The Session Initiation Protocol: Providing Advanced Telephony Services Across the Internet

Henning Schulzrinne*
Networking Techniques Research Department
Bell Laboratories
101 Crawfords Corner Rd
Rm. 4F-523
Holmdel, NJ 07733
phone: (732) 949-8344
fax: (732) 949-9118
hgs@cs.columbia.edu

Jonathan Rosenberg
High Speed Networks Research Department
Bell Laboratories
101 Crawfords Corner Rd.
Rm. 4C-526
Holmdel, NJ 07733
phone: (732) 949-6418
fax: (732) 834-5379
jdrosen@bell-labs.com

September 13, 1998

Abstract

Over the past few years, Internet telephony has evolved from toy for the technically savvy to a technology that may well replace the existing circuit-switched telephone network over the long run. To support widespread use of Internet telephony, a host of standardized protocols are required. These include protocols to ensure quality of service (QoS), transport audio and video data, provide directory services and enable signaling. Signaling protocols are of particular interest as they are the basis for providing advanced services, such as mobility, universal numbers, multi-party conferencing, voice mail and automatic call distribution.

Two protocols have emerged to fill this need: the ITU-T H.323 suite of protocols, and the IETF Session Initiation Protocol (SIP). We examine the usage of SIP for Internet telephony. We provide an overview of the protocol and its architecture, and then describe how it can be used to provide some advanced services. We then discuss some of the strengths of SIP: simplicity, scalability, extensibility,

*Also Prof. of Computer Science at Columbia University

and modularity, and argue why these are critical components for an IP telephony signaling protocol. We also indicate how SIP can be used not just for end-to-end IP services, but also to control existing phone services.

1 Introduction

The term Internet telephony has evolved to infer a range of different services. In general, it refers to the transport of real-time media, such as voice and video, over the Internet, to provide interactive communications between one or more parties. The parties involved may access the Internet via a PC, a standalone IP enabled device, or even by dialing up to a gateway from a traditional PSTN handset.

The advantages of IP telephony are quite sweeping [1]. It can offer high quality voice, improved multiplexing gains (an inherent advantage with any packet switched network), rich computer telephony integration, advanced services, an open market for providers, and reduced cost.

Supporting Internet telephony requires a range of protocols, including those needed for data transport (the Real Time Transport Protocol (RTP) [2]), Quality of Service (QoS) establishment and management (The Resource Reservation Protocol (RSVP) [3, 4], YESSIR [5]), billing and authentication (DIAMETER [6], RADIUS [7, 8]), gateway discovery [9], and directory access (Lightweight Directory Access Protocol (LDAP) [10], among others. In addition, Internet telephony requires a means for prospective communications partners to find each other and to signal to the other party their desire to communicate. We refer to this functionality as *Internet telephony signaling.*

We see Internet telephony signaling specifically encompassing a number of functions:

Name translation and user location involves the mapping between names which identify a callee and the eventual location of the callee, be it a telephone number, pager, voicemail, email address, or web page. This translation and location function can be more complex that a simple database lookup, but may depend on caller and callee preferences, media and codec support at various points of presence, and service provider or third party logic.

Feature negotiation allows a group of participants to agree on what media to exchange, and their respective parameters. In a multiparty IP telephony conference, the set and type of media need not be uniform. Different participants can exchange different media types with each other; some may only receive audio, while others may receive audio and video. Such a mix can occur due to limited computing facilities at an endpoint, desire to use a particular media format, or lack of a single media format common to all participants.

Call participant management allows any call participant to invite new users into an existing call and terminate associations with other participants. During the call, participants should be able to transfer and hold other participants.

Call feature changes make it possible to adjust the composition of media sessions during the course of a call, either because the participants require additional or reduced functionality or because of constraints imposed or removed by the addition or removal of call participants.

Two protocols have emerged to provide these functions, the ITU-T H.323 Series [11], and IETF Session Initiation Protocol (SIP) [12]. In this paper, we present SIP, and describe its use for various advanced services. See [13] for a description of how these services compare with H.323. The remainder of this paper is organized as follows. In section 2, we introduce SIP at a high level and present is basic functionality. In section 3, we describe several network services enabled by SIP. In section 4 we describe how SIP supports client to client services. Section 5 overviews the benefits of SIP.

2 Overview of SIP

SIP is used to establish, change, and destroy calls between one or more endpoints in an IP based network. It is based heavily on some of the most successful protocols to come out of the Internet Engineering Task Force (IETF), the group which standardizes all of the protocols used on the Internet. In particular, SIP is modeled after the Simple Mail Transfer Protocol (SMTP) [14, 15], the basis for email, and the HyperText Transfer Protocol (HTTP) [16], the basis of the web. Like both of these, SIP is a textual client-server protocol, with requests issued by the client and responses returned by the server. SIP actually reuses much of the syntax and semantics of HTTP, including its response code architecture, many message headers, and its overall operation. SIP is used for IP telephony functions by mapping each function to one or more *transactions*. A SIP transaction consists of a single request issued by a client, and one or more responses returned by one or more servers. SIP transactions are *idempotent*, just like HTTP transactions. Also like HTTP, each SIP request is an attempt to invoke some *method* on the server. There are six SIP methods. The most basic of these is the INVITE method, which is used to initiate a call between the client and the server.

Unlike HTTP and SMTP, SIP can run on top of either TCP or UDP. In the case of UDP, the protocol provides its own mechanisms for reliability. The use of UDP also means the SIP messages can actually be multicast. Multicast allows for, among other features, group invitations and basic Automatic Call Distribution (ACD) functions that don't require a distribution server. The use of UDP also provides for fast operation (avoiding the TCP SYN handshake) and better scalability (no need to maintain TCP connection state in the kernel). When used with TCP, SIP allows many requests and responses to be sent over the same TCP connection, as in HTTP 1.1.

2.1 Protocol Components

There are only two components in a SIP system - user agents and network servers. A user agent is an end system which acts on behalf of some person who wishes to participate in calls. A user agent contains, in general, both a protocol client (called a User Agent Client, or UAC) and a protocol server (called a User Agent Server, or UAS). The UAC is used to initiate a call, and the UAS is used to answer a call. The presence of both in a user agent allows for peer-to-peer operation using a client-server protocol.

In addition to user agents, SIP provides for network servers. These servers are of two different types - *proxy* and *redirect*. A SIP proxy acts in much the same way as an HTTP proxy or an SMTP Message Transfer Agent (MTA). It receives a request, makes a determination about the next server to send it to, and forwards the request, possibly after modifying some of the header fields. A SIP proxy has no way of knowing whether the next server to receive the request is another proxy server, redirect server, or user agent server. As such, SIP requests can traverse many servers on their way from UAC to UAS. Responses to a request always travel along the same set of servers the request followed, but in reverse order.

A redirect server receives requests, but instead of forwarding them to the next hop server, it tells the client to contact the next hop server directly. It does this by responding to the request itself using a *redirect response*, containing the address of the next hop server. This is analogous to iterative searches in the Domain Name System (DNS) [17, 18], just as a proxying is analogous to recursive searches.

2.2 SIP Network Servers

The main function of a SIP network server is to provide for name resolution and user location. When a user wishes to place a call, it must send an INVITE request to the UAS of that user. However, the caller in general will not know the IP address or hostname of the UAS for the given user; it will have only some name which represents the caller (usually an email address, but it can be a telephone number or some other local identifier). Using this name, the UAC can determine a network server which may be able to resolve the name to an IP address. This network server may, in turn, proxy or redirect the call to additional servers, eventually arriving at one which definitively knows the IP address the user is to be contacted at. The process of determining the next-hop server is known as *next-hop routing*. Similar to other dynamic routing protocols, like the Border Gateway Protocol (BGP) [19], SIP provides facilities for loop detection and prevention.

A SIP network server can use any means at its disposal to determine the next-hop server. These include using DNS, accessing databases, executing programs, prompting users, etc. The final UAC contacted by the caller is determined by the composition of the decisions made at all the servers from callee to caller. It is this fact which makes SIP servers the basis for powerful mobility and forwarding services.

A SIP network server may determine, as a result of its next-hop routing decision, that there are actually multiple next-hop servers which may be able to eventually contact the user. SIP allows for a proxy server to *fork* an incoming request, sending it in parallel to multiple next-hop servers. Each of these *branches* may generate responses; SIP provides rules for merging and passing back these responses upstream, towards the UAC.

Each SIP transaction can take a different path through servers in the network. In a typical call, the first request is an INVITE, which may traverse many network servers on its way to the callee. The response to the INVITE contains a reach address that the UAC can use to send transactions directly to the UAS. The implication of this is that SIP network servers do not maintain call state. Once a transaction is complete, a SIP server has no recollection of the caller or callee. This results in good scalability and reliability. A SIP server can crash and recover (or a backup swapped in), and none of the calls initiated through it are affected. This also means that the duration of and amount of state maintained at a server is small in comparsion to the Global Switched Telephone Network (GSTN), where a switch must maintain call state for the entire call duration.

Interestingly, it is not even required for a SIP network server to be stateful during a transaction. A proxy or redirect server can be completely stateless - it receives a request, and either generates a response, or proxies the request - and then forgets everything. Proxied requests contain all the state embedded inside of them which is required for correct protocol operation. This is nicely aligned with the Internet datagram architecture where packets contain sufficient information to be individually routed. Furthermore, a stateful proxy can decide to become stateless at any time during the transaction, and the system still operates correctly. The decision to be stateless or stateful can be made on a call by call basis as desired by the administrator. This allows for large, central SIP servers to be stateless, but for smaller, localized servers to be stateful.

Figure 1 depicts a typical SIP deployment of network servers, and message flow between them. Here, there three domains (A, B, and C), each of which has a single SIP server acting as an access point into and out of their networks. A user agent, Joe, in domain A makes a call to another user, Bob. The call invitation is forwarded to domain A's access server (1), which attempts to find the callee in domain B and C by forking the request. The request arrives at domain C (3), but the user is unknown at this location, so an error response is returned (4). The request at domain B (2), however, is forwarded to a local server internal to domain B (5), where it finally reaches the UAS (6). The response is then forwarded back along the same path to the caller (7,8,9,10).

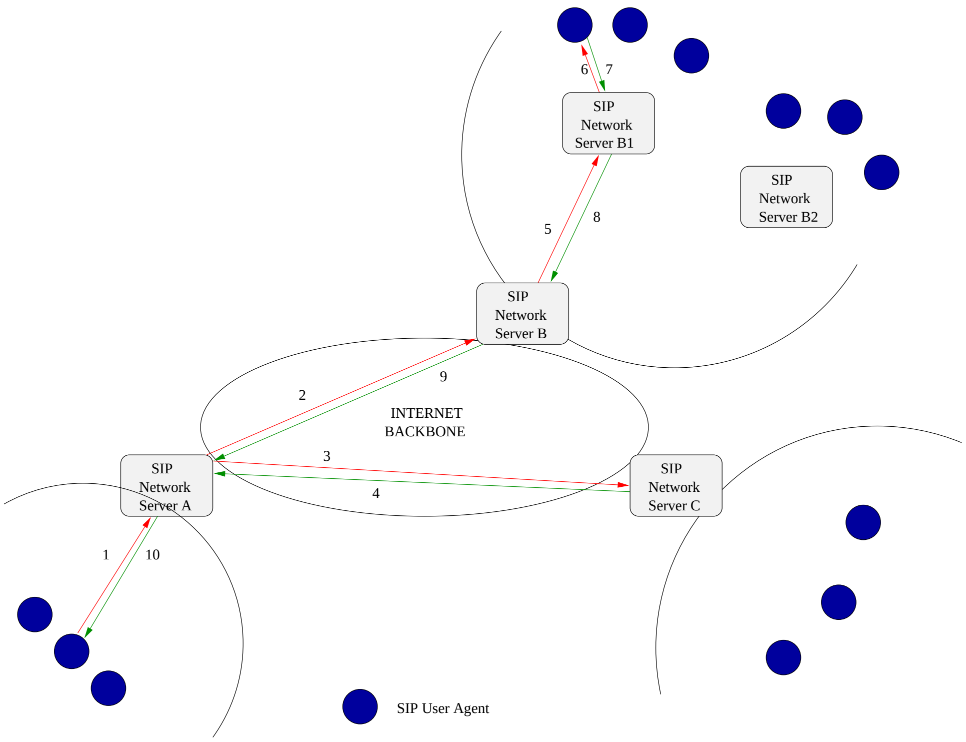

Figure 1: Typical SIP deployment

2.3 SIP Messages

A SIP request contains a header and a message body. The header contains various fields used to provide information on call services, addresses, and protocol features. The body is opaque to SIP, and can contain anything.

SIP defines several methods. These are INVITE, BYE, OPTIONS, ACK, REGISTER and CANCEL. INVITE is used to invite a user to a call. The header of an INVITE request contains the addresses of the caller and callee, subject of the call, call priority, call routing requests, caller preferences for user location, and desired features of the response, among others. The request body contains an opaque description of the media content of the session. Usually, this body is a Session Description Protocol (SDP) [20] object. SDP is a textual syntax for describing unicast and multicast multimedia sessions. It contains information on codecs, ports, and protocols to be used for sending media to the caller (such as parameters for the Real Time Transport Protocol (RTP) [2]). In the case of a broadcast-style multicast-based conference, it also contains information that describes the content, start and stop times, and administrator of the multicast session.

This allows a caller to invite a callee to participate in an existing mbone [21] session, for example. The SDP information is sufficient to allow a caller to begin sending and receiving audio immediately. The response to the INVITE contains the media information for the callee. SDP allows a user to indicate the ability to send and receive with multiple audio and video codecs, and can order those in preference of usage. This provides a basic, but based on experience, sufficient means for media capabilities exchange.

As the body is opaque to SIP, but described using MIME [22] syntax and semantics, SIP can use other media description formats besides SDP. Possibilities include H.245 capability descriptors, SMIL [23] presentation descriptions, or an XML [24] formatted descriptions of a new video codec. As these and other descriptions appear, they are as easily incorporated into SIP as a new image type is incorporated into HTTP.

REGISTER is used to convey location information to a SIP server. This allows a user to tell a SIP server how to map an incoming address into an outgoing address which will reach that user (or another proxy which knows how to reach that user). The body of a REGISTER message can be anything. Currently, the use of simple scripts to describe more complex programmatic name translations is being investigated, and is currently planned for standardization within the IETF iptel working group. This is much like service logic for switches. Furthermore, the body of a REGISTER response can contain configuration information useful to the user agent. These may include speed dial button configurations, additional addresses (allowing for PBX like functionality, whereby the server chooses the addresses used by each client), or a call log. Using the multipart MIME formatting rules, and the capability negotiation features of HTTP, new types of information can be added as time passes.

BYE is used to terminate a connection between two users in a conference. OPTIONS is used to solicit information about the capabilities of the callee, but does not set up a call. ACK is used for reliable message exchanges.

CANCEL is used to terminate a pending request, but does not undo a completed call. When received at a UAS, it has no effect if the UAS has already answered the call. If, however, the UAS has not answered, it indicates that the UAS should not bother responding, as the call has effectively been cancelled. This does not preclude a UAS from answering the call request anyway, however. It is simply an optimization.

As SIP is a textual protocol, generation and parsing of its messages is done trivially, particularly with powerful text processing languages such as perl. Furthermore, its compliance to RFC822 [25] formatting rules means existing HTTP or SMTP parsers can be used directly for lexigraphic analysis of messages. It also vastly simplifies debugging, development cycles, and extensions. A typical SIP INVITE is shown in Figure 2.

```
INVITE sip:ann@lucent.com SIP/2.0
Via: SIP/2.0/UDP 131.215.131.13;maddr=239.112.3.4;ttl=16
Via: SIP/2.0/TCP 10.0.1.1;received=128.13.44.52
From: John Smith <sip:jsmith@lucent.com>
To: Arun Netravali <sip:ann@lucent.com>
Subject: Raise
Call-ID: 132059753@mypc.domain.lucent.com
Content-Type: application/sdp
CSeq: 4711 INVITE
Content-Length: 187

v=0
o=user1 51633745 1348648134 IN IP4 128.3.4.5
s=Interactive Conference
c=IN IP4 224.2.4.4/127
t=0 0
m=audio 3456 RTP/AVP 0 22
a=rtpmap:22 application/g723.1
```

Figure 2: Typical SIP INVITE Message

2.4 Addressing and Naming

To be invited and identified, the called party has to be named. Since it is the most common form of user addressing in the Internet, SIP chose an email-like identifier of the form "user@domain", "user@host", "user@IP_address" or "phone-number@gateway". The domain name can be either the name of the host that a user is logged in at the time, an email address or the name of a domain-specific name translation service. Addresses of the form "phone-number@gateway" designate PSTN phone numbers reachable via the named gateway.

The use of email addresses as SIP addresses provides a scalable means by which a UAC can deliver a request to a SIP server which likely knows how to forward the request to the final callee - the Domain Name System (DNS). By using a series of DNS lookups, searching for service (SRV), mail exchange (MX), and address (A) records, the caller can determine the address of a server which has naming authority for all users within the domain.

The use of an email-like identifier also means that SIP addresses are easily transformed into Uniform Resource Identifiers (URI's) [26] such as sip:j.doe@example.com. As such, they can be embedded in web pages, so that clicking on the link initiates a call to that address, similar to a `mailto` [27] URL today.

2.5 Integration with Existing Protocols

One of the strengths of SIP is that it integrates well with existing protocols used on the Internet. In particular, SIP integrates well with the two dominant applications - web and email.

Integration with the web is on a number of levels. First, SIP carries around MIME content, as does HTTP. Thus, SIP can be used to return web content as a result of a call invitation. For example, a redirect response to a SIP invite can contain an HyperText Markup Language (HTML) document or a text document. This document could indicate detailed information on alternate places a user might be located (including pictures and sound). It could even return a Java applet which accepts input from the caller to determine where I might be reached. As a result, SIP would integrate extremely well with web browsers, bringing together the web and telephony for new, powerful services.

SIP uses URL's to decribe a user. These URL's can be embedded in web pages or in email, as can any other type of URL. Clicking on them can initiate calls, just like clicking on a web link gets a new web page.

One of the richest features of the web is its programmability. Web servers can make use of the Common Gateway Interface (CGI) [28], for example, to create dynamic content, customized for each user. This has resulted in many popular web services, such as stock information and movie listings. Since SIP looks like HTTP, we have begun to explore whether CGI can be applied to Internet telephony as well. Our investigations have indicated that CGI, as is, can be applied to SIP servers, and with backwards compatible extensions, provide a means for rich telephony programming. We have found that telephony services such as call-forward (and all its variants), mobility, and VPN, to name a few, are easily implemented with CGI. Furthermore, the wide array of software tools which exist to simplify CGI prototyping can be used for Internet telephony as well.

SIP integrates well with email and SMTP. A SIP INVITE message can be sent by email when all else fails. This is easily accomplished since a SIP address is the same as an email address. A SIP proxy server can easily reformat a SIP message into an SMTP message (the formatting is nearly identical), and forward it to an SMTP MTA. This provides integrated voicemail and email; call invitations to me when I was not available are simply delivered via email.

In addition, SIP headers can contain mailto URL's. This allows a caller to be redirected to contact the user by email, for example.

SIP also integrates nicely with the Real Time Streaming Protocol (RTSP) [29]. RTSP allows a client to have VCR-like controls over a media server. The client can instruct the server to play, record, fast-forward, and rewind, among other functions. The client can also specify details about what formats the server should record or play in. Like SIP, RTSP is a textual protocol, similar in format to HTTP. RTSP has several uses for IP telephony. First, voicemail is generally considered an important part of telephony. In essence, a voicemail server is nothing more than a media server. RTSP fits naturally as a means for controlling an IP telephony voicemail server. Secondly, it is often useful to record a call, or to playback some pre-recorded content into a conference call. RTSP could be used to control the recording and playback in these cases. In fact, RTSP has been engineered to allow a client to invite a media server to a session using SIP, and then control it using RTSP.

3 Network Services

As SIP's strength is in flexible user location and name mapping features, it provides a good framework for services such as personal mobility, 700, 800, and 900 services (note that billing is outside the scope of SIP, however), call screening, forward and transfer, and multiparty calls. Each of these are essentially based on programmed call routing, and thus fit well within SIP. SIP also introduces some new services not possible on the phone network, such as Caller Selection (see section 3.4). The sections which follow present just a taste of the services possible with SIP.

3.1 Personal Mobility

SIP supports advanced personal mobility services. An example of a mobility service is given in Figure 3.

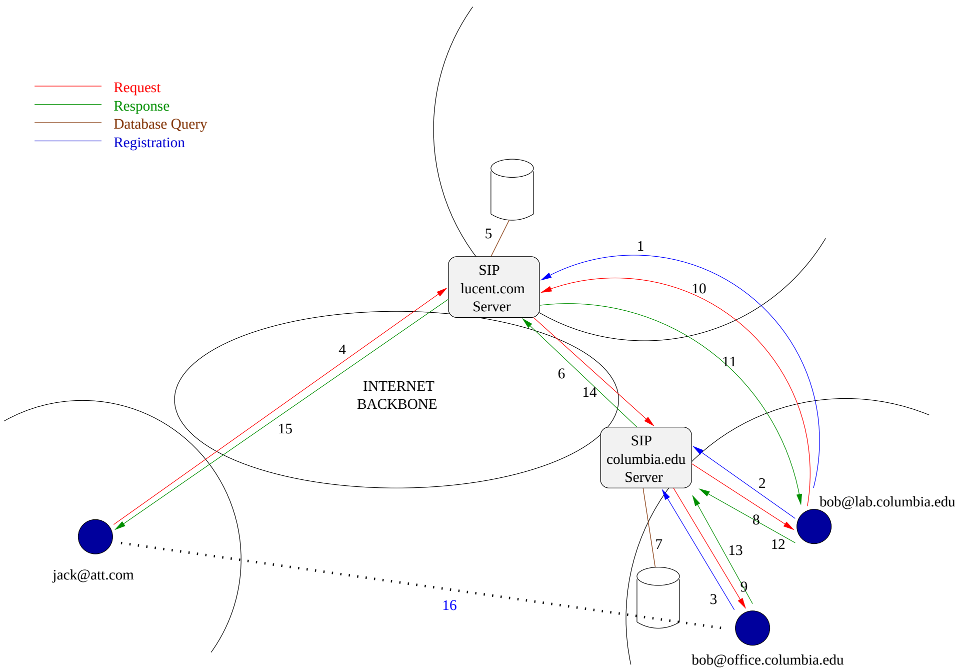

Figure 3: SIP Mobility Example

Here, a user of the system, Bob, maintains an office at Lucent. However, as an assistant professor at Columbia University, Bob also has a lab and an office there. Bob publishes a single IP telephony phone address for himself, `bob@lucent.com`. On a day when Bob is at Columbia, he sends a REGISTER message to the Lucent SIP server (1), listing his Columbia address, `bob@columbia.edu` as a forwarding address. Once at Columbia, Bob register's both his lab machine `bob@lab.columbia.edu` (2) and his office `bob@office.columbia.edu` with the Columbia registration server (3). Last time Bob was at Columbia, he set up his lab's computer to automatically forward calls back to his Lucent address. Forgetting about this, Bob restarts his client in the lab with the same configuration.

Later in the day, `jack@att.com` places a call to `bob@lucent.com`. Using DNS, the caller resolves `lucent.com` to the address of the Lucent SIP server, which receives the call request (4). The server checks its registration and policy databases (5), and decides to forwards the request to `bob@columbia.edu`. To do so, it looks up `columbia.edu` in DNS, and obtains the address of the main Columbia SIP server. It then forwards the request there (6). Once arriving at the Columbia server, it, too, does a policy and database lookup (7), and determines that there are two potential contacts for Bob. So, the server forks, and sends a call request to both the lab and office machines simultaneously (8,9). This causes the office phone to ring. The lab phone receives the request, and according to its outdated configuration, forwards it back to Lucent (10). Using the loop detection capabilities in SIP, the Lucent server determines that an error has occured, and returns an error response to the lab machine (11). It, in turn, returns an error code to the Columbia server (12).

In the meantime, Bob answers the phone in his office, sending an acceptance response back to the Columbia server (13). Having now received both responses, it forwards the call acceptance back to the Lucent server (14), which forwards the request back to the original caller (15). At this point, the Lucent and Columbia servers can destroy all call state if they so choose. Both future call transactions may proceed directly between the caller and Bob without passing through the intermediate servers (16).

The example illustrates a number of salient features of SIP. First, it shows how a call request can cause a "hunt" for a user to ensue, hopping between multiple servers until the final target of the call is found. Secondly, it demonstrates the loop detection features of SIP. Third, it demonstrates how a server can fork requests to speed up the process of contacting the desired user. Finally, it demonstrates how SIP network servers are used in a call only for the initial transaction. Note, however, that this does not lessen their importance in delivering rich mobility services.

3.2 Home Phone

One of the more interesting technical challenges for IP telephony is the ability to mimick the standard residential phone service. In particular, the following features are required:

1. When someone calls a particular number, all phones in the home ring.

2. When one of the lines is picked up, all other lines stop ringing.

3. A user can pick up from any other telephone in the home and join an existing call.

4. There can be multiple lines in the home, so that a user on another handset can initiate a new call, while one or more are currently in progress.

5. All users involved in a single call are all essentially involved in a multiparty conference, and are thus able to hear each other.

This basic service is easily emulated in SIP, as is exemplified in Figure 4.

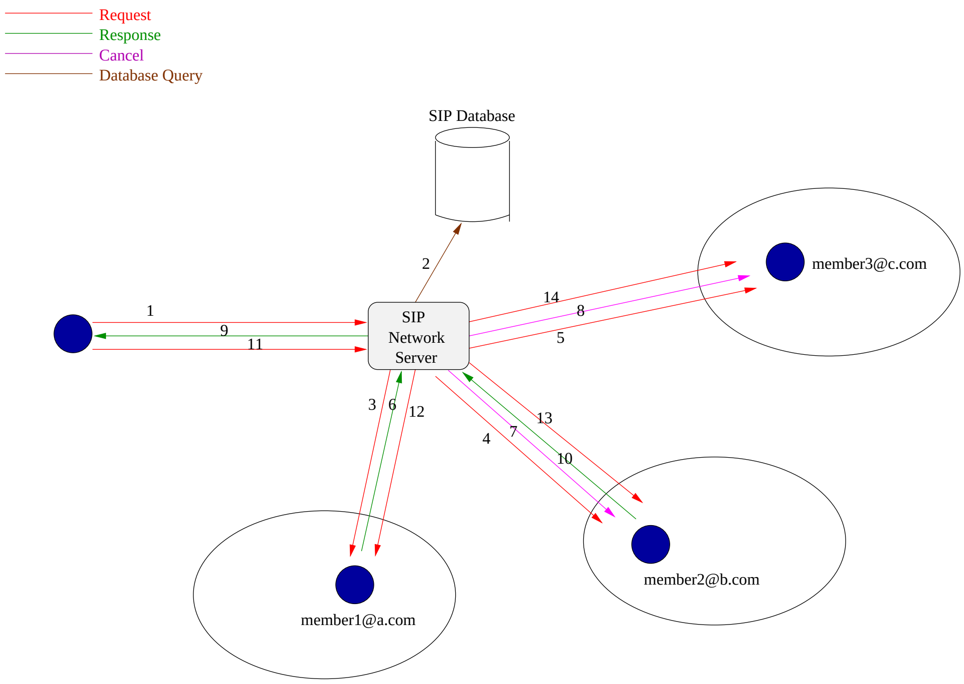

Figure 4: Residential Service Emulation

Here, the caller sends a SIP INVITE to `smith_family@isp.com`. The INVITE is sent to the main server for the ISP (1). The ISP consults its policy database (2), which indicates that the called address is a residential line. The database also includes a list of contact addresses, each of which constitutes a single extension on that line. The proxy server thus forks and sends out three INVITE's, (3,4,5), one to each address. Furthermore, the server modifies the SDP in the message, indicating the use of a multicast address for media exchange.

The lines ring at each address. A user picks up on the first line. This causes a call acceptance to be sent back to the server (6). According to the SIP specifications, when a forking proxy server receives a call acceptance on any one of its branches, it is to send a CANCEL request on all unanswered branches. So, the proxy sends out a CANCEL on the other two branches (7,8). As neither have yet answered, it causes the call request to be cancelled, effectively terminating the call. This causes the phones to stop ringing.

The server forwards the call acceptance back to the caller (9). It also acts as a multicast to unicast bridge, forwarding media received on the multicast group back to the caller, and vice a versa. The server also lists itself in the call acceptance as the main point of contact for continued signaling messages.

Later on, one of the other lines is picked up. This causes another call acceptance to be forwarded back to the server (10). This acceptance is acknowledged by the server. Using the multicast group, the new participant is now part of the call.

Later on, the caller hangs up the call, sending a BYE message to the server (11). The server sends the BYE message to all three branches (12, 13, 14), causing the two currently active lines to terminate. It also causes the unanswered line to destroy state associated with the call.

This example illustrates further the power of forked call requests, combined with the utility of the CANCEL request. The behavior described for the SIP clients in the above example is standard procedure, not specific for residential service. This means that the same piece of software can be used by clients for either residential or traditional business service. Only the server in the above example required special programming to direct it towards residential behavior (although this behavior was still compliant with the SIP specification).

These same mechanisms can be used equally well for ACD services.

3.3 Outsourced Call Screening

The usage of SIP proxy servers also enables for the outsourcing of various call logic services to third party vendors. Consider the case of outgoing call screening services. A parent wishes to prevent her teenage son from making calls to various adult telephone numbers. Instead of trying to keep track of the numbers of such services, and programming filters into the software in the home, the parent subscribes to a filering service, which keeps track of such numbers automatically.

The usage of SIP for this kind of service is depicted in Figure 5. The user's home software is configured to automatically forward all call requests to `service831@we_do_filters.com`. Whenever a call is made from the home, the call is forwarded to the server of the filtering provider (1). Once there, the service checks the URI of the incoming requests, and notices that it is service 832, adult line filtering. The server performs a database lookup (2), and finds that the number which is called is one of the ones in the database. The server then returns a call rejection to the caller (3), and includes a reason phrase that indicates a forbidden number. Had the called number not been in the database, the server would have proxied the call to the listed number, allowing it to complete normally.

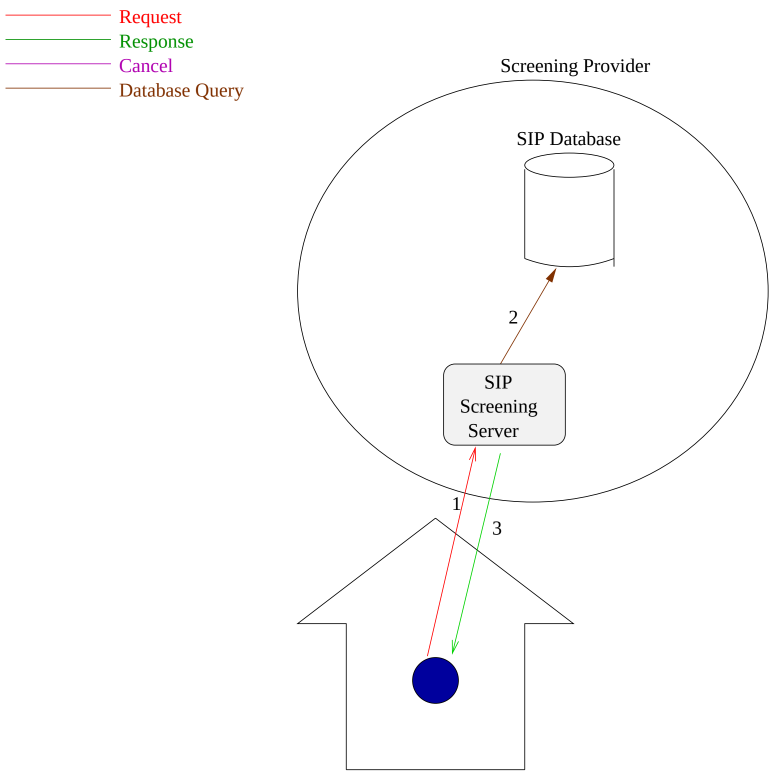

Figure 5: SIP Screening Service

3.4 Caller Selection

SIP also allows for multicast signaling and a new feature, called *caller selection*. Caller selection allows the call initiator to choose whom to talk to in the case where multiple parties answer a call. Figure 6 demonstrates an example scenario which makes use of both of these.

In this scenario, there are no network servers. The caller is an employee of a small company. The caller is in need of computer assistance, so he sends a SIP `INVITE` message to `sysadmins@company.com`. The callers' machine has been pre-configured to send messages destined for this address to a particular multicast address. All of the system administrators in the company have configured their software to listen for call requests on this address. The `INVITE` request is thus received by all three of the system administrators (1). One of them answers immediately, the response being multicast as well (2). This causes the other system administrator's software to effectively behave as if a `CANCEL` had been sent. Their lines stop ringing, and the call between the caller and the first administrator progresses. Later on, a second administrator answers, sending a call acceptance on the multicast group also (3). As SIP allows multiple call acceptances to a single request, the caller decides to accept the second response as well. Using SIP's third party call control mechanisms, the caller sets up a multicast media conference between all three.

When the second call acceptance arrived, the caller could have taken any one of a number of actions.

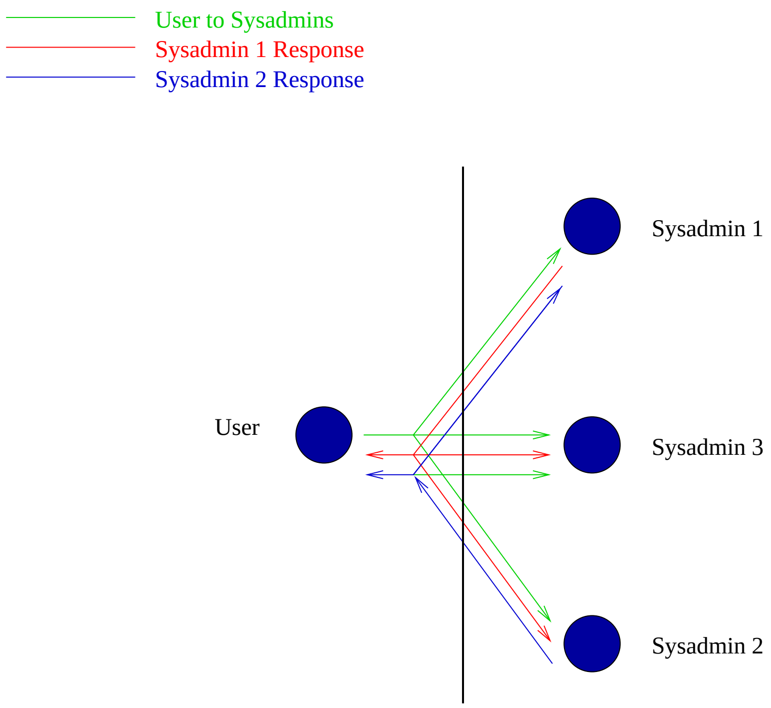

Figure 6: Caller Selection and Multicast Example

These include accepting the call, but not setting up a conference (in which case the caller is effectively in two calls), hanging up with one or the other, or both, or redirecting the second accepter to voicemail. Many other possibilities exist as well.

Note that in this example, a simple ACD type of service was provided, but with fully distributed mechanisms. No call distribution server was required.

4 Client Services

As the above examples have illustrated, SIP network servers are generally used to provide services related to address resolution, naming, user location, forwarding, and policy. Telephony services encompass more than just these, of course. They include features such as multiparty conferencing, transfer, hold, and mute. In an Internet environment, these services can be provided in the clients directly, without network server support. To this end, SIP supports these basic features with end to end protocol components. However, a SIP server can easily provide these services, if desired. This allows SIP to be used in environments ranging from the tightly controlled and managed PBX to fully distributed, serverless networks.

SIP supports multiparty and transfer services largely through two end-to-end headers, Also and Replaces. When including in either a request or a response, these instruct the recipient to place, or terminate respectively, a call to the parties listed in the header. This simple mechanism provides a framework for basic third party call control. This, in turn, is easily used to construct a variety of transfer and multiparty services [30].

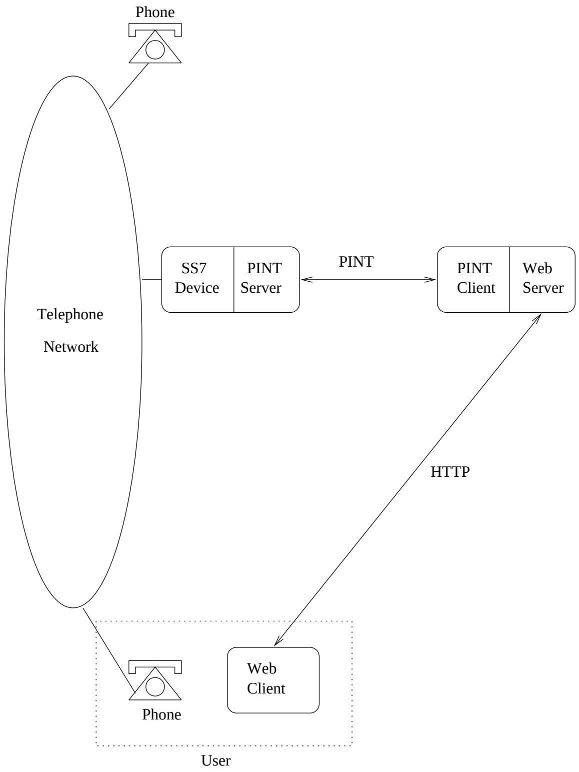

Figure 7: Click-to-Dial Service and PINT

As an example of this, the PSTN and IP Internetworking (pint) working group of the IETF is working to standardize a protocol for click-to-dial and click-to-fax services. Click-to-dial allows a user, while web browsing, to click on an icon, and receive a phone call from a service representative on the GSTN . The protocol to accomplish this allows a web server to pass third party call control commands to a "PSTN enabled" server, such as a PBX or SCP. See Figure 7 for an illustration. The requirements for a protocol to accomplish this include flexible addressing and naming, authentication, encryption, extensibility, and third party call control primitives. As SIP is ideal for this, the working group is standardizing some basic SIP extensions to provide the service.

5 Key Benefits

SIP offers a number of key benefits, which we discuss here in more detail. These include simplicity, extensibility, modularity, scalability, integration, and enhanced services.

5.1 Simplicity

SIP is a very simple protocol. The SIP specification is just 99 pages. SIP has only 42 headers, each with a small number of values and parameters. A basic, but interoperable SIP Internet telephony implementation can get by with four headers (To, From, Call-ID, and CSeq) and three request types (INVITE, ACK, and BYE) and is small enough to be assigned as a homework programming problem in the Advanced Internet Services class taught at Columbia University. A fully functional SIP client agent, with a graphical user interface, has been implemented in just two months.

Since SIP encodes its messages as text, parsing and generation is simple. This is especially true when done with powerful text processing languages such as Perl. The textual encoding also simplifies debugging, allowing manual entry and perusing of messages. Its similarity to HTTP also allows for code-reuse; existing HTTP parsers can be quickly modified for SIP usage.

5.2 Extensibility

Extensibility is a key metric for measuring an IP telephony signaling protocol. Telephony is a currently the one of the largest, economically important, widespread, critical service, and Internet telephony is poised to, over the long run, supplant the existing circuit switched infrastructure developed to support it. As with any heavily used service, the features provided evolve over time as new applications are developed. This makes compatibility among versions a complex issue. As the Internet is an open, distributed, and evolving entity, one can expect extensions to IP telephony protocols to be widespread and uncoordinated. This makes it critical to build in powerful extension mechanisms from the outset.

SIP has learned the lessons of HTTP and SMTP (both of which are widely used protocols that have evolved over time), and built in a rich set of extensibility and compatibility functions. By default, unknown headers and values are ignored. Using the Require header, clients can indicate named feature sets that the server must understand. When a request arrives at a server, it checks the list of named features in the Require header. If any of them are not supported, the server returns an error code and lists the set of features it does understand. The client can then determine the problematic feature and fall back to simpler operation. The feature names are based on a hierarchical namespace, and new feature names can be registered with the Internet Assigned Numbers Authority (IANA). This means that any developer can create new features in SIP, and then simply register a name for them. Compatibility is still maintained across different versions.

To further enhance extensibility, numerical error codes are hierarchically organized, as in HTTP. There are six basic classes, each of which is identified by the hundreds digit in the response code. Basic protocol operation is dictated solely by the class, and terminals need only understand the class of the response. The other digits provide additional information, usually useful but not critical. This allows for additional features to be added by defining semantics for the error codes in a class, while achieving compatibility.

The textual encoding means that header fields are self-describing. It is self-evident what the meaning of the To, From, and Subject fields are. As new header fields are added in various different implementations, developers in other corporations can determine usage just from the name, and add support for the field. This kind of distributed, documentation-less standardization has been common in SMTP, which has evolved over the years.

As SIP is similar to HTTP, mechanisms being developed for HTTP extensibility can also be used in SIP. Among these are the Protocol Extensions Protocol (PEP) [31], which contains pointers to the documentation for various features within the HTTP messages themselves.

5.3 Modularity

Another aspect of extensibility is modularity. Internet telephony requires a large number of different functions, some of which have been mentioned previously. One can be certain that mechanisms for accomplishing these functions will evolve over time (especially with regards to QoS). This makes it critical to apportion these functions to seperate, modular, orthogonal components, which can be swapped in and out over time. It is also critical to use seperate, general protocols for each of these functions. This allows for the function to be duplicated in other applications with ease. For example, it is more efficient to have a single QoS mechanism which is application independent, rather than invent a new QoS protocol or mechanism for each application.

SIP is very modular. It encompasses basic call signaling (initiation, termination, and change), user location, and basic registration (essential for user location). Advanced signaling is part of SIP, but within a single extension. Quality of service, directory accesses, service discovery, session content description, and conference control are all orthogonal, and reside in seperate protocols.

A key feature of SIP is that it separates the notion of a session from the protocol used invite a user to a session. SIP is just used for invitations; it doesn't know anything about the session itself. Protocols such as SDP (which describes media sessions) are used for this purpose. This means that SIP can also be used to invite users to control sessions, broadcast television sessions, document editing sessions, and even virtual reality sessions, once description formats for these sessions are developed. Proxy servers don't need to know or understand anything about these sessions. Thus, a proxy server deployed today will still work with these future applications. It is this kind of modularity that has allowed the web to flourish. Although it was initially only used to carry HTML, HTTP now carries a wide variety of content types. Had HTTP only carried HTML, there would surely be no web.

Interestingly, the session a SIP user gets invited to can even be described by a program. SIP can carry Java applets or Tcl code to define what a session means. This makes SIP a powerful tool for future Intelligent Network (IN) applications, where programmability is key.

5.4 Scalability

We can observe scalability on a number of different levels:

Large numbers of domains: SIP leverages off of the Domain Name System, and off of powerful routing protocols, such as BGP, to provide wide area operation. End systems can be located anywhere on the Internet. No additional name resolution services beyond those already used in the Internet are required.

Server processing: In SIP, a transaction through several servers and gateways can be either stateful or stateless. In the case of UDP, no connection state is even required. This means that large, backbone servers can be based on UDP and operate in a stateless fashion, reducing memory requirements and improving scalability. SIP servers at network edges, or within the enterprise, can be stateful, allowing more complex services. This model of "simple within the core, complex at the periperhy", has been the cornerstone of Internet scalability for some time.

Conference sizes: SIP scales to all different conference sizes. There is no requirement for a central multipoint controller; conference coordination can be fully distributed or centralized, at the discretion of the implementor. SIP also works for large, broadcast style conferences. In fact, SIP's original usage was to invite users to participate in mbone conferences. It is for this reason that SIP is implemented in the popular mbone session directory tool, sdr.

5.5 Integration

An important feature of SIP is that it integrates very well with the web and other networking services. Its critical to realize that the power of the Internet is its support for a vast array of applications. An Internet telephony protocol should therefore strive to integrate well with them, particularly the web.

As described previously, SIP integrates with the web, email, and streaming media applications and protocols, to name a few.

6 Conclusion

In this paper, we have presented the Session Initiation Protocol (SIP). SIP is used for the initiation, control, and termination of multimedia conferences. It builds upon the strengths of HTTP and SMTP, mapping telephony services into simple request-response transactions. SIP's strengths lie in its ability to support rich mobility, forwarding, address resolution, and naming services. We have shown how SIP can be used for such things as personal mobility, call screening, residential line services, and ACD. We have briefly mentioned the third party call control services available in SIP, and how they can be used to deliver click-to-dial.

A SIP server, user agent, and gateway are all currently under development within Bell Laboratories. Prototypes of all three were recently demonstrated at the Supercomm trade show in June of 1998, and beta code is shipping to customers in September 1998. The SIP client is a Java based application which makes use of the JTAPI [32] interface for allowing rapid development of other SIP applications. The server is a multi-platform software toolkit. It incorporates support for hierarchical policy databases to guide routing decisions, which can be used to construct many of the services described here. The SIP gateway interfaces to the SARAS call processing system, under development in BCS, which provides a universal platform for call services within SIP, H.323, ISDN, and POTS.

References

[1] H. Schulzrinne, "Re-engineering the telephone system," in *Proc. of IEEE Singapore International Conference on Networks (SICON)*, (Singapore), Apr. 1997.

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

[3] B. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation protocol (RSVP) – version 1 functional specification," Request for Comments (Proposed Standard) 2205, Internet Engineering Task Force, Oct. 1997.

[4] B. Braden and L. Zhang, "Resource ReSerVation protocol (RSVP) – version 1 message processing rules," Request for Comments (Proposed Standard) 2209, Internet Engineering Task Force, Oct. 1997.

[5] P. Pan and H. Schulzrinne, "Yessir: A simple reservation mechanism for the internet," Technical Report RC 20697, IBM Research, Hawthorne, New York, Sept. 1997.

[6] A. Rubens and P. Calhoun, "DIAMETER base protocol," Internet Draft, Internet Engineering Task Force, May 1998. Work in progress.

[7] C. Rigney, A. Rubens, W. Simpson, and S. Willens, "Remote authentication dial in user service (RADIUS)," Request for Comments (Proposed Standard) 2138, Internet Engineering Task Force, Apr. 1997. (Obsoletes RFC2058).

[8] C. Rigney, "RADIUS accounting," Request for Comments (Informational) 2139, Internet Engineering Task Force, Apr. 1997. (Obsoletes RFC2059).

[9] J. Rosenberg and H. Schulzrinne, "Internet telephony gateway location," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, (San Francisco, California), March/April 1998.

[10] T. Howes, S. Kille, and M. Wahl, "Lightweight directory access protocol (v3)," Request for Comments (Proposed Standard) 2251, Internet Engineering Task Force, Dec. 1997.

[11] International Telecommunication Union, "Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service," Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1996.

[12] M. Handley, H. Schulzrinne, and E. Schooler, "SIP: session initiation protocol," Internet Draft, Internet Engineering Task Force, May 1998. Work in progress.

[13] H. Schulzrinne and J. Rosenberg, "A comparison of SIP and H.323 for internet telephony," in *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, (Cambridge, England), July 1998.

[14] J. Postel, "Simple mail transfer protocol," Request for Comments (Standard) STD 10, 821, Internet Engineering Task Force, Aug. 1982. (Obsoletes RFC788).

[15] J. Klensin and D. Mann, "Simple mail transfer protocol," Internet Draft, Internet Engineering Task Force, May 1998. Work in progress.

[16] R. Fielding, J. Gettys, J. Mogul, H. Nielsen, and T. Berners-Lee, "Hypertext transfer protocol – HTTP/1.1," Request for Comments (Proposed Standard) 2068, Internet Engineering Task Force, Jan. 1997.

[17] P. Mockapetris, "Domain names - concepts and facilities," Request for Comments (Standard) STD 13, 1034, Internet Engineering Task Force, Nov. 1987. (Obsoletes RFC882); (Obsoletes RFC883); (Obsoletes

RFC973); (Obsoleted by RFC2065); (Updated by RFC1101); (Updated by RFC1876); (Updated by RFC1982); (Updated by RFC2181); (Updated by RFC2308).

[18] P. Mockapetris, "Domain names - implementation and specification," Request for Comments (Standard) STD 13, 1035, Internet Engineering Task Force, Nov. 1987. (Obsoletes RFC973); (Obsoleted by RFC2136); (Obsoleted by RFC2137); (Updated by RFC1348); (Updated by RFC1995); (Updated by RFC1996); (Updated by RFC2065); (Updated by RFC2181); (Updated by RFC2308).

[19] Y. Rekhter and T. Li, "A border gateway protocol 4 (BGP-4)," Request for Comments (Draft Standard) 1771, Internet Engineering Task Force, Mar. 1995. (Obsoletes RFC1654).

[20] M. Handley and V. Jacobson, "SDP: session description protocol," Request for Comments (Proposed Standard) 2327, Internet Engineering Task Force, Apr. 1998.

[21] H. Eriksson, "MBone – the multicast backbone," in *Proceedings of the International Networking Conference (INET)*, (San Francisco, California), pp. CCC–1 – CCC–5, Internet Society, Aug. 1993.

[22] N. Borenstein and N. Freed, "MIME (multipurpose internet mail extensions): Mechanisms for specifying and describing the format of internet message bodies," Request for Comments (Proposed Standard) 1341, Internet Engineering Task Force, June 1992. (Obsoleted by RFC1521).

[23] World Wide Web Consortium, "Synchronized multimedia integration language (smil) 1.0 specification," Recommendation PR-smil-19980409, World Wide Web Consortium, Apr. 1998.

[24] World Wide Web Consortium, "Extensible markup language (xml) 1.0," Recommendation REC-xml-19980210, World Wide Web Consortium, Feb. 1998.

[25] D. Crocker, "Standard for the format of ARPA internet text messages," Request for Comments (Standard) STD 11, 822, Internet Engineering Task Force, Aug. 1982. (Obsoletes RFC733); (Updated by RFC987); (Updated by RFC1327).

[26] T. Berners-Lee, L. Masinter, and R. Fielding, "Uniform resource identifiers (URI): generic syntax," Internet Draft, Internet Engineering Task Force, Mar. 1998. Work in progress.

[27] P. Hoffman, L. Masinter, and J. Zawinski, "The mailto URL scheme," Request for Comments (Proposed Standard) 2368, Internet Engineering Task Force, July 1998.

[28] D. Robinson and K. Coar, "The WWW common gateway interface version 1.1," Internet Draft, Internet Engineering Task Force, May 1998. Work in progress.

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

[30] H. Schulzrinne and J. Rosenberg, "Signaling for internet telephony," in *International Conference on Network Protocols (ICNP)*, (Austin, Texas), Oct. 1998.

[31] D. Connolly, H. Nielsen, R. Khare, and E. Prud'hommeaux, "PEP - an extension mechanism for HTTP," Internet Draft, Internet Engineering Task Force, Dec. 1997. Work in progress.

[32] Sun Microsystems, "The java telephony API," Feb. 1997.