communication through multicast and unicast bridges.
The major advantage of multicast used by Access Grid is the conceived scalability in the architecture. However, network administrators and service providers are reluctant to deploy multicast because of complicated management, the NAT/Firewall barrier and potential security issues. All these problems discourage people without access to high-speed networks from using multicast and Access Grid.

2.3 Skype

Kazaa [13] is one of the most popular and widely used p2p system, with over 85 million downloads worldwide and an average of 2 million users online at any given time. In the beginning, Kazaa only provided file sharing service. Kazaa nodes dynamically elect 'super-nodes' that form an unstructured overlay network and use query flooding to locate content. Regular nodes connect to one or more super-nodes to query the network content and use HTTP protocol to directly download the selected content from the provider. In 2003, Kazaa extended its service to VoIP world by launching Skype [14] p2p VoIP solution and gained a big success. Skype was trying to address the problems of legacy VoIP solutions by improving sound quality, achieving firewall and NAT traversal and using p2p overlay rather than expensive, centralized infrastructure. It also provided supplemental features like instant messaging service. The Skype nodes organize themselves into a peer-to-peer overlay, using a super-node architecture. Super-nodes are operated by the Skype Corporation, which also controls user names and authorization. All end-to-end communication, both voice and IM is encrypted for security. Skype uses iLBC [15], iSAC [16] implemented by GlobalIPSound [17]. These codecs are excellent audio codecs which have very good tolerance of packet loss and fancy echo cancellation algorithm.

2.4 Summary of Issues

Although all of these systems have advantages, they are not sufficient for building our proposed advanced service-oriented VoIP conferencing systems:

(1) SIP has very limited supported for conference control.
(2) In H.323 framework, AV collaboration and T.120 are not well integrated. Moreover, the AV communication services and T.120 overlay networks don't have very good scalability.H.323 and T.120 is designed in a relative complicated OSI model. It is not easy to understand and develop in their APIs.
(3) Access Grid heavily depends on multicast service and limited number of unicast bridge servers in the Internet 2. Since it doesn't have overlay substrate to support further deployment in heterogeneous Internet users, most users of Access Grid are institutions connected with Internet 2.
(4) Obviously, Skype peer-to-peer approach is quite promising. But it uses its own propriety protocols and can't interoperate with other legacy VoIP clients such as H.323 and SIP. Moreover it can only support audio conferencing and currently has no video service.

2.5 Our design principles

Internet has evolved from simple data communication network into an indispensable and sophisticated service delivery infrastructure. And many up-to-date software technologies such as XML, SOAP, Web-Service, Publish /Subscribe messaging as well as peer-to-peer computing have emerged and started to change the Internet applications. These new technologies enable the new service-oriented architecture for VoIP conference systems:

(1) A scalable, robust and QoS-aware "overlay" network is needed to support multimedia group communication over heterogeneous networking environments. Such an overlay network should be able to go through firewall and NAT, provide multicast service in whatever unicast and multicast networks, make intelligent QoS routing and offer reliable data delivery in whatever loss network. It also can be configured as P2P or distributed server-based overlay to provide differential services for VIP and regular users. This extends the very successful Skype messaging to a Grid and Web-Service environment.

(2) A common AV signaling protocol has to be designed to support interactions between different AV collaboration endpoints. For example, in order to get the H.323, SIP and MBONE endpoints to work in the same AV session, we have to translate their signaling procedures into our common procedure and build the collaboration session. A core conference control mechanism is required for establishing and managing the multi-point conference. The service of this part is quite like T.124 [18] (Generic Conference Control) in T.120 framework. However this mechanism will provide more flexible facilities to describe application

sessions and entities. And it can be designed in a more scalable approach based on the powerful publish/subscribe messaging services. All description information for the applications and sessions can be kept in XML format rather than binary format, which will lead to interoperability, easier development and ability to tap important useful capabilities like security and metadata frameworks being developed for web services.

(3) A distributed media and session service management mechanism has to be introduced to address the scalability issue in VoIP conferencing. And for heterogeneous clients, sometimes they need customized multimedia service to adapt media streams to their capability. The service overlay network allows the users to locate the suitable service resources and compose them into a service workflow for their purpose. Based on the scalable peer-to-peer service discovery, the service management mechanism can be considered together with the session management functionality.

3. Global-MMCS: A Service-Oriented Multimedia Communication System

Figure 1 shows the service-oriented architecture of Global-MMCS. NaradaBrokering offers the media delivery, storage services and service discovery to various users. Media processing service defines the specific data processes necessary for collaborations such as media adaptation and media mixing. Session management service can control the associated media service instance, maintain the session membership, and enforce floor control.

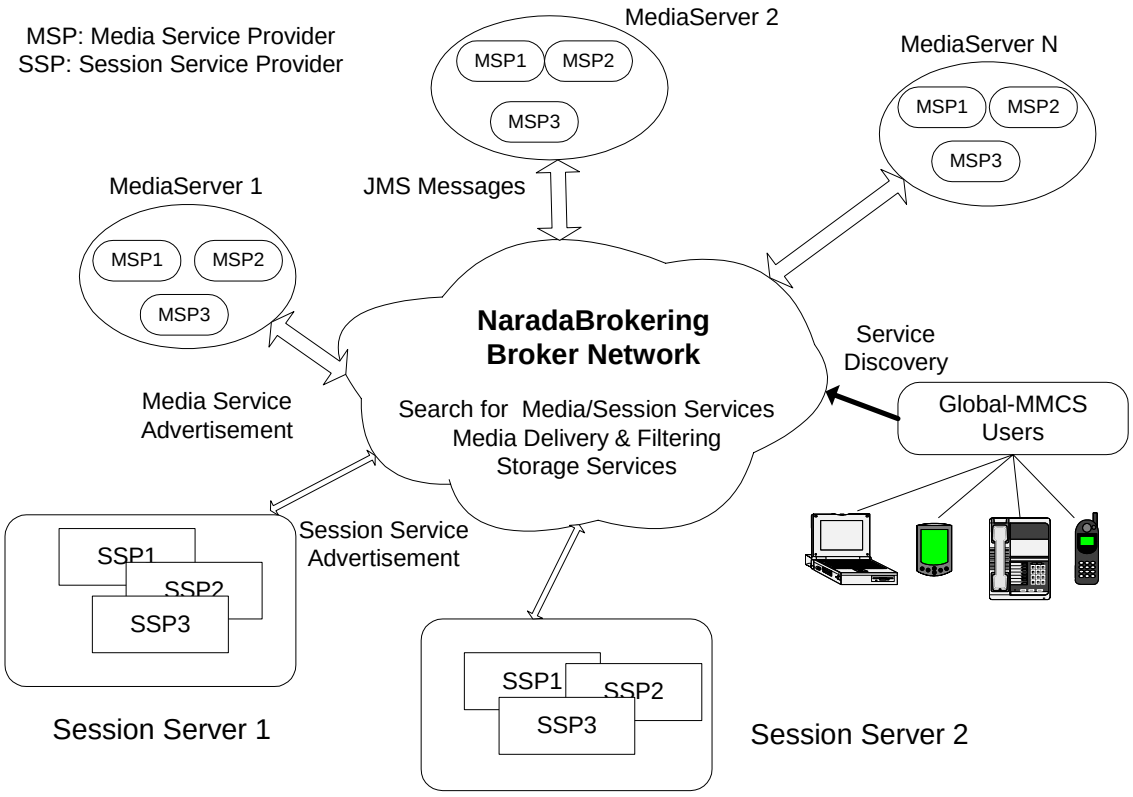

Figure 1. Global-MMCS: A Service-Oriented VoIP System

Publish/subscribe systems provide a messaging middleware that decouples producers and consumers on time, space and synchronization. In a wide-area VoIP conferencing system, the heterogeneity in clients is a big issue for the scalability, especially for video. A filter component in publish/subscribe model can make necessary media processes such as transcoding, traffic reshaping, resizing and color transformation to create the customized streams for receivers. The XGSP framework provides clients a rich XML tool to describe their capabilities and characteristics of network connections. Whenever a client subscribes a media

stream, Global-MMCS checks the format and bitrate of this stream against the customization specification and inserts a proper filter in the media delivery path.

3.1 Media Delivery Service

NaradaBrokering is publish/subscribe a messaging overlay network consisting of a dynamic collection of brokers. The brokers can be self-organized into a hierarchy topology to support fast routing. The performance of connections between brokers is also monitored for the QoS routing decision. The NaradaBrokering transport framework facilitates easy addition of transport protocols for communications between NaradaBrokering nodes. One of the most important elements in the transport framework is the Link primitive, which encapsulates operations between two communications endpoints and abstracts details pertaining to communications and handshakes. Currently TCP, UDP, RTP, Multicast, SSL and HTTP based implementations of the transport framework exist.

(1) RTP Topic and RTPEvent

Since publish-subscribe systems are not designed to serve real-time multimedia traffic, they are usually used to deliver guaranteed messages by employing reliable transport protocols. In addition, they do not focus on delivering high bandwidth traffic or reducing the sizes of the messages they transfer. It is more important for them to provide more services than saving bandwidth. In publish/subscribe messaging systems, messages tend to have many headers, related to the content description, reliable delivery, priority, ordering, distribution traces, etc. Since audio and video streams do not require them, many of these headers are unnecessary. For example, a message in Java Message Service [19] has at least 10 headers. Many of them are redundant in the context of audio and video delivery. These headers take around 200 bytes when they are serialized to transfer over the network. Then, there is the cost associated with serializing and de-serializing the multimedia content. Therefore, we need to design a new event type with minimum headers and minimum computational overhead.

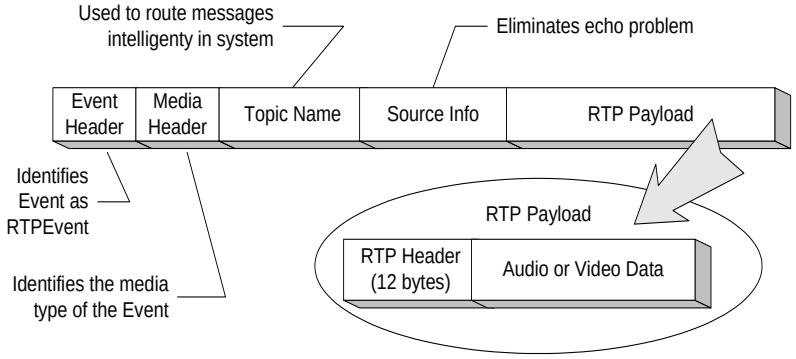

Figure 2. Serialized RTPEvent

RTPEvent encapsulates media content that comprises of 4 elements. There are two headers identifying the event type. Both headers are 1 byte. Event header identifies the event as RTPEvent among other event types in NaradaBrokering system. Media header identifies the type of the RTPEvent such as audio, video, RTCP, etc. To eliminate echo problems arising from the system routing content back to the originator of the content, information pertaining to the source is also included. This information can be represented in an integer, which amounts to 4 bytes. Finally, there is the media content itself as the payload in the event. Although, in Figure 2 an RTP package is seen as the payload, it can be any data type. Therefore, the total length of the headers in an RTPEvent is 14 bytes. 14 bytes is small enough to add to each audio and video package transferred in the system.

Because a lot of other RTP based clients from H.323, SIP and Access Grid need to be supported in Global-MMCS, we have developed a specialized implementation of the NaradaBrokering transport framework called RTPLink for both UDP and Multicast. This process entailed an implementation of the Link interface which abstracts the communication link between two entities. The RTPLink can receive raw RTP packages over UDP or multicast from legacy systems, wrap them in RTPEvents and propagate through the protocol layer in the broker node. Once it reaches the protocol layer, the event is routed within the distributed broker network.

The RTPLink deals with the initialization, registration and data processing on the communication link. During the initialization process, the RTPLink is provided a port number to listen for packages from the legacy client at the other end, and also the IP address and the port number of the legacy client to be able to send packages. For registration purposes, the RTPLink is assigned a NaradaBrokering-ID and the RTPLink subscribes to the topic corresponding to its meeting. In the data processing part, the RTPLink constructs the RTPEvents for processing within the broker network when it receives media packages. On the other hand, when an RTPEvent is ready to be sent to the legacy application, the RTPLink retrieves the RTP payload from the RTPEvent and sends it to the legacy application based on the parameters specified during initializations.

We are independently relating these ideas to work on fast XML and SOAP to properly place this work in a web-service framework [21].

(2)Communication Performance Monitoring and QoS Routing

Every broker in NaradaBrokering incorporates a monitoring service that monitors the state of the links originating from the broker node. An implementation of the Link primitive can measure and report performance factors such as bandwidth, jitter, transit delays, loss rates and system throughputs**.** The Monitoring Service cycles through this list of links at regular intervals to retrieve performance information from each link. The Monitoring Service encapsulates performance data gathered from each link in an XML structure and reports these retrieved information to the Aggregation Node**.** Each Aggregation Node aggregates performance data from monitoring services running at multiple nodes. We query the XML file using XPath to detect network conditions from the aggregated performance data. All the metrics can be checked for thresholds. If a certain threshold is reached then the broker node can be informed to take actions to correct the situation.

Individual brokers maintain weighted broker network maps (BNMs) that are used to compute routes to reach destinations. Aggregated information can be used to dynamically update the costs associated with traversal. Similarly, certain routes can be invalidated as a result of certain network conditions. Event routing is the process of disseminating events to relevant clients. This includes matching the content, computing the destinations and routing the content along to its relevant destinations by determining the next broker node that the event must be relayed to. As an event flows through the system, the associated trace is modified to snapshot the event's dissemination within the broker network. These routing traces indicate – and can be used to verify – an event's dissemination within various parts of the broker network. Routing decisions are made on the basis of this trace information and the computed destinations.

3.2 Media Processing Service

Computation and storage resources connected with NaradaBrokering brokers are service containers that can host both media processing and session management service. XGSP framework specifies the XML scheme for describing the media processing and session management services. Each service provider can advertise its service XML description to the connected broker. Allowing clients to use XPATH or SQL-like queries, NaradaBrokering offers scalable resource search for service discovery through its routing and event matching mechanism. A service requestor can specify the on-demand service in the form of XPATH and search scope. After the query result returns, it can choose the best one based the evaluation criteria and negotiate with the target service provider to create a service instance.

Each broker may have a registered media service container called MediaServer, which hosts various compute intensive media processing services. All service providers implement the interface required by the server container to be able to run inside. MediaServer can create, start and stop media service instances. In addition, it advertises these service providers and reports the status information to the broker regarding the load on that machine. Media service customers which are usually XGSP audiovisual session servers can locate the best container and request a service instance to execute in the container.

A media service is a functional entity which can receive one or multiple media streams, make some processing, and output one or multiple media streams. In most cases, a simple media service should be organized in a media filter chain. Filter can be either as simple as bit-stream parsing, or as complicated as decoding and encoding. Composite media services are usually acyclic computation graph consisting of multiple filter chains.

In VoIP field, typical media services include: stream adaption, transcoding, audio-mixing and video-mixing. Audio mixing is very important to those clients that can't receive multiple RTP audio streams and mix them. Video mixing service improves the visual collaboration especially for those limited clients which can only handle a single video stream. Transcoding helps to generate a new stream which is encoded in the

format wanted by the user. For examples, if a RealPlayer user needs to receive a video encoded in H.261 RTP, a RealStream producer is needed to first decode the H.261 video and generate a new RealFormat stream.

The media service container can also support the extension of third-party media processing services. A service description interface is defined for the third-party developers. Moreover, even if only codec implementation is presented, it is also possible for the intelligent engine in the container to adaptively organize the transcoding filter chain and automatically generate the media service template.

3.3 Session Management Service

Here the session is not only about the group of people using whatever clients to collaborate but also the associated media services. XGSP audiovisual sessions have five states: created, canceled, activated, deactivated and finished. The states are managed by the XGSP AV session server. A XGSP user can initiate an audio-visual session at first. This created session can be activated by the session server after the meeting time arrives and the needed service instances are created. Figure 3 shows the interaction between users and services components for the session management procedure.

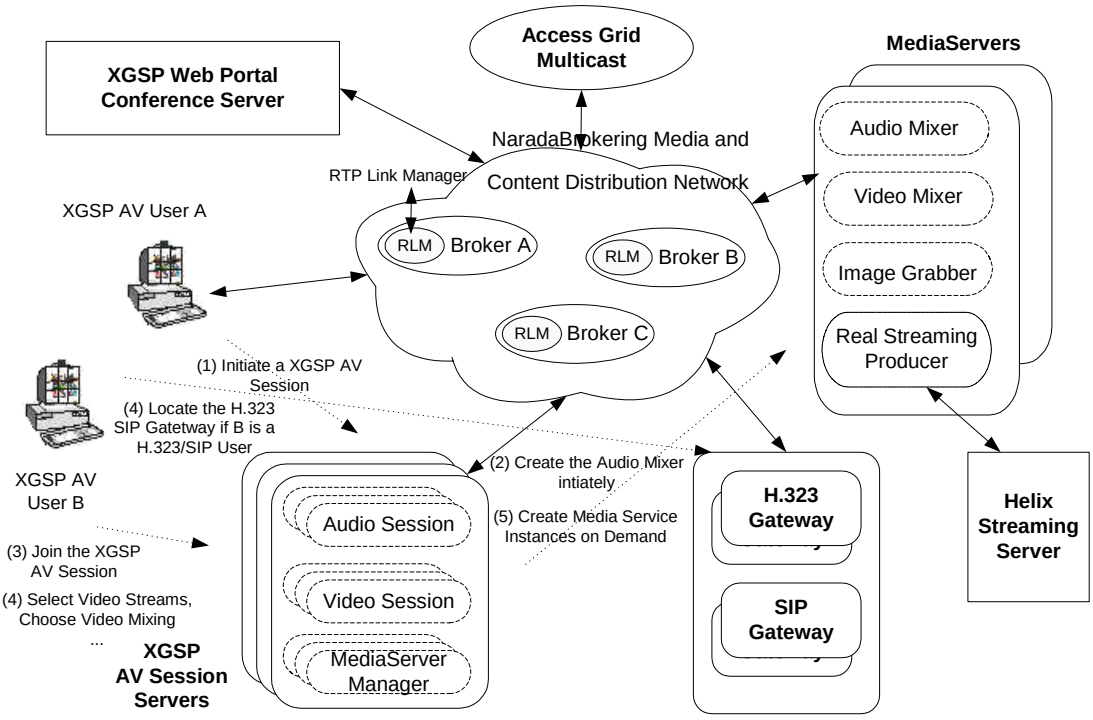

Figure 3 XGSP Session Management in a service-oriented way

Each session server may host limited numbers of active XGSP AV Session. The exact number depends upon the workload and the computation power of the machine. The session initiator will firstly locate the right session provider to create a session service instance – session server, for a particular XGSP AV session. Then, this session server will locate the necessary media service resource on demand. In the current implementation, a default audio mixer is created to handle all the audio in the session. Private audio mixers can be created for private sessions for subgroups in the session. And video mixers can be created by the session server by the request of the client. An image grabber is created when a new video stream is detected in the session. Customized transcoding services can be created when a user sends a matching request. For example, a mobile client like PDA connected to Wi-Fi, which only has limited processing power wants to choose a 24 4-CIF MPEG-4 video, a transcoding process pipeline consisting of frame rate adapter , video size downsampler and color transformation, is needed to adapt the stream. Another example

is if an H.323 terminal which can only handle H.261 and H.263 codecs wants to display a MPEG-4 video, it will ask the session server to start a MPEG-4-to-H.261 transcoder.

The AV Session Server also involves the management of XGSP sessions for other legacy clients like H.323, SIP and RealPlayer. Global-MMCS has H.323, SIP, and RealStreaming Gateways for adapting H.323, SIP terminals and RealPlayers. The XGSP AV Session Server needs to collaborate with these Gateway Servers to deal with the session control layer problems in this heterogeneous collaboration system. The H.323 and SIP gateway transform H323 and SIP messages into XGSP signaling messages. The RealStreaming Gateway gets the encoding jobs from the Session Server and generates the RealMedia streams from the selected conferencing AV streams. These session boarder services are organized in the slightly different way from media service. For H.323/SIP Users, the H.323/SIP Console firstly launches a search for the local H.323/SIP Gateways. So H.323/SIP session service is different from media service model, because of NAT/Firewall issues, each service should work in the local way. And let's users themselves to locate the right Gateway and register their terminals/UA to it. For celluar phone Users, they have to contact the wireless provider at first. Global-MMCS runs the mobileGateway at the edge of wireless network. For real-streaming Users, the XGSP session server needs to create a RealStreaming Producer instance which is able to encode up to 4 real-streaming encoding jobs according to the demand.

Since NaradaBrokering can publish the performance monitor data *in* the form of XML document, the AV Session Server should subscribe to this network performance topic. From these performance data and broker network maps, the Session Server can estimate the delay and bandwidth between the service candidates and the requesting user. Based on the workload of the media service providers and estimated the performance metrics, the Session Server can find the best service providers and initiate a media service instance. Furthermore, the AV Session Server has to monitor the running of each media service instance. Through a specific NaradaBrokering topic, an active media service instance can publish some status meta-data to notify the session server. If it fails to do within a period of time, the AV Session Server is responsible for restart it or locate a new service provider and start a new instance.

4. Implementation and Performance

We have released the initial version of Global-MMCS. It has several software components including: Media Server package, Session Server package, Web Server package, H.323 Server package, Real Streaming package. The whole system uses NaradaBrokering package for communication overlay and this software is also freely available in open source. All the audiovisual processing services, including the video mixer, audio mixer, the image grabber servers as well as the front end of RealStreaming Gateway are developed using Java Media Framework where we make available major performance improvements for this standard Java package. Extensive performance measurement have been made to evaluate the scalability of the system.

4.1 Performance of NaradaBrokering for Audio/Video Delivery

The performance of the event brokering network is critical to the success of our VoIP conference system. We tested single audio meetings with varying number of participants on a broker. Table 1 shows the summary of the test results.

Total Users	First user Latency (ms)	Last user Latency (ms)	Avg Latency (ms)	Avg Jitter (ms)	Avg Late Arrivals	Out BW (Mbps)
12	0.5	0.7	0.6	0.18	0	0.76
100	0.5	2.3	1.4	0.15	0	6.4
400	0.5	7.9	4.2	0.21	0	25.6
800	0.5	15.5	8	0.18	0	51.2
1200	0.5	22.6	11.6	0.22	0	76.8
1400	0.5	26.5	13.5	0.26	0	89.6
1500	3.3	32.3	17.8	0.44	25%	96.0
1600	2260	2290	2275	1.2	100%	102.4

Table 1 Test results of single audio meetings for one broker

All audio packets are the same size and they are evenly distributed in time (a packet every 30ms), and the routing of all packets in a stream takes the same amount of time on the broker. As long as the routing of a packet takes less than 30ms, the routing of one packet in the stream doesn't affect the routing of the next one. Therefore, the latency values of consecutive packets are almost the same for a user in a meeting. This results in very small jitter values. Moreover, the latency values for the first client are always constant and very small until the broker is overloaded. In addition, the routing time of an audio packet increases linearly by the number of participants in the meeting. The broker is overloaded when the routing of a packet takes more than 30ms. In that case, the next packet arrives before the routing of the current packet is completed. Each packet delays the routing of the next one and the latency increases constantly for the upcoming packets in the stream. In this test shown in table 1, the broker becomes overloaded when there are 1600 participants in the meeting.

We tested the performance of a single broker with single video meeting and multiple smaller meetings. The most important outcome of these tests was the fact that the broker was utilized much better for multiple smaller meetings than single large size meetings. It supported a higher number of participants with smaller latency and jitter values. Multiple meetings better balance the work of a broker within the 30ms inter-packet gap. Figure 4 shows the average latency values of single video meeting tests and multiple video meeting tests for the same number of participants. In multiple video meeting cases, all video meetings had 20 participants. As the latency values show the average latencies of multiple video meetings are much smaller. Similarly, the jitter values and loss rates are also much smaller. Therefore, the broker was able to provide services to 700 participants in 35 video meetings with very little late arriving packages. It was able to support only 400 participants in the single video meeting test.

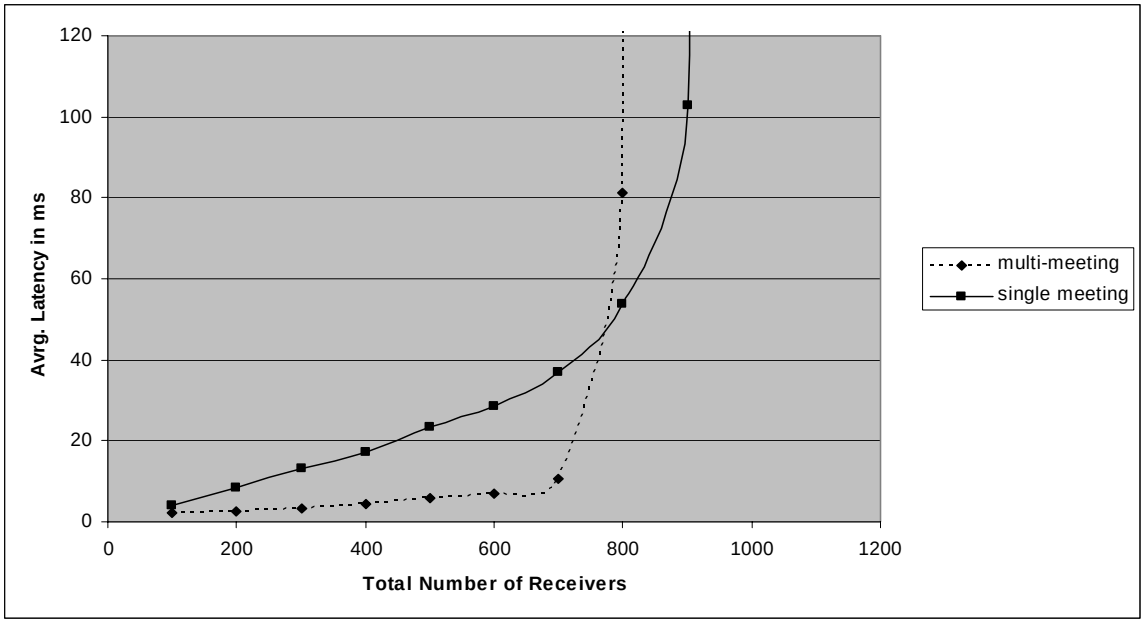

Figure 4 Average latencies of single and multiple video meetings

We also made experiments to investigate the performance of the delivery of audiovisual streams to geographically distant locations. Four sites include Indiana University in Bloomington Indiana, Florida State University in Tallahassee Florida, Syracuse University in Syracuse, New York and Cardiff University in Cardiff, UK. The test demonstrate that by running a broker in a remote site, significant bandwidth saving can be achieved and the bandwidth limitations can be overcome to support more participants. Moreover, running brokers in geographically distant locations can reduce the transit delays of packets considerably. Our analysis of the performance of the broker network shows that the NaradaBrokering has a very good scalability in supporting both voice and video communication. In our test setting, four brokers supported up to 6400 participants in an audio-conference and this number can be increased linearly by adding new

brokers. The quality of stream delivery for large size meetings can be improved significantly by distributing clients among multiple brokers.

4.2 Media Service Performance

We investigate the performance of media services in controlled experiments. All the service providers running in a single server are configured to create more service instance so that the overhead increase in the terms of CPU and memory usage in the server could be measured. Although there is some difference in the computation overhead generated by services, they are all computation intensive tasks. Since individual service instances can run independently and be attached to different XGSP AV sessions, it is quite easy to distribute them into many different distributed hosting servers with streams software multicast by the NaradaBrokering messaging infrastructure. Provided we instantiate enough media service resources, the scalability of the Global-MMCS can be guaranteed. We imagine that there is either a "farm of CPU's" in a Grid server based model or a P2P implementation, these media services are provided on the peer clients themselves. One can trade-ff and mix these models to get needed capability, performance and fault-tolerance.

Table 2 shows the typical overhead caused by media services in commercial PCs. In the audio mixing test, if all the audio streams are ULAW, a machine can support around 20 mixing sessions. When we made tests with another more computing intensive codec, G.723, one machine can run only 5 mixing sessions. Therefore, on the average one machine may support around 10 mixing sessions. If every mixed video stream is encoded in H.261 and has an average bandwidth of 150kbps, a machine can host up to 4 video mixers. For the image grabber which transcodes a H.261 stream into a series of JPEG pictures, an ordinary machine can run around 50 image grabbers. And the Real Streaming producer has the similar overhead to the video mixing service.

Media Services	Computation Overhead
Audio Mixing	46% while 20 audio mixers (six active speakers) are running
Video Mixing	94% while 4 video mixers (4-way mixing) are running
Image Grabber	70% while 50 image grabbers are running
RealStreaming Producer	90% while 4 23fps stream producers are running

Table 2. Typical computation overhead caused by media services

4.3 JMF Implementation

Java Media Framework (JMF) [21] provides a unified architecture and messaging protocol for managing the acquisition, processing, and delivery of time-based media data. By exploiting the advantages of the Java platform, JMF delivers the promise of "Write Once, Run Anywhere" to developers who want to use media such as audio and video in their Java programs. Although JMF has some good quality for building multimedia service, it is well known to have the poor performance compared to Tcl/C++ based VIC intensively used by AccessGrid community.

During the development, we introduced some new data copying mechanism to improve the rendering performance of JMF library. Table 3 shows CPU overhead of the rendering one to eight video stream by VIC, the Global-MMCS client based on Sun JMF library and the client based on our optimized JMF library. Each video stream is a CIF-size 30-second video sequence encoded in H.261, and has average bandwidth 500kbps. It is obvious that the performance of JMF rendering has been improved by 200% and even outperforms VIC.

Stream Number	**VIC**	**Sun JMF Client**	**Fast JMF Client**
1	8% - 9%	15% - 16 %	6% - 7%
2	13% - 14%	24% - 25 %	9% - 10%
3	17% - 18%	33% - 34 %	15% - 16%
4	23% - 24%	40% - 41%	17% - 18%
5	26% - 27%	46% - 47%	23% -24%
6	32% - 33%	51% - 52%	27% - 28%
7	35% - 36%	58% - 59%	31% - 32%
8	40% - 41%	62% - 63%	34% - 35%

Table 3 rendering performance of VIC, Sun JMF and enhanced JMF

4.4 Scalability Discussion

On the basis of measurement in controlled experiment environment which consists of at most 4 brokers, we can discuss the scalability of the system in a more general way and larger scale. Consider scenarios where users are connected in collaborative sessions such as supported in Access Grid rooms or text chat sessions. All communication is performed by NaradaBrokering installed as a mixture of standalone brokers, handlers/plug-ins for clients or Web services.

The software routing is performed by the handlers and the brokers. The routers fall into two groups: Geographical routers handle traffic in a particular region and Functional routers cope with particular capabilities – in this case the different rooms. Hardware multicast is one approach to geographical multicast. However note that we ignore the possibility of hardware multicast below and assume software multicast where this is needed. Note that we have very efficient communication between institutions and hardware multicast would only be useful internal to each organization where it could add quite a bit of value and be easier to implement than globally across institution. NaradaBrokering supports hybrid protocols that mix hardware unicast and multicast. The network architecture is designed to make communication traffic as reliable and low bandwidth as possible.

Note we replicate the routers for each room at each institution so as to minimize traffic between institutions by using software multicast only within each institution. We give a more detailed analysis below ignoring any Geographical routers. These could be implemented in P2P fashion using NaradaBrokering handlers on the clients.

Assume there are ***N*** Institutions, ***R*** XGSP AV sessions and ***M*** participants per "session". Let each person send out one native AV stream and receive ***4+X*** streams. Let each client mix the (first) ***4*** streams to make one composite mixed stream. The other ***X*** streams can either be other mixed streams or native webcam streams. Clients are also capable of generating these thumbnails of the received streams and publish over the NaradaBrokering. Note we are mixing and image grabbing in P2P mode as more CPU power on clients. One could mix on servers but currently low end Linux server can produce up to 4 mixed streams where each mixed stream made up of 4 native streams. Thus one needs ***M*R/4*** servers for mixing to make the MR composite streams imagined here. The clients can display any of received streams and/or the mixed stream that it makes itself. And it can send out 2 streams including the native stream from its webcam and mixed one it makes.

Assume each person can only in one room. We configure S servers at each institution each handling ***R/S*** rooms. So the traffic at each server is receiving ***2*M*** streams and dispatching ***(N-1)* 2*M/N*** streams to remote sites. Here we assume uniform distribution ***M/N*** people per session per site, so each session dispatches ***(4+X)*M/N*** streams to local clients. Thus total traffic per session for the server is ***4*M +(2+X)*M/N*** streams.

We can illustrate the result in a particular example: let R= 50, S= 25, N=2, M=20**,** X=4. Assume the bandwidth of each stream is 1/3 Mbps. Therefore each Server has 93 Mbps traffic summing input and output bandwidth which has basically reach the peak rate in a server with 100 Mbps network interface. Each client needs to handle up to 3.3 Mbps traffic. Each Institution has 50 servers and 500 clients, so we get totally 50 servers and 1000 clients.

Based on the measurement result in Table 1, one regular server can usually run up to 20 audio mixers. We need other 3 or 4 audio servers for 50 sessions. Many rooms can share a server for their session control Web Service and we can estimate that a safe number is 500 users per session server. Two session servers and two H.323/SIP gateways should be enough for 1000 users. If for each session we need to generate a RealMedia stream simultaneously, at least 12 servers are necessary for 50 sessions. To support thousands of streams and dozens of sessions, media services demands the computation resources in dozens.

5. Conclusion and Future Work

In this paper we present the design principle and experience of building a scalable service-oriented VoIP conferencing system. This system uses NaradaBrokering messaging middleware as QoS-aware service hosting and delivery infrastructure. Further the XGSP collaboration framework is defined for the media service and session management. Such a service-oriented and integrated collaboration environment greatly improves the scalability of traditional videoconferencing system, benefits customers using diverse multimedia terminals through different network connections and simplifies the further extension and interoperability.

We have built a prototype system to verify our ideas and integrate a few common media service for VoIP conferences. Based on the flexible framework, we are working on new services like audio-video annotation and new media codecs like MPEG-4 and H.264. Further extensions could include: extending the scalability of the system to over 10,000 users, adding archiving and replay services, and customizing the system for different application scenarios like collaborations in e-sports and e-science.

6. References

[1] ITU. Recommendation H.323 (1999), Packet-base multimedia communications systems.

[2] J. Rosenberg et al. (2002) "SIP: Session Initiation Protocol", RFC 3261, Internet Engineering Task Force, http://www.ietf.org/rfc/rfc3261.txt.

[3] Access Grid (2003), http://www.accessgrid.org

[4] Wu W, Fox G. C, Bulut H, Uyar A, Altay H. Design and Implementation of A Collaboration Web-services system. Journal of Neural, Parallel & Scientific Computations, Volume 12, 2004.

[5] Global Multimedia Collaboration System (Global-MMCS), http://www.globalmmcs.org

[6] Fox G. C, Pallickara S. The Narada Event Brokering System: Overview and Extensions. proceedings of the 2002 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'02)

[7] Handley, M., Crowcroft, J., Bormann, C. and J. Ott. The Internet Multimedia Conferencing Architecture, Internet Draft, draft -ietf-mmusic -confarch-03.txt.

[8] ITU. Recommendation H.225(2000), Calling Signaling Protocols and Media Stream Packetization for Packet-based Multimedia Communication Systems.

[9] ITU. Recommendation H.245(2000), Control Protocols for Multimedia Communication.

[10] ITU. Recommendation H.243(2000), Terminal for low bit-rate multimedia communication.

[11] ITU. Recommendation T.120(1995),Multipoint Data Conferencing and Real Time Communication Protocols, 1995

[12] Bormann, C., Kutscher, D., Ott, J., and Trossen, D.. Simple conference control protocol service specification. Internet Draft, Internet Engineering Task Force, Work in progress.

[13] Kazaa, http://www.kazaa.com

[14] Skype, http://www.skype.com/

[15] ILBC codec, http://www.globalipsound.com/pdf/gips_iLBC.pdf

[16] iSAC codec, http://www.globalipsound.com/pdf/gips_iSAC.pdf

[17] Global IP Sound. http://www.globalipsound.com/partners/

[18] ITU. Recommendation T.124 (1995) Generic conference control, 1995.

[19] Java Message Service (JMS), http://java.sun.com/products/jms/

[20] Sangyoon Oh, Hasan Bulut, Ahmet Uyar, Wenjun Wu, Geoffrey Fox Optimized Communication using the SOAP Infoset For Mobile Multimedia Collaboration Applications Proceedings of the International Symposium on Collaborative Technologies and Systems CTS05 May 2005, St. Louis Missouri, USA.

[21] Sun Microsystems, Java Media Framework 2.1, (2001), http://java.sun.com/products/javamedia/jmf/2.1.1/index.html.