Comprehensive Multi-platform Collaboration

Kundan Singh, Xiaotao Wu, Jonathan Lennox and Henning Schulzrinne
Department of Computer Science, Columbia University
{kns10,xiaotaow,lennox,hgs}@cs.columbia.edu

Abstract

We describe the architecture and implementation of our comprehensive multi-platform collaboration framework known as Columbia InterNet Extensible Multimedia Architecture (CINEMA). It provides a distributed architecture for collaboration using synchronous communications like multimedia conferencing, instant messaging, shared web-browsing, and asynchronous communications like discussion forums, shared files, voice and video mails. It allows seamless integration with various communication means like telephones, IP phones, web and electronic mail. In addition, it provides value-added services such as call handling based on location information and presence status. The paper discusses the media services needed for collaborative environment, the components provided by CINEMA and the interaction among those components.

(**keywords**: collaborative work; multimedia communication; conferencing; SIP)

1 INTRODUCTION

In many organizations, e-mail and tele-conferencing are the only means of collaboration. More recently, instant messaging (IM) is being used for short interactive communication. Even though these communication means are not designed for collaborative work, the limited set of available options causes them to put all their data such as meeting notes, documents, conference schedules and reminders into the email system.

We need a collaborative environment that seamlessly integrates with the existing communication means of email and phone as well as newer methods like IP telephony and instant messaging. Consider an IP telephony conference with some participants on phone, and some others using desktop audio/video clients. Late-arriving participants can browse through the past meeting proceedings, and non-participating group members can be automatically notified of meeting minutes and other important document locations via email.

One reason many earlier collaboration systems have not succeeded is that they were hard to use for people when the teams and groups span organizational boundaries. Also, they often require installing a lot of software, usually only available for limited set of platforms such as Windows, or work for only one vendor tools [69]. Collaboration tends to the "least common multiple" configuration supporting all needed tools and platforms, since groups can rarely say "sorry, since you cannot run this software, we will not include you in the committee".

There are two modes of collaboration. A "synchronous" or tightly coupled collaboration is highly interactive and requires the active presence of the other members of the group. On the other hand, an "asynchronous" or loosely coupled collaboration is part of some collective activity directed towards some shared goal or common purpose, but does not require the active presence of the other members of the group. A comprehensive collaboration environment provides both synchronous and asynchronous collaboration tools and integrates the two so that users can easily alternate between the two.

Our system is different from other conferencing applications in that it integrates the two modes of colloboration. For example, same group of people can be addressed by video conference, IM and email, with appropriate archival of interactions. Secondly, it provides device-transparency by allowing access and interaction even if participants temporarily have only a phone or email. Although it is not new, we also provide hybrid interaction such that one can use phone for audio and PC for IM and document sharing in the same conference.

Multimedia conferences on the Internet [56, 58] can be created using the Session Initiation Protocol (SIP). SIP [59] is a signaling protocol for Internet conferencing, telephony, presence, event notification and IM. One advantage of SIP over PSTN (Public Switched Telephone Network) based tele-conferencing is that it allows creating new flexible services including traditional PSTN services such as interactive voice response (IVR), call transfer, music-on-hold as well as more Internet-specific services like presence-enabled calls and integration of voice-mails and emails. Using SIP for instant messaging allows easy integration with other SIP-based telephony components. Various components such as media streaming server, call-routing engines and user location server can be combined in various ways to create new service architectures. One such architecture, CINEMA (Columbia InterNet Extensible Multimedia Architecture [34, 35, 70]) promotes collaboration using synchronous communications via multimedia conferencing, instant messaging, shared web-browsing, and asynchronous communications via file sharing, discussion forum, voice and video mails.

Our architecture provides building block tools for any type of multimedia collaboration, instead of focusing on specific types such as collaborative software development. We want to support three kinds of typical interactions: long-lived distributed groups that alternate between synchronous and asynchronous interactions, such as design teams, college classes, committees and work teams, asymmetric events such as lecture and lecture series, where interaction is mostly limited to asking questions to the speaker, and short-lived spontaneous interaction among groups of people.

Our collaboration tools are based on standard protocols such as SIP [59] and Real-Time Streaming Protocol (RTSP [68]) for signaling, Real-time Transport Protocol (RTP [66]) for media transport, VoiceXML [45] for voice-based interaction, Call Processing Language (CPL [40]) for network-based service creation, Language for End System Services (LESS [79]) for endpoint-based service creation and a web interface for asynchronous collaboration.

In this paper, we describe the architecture and implementation of our comprehensive multi-platform collaboration framework. While the previous work ([34, 35, 70]) focussed on the telephony aspects, this article focuses on collaboration. We describe the requirements for comprehensive multimedia communication and collaboration environments in Section 2. Some related work is listed in Section 3. Section 4 provides an overview of the architecture and the user interface. Section 5 describes the synchronous collaboration architecture whereas Section 6 details the asynchronous collaboration. Finally, we present the conclusions and future work in Section 9.

2 REQUIREMENTS

The basic requirements for the comprehensive collaboration system consist of a personalized view of the system, real-time or interactive multimedia collaboration (called *synchronous*) and loosely tuned sharing of information (called *asynchronous*). A web-based user interface provides a portable and personalized way to access the system.

2.1 Personalized view

People like personalized view of the system such as per-user calendar with appointments and conferences. However, the system should also allow sharing the view with other users or in a group after filtering. For example, Alice may not want to see the events posted by Bob. She can schedule a conference or discussion forum for her project group, and invite the members to join.

2.2 Synchronous collaboration

The system should allow multi-party audio, video and text conferencing. It may allow shared white-board, shared applications and screen sharing. It should allow restricted conferences with only authorized members as well as public unrestricted ones. The conferences may be pre-scheduled or created on the fly. It may support both dial-in and dial-out conferences, floor control, and telephone-based authentication. It should be possible to record, and later, playback the proceedings of a conference. It may allow time-positioned playback (e.g., play after first 30 min of recorded data), share files with other participants (e.g., agenda, slides or meeting minutes), play a media file in the conference, merge two conferences into one, or split one into two conferences.

2.3 Asynchronous collaboration

Most basic form of sharing information is via various forms of messaging, e.g., E-mail, voice and video mails. The system may allow recording and filtering of IM communications. It may allow storing messages in various folders, accessing remote email clients or servers for multimedia messages and listening to the messages via a telephone. The web-based message board may be accessed via email or telephone. It should be possible to share files and other information within or across groups.

3 RELATED WORK

The computer-supported collaborative work (CSCW) has been studied even before the web [74, 21, 23, 24]. ACM's special interest group on supporting group work, SIGGROUP [12], explores topics related to computer-based systems that affects team or group in workplace settings. However, the focus remained mostly on web-based document sharing and concurrent editing in systems such as BSCW [11], Lotus Domino [4], Hyperwave [3] or Livelink [6]. Many researchers have explored specific types of collaboration such as collaborative software development [37], electronic class rooms [46], network games and sharing health-care information.

Multimedia conferencing using audio, video, and data communication using IM and email, have independently evolved and become popular over the years [61, 61, 31, 50, 60]. Using audio and video for collaborative work is not new [62, 33]. There are a number of audio/video collaboration systems such as MBone tools [44, 38], MeetingPlace [5] and GnomeMeeting [1]. The ITU-T's H.323 [75, 48] provides video conferencing systems along with T.120 for data conferencing and T.128 for application sharing [13].

Most of the technologies used in our architecture, such as shared web-browsing [30], conference floor control [25], application sharing [7, 2] and web-based collaboration [9] have been investigated extensively. A number of web portals such as Yahoo! and MSN provide online calendaring, and sharing of information to some extent. However, the concept of group is rarely used. Our work is the first demonstration of a SIP-based comprehensive and extensible collaboration system. Our approach comes from a multimedia communication background, that extends the previous CINEMA communication suite [70] to support different kinds of collaboration across different platforms. Our architecture integrates together the conferencing and collaborative computing approaches.

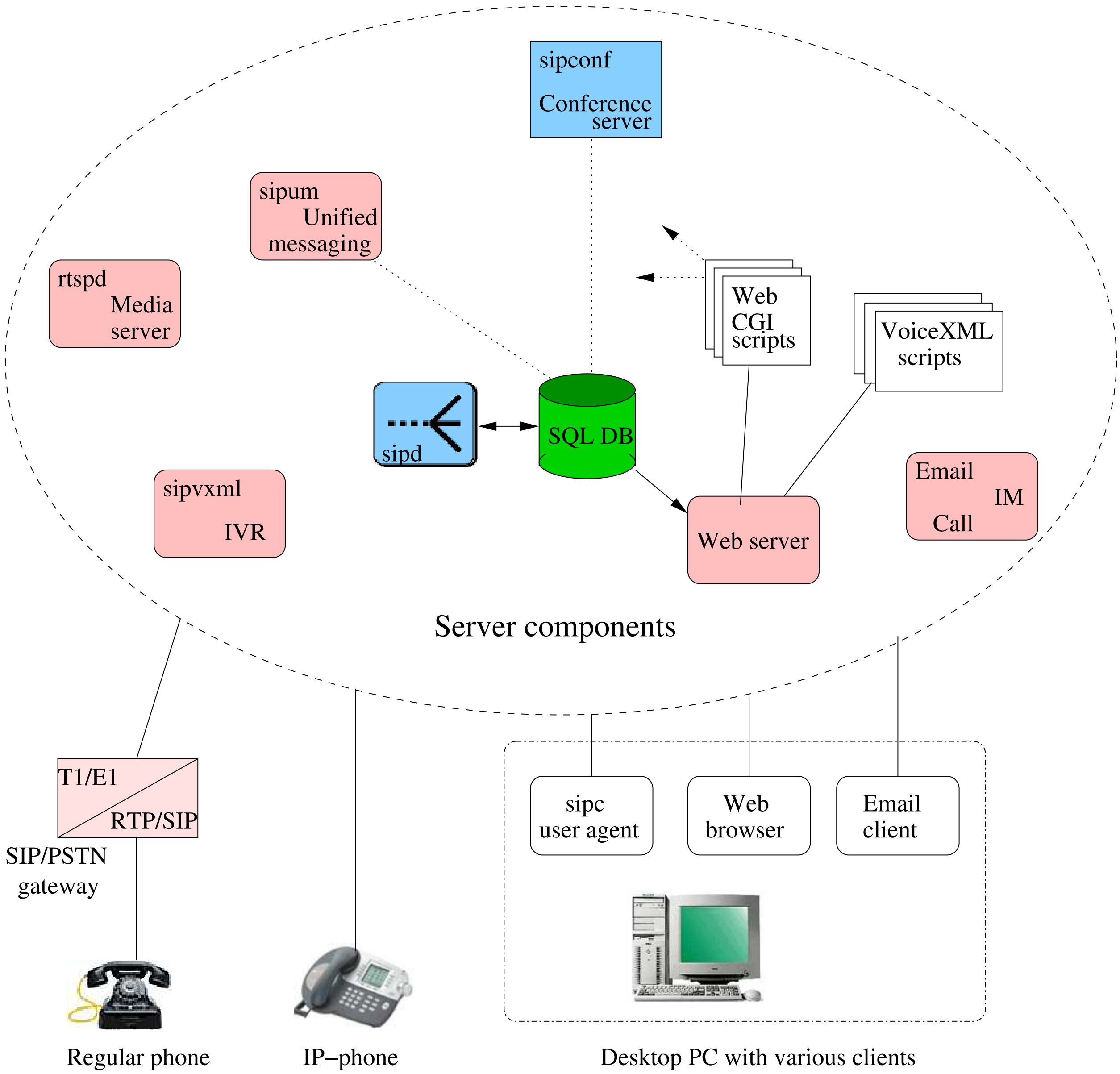

Figure 1: SIP-based collaborative work environment

The architecture consists of a set of distributed server components and user agents as shown in Fig. 1. The SIP registration and proxy server (sipd) is used for user location and forwading of signaling messages. The multi-party conference server, sipconf [71], forms the core of the synchronous collaboration infrastructure. The media server, rtspd, allows streaming of multimedia content for playback and recording. The unified messaging server, sipum, provides centralized answering machine, and multimedia mail service [73]. A web-based interface provides aynchronous collaboration support. User agents such as regular PSTN phone via a SIP/PSTN gateway, IP-phone, or desktop based SIP user agents like sipc are used for synchronous collaboration. Interactive voice dialogue via the VoiceXML browser, sipvxml [72], allows easy access to a telephone user. The SIP server and the SQL database [47] form the core of the infrastructure for basic call flow (Fig. 2).

4.1 Basic call flow

The system identifies the user by her *canonical user identifier* of the form *user@domain*. When a user, Bob, starts the SIP client on his PC, IP-phone or hand-held device, the client registers with the SIP server indicating the IP address of the device as shown in Fig. 2. The SIP server stores the mapping between the identifier *bob@home.com* and the address in the database. When another user, Alice, makes a call or sends instant message to *bob@home.com*, the server of the *home.com* domain proxies the request to the current device of Bob. Various call-routing preferences such as caller authentication can be configured from the web interface.

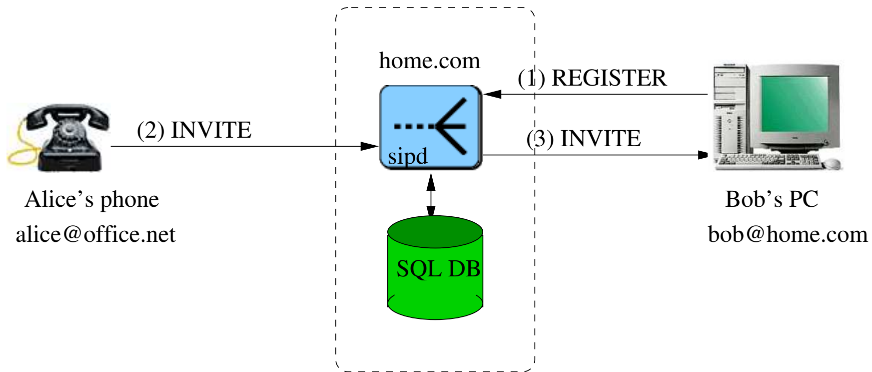

Figure 2: SIP call flow using proxy servers

4.2 Web interface

The web-based user interface allows managing user accounts, voice-mails and conferences. The web pages are generated using the HTTP CGI [55] scripts that access the SQL database for configuration and profiles. The web pages provide intuitive user interface components and context-sensitive help. There are multiple levels of details in different user expertise levels. For example, a beginner-level user accesses only basic features to get started whereas an advanced-level user can configure and manage detailed information. The interface allows configurable layout of the web pages so that a particular installation of the system can be adapted as per the service provider.

There are two types of users: *regular users* and *administrators*. An administrator has additional privileges compared to a regular user. The first user created during installation becomes the administrator. An administrator can add additional users as administrator or regular user, change the user type, or access profiles of other users. New users can also "sign-up" for the service from the web. The web interface functionality can be further classified as follows:

Call-routing profile: The user can manage profile information, current contact locations, alternative names for identification, on-line status of "buddies", access control as to who can call, and programmable call handling, e.g., based on time-of-day or caller-id.

Unified messaging: This includes the integrated interface for voice and video mails, emails and discussion forum on various topics.

Event calendar and conferencing: This provides the personalized calendar for each user. It allows managing various appointments, events and conferences.

Address book and access group management: The user can maintain an address book of his friends' profile such as name, email, department, birthday and postal address. The user can organize his address book entries into groups with different access privileges. For example, people in "my family" group can access his personal calendar whereas others cannot.

Administration and accounting: An administrator can manage several server configuration parameters, user privileges, as well as the visual layout of the web pages. He can also assign various tariff rates for the phone calls and configure the gateway locations for the telephone destinations.

The web interface is just a front-end to the user profile and system configuration information stored in the SQL database.

4.3 Personal calendar and address book

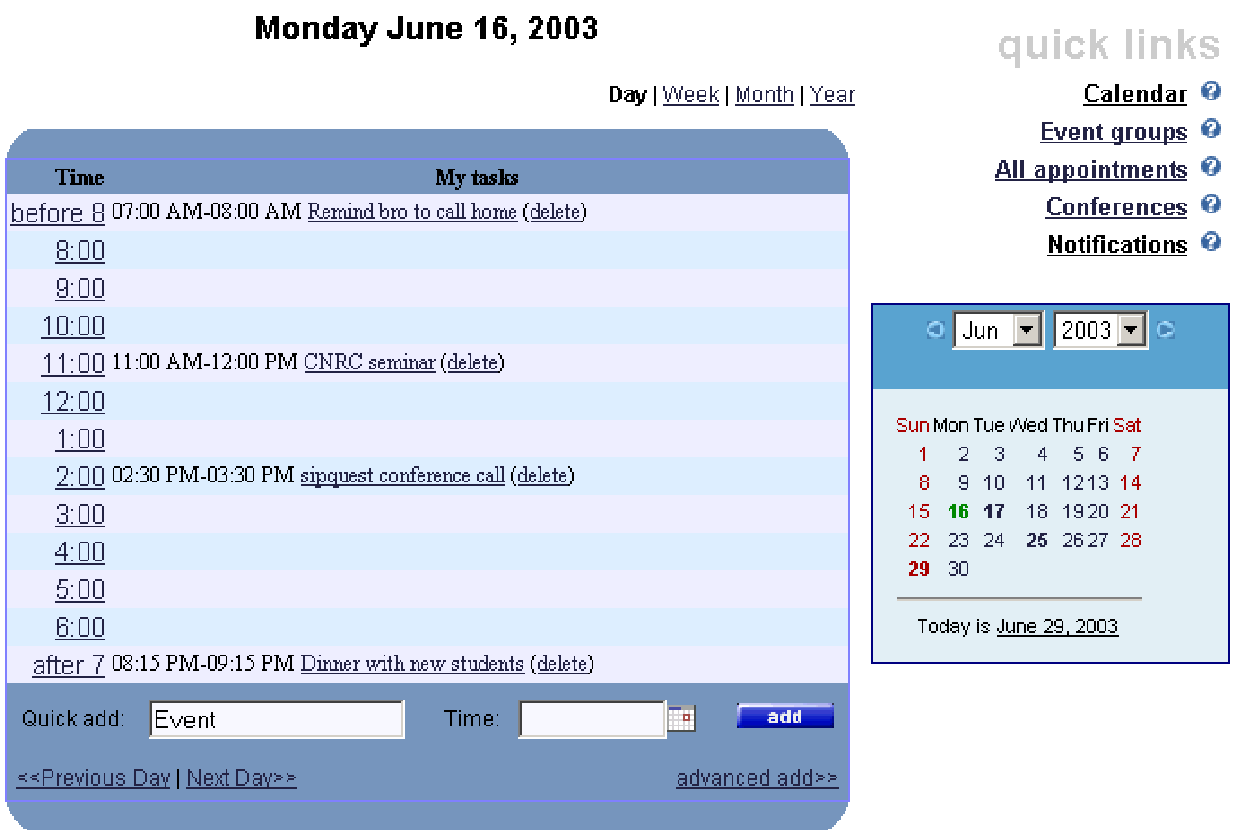

Figure 3: Personal calendar

When the user logs in from the web, it shows the most recent appointments and voice-mails. A personal calendar shows the various appointments or conferences scheduled for the user or his group (Fig. 3). The user can see the day, week, month or year view for different levels of information.

The per-user address book allows organizing the contacts into local or global access groups. A *local* group is visible only to the owner, e.g., "my friends", whereas a *global* group is visible to everyone, e.g., "network research group". An address book entry can belong to zero or more access groups. An event, such as an appointment or a class schedule, can have a group-name with given group-privileges. The read or write access privilege for an event can be owner, group or everyone, similar to Unix file permissions. The read access specifies who can view the description and details of the event. The write access tells who can modify the event attributes. A personal appointment typically has owner privileges for read and write, whereas a seminar series has group read access and owner write access.

The Windows registry (Fig. 4) can be configured for click-to-dial from the address book, so that the user can click on an HTML hyperlink, e.g., `<a href="sip:alice@nyu.edu">Alice</a>`, from the Internet Explorer web-browser to make a SIP call to Alice. On Unix systems, one can embed the data URL [43] of the form `<a href="data:application/sip,sip:alice@nyu.edu"> Alice </a>` in web pages.

```
REGEDIT4
[HKEY_CLASSES_ROOT\sip]
@="URL: SIP Protocol"
"URL Protocol"="RFC 3261 SIP"
[HKEY_CLASSES_ROOT\sip\DefaultIcon]
@="sipc.exe"
[HKEY_CLASSES_ROOT\sip\shell]
[HKEY_CLASSES_ROOT\sip\shell\open]
[HKEY_CLASSES_ROOT\sip\shell\open\command]
@="sipc.exe %1"
```

Figure 4: Windows registry configuration for click-to-dial

4.4 Events and event-groups

An event is an individual event or appointment. An eventgroup is a collection of related events, e.g., an university course for which individual classes, or events, happen weekly. Every event can belong to an eventgroup. An eventgroup can have zero or more events. An eventgroup can optionally have a repeat indicator, e.g., every month, every year. The repeat indicator is useful if one does not want to itemize individual events, e.g., yearly birthday reminders.

An event-group may be associated with an optional conference name, e.g., on-line lecture series. While an eventgroup defines a group of events used in calendar, a conference is strictly a synchronous collaboration with additional attributes like supported media-types, dial-in number, recording formats, default audio sampling rate, public or private conference type and public or private participant list. Various SQL tables for storing the information are explained in CINEMA technical report [70].

4.5 Database tables

The user information needed for call routing is stored in the Primary User Table (PUT), indexed by the user identifier. Personal address book information such as full name, organization, department, email address and biography, are stored in the person table. The relationship among various tables is shown in Fig. 5 and 6. The PUT and person table entries (Fig. 5) are kept for each user whereas other tables such as personnote or aliases are optional. For details see [70].

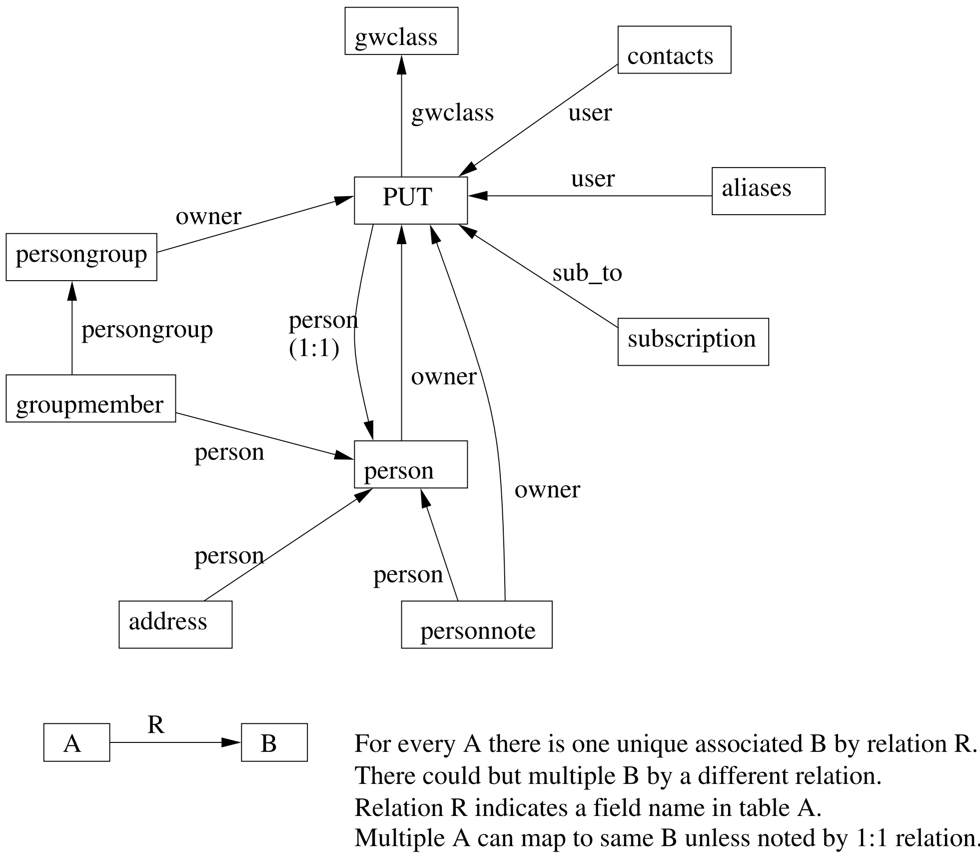

Figure 5: Primary user table, person, person group, and so on

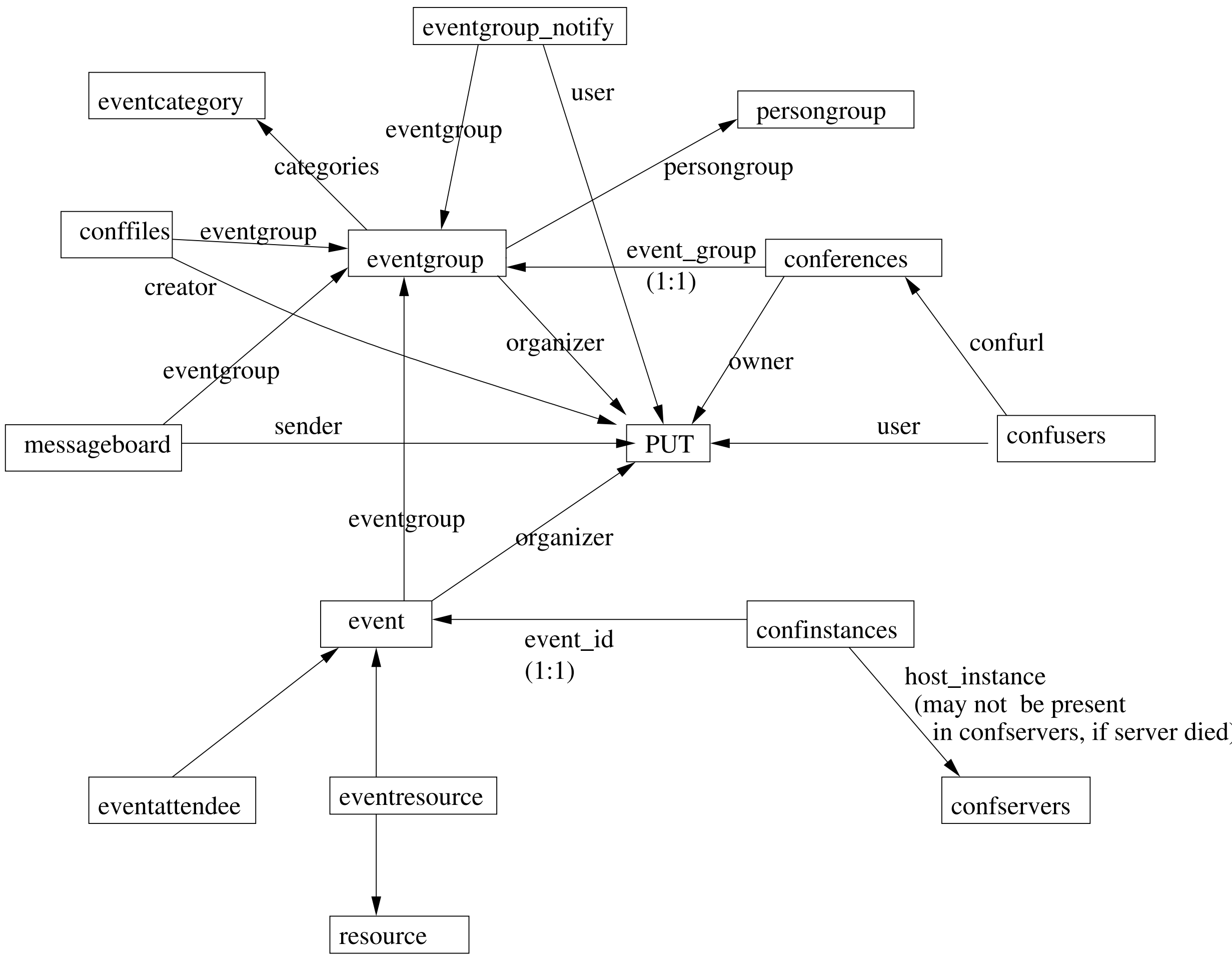

Figure 6: Events, event-groups, conferences and conference instances

A multi-party multimedia conference is the simplest form of synchronous collaboration. In the absence of real multicast, centralized conference servers provide an attractive solution for small to medium scale conferences [71]. Moreover, a centralized control integrates easily with other collaboration requirements such as floor control. For example, the organizer can control who gets to speak at any instant if there are multiple speakers, and enforce the policy at the server.

The participants dial the conference URL, e.g., *sip:staff-meet@cs.columbia.edu*, to join the dial-in conference. The conferences can be pre-scheduled from the web interface, or created on the fly, e.g., by dialing *sip:letsmeet.adhoc@conference-server*. For centralized conferencing, we need a central conference server such as sipconf and user agents such as sipc as described below.

Figure 7: Columbia SIP User Agent (sipc)

5.1 User agent

Sipc is a SIP user agent that can be used for Internet telephony calls, multimedia conferences, presence, instant messaging, and shared web browsing. It supports a range of media types, such as audio, video, text and white board (Fig. 7), and can be easily extended to handle additional media types. It uses external media tools such as vic [76], RAT [77] and wb [10]. We are currently developing our own low-latency audio tool. Beyond multimedia communication, it can also perform network appliance control, or act as SIP-based Internet radio or TV [32]. We are extending it to support emergency services [65].

The participants can also use other SIP-phones or regular telephones to join the conference. We are implementing another SIP user agent, sipz, for handheld devices to allow mobile multimedia participants.

The conference can have heterogeneous endpoints with different media capabilities. For example, some user agents connected to low bandwidth links can have only low bit-rate audio codec whereas others on high bandwidth links can support high-quality codecs along with video. When a user agent joins a conference, it indicates its capabilities to the server. The server selects a subset of capabilities based on the intersection of user agent capabilities and the server capabilities on per-participant basis.

5.2 Audio mixing

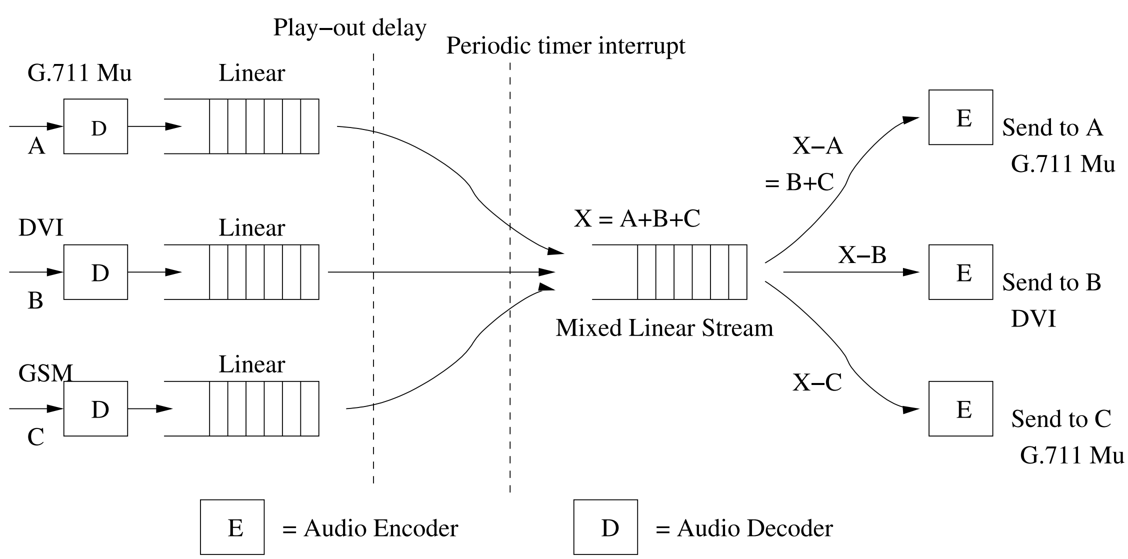

Figure 8: Audio mixing

When the participants join the conference, the server mixes and redistributes the audio such that a participant hears everyone else except herself from the server. The server decodes the incoming audio from the participant, and puts it in a per-participant queue as shown in Fig. 8. On periodic interrupt, the participant audio is mixed, and redistributed back to the participant after encoding. Optimizations reduce the number of encoders and decoders [71]. The server acts as an RTP mixer [66] for the audio. Each call leg in the conference forms an RTP session with the participant.

If the participants use different devices, it is possible that some users are heard louder whereas some others are hardly audible due to different speaker and microphone volumes. The server can do automatic gain control for both incoming and outgoing audio. However, this puts additional processing overhead on the server and reduces scalability. Alternatively, the server can indicate the volume level if its too high or too low to the participant, who can then adjust her microphone and speaker volume, or it can selectively implement automatic gain control for participants who want it. For example, a participant can press 6 to increase her microphone volume, or press 7 to reduce her speaker volume.

5.3 Video forwarding

Unlike audio, mixing does not make sense for video. Every participant may want video from everyone else in the conference. The server implements transparent packet forwarding for video. A video packet from a participant is distributed to every other participant in the conference without modification. In this case, the server does not implement the RTP stack for video session. The lip synchronization between the audio and video sessions is done at the participant's user agent on receiving the two streams.

Alternatively, the server can send only one video stream of the *active* speaker to all the participants, or the chair can decide whose video stream needs to be distributed. We believe that organizing the participants' video in NxN tile puts additional processing load on the server, degrades quality, and is undesirable.

The user agent indicates its receiving UDP port number for video when it joins the conference. In response, the server returns its own receiving port. The server first tries to allocate the same port as the user agent, because some video tools such as vic [76] do not allow different transmit and receive port numbers. However, if the desired port is not available at the server, the server selects a different port causing interoperability problems with such tools.

If the user agent does not indicate video capability, i.e., no video port, then video is disabled for this call leg. The participant can dynamically change the capability. For instance, she can start with audio, and later, switch to audio and video session.

5.4 White-board sharing

The White-board is a conference application for shared drawing. It allows synchronous collaboration using graphic information. The ITU-T Recommendation T.126 defines a protocol to manage the conference-wide synchronization of multi-plane and multi-view graphical workspace. However, In our system, we use a existing simple white-board application developed at UCL [10] in sipc and are planning to support sharing in sipconf. The server can forward the drawing commands to all the participants except the sender without internally maintaining a shared graphical workspace. However, the new joiners will lose historical drawings. To avoid this, the server can cache all the drawing commands for late-arriving participants.

5.5 Instant messaging

The instant message (IM) handling in the conference server is similar to video forwarding. When *alice@office.net* sends an IM to *bob@home.com*, the SIP server at *home.com* domain proxies it to the current location of Bob's phone. An IM sent to the conference URL *sip:staff-meet@servers.com* is intended for all the conference participants. If the conference is not active or there is no other participant, then the server indicates the error to the sender. If the sender is not already in the conference, then the server can either indicate an error to the sender, or still continue to distribute the IM to the participants. In a way, the server provides a group address to send IM to, similar to email-groups.

```
MESSAGE sip:alice@office.net SIP/2.0
From: <sip:staff-meet@servers.com>
To: <sip:alice@office.net>; tag=Uo18a
Content-Type: Message/CPIM
...                                         SIP headers
From: Bob Wilson <im:bob@home.com>
To: Alice <im:alice@office.net>
Content-Type: text/plain
...                                         IM headers
Meet me at Tom's at 8:00.
                                            IM text
```

Figure 9: Example SIP MESSAGE for instant messaging

An example SIP MESSAGE sent by the server is shown in Fig. 9 and has three parts: SIP headers, IM headers and IM text. Our sipc displays one message window for all the IM in the same conference.

Alternatively, a user agent can display one window per sender per conference. The server can also forward indications [63] that allows Alice's user agent to display status such as "Bob is typing a message".

The server should allow transitioning from an IM session to a full multimedia session, and vice-versa, when the participant changes her media capabilities accordingly.

5.6 Shared web browsing

The SIP MESSAGE method can be used not only for instant messaging, but also for some additional control. For example, sipc can capture the browser event on navigation and indicate that HTTP URL to the remote party. The server forwards the message like any other IM, thus, readily supports sharing among multiple participants. The message is similar to Fig. 9 except that the IM header Content-Type is `text/uri-list` and the IM text contains the HTTP URL. If the remote party understands this content, it can also invoke the browser pointing to the given HTTP URL. In sipc, we have implemented an application to control Internet Explorer and Netscape for shared browsing.

5.7 Screen sharing

We have added support for the open source Virtual Network Computing (VNC [51])-based screen sharing in both sipc and sipconf. VNC is a client server protocol, where the server shares its screen to a viewer or client. To avoid authenticating the client, we initiate the session from the VNC server to the listening client. If a participant shares her screen, her user agent invokes the VNC server application whereas all the other participants invoke the VNC client application. The conference server merely forwards packets similar to video forwarding. The data packets containing the screen buffers are forwarded from the VNC server to all the VNC client applications where as the control packets such as mouse and keyboard input are sent from the VNC client to the VNC server application. The VNC protocol can be tunneled through SSH for secure sessions.

5.8 Conference control

In a hybrid conference using phone for audio and PC for IM, it should be possible to control the conference from either phone or IM. Simple IM to the server can be used as control commands, e.g., if a participant sends IM text as "list", the server returns the IM text containing list of all the active participants. Similarly, when a new participant joins or one leaves, all the existing participants are notified by the server via IM.

Conference floor control means controlling who gets the exclusive access of the shared media channels or resources. For example, typically only one participant should speak in a conference. In case of multiple contenders, the conference chair can decide who gets to speak. There are many ways to do advanced floor control such as using Simple Object Access Protocol (SOAP) to run Remote Procedure Calls (RPC) [78] on the server, web interface, and via touch-tone phones. We are implementing the SOAP-based floor control in our server and user agent.

SIP and SOAP: Conference floor control consists of two parts: notifying the participants about who is the holding the floor [54], and allowing the moderator and the participants to remotely control the floor. For example, a moderator can grant or deny a floor request and a participant can claim or release a floor. We use XML-based platform independent SOAP [16]) for encapsulating and exchanging floor-control commands instead of creating a new RPC (remote procedure call) protocol. We have defined a SIP and SOAP-based floor control protocol [78].

Web interface: The control messages can be sent from the web via CGI scripts or Java applets. The moderator can grant or reject floor to other participants from the web. For the web-based floor control, the web components communicate with the conference server and indicate the appropriate control message.

Interactive voice response: This allows a telephone user to control the conference via limited touch-tone keys. For example, "press 1 to ask for floor". The DTMF (Dual-tone multiple frequency) digits are typically detected and translated to special RTP packets [67] at the telephony gateway.

5.9 Dial-in vs dial-out conferences

Although most of our earlier discussion focussed on dial-in conferences, dial-out mode is equally important. For example, if a participant wants to invite another user in the conference, or the server wants to send out call invitations to the intended participants at the scheduled time. Usually some form of audio and text announcement indicates the purpose of the call to the user. To avoid the dialed-out call going to answering machine, the server may prompt the user to press certain digits to actually join the conference.

There are a number of related events during or after the conference, that need to be shared with others even when the conference is not active. For example, the recorded conversation or meeting minutes may be needed in subsequent meetings, off-line discussion on the topics covered in the conference needs to be co-ordinated in the same way as the conference was controlled or the notes may be edited remotely using WebDAV [29]. The primary objectives of these collaboration mechanisms are to avoid duplicating shared data and to provide some form of change control on shared data.

Use of RTSP, instead of the traditional download-and-view web model, enables the recording of the content once and the use of the pointer or the URL when forwarding the content without actually forwarding the multimedia file. This is desirable for low bandwidth situations where downloading a whole media file is very expensive, particularly if the recipient decides that she doesn't want to listen to the audio after hearing the first few seconds. Moreover, the multimedia content can be accessed with any RTSP based media player, e.g., Apple's QuickTime.

As mentioned earlier, every conference is associated with some eventgroup. An eventgroup can be associated with various forms of asynchronous collaboration mechanisms, such as file sharing and discussion forum. Conference participants can share meeting notes, agenda or other documents via the web.

6.1 File sharing

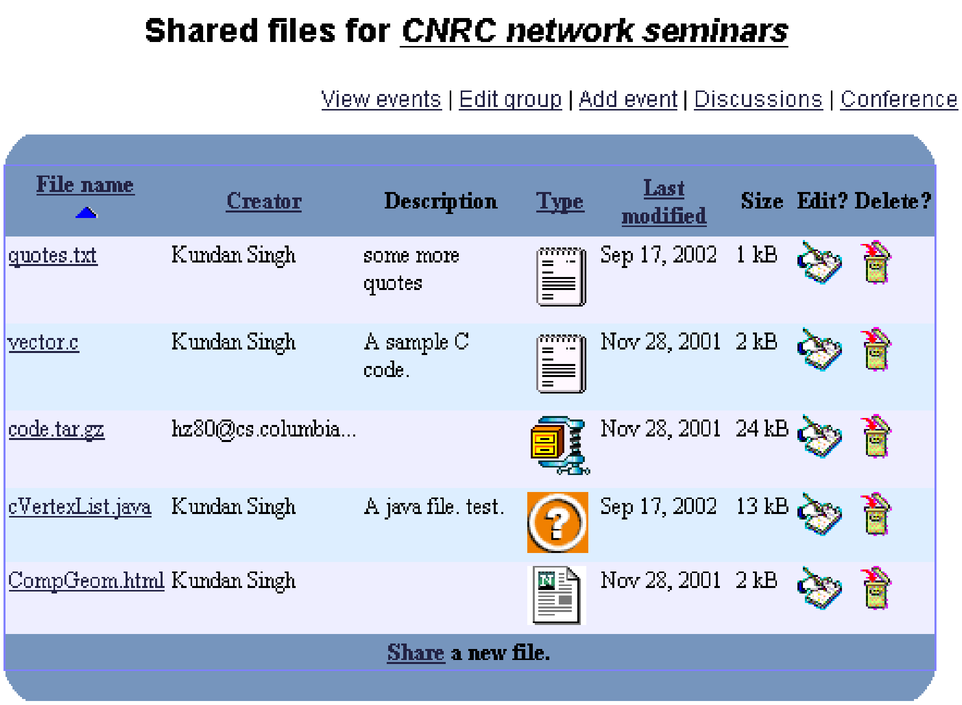

Figure 10: File sharing

The web interface allows uploading shared files as shown in Fig. 10. The shared file attributes consist of the creator's user identifier, name of the file, MIME-type [28] for display, a brief textual description, date of creation and last modification, and the access privileges for read, write and delete. The read access can be for the group or public, whereas the write and delete access can be for the group or owner. The group name of the file is inherited from the associated eventgroup. The users can register to get notified via email when the shared file is modified or deleted.

6.2 Discussion forum

Message boards and discussion forums facilitate asynchronous discussion on a particular topic. One advantage over email-based discussion is that it can systematically display the various discussion threads, postings and replies. The messageboard SQL table stores the message subject, content, sender identifier, date and time, associated eventgroup identifier, a unique message identifier and the message identifier of the parent message. If there is no parent message, then this message is the start of some thread. If there is a parent message, then this message is a reply to that parent message. The associated eventgroup specifies the read and write access attributes for the message board.

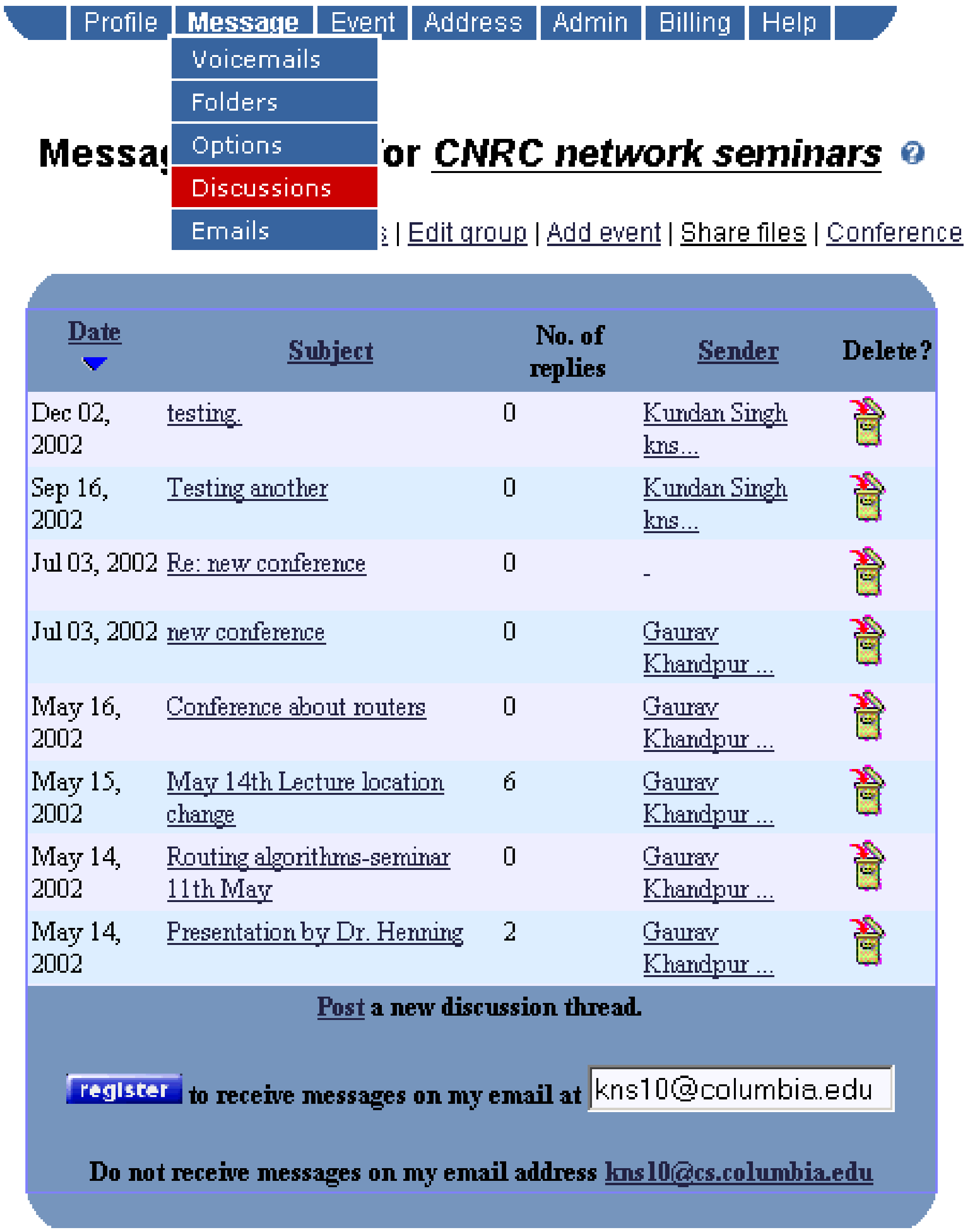

Figure 11: Web-based discussion forum

The users can also register to receive new posts or replies in their email. They can use email to post a message or reply to the discussion thread. Fig. 11 shows an example web interface. Integrating email with the system is discussed in detail in Section 7.7.

6.3 Conference event recording

The system allows recording of the audio, video and IM communications in a conference. The audio recording at the conference can be done either when the media packets (RTP) are received from the participant or when the mixed stream is created as in Fig. 8. In the former case, the recording is done by dumping the raw RTP (and RTCP) packets along with packet size and time-stamp, in a file. This "rtpdump" format can later be played out using our media server, rtspd. The server does not need to understand any specific media file-formats, such as MPEG or "wav", but works as longs as the playing client understands the codec used by the recording client. On the other hand, a mixed audio stream can be recorded in standard Sun "snd" or Microsoft "wav" file format. Only rtpdump recording format is needed for video, since the server does not generate any mixed video stream. The system allows recording in a local file or to remote media server using an RTSP URL. A per-conference quota on maximum recorded file size can be imposed. The recorded file or URI information is stored in the confinstances table for each conference instance, whereas recording format preference is indicated in the conferences table.

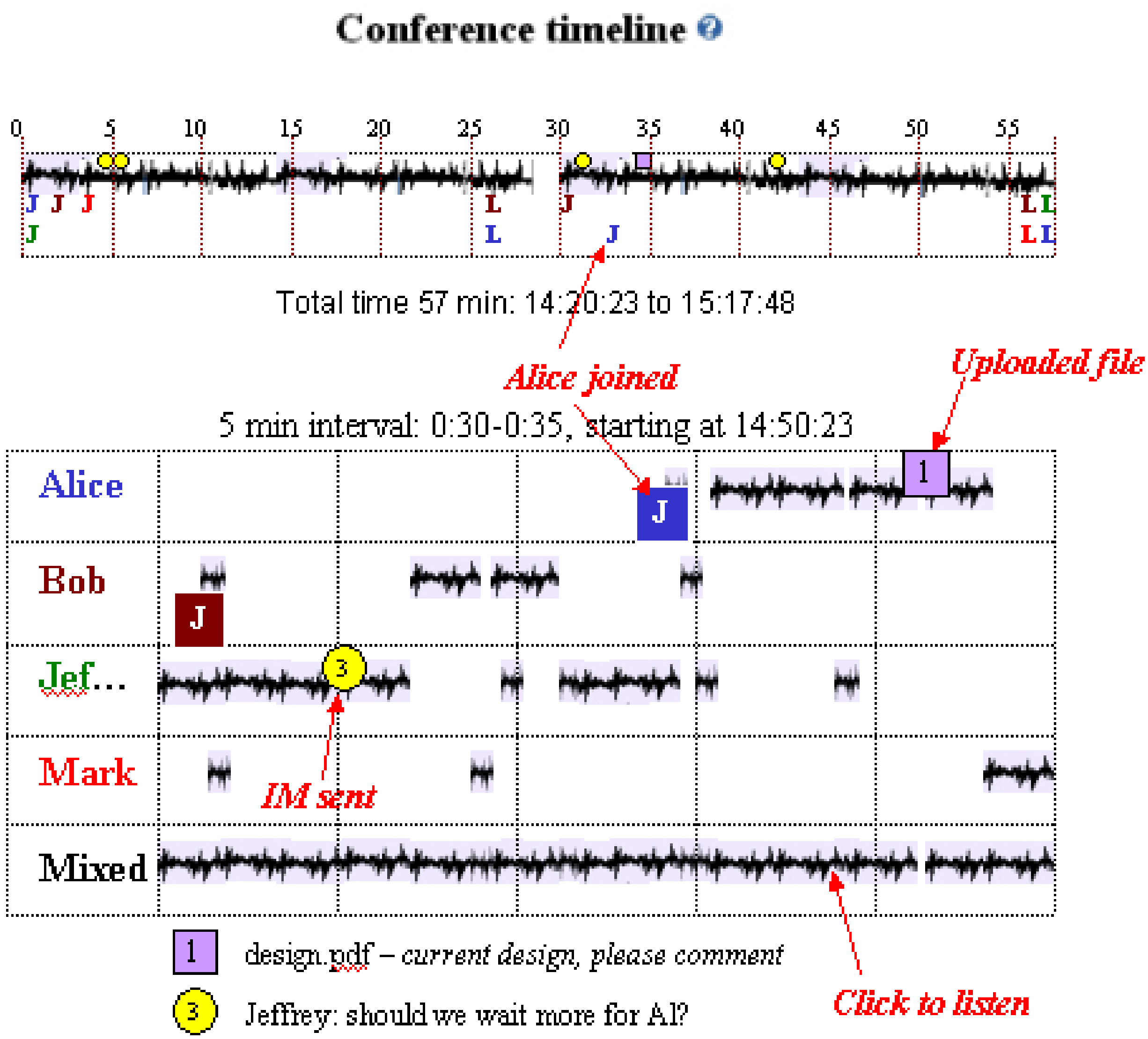

Figure 12: Web interface for conference recording

We are developing a web interface to display the conference proceedings in a time-line as shown in Fig. 12. The first time-line indicates the complete conference duration with the important events, such as the new user join, leave, file uploads and instant message interaction. The second time-line is the zoom-in view of a part of the conference duration as selected in the first time-line. User can click on the appropriate icon to playback the recorded media (audio, video), instant message or view the uploaded file. User can click on the time axis to jump to that location. Different colors are used to identify a small number of active participants.

6.4 Unified messaging

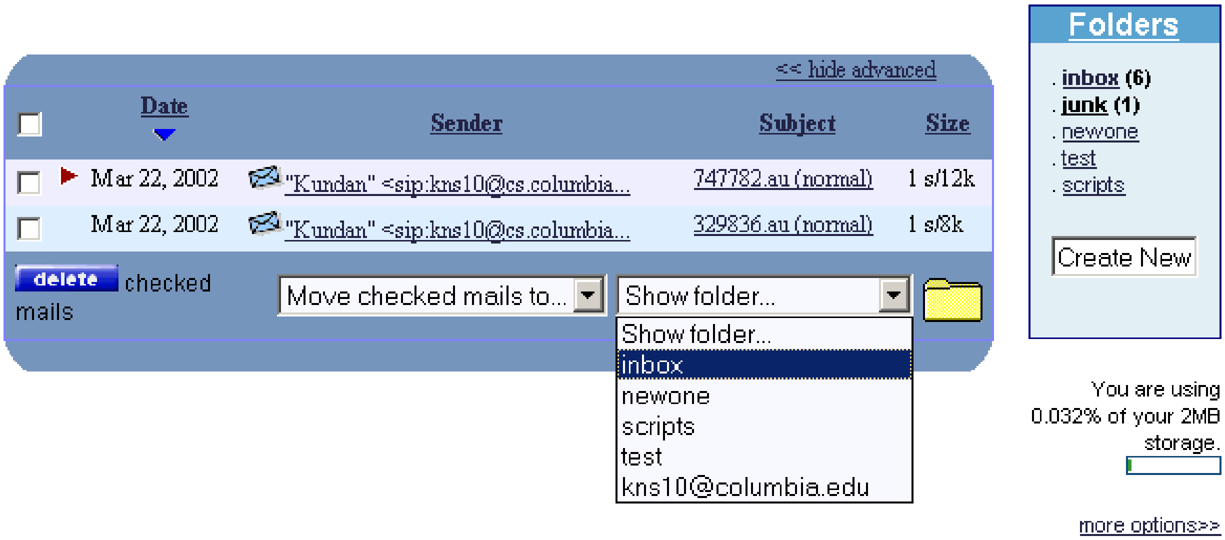

Figure 13: Voice messages

The ability to send multimedia messages to other individuals or a group is an important feature of collaboration systems. Registered users can listen to their voice/video messages, recorded conference proceedings or view their emails from the web. An example web page is shown in Fig. 13.

The voice/video mail is recorded at the media server, rtspd, by the centralized answering machine and voice mail server, sipum [73]. The server notifies the user of new incoming messages, e.g., using email, and indicates the pointer or URL to listen to the message. It should allow sending the media content instead of the pointer in the email, if the user wants that way.

The system provides an integrated set of facilities to ease user administration and to share common resources such as address books, calendar and group messages, and can also be extended to use other email and calendaring tools if the user prefers. For example, it would be nice if we could embed conference invitations in email that is automatically added to user's personal calendar if she accepts the invitation. The web interface can be extended with a simple IMAP-based client to fetch and display the email from her other email accounts. Alternatively, she can use her own email client to read these messages. She should be able to send multimedia messages to a group of people, such that the message appears in the group member's "inbox" folder. The user can delete or move the message pointer to another folder, and still share the media content across the group.

6.5 Notifications and announcements

The system can notify the user of various appointment reminders, conferences schedules or changes in shared files, message board or incoming multimedia message. The notification information is stored in the eventgroup_notify table and can be associated with an event and an eventgroup. The table also contains the destination for notification such as phone number, SIP URI, email address or IM address, time relative to the event in seconds (e.g., notify 60s before the event), and the identifier of the scheduled notification. The notifications are scheduled using the "at" command on the Unix platform. The user can schedule the same notification to multiple destinations. It supports different kinds of notifications:

- Birthday, appointment or other event reminders for which the notification is sent before the event occurs.

- Scheduling any text or media as a notification (e.g., wake-up call) that automatically creates an associated event. The notification is sent when the event occurs.

- Notification for the eventgroup, in which case the notification is sent for every individual event in that eventgroup.

While an email or IM is an one-time event with no interaction, a phone-based notification can prompt the user with more options via IVR. For example, "press 1 to get notified again after 5 min, or press 2 to listen to the details of the event". The system can allow scheduling the notifications from the web interface or via telephone using the touch-tone input.

It is possible to send a phone announcement to an eventgroup, in which case all the group members get the announcement, or to a set of SIP addresses or phone numbers. For example, an announcement to 1-212-93970?? will be received by all the valid telephone subscribers in the range 1-212-9397000 to 1-212-9397099. The announcement server makes SIP calls to all the numbers specified, and if successful, speaks out the announcement. It attempts multiple times on busy or no-answer. To avoid leaving the announcement to an answering machine, the server can prompt the recipient to press some digit to confirm user presence. Such announcement system will also be useful in the case of an emergency.

So far we have discussed the synchronous and asynchronous collaboration tools in CINEMA. There are other interesting services that assist both kinds of collaboration. For example, a conference server can dial-out a scheduled meeting only when all the required participants are on-line. An IM user can join a tele-conference and interact via speech-to-text and text-to-speech conversion between the IM text and other participants' audio. The location information published by the user can determine her availability. We describe some of these enabling technologies in this section.

7.1 Presence

The presence information gets used quite often in people's daily life. People are used to checking online status before starting a conversation with their IM "buddies".

In our system, we base our presence information handling on the SIP event notification architecture [54]. Beyond the existing implementations' presence status of online, offline or away, we consider both current and future communication availability via timed-status, a number of place-types, such as "home", "office", and "public" as well as a privacy classification into "public", "private" and "quiet" [64]. The user can also indicate whether the communication is likely to be overheard or whether audio is considered undesirable.

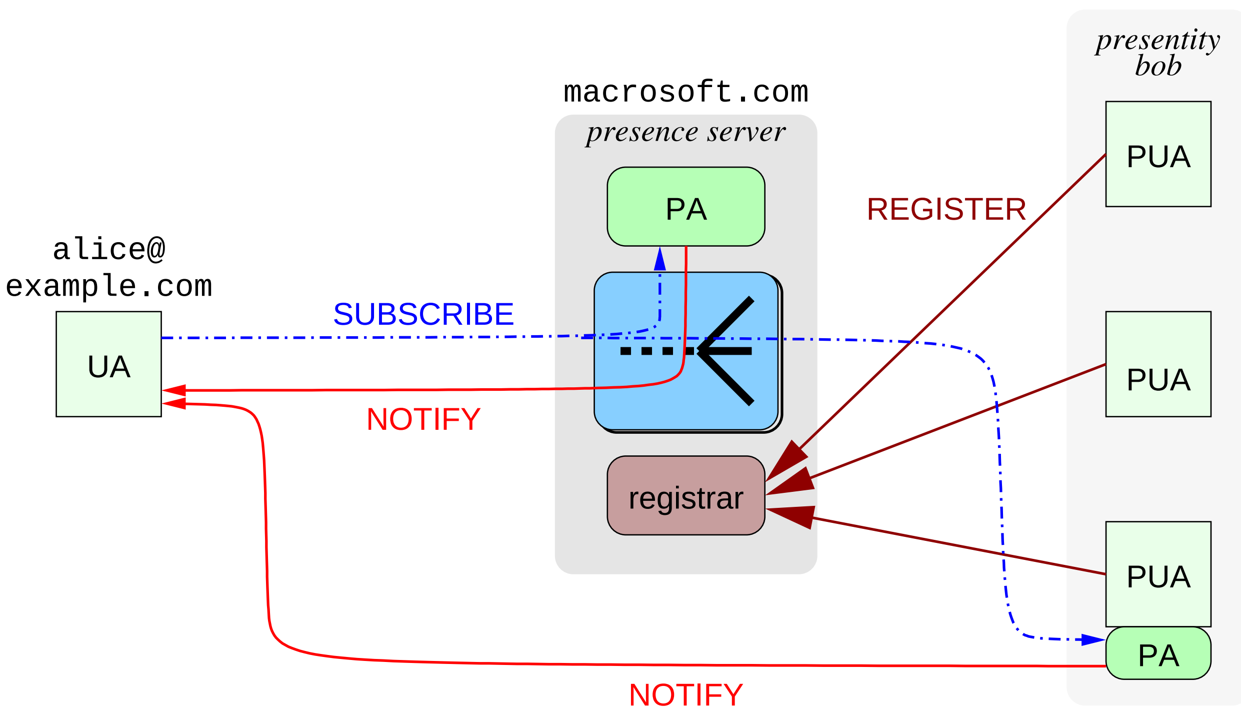

Figure 14: SIP-based presence

The presence information is maintained either on the SIP servers residing in networks or on the presence-enabled SIP user agents as shown in Fig. 14. If a user Alice is interested in the presence status of another user Bob, then she subscribes to his address *sip:bob@macrosoft.com* for the event package of "presence". Our SIP server, sipd, proxies the SUBSCRIBE message to the registered user agents with presence capability of Bob. If the user agent wishes to handle the subscription, it sends an accept (200-class SIP response) to sipd. Therefore, sipd disables the internal presence agent for this subscription. On the other hand, if the user rejects (600-class response), then the sipd stores the decision in the SQL database so that future subscription from Alice to Bob are also rejected even if Bob's user agent is off-line. For other responses such as the user agent is not presence-enabled, the sipd enables the built-in presence agent for this subscription.

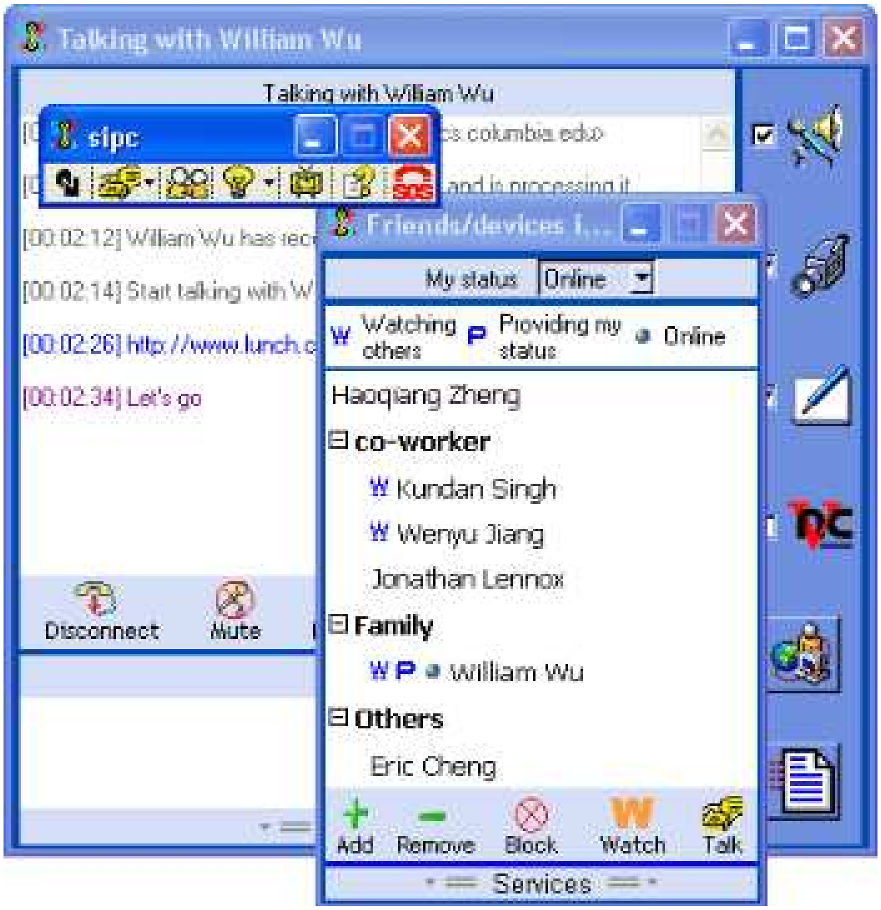

Figure 15: IM and Presence support in sipc

However, before the actual presence information can be conveyed to Alice, the subscribee Bob must approve the subscription from the web.

The web interface displays the list of subscribed users (buddies) as well as all the others who are interested in knowing the presence status of this user as shown in Fig. 16. Note that a subscription can be handled only by the server or the user agent but not simultaneously by both [57]. It is possible to transfer the subscription from the user agent to the server and vice-versa.

The network-based presence agent is useful when the end devices are not presence-enabled such as location sensors or magnetic swipe-card reader. For example, Bob can use a passive device, such as a magnetic swipe-card or an iButton [22] and the card or button reader delivers the location information to the server. Alternatively, when Bob's wireless phone REGISTERs with the server, the server can publish his on-line status to the subscriber, Alice.

Pushing the presence information to the end systems also has some benefits. In Internet telephony, end systems are the only entities where signaling and media flows converge whereas intermediate proxies only handle signaling. The means that several services can only be performed in the end system. We have defined a scripting language named Language for End System Services (LESS) [79] that includes a language package specifically for presence information handling. The example script in Fig. 17 places a call to Bob when he is on-line.

The SIP event notification can be applied to non-presence events, e.g., the voice-mail server can notify the user's phone of any waiting messages [42].

7.2 Interactive voice response (IVR)

We have discussed a number of examples involving user interaction via touch-tone input from a telephone. VoiceXML [45] is an XML-based language developed by the W3C to create voice dialogs that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input and recording of audio for telephony

Figure 16: Web-based presence

applications. It enhances the traditional proprietary and closed IVR systems to an open programmable architecture. The back-end programmable web CGI scripts can perform the application logic, such as voice-mail access or email access by phone. Our sipvxml is a SIP-based VoiceXML browser that allows a SIP-phone, or a regular telephone via a gateway, to interact with the back-end application logic [72]. Fig. 18 shows an example script that prompts the user for her identifier and invokes the next script `login.cgi` with the user input. The `login.cgi` script can generate the VoiceXML document for the the next set of dialogues.

We have developed some CGI-based applications for voice-mail access and conference participation. Each registered user gets a unique telephone PIN (personal identification number) for authentication. The voice-mail script announces the number of new and old messages, and prompts the caller to listen to the messages. The conferencing application prompts for the conference number and transfers the call to that conference.

There are two ways to transfer a call from the browser to another phone in VoiceXML. The *blind* transfer sends the SIP REFER message, and terminates the original call leg between the caller and the browser. The caller user agent is then responsible for placing another call to the Refer-to location. The *bridged* transfer causes the browser to place another call to the destination, and join the media path between the original caller and the destination. The advantage of bridged transfer is that the browser remains in the media path, and can terminate the call, e.g., if the calling-time exceeds the quota for the long-distance destination.

To allow PC-based SIP phones that do not have touch-tone dial-pad, the browser may accepts input via IM text. The prompts can be sent both in audio and IM for such phones. It will be interesting to explore how IM can be used for accessing voice-mails from sipc.

Interaction between VoiceXML and LESS is for further study. For example, sipconf can include a VoiceXML browser and invoke the LESS procedures for presence-enabled calls on voice-input.

```xml
<LESS:LESS
 xmlns:LESS="urn:ietf:params:xml:ns:LESS"
 xmlns:Generic="...:ns:LESS:generic"
 xmlns:Presence="...:ns:LESS:presence"
 xmlns:UI="...:ns:LESS:ui"
 xmlns:xsi="http://.../XMLSchema-instance"
 xsi:schemaLocation="......"
 name="bobOnlineCall" priority="0.8">
 <subaction name="generalCall">
  <Generic:call/>
  <UI:alert message="Calling %address%"/>
 </subaction>
 <Presence:notification
   direction="incoming" package="presence">
  <Presence:presence-switch>
   <event package="presence" is="OPEN">
    <LESS:address-switch field="origin">
     <address is="sip:bob@macrosoft.com">
      <sub ref="generalCall"/>
     </address>
     <otherwise>
      <UI:alert message="%address% online"/>
     </otherwise>
    </LESS:address-switch>
   </event>
  </Presence:presence-switch>
 </Presence:notification>
</LESS:LESS>
```

Figure 17: Example LESS script for presence-enabled call

```xml
<?xml version="1.0"?>
<vxml version="1.0">
 <form>
  <field name="userid">
    <prompt>
     Please enter your user identifier.
    </prompt>
    <filled>
      <prompt>
        You said <value expr="userid"/>.
      </prompt>
    </filled>
  </field>

  <block>
    <submit next="login.cgi"
            namelist="userid"/>
  </block>
 </form>
</vxml>
```

Figure 18: Example VoiceXML page

In a distributed component architecture it may be desirable to separate the text-to-speech and speech-to-text functions from a VoiceXML browser, as different modules. For example, an RTSP server can convert the text supplied in URL to speech and stream to the client. Alternatively, a SIP "text-audio" converter can convert between the text in IM and the audio in the call session. Such external components can be invited in the existing sessions for applications such as email by phone, as we describe next.

7.3 Interaction among email, telephone and IM

Today, email is the most common form of electronic communication. However, the convenience of email is limited by the necessity of an Internet connected computer. A system that allows interworking of email with other communication means such as telephone or IM, will enhance user experience. Such system can be used to reach those users who only have email access via IM, define certain incoming emails as important and forward them to IM, get a virtual-IM account to interact with other IM users via email, access emails via phone, get notified of any important email on phone, and text-chat with other IM users or in a conference via phone. We describe the SIP-based architecture and on-going implementation of such interactions in CINEMA.

7.4 Email by phone

Assuming that wherever you go there is a telephone, using a telephone to check email is a sensible solution. Our *email-by-phone* system provides a way to check and even send email from a touch-tone telephone [41]. The application runs as Java Servlet on a web server, generating VoiceXML pages for telephony dialogues, and interacting with back-end IMAP or POP3 email-servers as shown in Fig. 19.

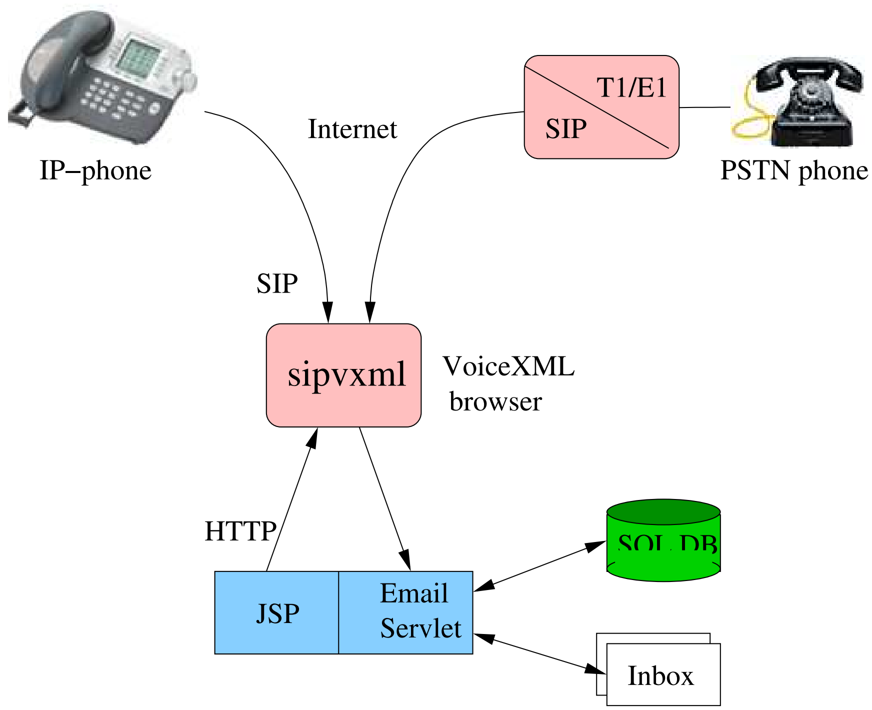

Figure 19: Email-by-phone architecture

The caller is prompted to listen to old or new messages, compose a new message, reply to an existing message, delete a message, forward a message, advance through the messages, and switch between new and old messages for playback.

7.5 Email to phone

Unlike the email-by-phone system that needs caller-based initiation, the system can also asynchronously notify the telephone when the user receives a new important email. The definition of "important" email can be programmed by the user. The architecture is shown in Fig. 20.

Incoming email filtering on Unix is relatively simple using procmail [8]. On other platforms, such as windows, one can periodically poll the IMAP-based email-server for new emails. An example procmail script shown in Fig. 21 that treats the subject with "important" or "Important" keyword, or sender as "Alice", as important and forwards to the siptc script.

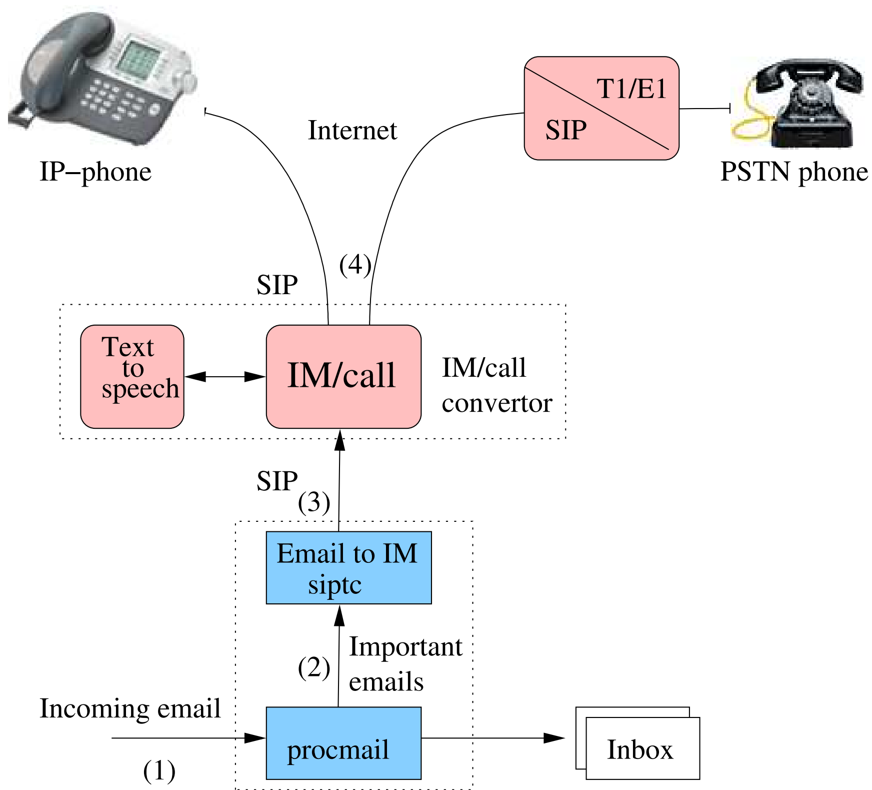

Figure 20: Email notification to phone

```
:0c
* ^Subject:.*[iI]mportant.*
| siptc

:0c
* ^From:.*Alice.*
| siptc
```

Figure 21: Example procmail script

The siptc script extracts the email body and other information, e.g., subject, priority and sender address, creates a SIP instant message, and sends it to the IM/call converter. The IM text is truncated if it is too big. The forwarded-from, replied-to or signature part in the email should be ignored as shown in Fig. 22.

The IM/call converter acts as a translator between the SIP-based IM and audio call. In the reverse direction, it can notify the IM over phone. In the absence of session-based IM, it uses the various headers in the SIP MESSAGE to associate the IM session with the audio call. This avoids making a new SIP call, if IM is received for an existing session. The SIP call destination can be pre-configured or derived from the IM destination address, which can be an IP user or a telephone subscribe via a gateway. Separating the converter from the email to IM translation allows running the email system and speech system on different hosts in the network.

Once the call is established the converted text is spoken out to the destination phone. After that, the system may transfer the phone call to an IVR system, e.g., to prompt the user to repeat the message or connect to the *email-by-phone* system.

7.6 IM/call converter

The IM/call converter described in the previous section, can be extended to a generic translator to allow a phone user to initiate an IM conversation. For example, Bob's IP telephony service provider allocates a telephone extension such as 7155 for his IM address. When Alice dials the extension, the service provider

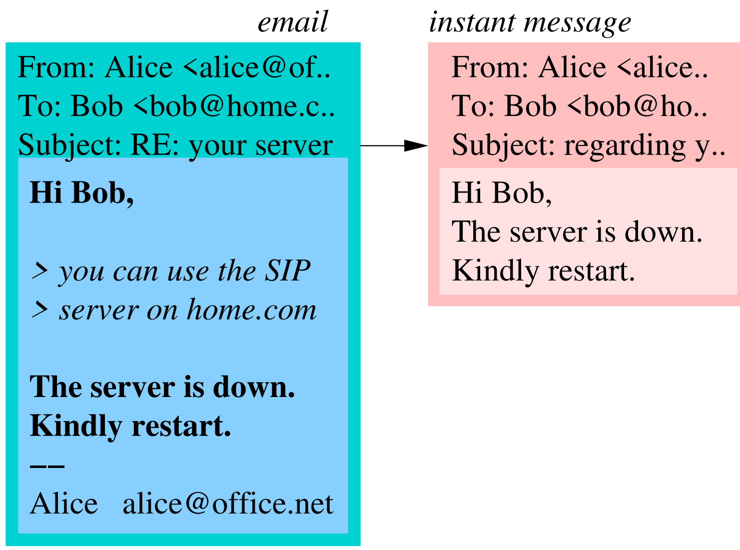

Figure 22: Example translation used in email to phone system

maps the destination to *sip:Ym9iQGhvc3RC@server*. Our simvoice server recognizes the final destination Ym9iQGhvc3RC as Base64-encoding [27] of *bob@hostB*. It sends an initial IM greeting to *sip:bob@hostB*. It maintains the association between the caller and the final IM destination for the duration of the call.

When Alice speaks, the audio is converted to text using CMU Sphinx speech recognition engine [20]. To send an IM, the user can indicate the end of a speech message by pressing a DTMF key. Alternatively, the converter can assume the end of a speech message, when it receives some audio followed by a few seconds of silence.

In the reverse direction, when Bob sends an IM text to simvoice, it invokes the Flite text-to-speech engine [14] to convert and send it to Alice as voice.

If Alice hangs up, the association is lost. If the simvoice can not find an associated call for an incoming instant message, it replies with another instant message indicating the error. The IM user should be allowed to initiate the session using session-based IM [18]

In a collaboration environment, the converter allows the users with different system capabilities such as telephone and IM to interact. This helps the deaf, hard of hearing and speech-impaired individuals to collaborate easily in a multimedia conference [19].

Often we have found that the speech-recognition quality is poor. The converter should provide feedback to the speaker by sending the converted IM text also to the phone using text-to-speech.

7.7 Email to IM

In the email to IM direction, Alice can email to a special address such as *myserver+bob@office.net*[1] which is received by Bob as IM. The procmail script of *myserver@office.net* receives the email, finds the IM destination as bob, translates the email's text content to IM and sends it to *bob@office.net*. Alternatively, Bob can advertise his email address as *bob+im@office.net* to send him an IM. The difference is that the procmail is configured at *myserver* in the former and at Bob's email in the latter case. A third approach is that Bob defines certain emails as important as described in Section 7.5 and automatically forward them to his IM address.

7.8 IM to Email

In the reverse direction, Alice can also use the services from *myserver* such that Bob can send IM to *myserver+alice@home.com*. The server should put the appropriate email Reply-to header pointing to the sender via *myserver*, so that the email replies can be sent back to the IM user correctly. Alternatively, Alice

[1] Note that some email servers allow sending email to user+something@domain, which will be delivered to the inbox of user@domain.

can sign-up with her SIP-provider to run a programmable call-routing using SIP-CGI [39] that identifies important IM and sends her an email with the content when she is not on-line, as shown in Fig. 23.

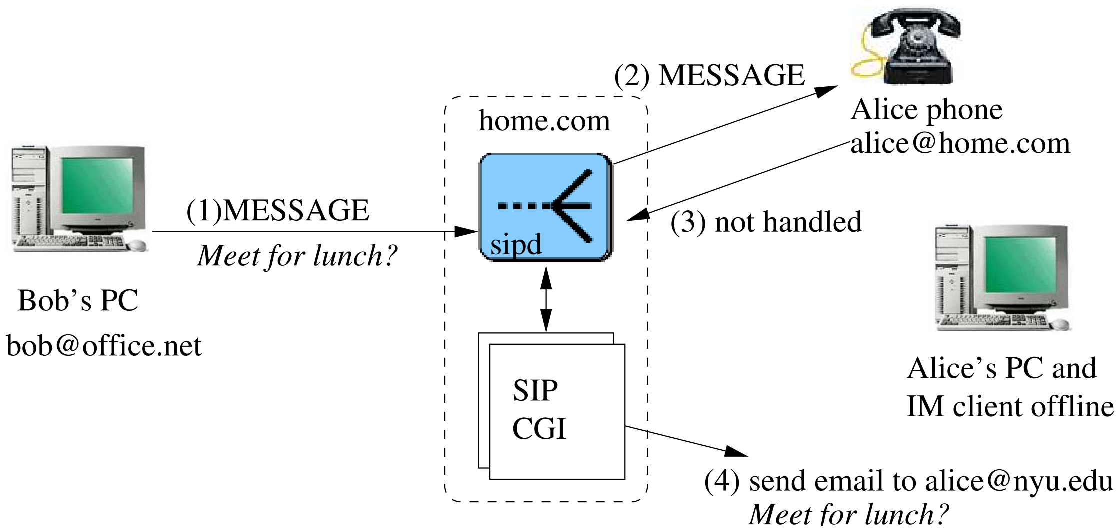

Figure 23: SIP-CGI for IM to email translation

8 SECURITY CONSIDERATION

In an Internet-based collaborative environment, such as CINEMA, there are many security issues that needs to be addressed. For example, secure web access, conference calls and emails, protection against malicious emails and calls, and unauthorized access to gateways and servers. All the modules in the system including the web access, SIP servers and SIP/PSTN gateway need to be protected.

The system can use existing transport layer security (TLS [15]) protocols for web access and conference calls. Digest authentication [26] prevents unauthorized access. For browsers that do not implement digest authentication, some protection can be provided using the Javascript-based implementation [36]. We store digest hash [53] of the password instead of clear-text.

Two security models are plausible. For users trying to authenticate with their home domain server, it's desirable that the users and the SIP server have shared secret or, use public-key-infrastructure if it exists. This model requires manually enrolling and removing users. The second mechanism employs cross-domain AAA. It's usually good for visitors temporarily using the resources in the visited domain. For example, when the user `alice@example.com` visits `visited.com`, the visited domain queries the AAA server, using RADIUS [52] or DIAMETER [17], in the `example.com` domain and ascertains that Alice is a valid user in her home domain. This approach requires some kind of roaming agreement between the collaborating domains.

Longer telephone PIN should be used to prevent guessing. Gateway's access control list (ACL) can further help is preventing unauthorized phone calls. Various access classes allow differential access to the gateway, e.g., restrict the students to only local calls, but allow the faculty to make long distance calls.

Various forms of programmable scripts such as CGI [39], CPL [40] and Servlet [49] can help reduce unwanted phone calls or instant message spams to the user through the SIP server. Moreover, users can configure their profile to authenticate all incoming calls. The primary limitation is that it requires the caller to be registered with the system. It should allow the unsafe scripts (CGI and Servlet are unsafe whereas CPL is safe) only to the trusted users.

The system can attempt to detect and report attacks by monitoring the frequency of denied access attempts on web as well as SIP servers.

9 CONCLUSIONS AND FUTURE WORK

We have discussed seamless integration between two types of collaboration modes: synchronous and asynchronous. The conference server and user agent in our CINEMA infrastructure allow synchronous multi-party multimedia collaboration via audio, video, instant message, screen sharing and shared web-browsing. The personalized user profile, calendaring, address book management, event and conference management, and system configuration can be done from the web interface. It also facilitates document sharing and asynchronous discussions among the group members. Moderators can monitor and control various synchronous and asynchronous activities. The messaging and notifications are used to reach the users when they are off-line.

A SIP-based architecture allows easily extending the infrastructure with new features, e.g., presence-enabled calls and programmable call routing. Interactive voice response provide easy access to the system from a telephone, whereas various text-to-speech tools allow interaction via plain email. This facilitates access to the system transparent to the end user device. Hence, we claim CINEMA to be a comprehensive multi-platform collaboration architecture. Moreover, the system allows hybrid interaction, e.g., phone for audio, PC for IM and document sharing in the same conference.

We have described the various collaboration tools in CINEMA and how they interact to achieve new services. Collaborative work is a vast research area incorporating numerous technologies such as networks, multimedia, object oriented concepts, virtual reality and artificial intelligence. Our aim is to complement these research innovations by providing a framework over which other collaboration tools can be built or integrated. Although, CINEMA's main focus is on real-time synchronous communication, we also correlate the two modes of collaboration for an enhanced end-user experience. CINEMA can be used within an organization as well as in portal mode by application service providers.

We have not implemented everything described in the paper. In particular, we are working on recording of conference events such as join or leave, programmable end system services, conference floor-control, performance measurement and improvement of the multimedia conference server, load balancing and scaling techniques for the servers, and optimizations for distributed multi-site collaboration. This paper is a continuation of our earlier work [34, 35, 70].

10 Acknowledgments

Wenyu Jiang and Sankaran Narayanan contributed in the core design and implementation of CINEMA. A number of other students have contributed to various components in the architecture as follows. Salman Baset implemented the event notification web interface. Joseph Gagliano helped in email notification to phone. Gaurav Khandpur implemented the web-based discussion forum. Ali Khwaja integrated the high-quality codec and sampling rate conversion in the conference server. Chin-hong Lin and Agung Suyono implemented the conference recording. Daniel Liu and Naho Ogasawara implemented the email-by-phone system. Gautam Nair helped in the initial implementation of the conference mixer. Ajay Nambi implemented some VoiceXML browser enhancements, voice-mail access and conference joining scripts. Sankaran Narayanan added TLS and IPv6 support. Eva Nautiyal and Manica Piputbundit helped in the implementation of phone announcement service. Anurag Pant integrated the text-to-speech support in our media server. Mark Pimentel helped in implementing the conference timeline display. Joe Rosen implemented automatic gain control in the server. Jeffrey Schnurmacher designed the initial web interface layout. Naoya Seta implemented the IM/call converter. Visda Vokhshoori and Sean Mandel helped in initial VoiceXML browser implementation. Huwei Zhang contributed in file-sharing and conference load balancing.

The work[2] is supported by grant from SIPquest Inc.

[2]More information about CINEMA is at http://www.cs.columbia.edu/IRT/cinema

References

[1] GnomeMeeting. http://www.gnomemeeting.org.

[2] GoToMyPC by Expert City, Inc. http://www.gotomypc.com/.

[3] Hyperwave. http://www.hyperwave.com.

[4] Lotus domino. http://www.lotus.com.

[5] MeetingPlace. http://www.meetingplace.net/.

[6] Opentext corporation. http://www.opentext.com/livelink.

[7] pcAnywhere by Symantec, Inc. http://www.symantec.com/pcanywhere.

[8] Procmail home-page. http://www.procmail.org/.

[9] VirtualPlaces. http://www.vplaces.com/vpnet/index.html.

[10] Wbd: Whiteboard from University College London. http://www-mice.cs.ucl.ac.uk/multimedia/software/wbd/.

[11] W. Appelt. WWW based collaboration with the BSCW system. In *SOFSEM (SOFtware SEMinar)*, pages 66–78, Milovy, Czech Republic, Nov. 1999. Springer-Verlag in the Lecture Notes in Computer Science 1725. http://bscw.gmd.de/Papers/SOFSEM99/sofsem.pdf.

[12] Association for Computing Machinery (ACM). ACM special interest group on supporting group work (SIGGROUP), 1996. http://www.acm.org/siggroup/.

[13] P. Balaouras, I. Stavrakakis, and L. Merakos. Potential and limitations of a teleteaching environment based on H.323 audio-visual communication systems. *Computer Networks*, 34(6):945–958, Dec. 2000.

[14] A. Black and K. Lenzo. *Flite: a small, fast run time synthesis engine*. Speech Group at Carnegie Mellon University, 1.0 edition, Aug. 2001. http://fife.speech.cs.cmu.edu/flite/.

[15] S. Blake-Wilson, M. Nystrom, D. Hopwood, J. Mikkelsen, and T. Wright. Transport layer security (TLS) extensions. RFC 3546, Internet Engineering Task Force, June 2003.

[16] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. Simple object access protocol (SOAP) 1.1. Technical report, World Wide Web Consortium, W3C, May 2000.

[17] P. Calhoun et al. Diameter base protocol. Internet draft, Internet Engineering Task Force, Jan. 2003. Work in progress.

[18] B. Campbell et al. Instant message sessions in SIMPLE. Internet draft, Internet Engineering Task Force, July 2003. Work in progress.

[19] N. Charlton, M. Gasson, G. Gybels, M. Spanner, and A. van Wijk. User requirements for the session initiation protocol (SIP) in support of deaf, hard of hearing and speech-impaired individuals. RFC 3351, Internet Engineering Task Force, Aug. 2002.

[20] CMU Sphinx group. CMU sphinx open source speech recognition engines, 2000. http://www.speech.cs.cmu.edu/sphinx/index.html.

[21] J. Conklin. Hypertext: An introduction and survey. In D. Marca and G. Bock, editors, *Groupware — software for computer-supported cooperative work*. IEEE Computer Society Press, 1992. IEEE Computer, September 1987.

[22] Dallas Semiconductor Corp. ibutton, 2002. http://www.ibutton.com.

[23] A. Dix. Computer-supported cooperative work - a framework. In *Design Issues in CSCW, Eds. D. Rosenburg and C. Hutchison*. Springer Verlag, 1994. http://www.comp.lancs.ac.uk/computing/users/dixa/papers/cscwframework94/.

[24] A. Dix. Challenges and perspectives for cooperative work on the web. In *An International workshop on CSCW and the Web*, Sankt Augustin, Germany, Feb. 1996. ERCIM/W4G. http://orgwis.gmd.de/projects/W4G/proceedings/challenges.html.

[25] H.-P. Dommel and J. J. Garcia-Luna-Aceves. Floor control for multimedia conferencing and collaboration. *Multimedia Systems*, 5(1):23–38, 1997.

[26] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. J. Leach, A. Luotonen, and L. Stewart. HTTP authentication: Basic and digest access authentication. RFC 2617, Internet Engineering Task Force, June 1999.

[27] N. Freed and N. Borenstein. Multipurpose Internet mail extensions (MIME) part one: Format of Internet message bodies. RFC 2045, Internet Engineering Task Force, Nov. 1996.

[28] N. Freed and N. Borenstein. Multipurpose Internet mail extensions (MIME) part two: Media types. RFC 2046, Internet Engineering Task Force, Nov. 1996.

[29] Y. Goland, E. Whitehead, A. Faizi, S. Carter, and D. Jensen. HTTP extensions for distributed authoring – WEBDAV. RFC 2518, Internet Engineering Task Force, Feb. 1999.

[30] S. Greenberg and M. Roseman. Groupweb: A web browser as real-time groupware. In *Conference on human factors in computing systems, companion, proceedings*, pages 271–272, Vancouver, Canada, Apr. 1996. ACM SIGCHI'96.

[31] M. Handel and J. Herbsleb. What is chat doing in the workplace. In *Proceedings of ACM Conference on computer supported cooperative work(CSCW)*, New Orleans, Louisiana, USA, Nov. 2002.

[32] M. Handley, C. E. Perkins, and E. Whelan. Session announcement protocol. RFC 2974, Internet Engineering Task Force, Oct. 2000.

[33] E. A. Isaacs and J. C. Tang. What video can and can't do for collaboration: a case study. In *ACM Multimedia*, pages 199–206, Anaheim, California, Aug. 1993.

[34] W. Jiang, J. Lennox, S. Narayanan, H. Schulzrinne, K. Singh, and X. Wu. Integrating Internet telephony services. *IEEE Internet Computing*, 6(3):64–72, May 2002.

[35] W. Jiang, J. Lennox, H. Schulzrinne, and K. Singh. Towards junking the PBX: deploying IP telephony. In *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Port Jefferson, New York, June 2001.

[36] P. Johnston. Javascript MD5. http://pajhome.org.uk/crypt/md5/.

[37] G. Kaiser and S. M. Kaplan. CSCW and software process. session summary in ninth international software process workshop: The role of humans in the process. In *Ninth International Software Process Workshop*, pages 9–11, Oct. 1994.

[38] V. Kumar. *MBone: Interactive Multimedia On The Internet*. Macmillan Publishing (Simon & Schuster), 1995.

[39] J. Lennox, H. Schulzrinne, and J. Rosenberg. Common gateway interface for SIP. RFC 3050, Internet Engineering Task Force, Jan. 2001.

[40] J. Lennox, X. Wu, and H. Schulzrinne. CPL: a language for user control of Internet telephony services. Internet draft, Internet Engineering Task Force, Aug. 2003. Work in progress.

[41] D. Liu and N. Ogasawara. *Email by phone using VoiceXML*. Columbia University, New York, May 2001.

[42] R. Mahy. A message summary and message waiting indication event package for the session initiation protocol (SIP). Internet draft, Internet Engineering Task Force, July 2003. Work in progress.

[43] L. Masinter. The "data" URL scheme. RFC 2397, Internet Engineering Task Force, Aug. 1998.

[44] S. McCanne and V. Jacobson. vic: A flexible framework for packet video. In *ACM Multimedia*, Nov. 1995.

[45] S. McGlashan, D. Burnett, J. Carter, S. Tryphonas, J. Ferrans, A. Hunt, B. Lucas, and B. Porter. Voice extensible markup language (voicexml) version 2.0. Technical report, World Wide Web Consortium (W3C), Feb. 2003. http://www.w3.org/TR/voicexml20/.

[46] M. Mühlhäuser. Interdisciplinary development of an electronic class and conference room. *Journal of Universal Computer Science (J.UCS)*, 2(10):694–710, Oct. 1996.

[47] MySQL AB Co. MySQL home page. http://www.mysql.com.

[48] J. Ott. Teleconferencing in the ITU-T. In *IETF*, San Jose, California, Dec. 1994. Multiparty Multimedia Session Control WG (MMusic), Talk (c).

[49] J. C. Process. SIP servlet API. Java Specification Requests JSR 116, Java Community Process, May 2002.

[50] P. V. Rangan and D. C. Swinehart. Software architecture for integration of video services in the etherphone environment. *IEEE Journal on Selected Areas in Communications*, 9(9):1395–1404, Dec. 1991.

[51] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, January/February 1998.

[52] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote authentication dial in user service (RADIUS). RFC 2865, Internet Engineering Task Force, June 2000.

[53] R. Rivest. The MD5 message-digest algorithm. RFC 1321, Internet Engineering Task Force, Apr. 1992.

[54] A. B. Roach. Session initiation protocol (sip)-specific event notification. RFC 3265, Internet Engineering Task Force, June 2002.

[55] D. Robinson and K. Coar. The common gateway interface (CGI) version 1.1. Internet Draft draft-coar-cgi-v11-04.txt,.ps,, Internet Engineering Task Force, Oct. 2003. Work in progress.

[56] J. Rosenberg. A framework for conferencing with the session initiation protocol. Internet draft, Internet Engineering Task Force, May 2003. Work in progress.

[57] J. Rosenberg. A presence event package for the session initiation protocol (SIP). Internet draft, Internet Engineering Task Force, Jan. 2003. Work in progress.

[58] J. Rosenberg and H. Schulzrinne. Models for multi party conferencing in SIP. Internet draft, Internet Engineering Task Force, July 2002. Work in progress.

[59] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. R. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: session initiation protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[60] S. Sarin. Computer-based real-time conferencing systems. *IEEE Computer*, 7(10):33–45, Oct. 1985.

[61] E. Schooler, S. Casner, and J. B. Postel. Multimedia conferencing: Has it come of age? In *24th Hawaii International Conference on System Science*, volume 3, pages 707–716, Hawaii, Jan. 1991. IEEE.

[62] H. Schulzrinne. Conferencing and collaborative computing. In *Dagstuhl Seminar on Fundamentals and Perspectives of Multimedia Systems*, Dagstuhl Castle, Germany, July 1994.

[63] H. Schulzrinne. is-composing indication for instant messaging using the session initiation protocol (SIP). Internet draft, Internet Engineering Task Force, Feb. 2003. Work in progress.

[64] H. Schulzrinne. RPIDS – rich presence information data format for presence based on the session initiation protocol (SIP). Internet draft, Internet Engineering Task Force, July 2003. Work in progress.

[65] H. Schulzrinne and K. Arabshian. Providing emergency services in Internet telephony. *IEEE Internet Computing*, 6:39–47, May 2002.

[66] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: a transport protocol for real-time applications. RFC 1889, Internet Engineering Task Force, Jan. 1996.

[67] H. Schulzrinne and S. Petrack. RTP payload for DTMF digits, telephony tones and telephony signals. RFC 2833, Internet Engineering Task Force, May 2000.

[68] H. Schulzrinne, A. Rao, and R. Lanphier. Real time streaming protocol (RTSP). RFC 2326, Internet Engineering Task Force, Apr. 1998.

[69] J. Schwartz. Collaboration: More hype than reality. *InternetWeek (online newsletter)*, Oct. 1999. http://www.internetweek.com/trans/tr99-bp1.htm.

[70] K. Singh, W. Jiang, J. Lennox, S. Narayanan, and H. Schulzrinne. CINEMA: columbia internet extensible multimedia architecture. technical report CUCS-011-02, Department of Computer Science, Columbia University, New York, New York, May 2002.

[71] K. Singh, G. Nair, and H. Schulzrinne. Centralized conferencing using SIP. In *Internet Telephony Workshop*, New York, Apr. 2001.

[72] K. Singh, A. Nambi, and H. Schulzrinne. Integrating VoiceXML with SIP services. In *Conference Record of the International Conference on Communications (ICC)*, May 2003.

[73] K. Singh and H. Schulzrinne. Unified messaging using SIP and RTSP. In *IP Telecom Services Workshop*, pages 31–37, Atlanta, Georgia, Sept. 2000.

[74] M. Stefik, G. Foster, D. G. Bobrow, K. Kahn, S. Lanning, and L. Suchman. Beyond the chalkboard: computer support for collaboration and problem solving in meetings. *Communications ACM*, 30(1):32–47, Jan. 1987.

[75] J. Toga and J. Ott. ITU-T standardization activities for interactive multimedia communications on packet-based networks: H.323 and related recommendations. *Computer Networks and ISDN Systems*, 31(3):205–223, Feb. 1999.

[76] UCB/LBNL. vic – video conferencing tool. http://www-nrg.ee.lbl.gov/vic/.

[77] UCL Multimedia. Robust audio tool (RAT). http://www-mice.cs.ucl.ac.uk/multimedia/software/rat/.

[78] X. Wu et al. Use of session initiation protocol (SIP) and simple object access protocol (SOAP) for conference floor control protocol (SOAP) for conference floor control. Internet draft, Internet Engineering Task Force, Mar. 2003. Work in progress.

[79] X. Wu and H. Schulzrinne. Programmable end system services using SIP. In *Conference Record of the International Conference on Communications (ICC)*, May 2003.