# Unified Messaging using SIP and RTSP*

Kundan Singh and Henning Schulzrinne
Dept. of Computer Science
Columbia University
New York, USA
{kns10,hgs}@cs.columbia.edu

October 11, 2000

**Abstract**

Traditional answering machines and voice mail services are closed systems, tightly coupled to a single end system, the local PBX or local exchange carrier. Even simple services, such as forwarding voice mail to another user outside the local system, are hard to provide. With the advent of Internet telephony, we need to provide voice and video mail services. This also offers the opportunity to address some of the shortcomings of existing voice mail systems.

We list general requirements for a multimedia mail system for Internet telephony. We then propose an architecture using SIP (Session Initiation Protocol) and RTSP (Real-Time Streaming Protocol) and compare various alternative approaches to solving call forwarding, reclaiming and retrieval of messages. We also briefly describe our prototype implementation.

## 1 Introduction

Current voice mail systems are closed. It is hard to perform simple operations, like forwarding voice mail outside the local PBX, filtering or sorting messages. Configuration is tedious, e.g., one can not readily switch between a set of outgoing messages. Moreover, voice mail and call answering services are implemented as stand-alone proprietary systems. The service must be provided by the PBX, local phone company or the local handset or one must obtain a separate voice mail number.

*This work was supported by a grant from Sylantro Corp. An earlier version of this paper appeared in the *IP Telecom Services Workshop*, Sept 2000, Atlanta, Georgia.

On the other hand, Internet protocols, such as electronic mail for internet messaging and SIP (Session Initiation Protocol [1, 2]) for Internet telephony, have an open architecture. Configuration is simple and the protocols are extensible. There can be a separation between the internet service provider and the messaging or telephony service provider. Internet telephony is likely to replace the old telephone systems in the near future, particularly in corporate and institutional environments. So, it is important to design a voice mail system for Internet telephony.

We have built a modular Internet voice/video mail system using existing Internet protocols, in particular, SIP and RTSP (Real-Time Streaming Protocol [3]), that allows users message access from any Internet connected device, using standard media players or user agents. SIP is used for setting up multimedia calls over the Internet. RTSP controls the delivery of streaming media and provides facilities to play back, record, or perform other operations on multimedia content. We propose to use a SIP-PSTN gateway to access the voice mail service from a PSTN phone.

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

Having a SIP interface to the multimedia mail system makes it readily usable in Internet telephony. A SIP proxy/redirect server can be configured to forward the call to voice mail depending on caller address, time of day, etc.

### 1.1 Outline of the rest of the paper

The remainder of the paper is organized as follows. In Section 2, we list the general requirements for an Internet multimedia mail service. Section 3 describes and discusses the limitations of existing voice messaging systems. In Section 4, we describe a mechanism to use SIP and RTSP to provide voice (and multimedia) messaging. Section 5 gives an insight into our implementation. Finally, we describe future work in Section 6.

## 2 Requirements

In this section we list the requirements for a simple multimedia mail system.

**Recording and playback:** The system must be able to record the multimedia message. A user should be able to play back the message intended for him. System should provide security and privacy of messages.

**Ease of access:** The system should provide a framework in which a user can easily browse through his messages. In other words, user should be able to skip a complete message or part of it, read only the header for a message, pause, delete, rewind or fast-forward a message. It should be possible to have different configurable mail folders. The system should provide a web interface for accessing the messages.

**Telephony interface:** The system should be able to support DTMF (Dual Tone Multi Frequency) based navigation and message retrieval. This allows the messages to be accessed using PSTN phones. It should be possible to use the voice mail system in an Internet telephony environment as well as in a traditional PSTN environment.

**Notification:** The system must be able to notify the user on arrival of a new message. The notification may be sent using electronic mail, instant messaging, or any other suitable means. It should handle different message priorities, for example, messages from important people trigger a notification to a pager, while the remainder just generate regular email.

**Call reclaiming:** The system should support reclaiming the call by the callee, i.e., if the callee picks up the phone while the caller is recording the message, he should be able to talk to the caller.

**Unified messaging:** The system should handle voice messages as attachments in emails. It should be possible to forward a multimedia mail. For voice-only messages, VPIM [4, 5] should be used. This helps in integrating it with the existing Internet mail infrastructure.

**Scaleability:** The system should be scaleable and should be able to handle thousands of users. It should also allow redundant servers.

**Protocol compliance:** The system must be compatible with existing Internet multimedia protocols, and should be designed with little or no modification to the current infrastructure.

Other, less important, requirements include support for mailing lists, fax, text-to-speech conversion and automatic expiration of messages.

## 3 Related work

There is a fair amount of early messaging work, in particular, the Etherphone work done at Xerox PARC [6, 7, 8]. Profiles have been defined for Internet messaging to support voice. In particular, VPIM [5] supports the interchange of voice messages between voice mail systems, unified messaging systems, email servers and desktop client applications. The basic architecture is to carry the voice attachments in the electronic mail. None of these address the integration of Internet telephony with the voice messaging system. Moreover, carrying the voice bits across low-bandwidth links while forwarding the messages is not desirable. It also requires special-purpose client applications which can understand the profiles.

Various schemes have been proposed to forward a call to a voice mail server in SIP-based Internet telephony systems. The Common Gateway Interface for SIP [9] or the Call Processing Language [10, 11] can be used to configure the SIP server to use an external voice mail service. Campbell and Sparks [12] suggest the use of SIP Request-URI to carry service control information related to voice mail. However, integration of SIP based Internet telephony with the voice mail service is still an active research topic.

## 4 Architecture

The multimedia mail system, we propose, uses a voice mail server, an RTSP media server, and possibly a SIP proxy server. The RTSP media server is a storage server which handles the recording and playback of messages. The voice mail server is a SIP interface to the media server, to allow connections from a SIP user agent. These components can be located in different domains and operated by different entities. Moreover, since a SIP server can handle many more users than a typical multimedia server, this separation allows distributed message storage accross a number of RTSP media servers.

A SIP server acts as an address-look-up server. SIP user agents register their current locations with the SIP server and the server either informs the current location to any subsquent caller or forwards any subsequent call to the current location. A SIP server can operate in either redirect or proxy mode. In redirect mode the server redirects the caller to the current location of the registered user. The caller, then calls the new address. In proxy mode the server acts as an intermediate node between the caller and the callee. It forwards the call request to the current location of the registered user, and the response back to the caller. Multiple locations can be registered for a single user, for instance, if the user has many SIP phones. The SIP server, in proxy mode, forwards the call request to all the registered locations.

If the user picks up one of the phones, the server cancels branches to all the other phones. We have used Columbia SIP server, sipd, which supports both modes of operation.

Fig. 1 shows an example of recording voice mail. A SIP server handles all the users in a particular domain, e.g., cs.columbia.edu. Different users register their current location with the SIP server, so that the server can contact the user on receipt of an incoming call. The voice mail server also registers its location on behalf of all the users it is serving. From the SIP server's perspective, there are two active locations for every user, one is his actual SIP based phone and the other is the voice mail server.

When a user Alice, (`alice@home.com`) calls Bob, `bob@cs.columbia.edu`, the SIP server proxies the call to both locations. If the user picks up the phone, the branch to the voice mail server is cancelled and a normal SIP call proceeds between `Alice` and `Bob`.

The voice mail server is configured to wait for some time, say 10 seconds, before accepting the call. So, if `Bob` does not pick up the phone in 10 seconds, the voice mail server is going to accept the call on his behalf. Before accepting the call, the voice mail server sets up the media path with the RTSP server. It sends an RTSP SETUP message to the RTSP server to play back the voice prompt to `Alice` for leaving a voice message. The voice prompt for the outgoing message can be generated using a recorded media file, or, if configured, by converting the text of `Bob`'s vacation message to speech. The voice mail server sends another SETUP message to the RTSP server to record the message.

Once the caller has finished recording, he hangs up, triggering a SIP BYE request. The voice mail server informs the RTSP server to stop recording.

Media data for the outgoing and the recorded message is exchanged directly between the caller (`Alice`) and the RTSP media server using RTP (Real-time Transport Protocol [13]).

The figure does not show how the callee is notified about the arrival of voice messages. The multimedia mail server can notify `Bob` about the new message using electronic mail, or some other means, e.g., instant messaging. SIP can also be used to notify `Bob`'s user agent about a new message using the MESSAGE method [14], or using SUBSCRIBE and NOTIFY methods [15, 16].

There are various ways to forward an incoming call to a multimedia mail server:

- One approach is to use the parallel registration in the SIP proxy, as described in the previous example. Having the multimedia mail server register with the SIP server on behalf of the user is very simple and does not need any intelligence in the SIP user agent or the SIP server. However, there is a race

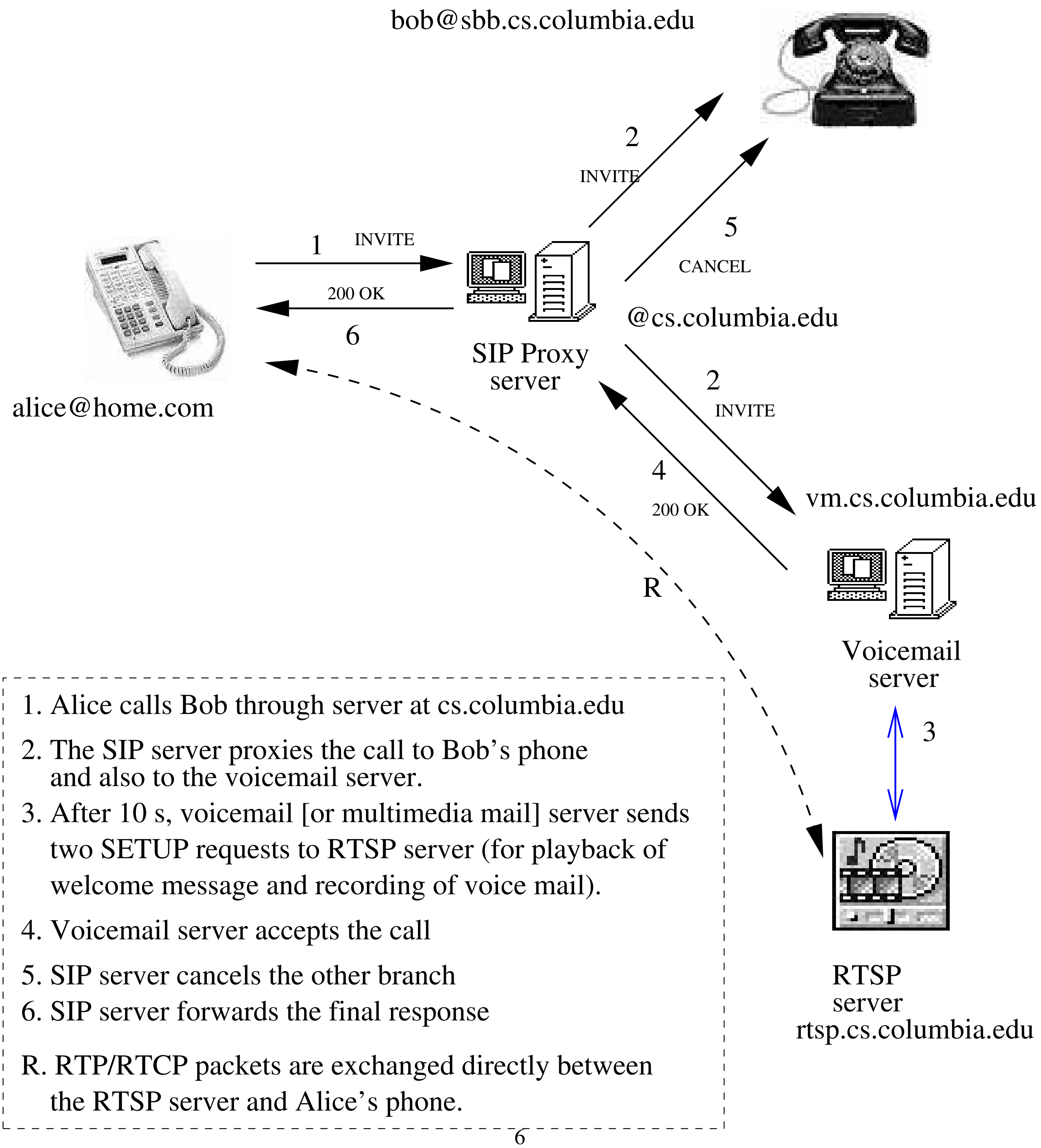




Figure 1: Forwarding the call to voice mail

condition, as to whether the user `Bob` or the voice mail server picks up the call first. If both pick up the call at approximately the same instant, `Alice` will receive two final responses. It is up to the caller to keep one or both the call legs. The response should indicate whether it is from a multimedia mail system or a human user. This will help the caller's user agent automatically send SIP BYE request to one of the call legs.

This approach does not distinguish between a busy callee and no response from callee. In either case the multimedia mail server will wait before accepting the call. This might be desirable if the callee's user agent implements call waiting service.

- `Bob`'s SIP user agent can be configured to redirect any incoming call to the multimedia mail server if there is no response for 10 seconds or if `Bob` is already in a call. A different request-URI can be used to specify the purpose of calling the multimedia mail server [12]. For instance, `bob-deposit@vm.cs.columbia.edu` can be used to leave a message, and `bob-6532-retrieve@vm.cs.columbia.edu` to retrieve the message number 6532. The advantage of this approach is to allow per-user configuration of the service. The disadvantage is that `Bob` won't be able to receive any message if his phone is dead. Secondly, such intelligence in an user agent is not always possible, particularly in low cost SIP enabled embedded devices.

- Another approach is to use CPL or sip-cgi. An example CPL script is shown in Fig. 2. This approach allows a more precise control over the service. `Bob` can now selectively forward the call to his voice mail depending on caller address, time of day, etc. Using CPL seems to be the best approach, providing much finer control on the call processing, however it needs CPL enabled servers.

- The SIP proxy server may be hard-configured to forward the call to a voice mail address if the callee is busy or there is no response.

### 4.1 Voice mail server

The voice mail server has both SIP and RTSP parts. On one side it can receive Internet telephony calls using SIP, and on the other side it behaves as a RTSP client and can perform playback, recording and other control on the multimedia mail residing at the remote RTSP server.

The RTSP server acts as a storage server for the multimedia mails. Separating the voice mail server from the storage server helps in building scaleable systems.

```
<?xml version="1.0" ?>
<!DOCTYPE cpl SYSTEM "cpl.dtd">

<cpl>
 <subaction id="voicemail">
  <location url=
    "sip:bob@vm.cs.columbia.edu">
   <redirect />
  </location>
 </subaction>

 <incoming>
  <address-switch field="origin"
     subfield="host">
   <address
     subdomain-of="cs.columbia.edu">
    <location url=
      "sip:bob@sbb.cs.columbia.edu">
     <proxy>
      <busy>
        <sub ref="voicemail" />
      </busy>
      <noanswer>
        <sub ref="voicemail" />
      </noanswer>
      <failure>
        <sub ref="voicemail" />
      </failure>
     </proxy>
    </location>
   </address>
   <otherwise>
    <sub ref="voicemail" />
   </otherwise>
  </address-switch>
 </incoming>
</cpl>
```

Figure 2: CPL script for forwarding a call to voice mail

For example, a single voice mail server can serve all students of an university, while using the departmental RTSP servers for load balancing. Since the voice mail server does not have to handle the media stream, processing speed is not a bottle-neck.

POP3 [17] and IMAP (Internet Message Access Protocol [18, 19]) are not used directly because they do not support media streaming. One can implement a POP3 or IMAP interface to fetch the voice message similar to text based electronic mails.

## 4.2 Retrieving voice mail

Our system offers four choices for retrieving multimedia mail messages: RTSP streaming media tool, SIP user agent, email attachment or web page.

Existing RTSP based media players can be used to directly play the voice messages from the RTSP server. For instance, the URI `rtsp://rtsp.cs.columbia.edu/bob/inbox/6532.au` can be used to retrieve the message number 6532 from the RTSP server for user `Bob`.

`Bob` can also use his SIP phone and call the URI `sip:bob-6532-retrive@vm.cs.columbia.edu` to retrieve his message. The call is received by the voice mail server which in turn contacts the RTSP media server and retrieves the message. The media data for the message is directly sent from the RTSP server to `Bob`'s SIP phone. This approach also allows for retrieval of voice messages from a PSTN phone through a SIP-PSTN gateway.

Alternatively, the multimedia mail server can be configured to send the message as an attachment to `Bob`'s email address.

The most preferred approach is to access the voice mail from a web page using a web browser.

## 4.3 Telephony interface

With the existing widespread use of PSTN, it is likely that most of the voice mail systems serve the telephone users, and not the Internet users. This places a new challenge to allow configuration, and retrieval of voice mails from a telephone using DTMF commands.

A SIP-PSTN gateway can be used to provide the telephony access to the voice mail server as shown in Fig. 3. We need another module, a DTMF-to-RTSP translator, which will translate the DTMF commands to the appropriate RTSP messages to be sent to the storage server. The gateway converts DTMF into special RTP packets [20], so that the DTMF-to-RTSP translator does not have to perform audio analysis. The media stream from the gateway goes to the RTSP server through the translator. The translator receives the signaling information for the call from the voice mail server.

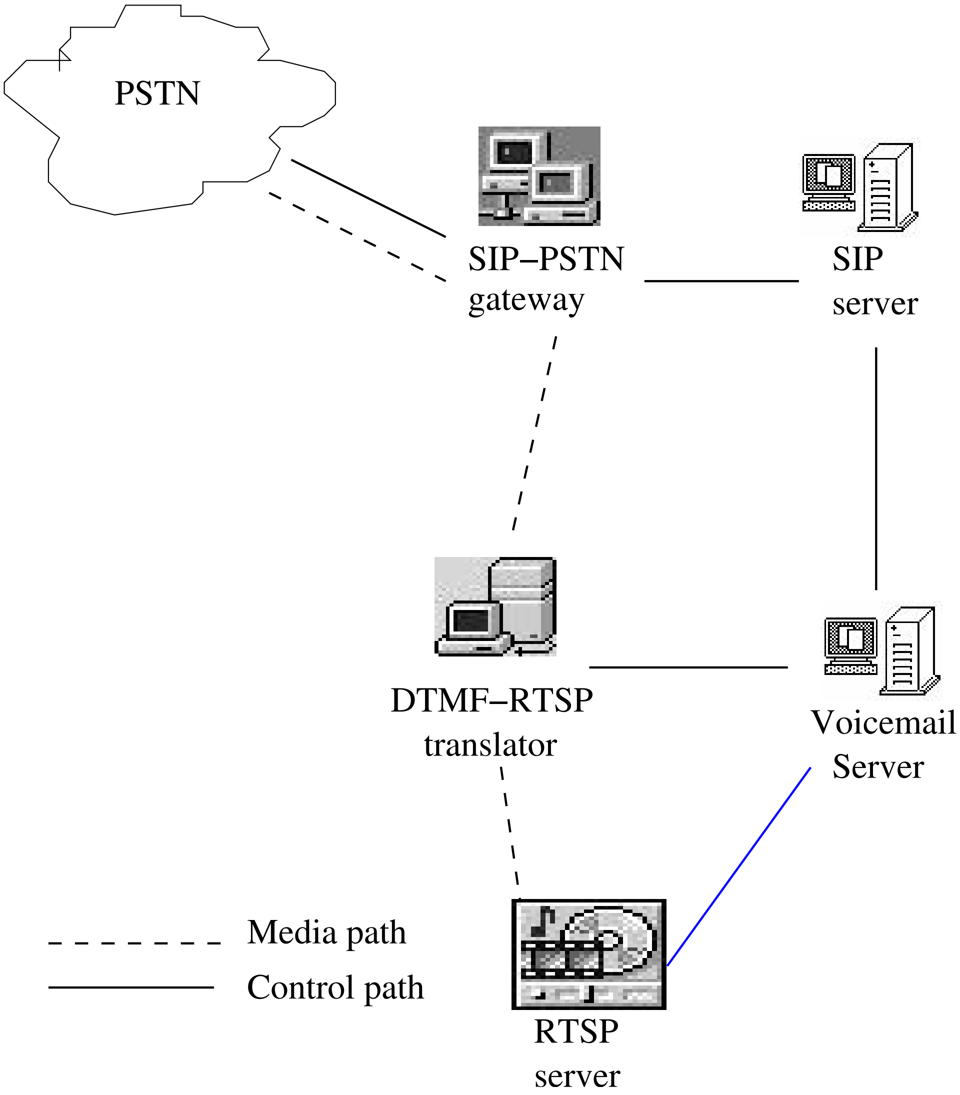


Figure 3: PSTN access to voice mail

It has also been proposed to send DTMF using SIP INFO messages. However, that will not change the basic architecture described here.

### 4.4 Call reclaiming

Another implicit requirement for the voice mail system is to allow reclaiming an already transferred call. If the callee arrives and picks up the phone when the voice mail is being recorded, the system should provide an option for the users to stop the recording and continue talking in a normal call. This is not trivial if the voice mail system is not part of the user agent.

One approach is to use SIP call control [21] to support call reclaiming. In the previous example (in Fig. 1), when the call gets transferred to the voice mail server, the voice mail server invites the intended user, `bob@sbb.cs.columbia.edu`, in the existing call. If `Bob` picks up the phone while the voice mail is being recorded, he joins the existing call to form a three party conference between the caller (`Alice`), the voice mail server and himself. The voice mail server then drops out of the conference by sending a SIP BYE. If `Bob` does not pick up the phone, the voice mail server cancels the call once the message from `Alice` has been recorded. However, it is not clear how the voice mail server can call `Bob` without having the SIP server fork a branch back to the voice mail server. One can extend the caller preference [22] to include a description of the user agent picking up the phone.

Another approach is to use third party call control, with the voice mail server as the third party. It simply sends an INVITE to `Bob`, with `Alice`'s session description. If `Bob` picks up, it also changes `Alice`'s session description via re-INVITE, so that the two now talk directly to each other media-wise. To avoid any confusion to `Bob`, the voice mail server may prompt him that `Alice`'s message was being recorded.

A third approach uses call state notification. `Bob` subscribes to call events from the voice mail server and can INVITE himself to the call. This requires further study.

It might be desirable to have the user decide whether to stop the recording or not. The caller may not want to repeat the long message if he has already recorded most of it.

The voice mail server uses the SIP request-URI to identify the purpose of the call. For instance, if the call is directly made to the voice mail server to leave an announcement or a reminder in user's mail box, the server should not try to contact the intended recipient.

### 4.5 Deletion of messages

The architecture assumes that the RTSP media server stores the multimedia messages. However, there is no explicit mechanism to delete a resource in RTSP, in its current form.

One option is to define a new method, say DELETE, to delete a resource or a media file on the RTSP server.

The other approach is to pretend as if you are recording the file, but terminate the RTSP connection without actually recording anything. To be more specific, an RTSP SETUP with record mode is sent to the server, immediately followed by an RTSP TEARDOWN, without sending a RECORD message. Our RTSP server interprets this as a command to delete the file. Even otherwise, the recorded file will be empty, and of no use.

While the first method is explicit, it requires modifying RTSP. We have implemented both approaches.

# 5 Implementation

We have successfully demonstrated an Internet voice mail system based on the above architecture using our SIP and RTSP servers, sipd and rtspd respectively. The system is likely to serve the Computer Science Department at Columbia University, replacing the existing PSTN based voice mail and answering machines. In this section we discuss some of the implementation features.

The voice mail server registers with the SIP server on behalf of every user, as discussed earlier. This allows for centralized configuration at the server, when serving different types of user agents (Desktop based client or SIP enabled ethernet phone device). The voice mail server sends email notification to the intended recipient regarding new voice mail arrivals. The template of the notification mail is configurable. This allows the user to modify the email message format as per his taste. For instance, user may just put a clickable RTSP URL in his email notification if he prefers to use a RTSP client, or the HTTP link to his voice mail inbox, or both.

Every user is given a voice mail account and a voice mail web page. The web interface is similar to existing web based mail services, like Hotmail. Fig. 4 shows a screen dump of the system.

Each user has a directory storing a configuration file and the voicemail messages. A unique URI is assigned to every user, for example, `alice@voicemail` is assigned to a user with login id `alice`. The voice mail server, when started, registers all these URIs with the SIP server.

The web interface is through CGI (Common Gateway Interface) and is written in Tcl (Tool Command Language).

Basic features like folder management, password change, customizing the voice response, deleting messages and sorting the messages based on different parame-

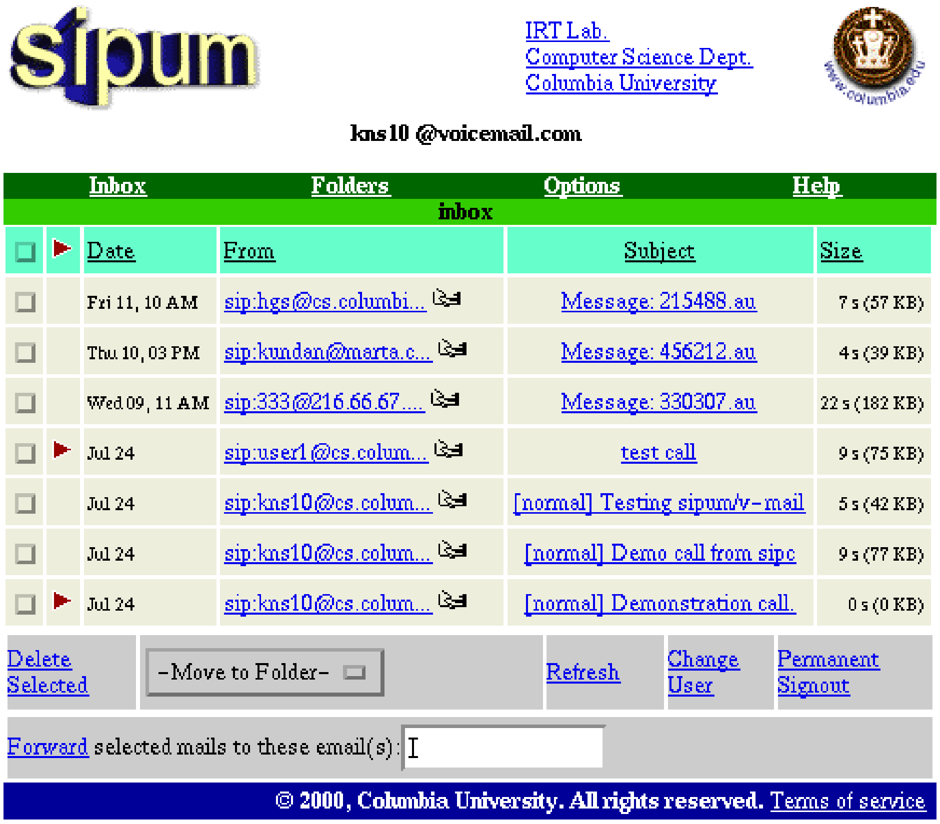


Figure 4: Example voice mail inbox

ters (e.g., date, subject, size) are already implemented. Forwarding of the voice message as a MIME (Multipurpose Internet Mail Extensions [23, 24, 25, 26, 27]) attachment to electronic mail is also supported. This helps in integration of electronic mail and voice mail services.

Our multimedia mail implementation can be accessed at http://www.cs.columbia.edu/vmail.

## 6 Conclusion and future work

We have described a multimedia mail architecture for Internet telephony, using SIP and RTSP, and shown how it meets the general requirements for a voice mail service. The architecture applies to any kind of multimedia mail, including video, because both SIP and RTSP are designed to support multimedia. Use of these protocols simplifies deployment, as related tools (SIP user agents, RTSP media players) are already available.

Various approaches are possible to utilize the voice mail service in the Internet telephony environment. Applicability ranges from a single user subscribed to a voice mail service to a whole university using the campus wide service. Separation of the voice mail server from the signaling and the storage servers helps in building scaleable systems.

We have also described some of the protocol issues, in particular, reclaiming a transferred call using SIP call control and deleting a mail using RTSP methods.

We have developed a prototype voice mail system, and will continue towards further university-wide deployment of the system. Other implementation issues related to DTMF based voice mail access and video mail will be addressed in the future versions of our system. We can easily include fax and email messages in the web mailbox, similar to some of the IMAP (Internet Message Access Protocol [18, 19]) web interfaces.

We are currently investigating the integration of voice mail, answering machine, electronic mail, MIME attachments for various media types, Internet telephony and other Internet-based services.

## 7 Acknowledgments

We would like to thank Tarun Kapoor for his help in the implementation and discussion, and Jonathan Lennox, who implemented the SIP proxy server sipd and its sip-cgi support.

## References

[1] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: session initiation protocol," Request for Comments 2543, Internet Engineering Task Force, Mar. 1999.

[2] H. Schulzrinne and J. Rosenberg, "Internet telephony: Architecture and protocols – an IETF perspective," *Computer Networks and ISDN Systems*, vol. 31, pp. 237–255, Feb. 1999.

[3] H. Schulzrinne, A. Rao, and R. Lanphier, "Real time streaming protocol (RTSP)," Request for Comments 2326, Internet Engineering Task Force, Apr. 1998.

[4] G. Vaudreuil and G. Parsons, "Voice profile for internet mail - version 2," Request for Comments 2421, Internet Engineering Task Force, Sept. 1998.

[5] G. Vaudreuil and G. Parsons, "Voice profile for internet mail - version 3," Internet Draft, Internet Engineering Task Force, Feb. 1999. Work in progress.

[6] D. B. Terry and D. C. Swinehart, "Managing stored voice in the Etherphone system," *ACM Transactions on Computer Systems*, vol. 6, pp. 3–27, Feb. 1988.

[7] P. T. Zellweger, D. B. Terry, and D. C. Swinehart, "An overview of the Etherphone system and its applications," in *Proc. of 2nd IEEE Conference on Computer Workstations*, (Santa Clara, California), pp. 160–168, Mar. 1988.

[8] 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*, vol. 9, pp. 1395–1404, Dec. 1991.

[9] J. Lennox, J. Rosenberg, and H. Schulzrinne, "Common gateway interface for SIP," Internet Draft, Internet Engineering Task Force, May 1999. Work in progress.

[10] J. Lennox and H. Schulzrinne, "CPL: a language for user control of internet telephony services," Internet Draft, Internet Engineering Task Force, Mar. 1999. Work in progress.

[11] J. Rosenberg, J. Lennox, and H. Schulzrinne, "Programming internet telephony services," *IEEE Network*, vol. 13, pp. 42–49, May/June 1999.

[12] B. Campbell and R. Sparks, "Control of service context using SIP Request-URI," Internet Draft, Internet Engineering Task Force, Jan. 2000. Work in progress.

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

[14] J. Rosenberg *et al.*, "SIP extensions for instant messaging," Internet Draft, Internet Engineering Task Force, June 2000. Work in progress.

[15] J. Rosenberg *et al.*, "SIP extensions for presence," Internet Draft, Internet Engineering Task Force, June 2000. Work in progress.

[16] R. Mahy and I. Slain, "SIP extensions for message waiting indication," Internet Draft, Internet Engineering Task Force, July 2000. Work in progress.

[17] J. Myers and M. Rose, "Post office protocol - version 3," Request for Comments 1939, Internet Engineering Task Force, May 1996.

[18] M. Crispin, "Internet message access protocol - version 4," Request for Comments 1730, Internet Engineering Task Force, Dec. 1994.

[19] M. Crispin, "Internet message access protocol - version 4rev1," Request for Comments 2060, Internet Engineering Task Force, Dec. 1996.

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

[21] H. Schulzrinne and J. Rosenberg, "SIP call control services," Internet Draft, Internet Engineering Task Force, June 1999. Work in progress.

[22] H. Schulzrinne and J. Rosenberg, "SIP caller preferences and callee capabilities," Internet Draft, Internet Engineering Task Force, July 2000. Work in progress.

[23] N. Freed and N. Borenstein, "Multipurpose internet mail extensions (MIME) part one: Format of internet message bodies," Request for Comments 2045, Internet Engineering Task Force, Nov. 1996.

[24] N. Freed and N. Borenstein, "Multipurpose internet mail extensions (MIME) part two: Media types," Request for Comments 2046, Internet Engineering Task Force, Nov. 1996.

[25] K. Moore, "MIME (multipurpose internet mail extensions) part three: Message header extensions for Non-ASCII text," Request for Comments 2047, Internet Engineering Task Force, Nov. 1996.

[26] N. Freed, J. Klensin, and J. Postel, "Multipurpose internet mail extensions (MIME) part four: Registration procedures," Request for Comments 2048, Internet Engineering Task Force, Nov. 1996.

[27] N. Freed and N. Borenstein, "Multipurpose internet mail extensions (MIME) part five: Conformance criteria and examples," Request for Comments 2049, Internet Engineering Task Force, Nov. 1996.