A WEB-SERVICES BASED CONFERENCE CONTROL FRAMEWORK FOR HETEROGENOUS A/V COLLABORATION

Wenjun Wu, Hasan Bulut, Ahmet Uyar, Geoffrey C. Fox
Community Grid Computing Laboratory, Indiana University
wewu@indiana.edu, hbulut@indiana.edu, auyar@mailbox.syr.edu, gcf@indiana.edu
Indiana Univ Research Park, 501 North Morton Street, Suite 222, Bloomington, IN47404, USA

Abstract

Conference control has been studied for years but most researches focus on homogenous A/V collaboration. There is no conference control framework for integration of multiple A/V systems such as H.323, SIP and AccessGrid. In this paper, we propose a web-serviced based scalable conference control framework for such heterogeneous collaboration system. We also implemented a prototype to verify and refine our framework.

Keywords

Conference Control, Web Services, XGSP

1. Introduction

Collaboration and videoconferencing systems have become a very important application in 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. However, at present they have features that sometimes can be compared but often they make implicit architecture and implementation assumptions that hamper interoperability and functionality. Therefore it is very important to create a more general framework to cover the wide range of collaboration solutions and enable users from different communities to collaborate. In this paper, we attempt to define such a common, interoperable framework based on Web services [4] technology for creating and controlling multipoint video & audio collaborations.

The paper is organized in the following way: Section 2 introduces related work and our research issues. Section 3 describes the XGSP conference control framework. Section 4 presents the implementation of XGSP prototype system. And we give the conclusion and future work in section 5.

2. Related Work and Problem Statement

Problems related to conference control have been studied extensively over the years [5, 6, 7, 8, 9, 10]. However, most of the works discuss only homogenous videoconferecing, including H.323, SIP and MMUSIC [11]. ITU-T developed conference control protocols as a part of the H.323 series of recommendations. It is reported [12] that T.124 [13, 14] has the scalability issue because of the inefficient database replication algorithm. And H.323 Audio/Video collaboration takes the simple protocol in H.243 [15] rather than T.124. The IETF's Multi-Party Multimedia (MMUSIC) working group has also proposed its own solution SCCP [5]. But in the year 2000, MMUSIC WG gave up and removed conference control from the WG charter. Recently SIP research group begun to develop their framework and produced a few drafts [9, 10]. But SIP work is still in the beginning phase and has not been widely accepted.

Our job is to define web serivces framework in which H.323, SIP as well as MMUSIC could be integrated. We divide an A/V collaboration system into three parts: A/V application endpoints, session servers and multipoint communication channels provider. Each collaboration system has a different implementation for the application endpoints, server components and different communication protocols between them. For example, in an H.323 based system, an A/V application endpoint refers to a H.323 terminal that is capable of sending audio and video. A session server refers to the Multipoint Controller that can create multipoint session. A multipoint communication channel provider is the Multipoint Processor that can mix audio and video from endpoints. In an Access Grid system, a client is based on the MBONE audio/video tools such as RAT and VIC. Further there is a venues server in Access Grid, which is responsible for scheduling meetings. Multicast RTP channels are the communication infrastructure for Access Grid.

To integrate all these heterogeneous systems into one collaboration system, we need to reach the following goals:

(1) Different kinds of application endpoints can join / leave in the same A/V collaboration session.

(2) Different multipoint A/V providers can be connected together to build unified A/V multipoint channels.

(3) A common user interface is present for all the users over different A/V application endpoints.

The first goal requires a common signaling protocol, which specifies the operation procedure between different types of A/V endpoints and session servers. The conference control framework and multipoint messaging middleware are required for the second and third goal to integrate various RTP multipoint communication servers. Web-service seems to be the best candidate for this conference control framework since it can run across various platforms and is easy to be extended and understood. Conference control consists of three parts: user session management, application session management as well as resource contention management, also known as floor control. Since there are different kinds of conference control protocols for different collaboration technologies, we have to wrap them into web-services and integrate these services in a more general framework.

3. XGSP Framework for Conference Control

Figure 1 shows the architecture of XGSP framework. The A/V media channel service provides multipoint A/V RTP channels for various A/V application endpoints. This service can be implemented on top of distributed messaging middleware, Narada [16] to create a unified, robust and scalable multipoint communication platform over hetergenous networking environment. Various collaboration systems including AccessGrid, H.323 and SIP are regarded as Web-services components in XGSP framework. They provide Web-services interface of their conference control protocols to the XGSP collaboration manager servers so that their RTP channels can be connected with the XGSP A/V media channel service under the control of the manager servers.

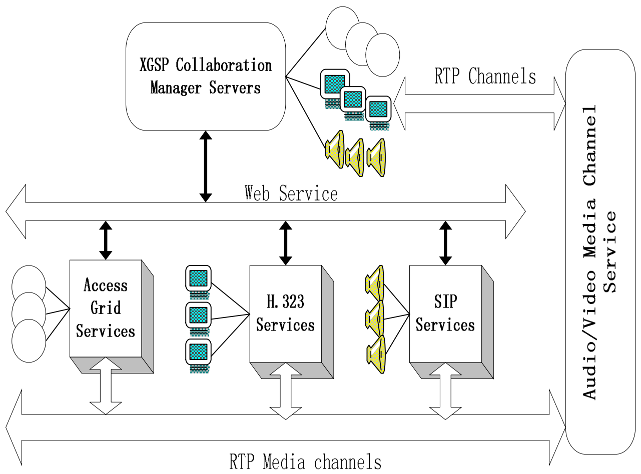

Figure 1 XGSP Conference Control Framework

Under such a framework, all kinds of A/V application endpoints can communicate with each other whether they are directly connected to XGSP A/V media channel service or to different collaboration systems. Different collaboration systems are regarded as XGSP communities having multiple collaboration rooms. A collaboration room is the abstraction of multipoint A/V RTP channels. In centralized conferencing, A/V endpoints have to enter the collaboration room to attend the videoconferencing. It is noted that the concept of “room” is widely used in current collaboration systems. Based on these rooms from different communities, XGSP can create a collaboration session for all the endpoints. The XGSP session occurring in a community room is referred to XGSP sub-session. Users have the opportunity to enter either the XGSP session or the XGSP sub-sessions in local communities.

To support such a collaboration model, we have to define a XGSP conference control framework, which should be generic, easy to extend, reliable and scalable. 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 service to A/V application endpoints and communities, controlling multipoint A/V RTP channels. Floor control manages the access to shared collaboration resources. Although there are various floor control policies for different collaboration applications, XGSP should offer basic floor control mechanisms to support all these policies.

XGSP is a two-level control framework which includes top conference control servers and servers from other communities. Therefore the three components can be designed in a hierachy and distributed model to improve the scalability, which means that the top XGSP servers manages the whole XGSP session and the local servers only control XGSP subsessions. SOAP [17] RPC commands can be used for the communication between the top XGSP servers and the local servers. It is noted that the servers from different communities may have different capabilities of handling sessions. For example, in AccessGrid communities, there is almost no much control for session membership and floors. So the services of the community should be described in WSDL which may help to generate the interface between the top XGSP servers and the local servers.

3.1 XGSP User Session Management

In a XGSP session, users have different roles and access rights to the collaboration resources. Each user should have an ACL (access control list) to describe his rights in the session. We can define multiple ACL templates according to the role of the user. Since the definition of the role and ACL groups usually depends upon the style and policy of the collaboration, XGSP provides some basic operations and simple definition of user roles to support different collaboration policies. XGSP provides the API for defining new user role for the specified collaboration scenario. When a user signs in XGSP system, the XGSP user session servers will create the ACL list for the user according the role template and its user profile.

XGSP users can be divided into three categories: administrator, chairman, normal users and anonymous users. In addition, normal users can be divided into: top normal user, local chairman, and local normal user. Anonymous users who are just audiences in the XGSP session, usually don't show up in the member list of a user session and have a very limited right of accessing the conference. For example an audience can't speak or send video to the meeting. The administrator user is a very special user which can be regarded as a super user in the conference. A chairman user usually has the power of controlling floor tokens.

The session membership containing a list of the participants is shared by all the participants and the user session server in this collaboration session. Whenever there is some change in the membership, for example a new member joins in the session, the membership has to be updated and distributed to all the participants. So we need a scalable mechanism to maintain the consistency of membership information shared among the participants. Since XGSP framework has a two-tier tree structure, the session servers in XGSP sub-sessions can play as an intermediate node to implement the distributed membership maintenance. XGSP top session server collects the membership report from the XGSP sub-session servers and the top normal users. And the sub-session servers collect the local membership reports from the local users. And the XGSP top session server announces the change of the membership to all the users and sub-session servers.

In such a two-tier structure, a user has the options to enter the system through either the top session server or the sub-session servers. Top users should sign in through the top session server, while local users should do it through the sub-session servers. This distributed algorithm requires that a XGSP sub-session server should provide the services of reporting local membership and showing the global membership to its local users. It is noted that if some community server has no capability of managing the membership, we may also need to deploy a XGSP sub-session server for this community.

3.2 XGSP A/V Application Session Management

XGSP application session management has two tasks: control the XGSP session over all the media servers and help application endpoints to join and leave the session. The A/V endpoints of top XGSP users should directly attach to the XGSP A/V Channel Service (called top A/V application session). And the endpoints of the local XGSP users should connect with the local media servers (called local A/V application session). XGSP defines the methods of create/activate/terminate to manage XGSP A/V application sessions. XGSP system will create a XGSP A/V session when a user schedules a XGSP session and defines the profile of the A/V session. The profile specifies the audio/video codec and the list of "rooms" from the communities involved in the session.

When the XGSP session is activated, the XGSP session server will link all the "rooms" in the session together by connecting multipoint A/V channels from different communities to the XGSP A/V Media Channel Service. For H.323 and SIP communities, they connect with the XGSP A/V channel Service by dialing in the H.323 and SIP gateway. Since a MBONE community like AG, has no signaling procedure, the XGSP servers will launch an AG agent that joins in the multicast A/V groups and forwards the packets between the top XGSP session and the AG multicast groups.

There are two steps in the join procedure of A/V endpoints, including negotiation of capabilities and establishment of UDP channels. When an application endpoint joins the XGSP application session, it has to make codec negotiation with the media server to ensure that it can support the audio/video codec used in the session. Since different A/V application endpoints have their own signaling procedures for joining and leaving session, we have to define a XGSP signaling protocol for H.225[18], H.245[19] (H.323 signaling protocols) and SIP as well as AccessGrid. The H.323 and SIP gateway transform these protocols into XGSP signaling protocol so that H.323 and SIP endpoints could communicate with the XGSP application session server. AccessGrid tools like VIC and RAT can join in the multicast XGSP subsession without using XGSP signaling procedure. But if they are running in a unicast environment, they have to reply upon XGSP signaling protocols to connect with the XGSP application session server. For those local users, their endpoints can directly connect with the local session servers.

3.3 XGSP Floor Control

Conference applications often have shared resources such as the right to talk, access to a limited-bandwidth video channel, a pointer or input focus in a shared application, access to shared lesson or game rooms. Floor control enables applications or users to gain safe and mutually exclusive or non-exclusive access to the shared object or resource. Floor control should support different floor control policies such as moderator-controlled or first-come-first-served. All these floor control policies can be implemented on the floor control primitives, including: request floor, release floor, grant floor, cancel floor, remove floor request. These primitives are exchanged between the conference participants, the conference server and the chairman moderator.

XGSP framework mainly focuses on dealing with audio and video floor control. Note that in the XGSP session, we have the XGSP top session and XGSP sub-sessions who may have some different floor control policies. For example, AccessGrid multicast session only supports free seminar policy which requires no floor control and some simple H.323 MCU only shows the video of current speaker. So XGSP offers a more general solution to this issue, which doesn't control the video and audio stream from senders, but the streams to the receivers in all the endpoints.

In the following, we mainly discuss on how to implement chair guided floor control policy. In the XGSP top session and XGSP sub-sessions, there may be chairmen for floor control. A local chairman controls the floor in the local XGSP session and requests floors from the chairman in the top session. A local chairman can be either a real human user or a running agent which provides the web-services of floor control to the top session chairman.

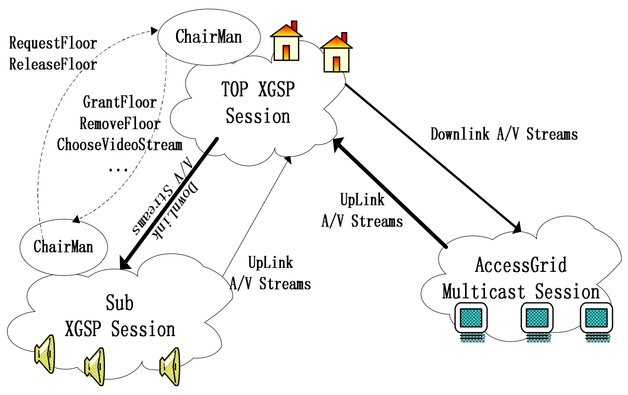

Figure 2 XGSP floor control

(1) Video control Policy

We define "**TVSSS**" as the streams that can be received by the endpoints in the XGSP top video session. TVSSS includes uplink streams from the XGSP sub-sessions and streams from the endpoints in the XGSP top video session. Not all the streams from the video sub-sessions can be received by XGSP video servers because the number of uplink streams is limited by the capacity of local video servers and the RTP channels with the XGSP video servers. In order to choose the uplink video streams in TVSSS, the chairman or administrator in the XGSP top video session can send XGSP floor control commands to the server of the local community. The local chairman can also send requests to the top session chairman to add some streams into TVSSS. If the local servers only support voice-activated video switch, the top session servers have no way to choose the uplink video streams.

XGSP doesn't support the function of disabling the video sender because a lot of video endpoints don't provide such a service. So we rely on the XGSP video servers to block the transmission of some video streams in the XGSP video session. The XGSP servers will only allow some streams from TVSSS for all the downlink video streams. It is noted that the XGSP top session servers can't control the transmission of local video streams in the local community. For example, in AccessGrid multicast session, users are free to watch any streams in this local session.

XGSP allows video endpoints to choose the video streams from TVSSS, which is very useful for those unicast-only endpoints in the XGSP top session. Because they can not receive and render multiple video streams. Local endpoints can only make choice from downlink streams and local video streams. But the local video servers may not support their choices on local video streams. In the case, the users on these endpoints can still choose the downlink video streams. It is the local chairman that makes the final decision on which video should be included in the downlink. The video selection service is not useful to the endpoints in the Access Grid session since they can receive and render multiple video streams.

In some video control policies, the chairman can force other users to watch a specified video. When such a policy is applied, the choices of other users will be disabled. For all the downlink video streams, the top chairman will specify a video from TVSSS.

(2) Audio control Policy

XGSP audio servers mix the audios and forward the mixed stream to audio endpoints. We don't have the same problem as video since there is only one mixed audio stream for all the downlink audio in the XGSP sessions. Just like video, XGSP audio floor control has to co-ordinate with the floor control mechanism working in local communities. A local user can get the audio floor only after he gets floor grant from the local chairman and the XGSP top session chairman. A local chairman collects audio floor requests from local users and forwards audio floor grants from XGSP top session chairman. Since there is neither audio floor control nor audio mixing in Access Grid sessions, the XGSP audio server can enforce the audio floor control by filtering out the extra AG audio streams from the mixed stream.

In summary, the floor control in XGSP session works in two levels. In the XGSP top session, the policy whether it is the style of free seminar or guided meeting, can be applied to all the endpoints. But in the XGSP sub-sessions, only the local policy can be applied. XGSP servers can control the uplink and downlink video and audio streams to partly apply the policy to local communities.

4. Implementation of the Prototype based on the XGSP framework

We have developed a prototype system called Global-MMCS (Global Multimeda Collaboration System) to verify and refine our XGSP conference control framework. In this prototype, three different kinds of endpoints and communities are integrated. We have OpenH323 MCU [20], HearMe [21], and AccessGrid as communities. The figure shows the prototype of XGSP framework.

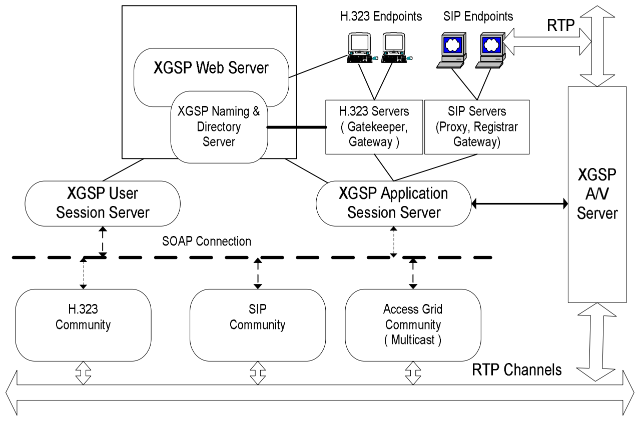
H.323 Endpoints
SIP Endpoints
RTP
XGSP Web Server
XGSP Naming & Directory Server
H.323 Servers (Gatekeeper, Gateway)
SIP Servers (Proxy, Registrar Gateway)
XGSP User Session Server
XGSP Application Session Server
XGSP A/V Server
SOAP Connection
H.323 Community
SIP Community
Access Grid Community (Multicast)
RTP Channels

Figure 3 XGSP Prototype Systems

For H.323 clients, we have the H.323 Gateway and H.323 gatekeeper, which interact with H.323 terminals and retrieve the information for XGSP protocol. For SIP clients, we implemented the SIP Gateway, SIP Proxy and SIP registrar, which supports SIP-to-XGSP transformation and SIP registration. The XGSP A/V server provides the services of bridging multicast and unicast, video-switching, video-Mixing and audio-Mixing to H.323, SIP as well as AG endpoints. The XGSP application session server builds XGSP connections for various A/V application endpoints, activates XGSP application sessions in the XGSP media server, and controls the A/V channels between the XGSP A/V server and other technology communities.

OpenH323 MCU and AccessGrid only provide simple application session services to their users. So their user sessions are managed by XGSP user session server. HearMe community has more sophisticated user and application session management. So our works focus on implementing HearMe web service by using its Voice conferencing SDK API. The HearMe ASCP interface can be used to support the API of user and application session management. Since HearMe can support Moderated conference, we run a HearMe client as a local chairman to implement the floor control service. All these HearMe services are described in WSDL format and integrated into a HearMe wrapper, through which XGSP session servers and invoked.

The XGSP web server implements XGSP user session management and floor control. The function of user session server and floor control is implemented in the form of servlet in the web server and dedicated session server threads run for communicating with the XGSP application session server and the local session servers of HearMe and OpenH323 MCU. The XGSP web server also provides the user portal and system administrator portal. Users need the web portal to participant in the audio video collaboration. The system administrator can use the portal to manage the system, for example configure the components of the system, and upload some new services and so on.

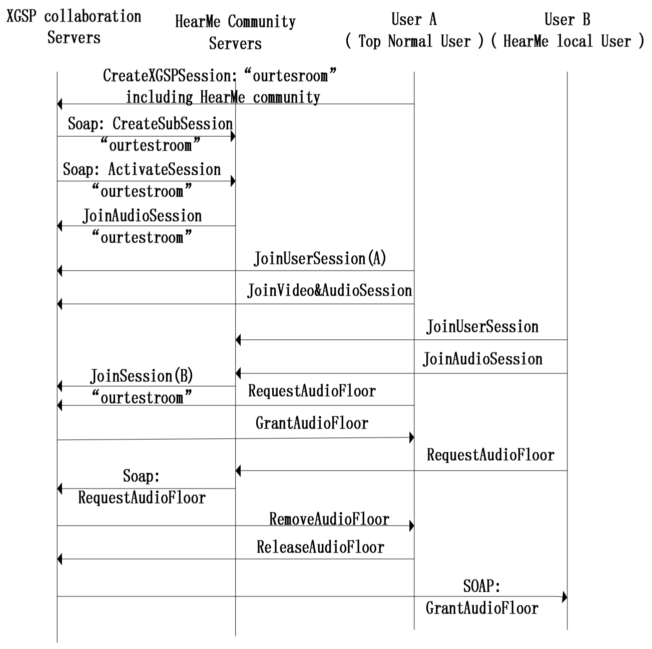
XGSP collaboration Servers
HearMe Community Servers
User A (Top Normal User)
User B (HearMe local User)
CreateXGSPSession; "ourtesroom" including HearMe community
Soap: CreateSubSession "ourtestroom"
Soap: ActivateSession "ourtestroom"
JoinAudioSession "ourtestroom"
JoinUserSession(A)
JoinVideo&AudioSession
JoinUserSession
JoinAudioSession
JoinSession(B) "ourtestroom"
RequestAudioFloor
GrantAudioFloor
RequestAudioFloor
Soap: RequestAudioFloor
RemoveAudioFloor
ReleaseAudioFloor
SOAP: GrantAudioFloor

Fig 4 an example of XGSP conference control procedure

Figure 4 shows an example of the fictional conference in which a HearMe client joins and gets audio floor to speak. There are two users in the conference: User A is a top normal user and User B is a HearMe local user. At the beginning, User A creates a XGSP session named "ourtestroom" which includes HearMe community. When the XGSP web server gets the XGSP command from User A, it sends SOAP commands to HearMe session server to create a XGSP sub-session and activate the session. After the session is ready, User A and User B join the XGSP top session and the XGSP HearMe sub-session separately. The A/V endpoint of User A directly connects with the XGSP A/V server, while the audio endpoint of User B connects with the HearMe audio MCU. Suppose initially the conference moderator allows user A to get the audio floor after it sends a request to the XGSP server. And then User B contacts the HearMe servers to become the speaker. The Hearme server notifies the XGSP moderator about the request by sending a SOAP command RequestAudioFloor to it. The XGSP moderator removes the floor from User A and grants it to User B.

Global-MMCS prototype system has been tested by developers and a small group of users across US and China. It gets positive feedbacks from the users who can use unicast and low bandwidth networks to attend AccessGrid conferences. Since the prototype includes a single XGSP A/V Server, it can only support limited scale of collaborations. For example it can support 5 conferences with 50 participants in each of them. Now we are working on the new prototype in which XGSP A/V servers will be distributed upon Narada broker infrastructure to improve the scalability and robustness of our system.

5. Conclusion

In this paper, we have described a web-service based framework XGSP for conference control. Under the XGSP framework, not only various audio/video endpoints but also communities can be integrated into a single A/V collaboration environment. This framework implements user and A/V application session management and floor control function in a scalable structure over heterogeneous collaboration systems. The XGSP framework is not designed for replacing the frameworks of H.323, SIP as well as AccessGrid, but for bridging them based on web-services technology.

Because XGSP signaling procedure is built on XML encoding and SOAP communication, its performance may be a little slower than those protocols based on text or binary encoding schema. But the cost caused by the SOAP engine and XML parsing will only increase the delay of session creation, user joining and so on. It will not affect the QoS of audio and video communication.

We also built a prototype system based on the XGSP framework. Such an integrated collaboration environment greatly benefits those users that want to enter Access Grid world via H.323 and SIP clients, and creates channels for interconnecting different collaboration communities.

6. References

[1] International Telecommunication Union, Packet-base multimedia communications systems, Recommendation H.323, Sep, 1999

[2] Session Initiation Protocol (SIP), RFC 2543, http://www.ietf.org/rfc/rfc2543.txt.

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

[4] Steve Graham, Simeon Simeonov, etc, *Building Web Services with Java* (Sams publishing, ISBN0-672-32181-5, 2002).

[5] Bormann, C., Kutscher, D., Ott, J., and Trossen, D. Simple conference control protocol service specification. Internet Draft, Internet Engineering Task Force, Mar. 2001, Work in progress.

[6] Dommel, H.-P., and Garcia-Luna-Aceves, J.Floor control for activity coordination in networked, multimedia applications, *Proc. of 2nd Asian-Pacific Conference on Communications (APCC)* .Osaka, Japan, June 1995.

[7] Handley, M., Wakefield, I., and Crowcroft, J. CCCP: conference control channel protocol-a scalable base for building conference control applications, *ACM Computer Communication Review 25*, 4 , Oct, 1995, 275-287.

[8] Kausar, N., and Crowcroft, J. An architecture of conference control functions, *Proc. of Photonics East, Boston*, Massachusetts, Sept 1999.

[9] Koskelainen P., Schulzrinne H. and Wu X., A SIP-based Conference Control Framework, *NOSSDAV'02*, May 12-14, 2002, Miami Beach, Florida, USA.

[10] Wu, X., Koskelainen P., Schulzrinne H., Chen C. Use SIP and SOAP for conference floor control. Internet Draft, Internet Engineering Task Force, Feb. 2002, Work in progress.

[11] Handley, M., Crowcroft, J., Bormann, C., and Ott, J. Very large conferences on the internet: the internet multimedia conferencing architecture, *Computer Networks 31*, 1999.

[12] Trossen, D. *Scalable Group Communications in Tightly Coupled Environments, PhD thesis, University of Technology*, Aachen, Germany, Sept 2000.

[13] International Telecommunication Union, Data protocols for multimedia conferencing. Recommendation T.120, July 1996.

[14] International Telecommunication Union, Generic conference control, Recommendation T.124, Feb. 1998.

[15] International Telecommunication Union, Terminal for low bit-rate multimedia communication, Recommendation H.243, Feb, 1998.

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

[17] Simple Object Access Protocol (SOAP) 1.1, http://www.w3.org/TR/SOAP/.

[18] International Telecommunication Union, Calling Signaling Protocols and Media Stream Packetization for Packet-based Multimedia Communication Systems", Recommendation H.225.0, Feb, 2000

[19] International Telecommunication Union, Control Protocols for Multimedia Communication", Recommendation H.245 Feb, 2000

[20] OpenH323 Project, http://www.openh323.org.

[21] HearMe Audio conference system, http://www.hearme.com.