A Comparison of H.323 vs SIP

Pavlos Papageorgiou
pavlos@eng.umd.edu

University of Maryland at College Park

June 4, 2001

Unpublished and incomplete manuscript. Missing experiments.

Contents

1 Introduction

This report outlines the major guidelines in my master thesis research. My primary focus is to provide a rigorous, correct and unbiased comparative analysis of two signaling protocols: the ITU recommendation H.323 and the Session Initiation Protocol (SIP) of the IETF.

1.1 Overview

H.323 and the Session Initiation Protocol have emerged as competing protocol standards for the signaling and call control of IP telephony. The former is a ITU recommendation and has evolved into an umbrella of specifications for packet based multimedia communication systems. H.323 is based on the Q.931 ISDN protocol and as such it embraces a traditional circuit switched approach. SIP on the other hand, is a IETF protocol which aims at providing equivalent services through a simpler and more lightweight web based approach.

We will first present an overview of the approach each protocol follows to address signaling and call control issues which arise in multimedia calls caried over a packet network. Then we will proceed to summarize previous work on comparing the two protocols. Finally, we conclude by explaining how we intend to supplement and contribute to this work by analyzing, evaluating and measuring the performance of the two protocols in a rigorous and thorough comparative performance analysis.

2 H.323 Protocol Overview

H.323 is a ITU recommendation based on the H.320 family of standards. The current version of the recommendation is version 4 [1]. Initially, the protocol (version 1) was designed to provide signaling for a multimedia conferencing system for LAN environments with no quality of service provisions. However, in is current state, it has evolved into an umbrella of specifications that define the complete architecture and operation of a multimedia conferencing system over a wide area packet network. In contrast to its original scope, it has become a scalable solution that can be interworked with managed large scale networks.

2.1 System Description

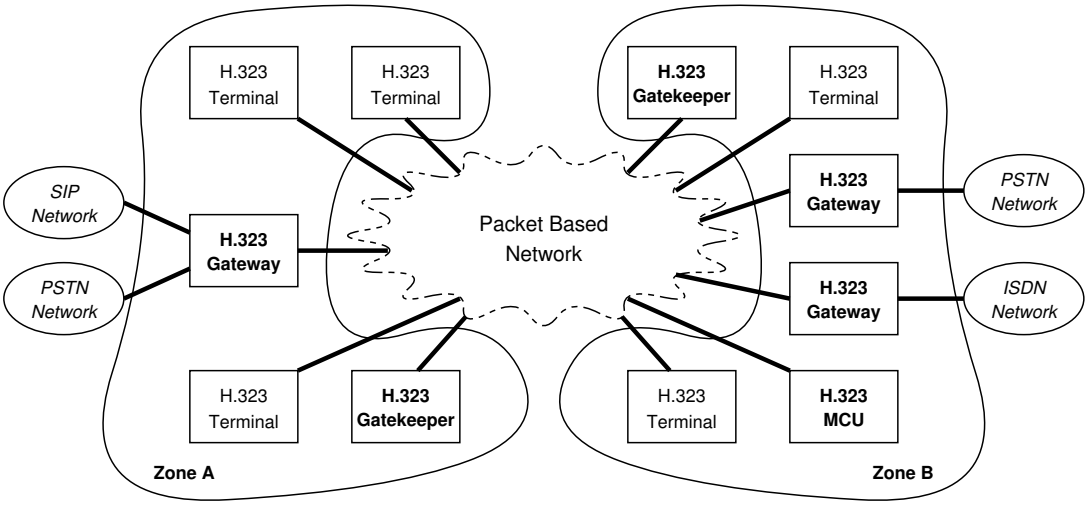

Figure 1: A H.323 System

A H.323 system provides the necessary signaling and control operations for performing multimedia communications over an underlying packet based network which may not provide a guaranteed quality of service. The actual network interface, the physical network and the transport protocols used on the network are not included in the scope of H.323.

A H.323 system comprises of the following entities: Terminals, Gatekeepers, Gateways, Multipoint Controllers, Multipoint Processors and Multipoint Control Units.

- ***Terminals*** provide the audio/video/data communications capability in point-to-point or multipoint conferences, as well as handling the H.323 signaling issues on behalf of the user.
- ***Gatekeepers*** provide admission control and address translation services
- ***Gateways*** are needed to provide interworking with terminals using other signaling protocols, such as PSTN terminals, ISDN terminals, SIP terminals, etc.
- ***Multipoint Controllers, Multipoint Processors and Multipoint Control Units*** provide support for multipoint conferences.

A central aspect of H.323 is the *H.323 call*. It is defined as the point-to-point multimedia communication between two H.323 endpoints. If the H.323 endpoint communicates with an endpoint which uses a different signaling protocol, then the H.323 call is defined as the call segment between the H.323 entity and the gateway that provides interworking with the foreign network.

. The call can have multiple participants. It begins with the call setup procedure and ends with the call termination procedure. It consists of a collection of reliable and unreliable channels between the endpoints. The signaling and control messages can be exchanged directly between the two endpoints or through one or more H.323 entities, such as gatekeeper, gateways or multipoint controllers.

The H.323 system is partitioned into zones. Each zone is comprised by the collection of all terminals, gateways and multipoint controllers managed by a single gatekeeper. A zone is not necessarily restricted within a single network segment; it may span through multiple network segments that are interconnected. Membership in a zone does not imply a specific network topology.

The H.323 protocol is a tightly coupled family of subprotocols which must all interoperate in order to complete successfully a multimedia call session. The subprotocols are described in ITU recommendations. The main ones are:

- *H.225:* Subprotocol for messages exchanged between H.323 endpoints for setting up and tearing down a call as well as for messages between an H.323 enpdoint and its controlling H.323 entity, such as a gatekeeper.
- *H.245:* Subprotocol for messages exchanged between endpoints in order to control the call session, exchange resource capabilities and establish media channels.
- *H.235:* Subprotocol for security and encryption for H.323 terminals.
- *H.450:* Subprotocols for supplementary services, such as Call Transfer, Call Park, Call Waiting etc.
- *Annexes:* Specific issues that arise in the H.323 protocol are clarified in the annexes, such as Annex C, Annex D, Annex H, etc.

2.2 Signaling Approach

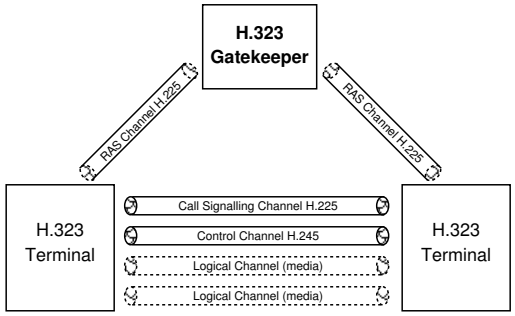

Figure 2: Direct Endpoint Call and Control Signaling

The H.323 protocol is implemented by exchanging messages between the protocol endpoints and intervening entities. These messages are all encoded using ASN.1 [2] (Abstract Syntax Notation) which is a binary format for defining the syntax of information data.

Even though H.323 is a packet based protocol, it remains tightly coupled with most traditional telecommunication standards. As such, it defines various communication channels, each using its own subprotocol for the communication between the various H.323 entities. Specifically, the protocol distinguishes between three communication channels:

1. The *RAS Channel (Registration Admission Status)* is an unreliable channel between the H.323 endpoint and the gatekeeper. It is used to exchange registration, admission, bandwidth change and status messages. The messages exchanged in this channel follow the recommendation H.225.

2. The *Call Signaling Channel* is a reliable channel between H.323 endpoints (direct or routed through gatekeepers). It is used to perform the call setup and teardown phases. The messages exchanged in this channel follow the recommendation H.225.

3. The *H.245 Control Channel* is a reliable channel between H.323 endpoints (direct or routed through gatekeepers). It is used to exchange the H.245 call control messages. The messages exchanged in this channel follow the recommendation H.245.

The signaling of the whole call session is performed through messages exchanged on these three channels. H.323 defines one additional type of channel; the logical media channel. This channel carries the media content and each session can have one or more channels established through the the H.245 control channel. The actual way the media content is transferred through those channels lies outside the scope of the H.323 recommendation.

2.3 Basic Functionality

In this report, our intention is not to duplicate the H.323 recommendation Instead we attempt to provide an insightful description of the protocol's behaviour and the main approach it chooses to adopt in order to tackle a broad range of signaling issues arising in multimedia call sessions. In H.323, a call session is perceived as consisting of five phases:

1. Call Setup

2. Initial Communication and Capability Exchange

3. Establishment of Audio Visual Communication

4. Call Services

5. Call Termination

H.323 makes use of three different signaling channels in order to complete these five phases. The sequence of events which take place during a typical call session can be briefly outlined as follows. Initially, an H.323 endpoint conveys registration information to an H.323 entity, the gatekeeper, through the RAS channel on which H.225 messages are exchanged. The registration is either performed only once or periodically refreshed according to the gatekeeper's policy; in any way, the registration process is not related to a particular call.

Whenever a terminal wishes to place a call, it requests permission from the controlling H.323 gatekeeper through the RAS channel. If permission is granted, the endpoint discovers in some way the transport address of the call signaling channel on the endpoint it wishes to call. Then it attempts to to setup a call session through this channel, which uses H.225 messages as well .

Once the call signaling channel has been established, the two endpoints proceed to setup the control signaling channel, which uses H.245 messages. It is this channel that then takes over the call completely and through which all signaling services are performed. At this point any of the endpoints can close the call signaling channel H.225 since it has served its purpose.

The two endpoints proceed to exchange and negotiate resource capabilities and set up the media channels. Usually, three media channels are established; one for audio, one for video and one for data. The two endpoints can now engage into a multimedia conversation. Unless any call services are requested during the conversation (modification of call characteristics, establishment of additional media channels, addition of an endpoint, etc.) there is no more traffic on the control channel H.245 before the call termination phase.

When the conversation is over, the endpoints close all the media channels they had opened and one of the endpoint initiates the tear down of the call control signaling channel H.245. If the call signaling channel H.225 is still open, then it is torn down as well.

The sequence of events presented here is quite indicative of the functionality of earlier versions of H.323. However, recent versions incorporate various optimizations aimed at minimizing the long call delay which obviously such a sequence of messages introduces. For example, H.323 v3 and v4 use Fast Connect and H.245 Tunnelling to decrease dramatically the delay.

These techniques succeed in greatly reducing the protocol setup delay; however, they do not really modify the essential way the protocol works. As a result, instead of confusing the unfamiliar reader by presenting from the onset the optimizations of the protocol, we choose to present initially the functionality of the protocol in the straightforward manner in which it was originally implemented. The optimizations introduced in later versions aim at condensing and tunnelling the required messages in order for the protocol implementations to be more capable to meet the strict call setup delay bounds.

2.3.1 Gatekeeper Discovery and Endpoint Registration

Before placing any call with the help of a gatekeeper, each endpoint must make its presence known to the gatekeeper. This is accomplished through a short registration process. However, in order to register, the endpoint must find in some way the transport address of at least one gatekeeper. This phase is called Gatekeeper Discovery.

During gatekeeper discovery, an endpoint determines which gatekeeper it can register with. If this is not performed manually (i . e .the endpoint is configured with the transport address of the associated gatekeeper), H.323 provides an automatic method.

We demonstrate the auto discovery method with an example where one endpoint and three gatekeepers are involved. After the gatekeeper discovery phase completes, the endpoint proceeds to register with one of the eligible candidates. We also show the symmetric deregistration procedure.

Once registered an endpoint can proceed to place calls with the help of the gatekeeper. The degree of involvement of the gatekeeper in the call varies between very small and completely involved, depending on the portion of the call signaling that gets routed through it.

In our example, the gatekeeper will be minimally involved by translating endpoint aliases to transport addresses and granting requests for the placement of a call (i . e .direct call and control signaling will be used).

2.3.2 Phase A: Call Setup

The primary goal of this phase is to locate the called endpoint, establish whether the user accepts the call and proceed with the setting up of the call signaling channel H.225. Then, in

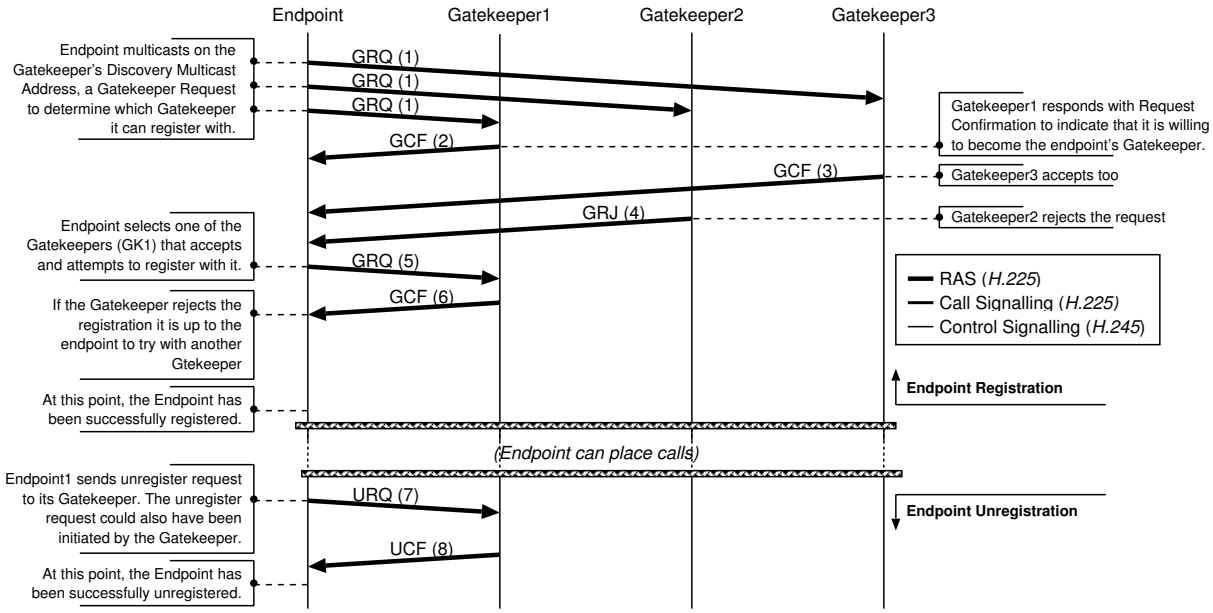

Figure 3: Gatekeeper Discovery and Endpoint Registration

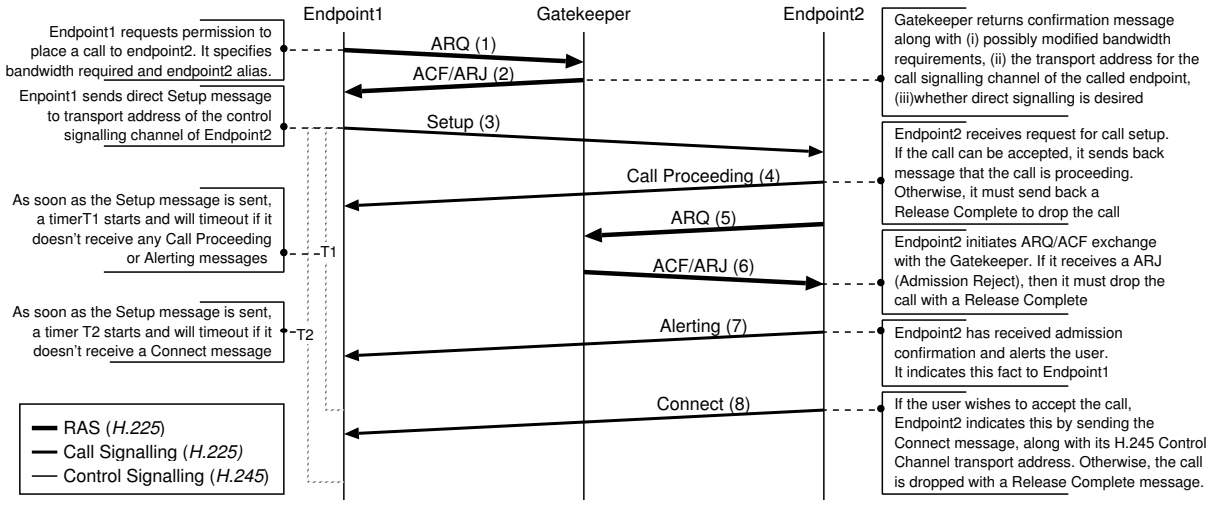

Figure 4: Phase A: Call Setup – Direct Signaling

the next phase, the call signaling channel will be used to open the call control channel H.245, which is the protocol that actually controls most aspects of the call.

Note that if the endpoints knew the transport addresses of each other, they could in principle bypass the gatekeeper and not request permission for placing the call. However, most of the times only the alias is known, and the translation services of the gatekeeper would be needed. But even in the case where alias translation is not required, an endpoint can be forced to use the getekeeper, by other means, such as placing it behind a firewall.

2.3.3 Phase B: Initial Communication and Capability Exchange

Following the setup of the call signaling channel, the endpoints proceed to establish a H.245 control channel. This is the most important channel, since messages exchanged through it control all aspects of the call from this point of the call session.

After Endpoint1 has received the Connect message, it must have received (in the Connect or a previous H.225 message) the transport address where Endpoint2 is listening for H.245 control channel messages. Thus, Endpoint1 sets up the outgoing H.245 channel from its side by sending the first message, TerminalCapabilitySet. This message advertises Endpoint1's resource capabilities and contains its transport address for the channel, which in turn enables Endpoint2 to open its outgoing side of the channel.

Note that the Control Channel H.245 could have been opened as soon as the transport address of the Endpoint2 H.245 channel was sent. This can happen in any of the Call Proceeding, Alerting and Connect messages. The H.245 channel may also have been established by the Endpoint2 in the case that Endpoint1 had sent its H.245 transport address during setup.

The requirement is that one endpoint starts listening for H.245 messages on some transport address and advertises this address, through the H.225 call signaling channel (since only this exists so far) to the other endpoint. The other endpoint does the same, but instead of sending back its address through the H.225 channel, it sends it on the just created half-duplex (only one endpoint listens) H.245 channel. This first message is the TerminalCapabilitySet, which contains other information as well and enables the receiving endpoint to establish its side of the H.245 channel.

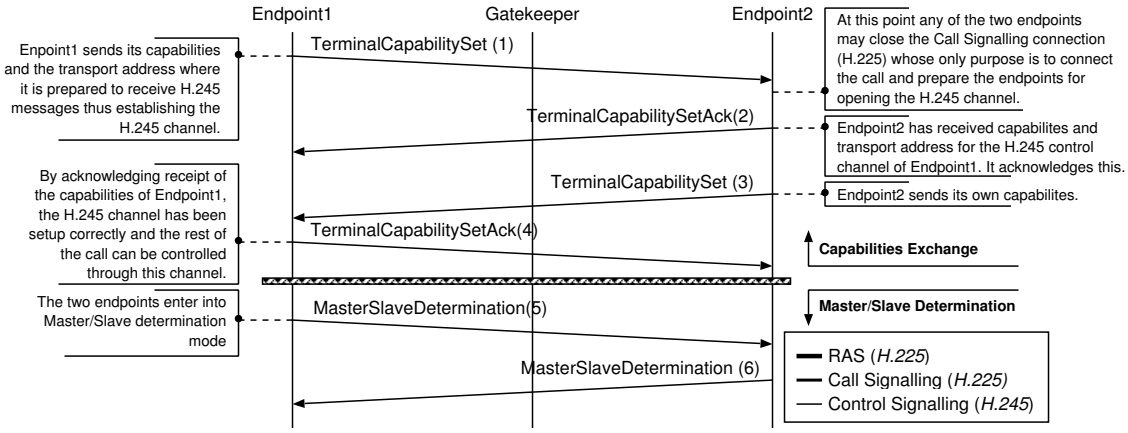

Figure 5: Phase B: Initial Communication and Capability Exchange – Direct Signaling

2.3.4 Phase C: Establishment of Audiovisual Communication

At this point, the control channel H.245 has been established, the endpoints know each other's capabilities and they have determined master/slave relations. The endpoints proceed to open logical channels for the exchange of media streams.

2.3.5 Phase D: Call Services

Everything is ready for the actual transfer of media packets, whether they are audio, video or data. The protocol itself is not involved in the exchange of the actual media streams. It lets

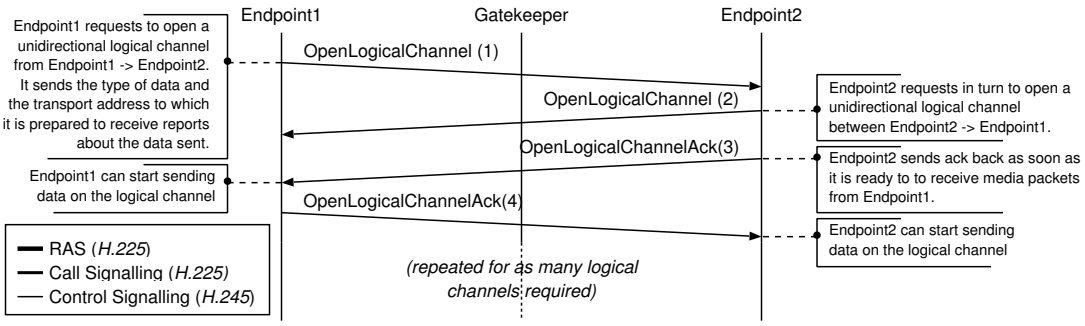

Figure 6: Phase C: Establishment of Audiovisual Communication – Direct Signaling

other protocols, namely RTP [3], to handle the transfer.

However, for any other service during the call, H.245 messages are used between the two endpoints and RAS messages when talking to the gatekeeper. Such services include:

- *Bandwidth Changes.* At any time during the call session, either the endpoints or the gatekeeper can request bandwidth increases/decreases for the media streams.
- *Status.* The gatekeeper needs to query the endpoints occasionally for liveness.
- *Conference Expansion.* Endpoints may be added/removed from the conference session any time during the call.
- *Supplementary Services.* These services are described in H.450 and are similar to services provided by traditional circuit switched networks, such as call hold, call transfer, call park, etc.

2.3.6 Phase E: Call Termination

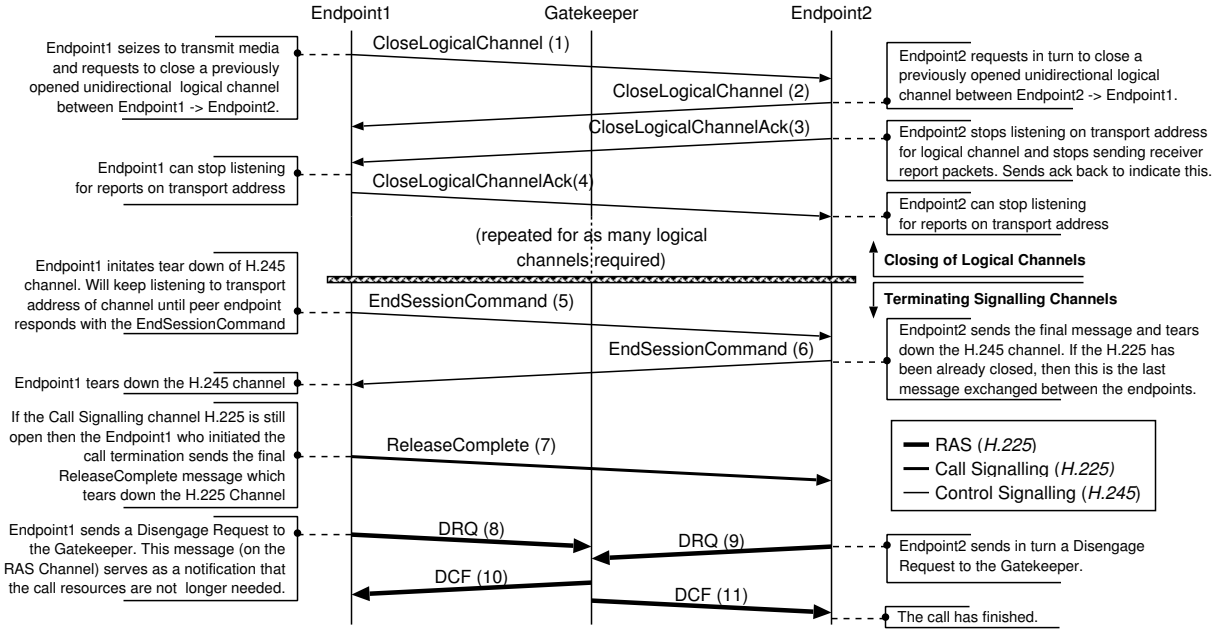

Figure 7: Phase E: Call Termination – Direct Signaling

At any point, either endpoint may terminate the call. The endpoints must close all the logical channels they opened for media exchange, close the H.245 channel, close the H.225 channel if still open and inform their respective gatekeepers about the end of the call.

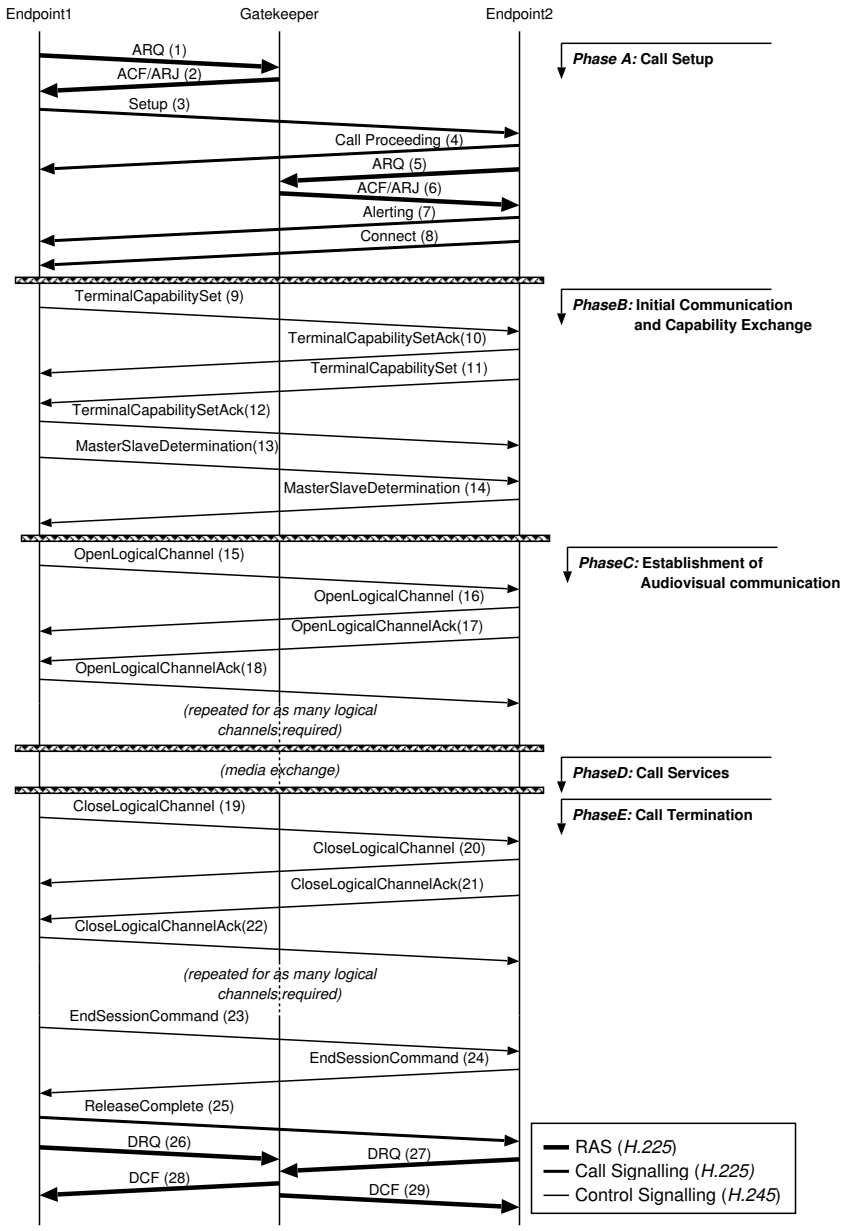

Figure 8: Complete Call (All Phases) – Direct Signaling

2.4 Advanced Functionality

In the previous section, we presented the essential functionality of the protocol in a straightforward but also a somewhat simplistic manner. Without altering the spirit of the protocol, advanced features were introduced at different stages of the protocol's evolution. These features have made it more flexible to administrative control, especially when multiple administrative domains are involved in a single call, and have succeeding in minimizing the rather large delay incurred during call setup.

2.4.1 Routing Signaling through Gatekeeper

So far, we have demonstrated the protocol with direct endpoint call signaling and direct control channel (fig. 2), i . e .both the call signaling channel H.225 and the call control channel H.245 were established directly between the two endpoints, without any intervening entity. The gatekeeper was restricted into admitting and releasing calls.

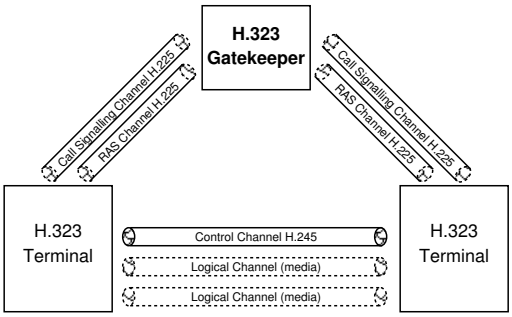

Figure 9: Gatekeeper routed call signaling with direct endpoint control channel.

It is, however, possible to route either or both of the channels through the gatekeepers involved in the call. One of the possible alternatives is to route the call signaling channel H.225 through the gatekeepers while still establishing a direct call control channel H.245 (fig. 9). In this case, the endpoints establish the call signaling channel H.225 with their respective gatekeepers, which are responsible for forwarding any call signaling messages received by them.

Another possible scenario is to route the call control channel H.245 as well through the gatekeepers (fig. 10). In this case, the two endpoints set up both the call signaling and the call control channels with their respective gatekeepers and route all messages through them.

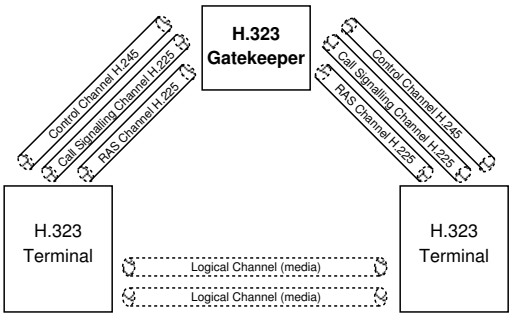

Figure 10: Gatekeeper routed call signaling and control channel.

In the case where the two endpoints are in different zones, and as a result have different controlling gatekeepers, more alternatives are possible. For example, one of the endpoints may route all its messages through the gatekeeper of the zone it belongs to (thus using the gatekeeper routed model) while the other endpoint may exchange its messages directly with the gatekeeper of the first endpoint (thus using the direct model because it bypasses its own gatekeeper).

The reasons for choosing among different routing schemes are multiple. In many cases, imposing constraints on the message routing scheme is necessary for billing and accounting

purposes. Or, the gatekeepers need to enforce tight bandwidth policing , or even for traversing firewalls behind protected domains, etc. We should note here, that it is the gatekeeper which specifies the routing scheme, during the call admission phase, that an endpoint must abide to during a call.

2.4.2 Optimization Techniques

The functionality of the protocol, as presented in section 2.3 involves a large overhead for a single call, even in the case of a basic point-to-point voice call. The setup of the call session, before which no media content can be exchanged, takes almost 6 to 7 roundtrips. This delay is most of the times unacceptable.

The call setup overhead incurs because, in order to open a logical channel for the transfer of media content, even of the most basic form, the two endpoints must exchange OpenLogicalChannel messages through an established H.245 call control channel. This channel, in turn, must be established by H.225 messages through the call signaling H.225 channel, which is the first channel that the two endpoints attempt to establish.

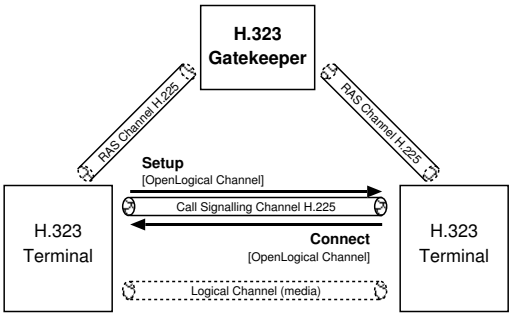

Figure 11: Fast Connect procedure

The first optimization, the Fast Cconnect procedure, introduced in H.323v2, allows endpoints to establish logical channels as soon as they exchange the first two messages that initiate the call signaling channel. This can be achieved early in the call setup phase (section 2.3.2 during which the calling endpoint is allowed to include OpenLogicalChannel elements in the initial Setup message. On receipt of the Setup message, the called endpoint can immediately start sending media content on the indicated channels and specifies which channels it accepts by including a special element in any message during this phase (i . e .until the Connect message is sent).

In this fashion, the two endpoints have established media channels before even the completion of phase A, i . e .before the receipt of the Connect message. This means that they can establish a basic point-to-point call with as few as one roundtrip message exchange.

If the two endpoints need to open additional channels, or need to use enhanced call features, they need to open a H.245 control channel through the usual procedure (section 2.3.3). Otherwise, they can continue using the channels they opened during FastConnect throughout the call and close them when they tear down the call signaling channel at the end of the call.

Another optimization, introduced in H.323v3, is to encapsulate all H.245 control messages in H.225 messages and send them through the call signaling channel, thus not requiring to establish a separate H.245 call control channel. This is also known as H.245 tunneling (fig. 12).

In both cases, the H.323 call signaling channel must remain open for the duration of the call. Note that FastConnect allows only the exchange of specific H.245 messages (only OpenLogicalChannel messages) and only during the first phase of call setup. In contrast, when

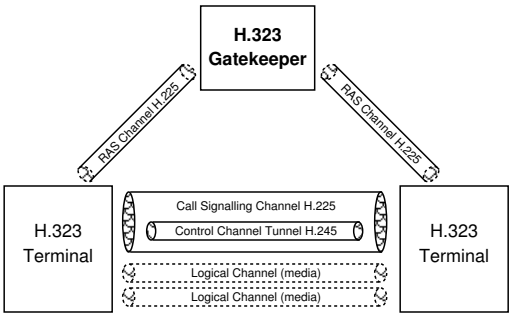

Figure 12: H.245 tunneling.

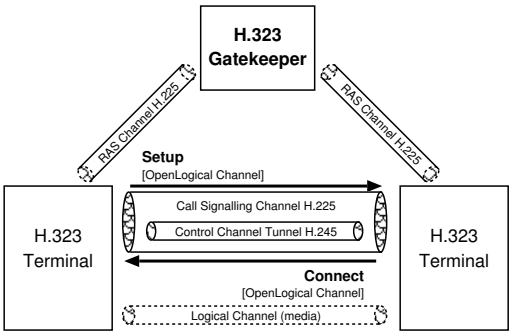

Figure 13: Fast Connect in parallel with H.245 tunneling.

using H.245 tunneling, the two endpoints behave as if they had indeed established a separate H.245 control channel, with the only difference that before sending a given H.245 message, it has to be encapsulated into a H.225 message and sent through the call signaling channel.

Occasionally, it may be desirable to combine those two methods and initiate H.245 tunneling in parallel with FastConnect (fig. 13). Since with FastConnect the two endpoints exchange only OpenLogicalChannel messages, there is no capability exchange and there are occasions that this may be required. Another reason is to have exchanged H.245 setup messages through the H.245 tunnel as fast as possible in case the FastConnect procedure fails.

2.4.3 Multiparty Conferencing

H.323 can be used to establish multipoint conferences. *–Needs more work.*

3 SIP Protocol Overview

SIP, which stands for Session Initiation Protocol, is an IETF application layer control protocol, defined in RFC 2543 [4], for the establishment, modification and termination of multimedia sessions with one or more participants. SIP makes minimal assumptions about the underlying transport and network layer protocol, which can provide either a packet or byte stream service with either reliable or unreliable service.

3.1 System Description

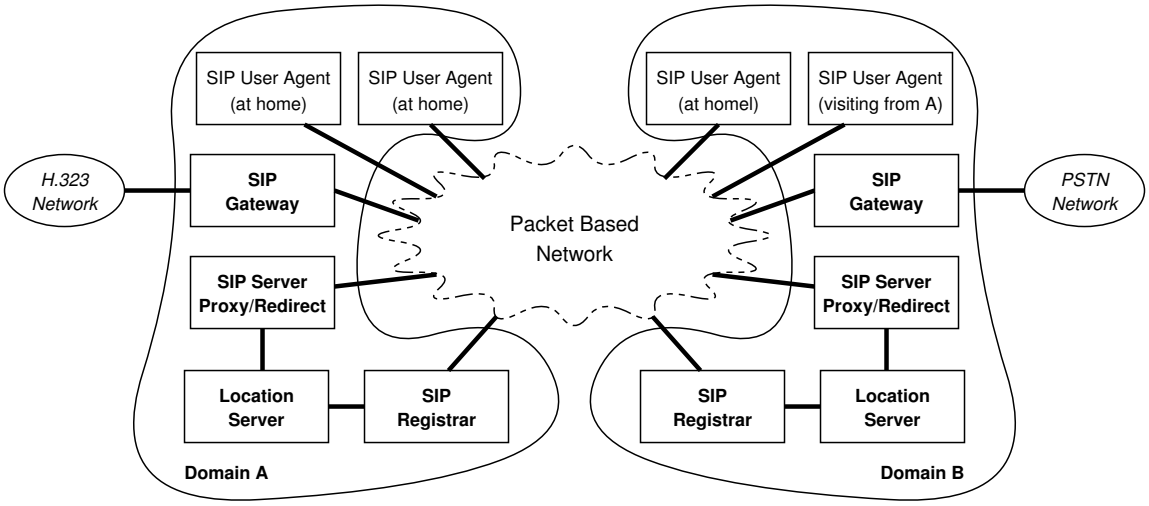

Figure 14:

A SIP system is based on a client/server model and is comprised of the following logical entities:

- A *User Agent (UA)* is an application that acts on behalf of the user, both as a client (User Agent Client) and as a server (User Agent Server). As a client it initiates SIP requests and as a server it accepts calls and responds to SIP requests made by other entities. The user agent is usually part of a multimedia terminal whose media capabilities it controls without having any media capabilities of its own.
- A *Registrar Server* is a SIP server that accepts only registration requests issued by user agents. A registrar server never forwards requests.
- A *Location Server* is a server which provides information to a proxy/redirect server about the possible current locations of a user. Usually, this entity is part of the proxy/redirect servers.
- A *Redirect Server* is a SIP server that provides address mapping services. It responds to a SIP request destined to an address with a list of new addresses. A redirect server doesn't accept calls, doesn't forward requests nor does it initiate any of its own.

- A *Proxy Server* is a SIP server that acts both as a server to user agents by forwarding SIP requests and as a client to other SIP servers by submitting the forwarded requests to them on behalf of user agents or proxy servers.

With the exception of the user agent, which is usually part of a multimedia terminal, the rest of the logical entities (registrar, redirect and proxy servers)a may be combined in a single application. Therefore, a single entity can act either as a proxy or as a redirect server, according to the SIP request, and at the same time accept registration requests. A SIP call is defined as the multimedia conference consisting of all participants invited by a common source.

Although not partitioned formally, the SIP system can be viewed as divided into domains each serviced by one redirect/proxy server and one registrar. A user agent has usually a home domain, which is specified by its address, but it can roam and use services in other domains as well, in which case it is considered to be 'visiting'. Otherwise it is considered to be "at home".

3.2 Signaling Approach

The SIP protocol follows a web based approach to call signaling, contrary to traditional telecommunication protocols. It resembles a client/server model, where SIP clients issue requests and SIP servers return one or more responses. The whole signaling protocol is built on this exchange of requests and responses, which are grouped into "transactions". Many of the SIP entities are comprised of both a client and a server side and the protocol has been designed in such a way that the entities can be either stateful or stateless.

SIP doesn't establish separate signaling channels for setting up and controlling the call. Instead, it defines the notion of transactions which consist of one request, sent by a client to a server, followed by zero or more provisional responses and one final response from the server. All the messages of a transaction share a common unique identifier and traverse the same set of hosts.

There are two types of messages in SIP; requests and responses. Both of them use the textual representation of the ISO 10646 character set with UTF-8 encoding. The message syntax follows HTTP/1.1, but it should be noted that SIP is not an extension to HTTP.

SIP defines a handful of request messages and a hierarchy of SIP responses. Each request and response method is comprised of header fields, which are either required, optional or not applicable, and a message body, which may be optional. Most of the times, the header fields are the ones that hold most of the information exchanged in the protocol. A subset of the header fields can be abbreviated by single letters, thus condensing the size of the messages. This form of compression is referred to as "tokenization".

When setting up a session, SIP messages need to describe the session characteristics to the peer user agent. SIP recommends but does not mandate the use of the the Session Description Protocol SDP, defined in RFC 2327 [5]. The session description is used for communicating the parameters required to establish the media channels for the transfer of the media content of the call session.

3.3 Basic Functionality

SIP is used for setting up, managing and tearing down multimedia conferences. It should be stressed that the actual delivery of the media content lies outside the scope of the SIP specification. SIP addresses the following aspects of multimedia communications:

1. *User Location.* The system provides means to determine the transport address where the user agent server of the called SIP endpoint listens for SIP requests.

2. *User Capabilities.* The system is responsible for determining the multimedia capabilities of each endpoint participating in the call and should ensure that they can communicate with each other if their capabilities are compatible.

3. *User Availability.* The system must determine if the called user is willing to engage into communication with the requesting endpoint.

4. *Call Setup.* The system should alert the user and configure both endpoints in such a way that the call can proceed.

5. *Call Handling.* The system, while not responsible for the transfer of the media content, should provide the means to modify the characteristics of the call session, such as adding/deleting media channels or call participants.

6. *Call Termination.* Finally, the system must terminate the call session upon the user's request and ensure that even when some endpoint doesn't follow the proper termination procedure, the call resources are released and the call session terminated.

A typical call session consists of a number of transactions between the user agents and the intervening protocol entities. The requests are issued from a user agent client, a proxy server acting on behalf of a user agent client or another proxy server. Each request prompts one or more responses from a user agent server or a proxy/redirect server that received the request. All the messages from the request until the final response constitute a transaction and can be exchanged directly by two entities or traverse one or more proxy servers along the way.

3.3.1 Methods

SIP makes use of the six request methods: INVITE, ACK, OPTIONS, BYE, CANCEL and REGISTER. The information of these methods lie within the header fields used. As an indication, the number of different header fields used in the various SIP requests are of the order of 50.

INVITE	This request is used to invite a user to participate in a multimedia session, with specific media characteristics. It is also used to modify an already established call session.
ACK	This request confirms that a user agent client has received the final response to an INVITE request.
OPTIONS	This request is sent to a server to query its capabilities.
BYE	This request indicates to the user agent server that the user agent client of another endpoint wishes to leave the call session.
CANCEL	This request is sent to abort a previous request.
REGISTER	This request informs the registrar of the user agent's current location, so it can be bound to the user agent's well known home address.

Table 1: SIP Request Methods

SIP requests are followed by one or more SIP responses. A response can be either final or provisional, the latter meaning that more responses should follow in order to complete the transaction. The responses are classified into six categories, based on an hierarchical error type classification. A minimal implementation

1xx Informational (provisional)	Request received, continuing to process request. The client should wait for further responses from the server.
2xx Success (final)	The action was successfully received, understood and accepted. The client must terminate any search.
3xx Redirection (final)	Further action must be taken in order to complete the request. The client must terminate any existing search but may initiate a new one.
4xx Client Error (final)	The request contains bad syntax or cannot be fulfilled at this server. The client should try another server or alter the request and retry with the same server.
5xx Server Error (final)	The request cannot be fulfilled at this server because of server error. The client should try with another server.
6xx Global Failure (final)	The request is invalid at any server. The client must abandon search.

Table 2: SIP Request Methods

3.3.2 Registrar Discovery and User Agent Registration

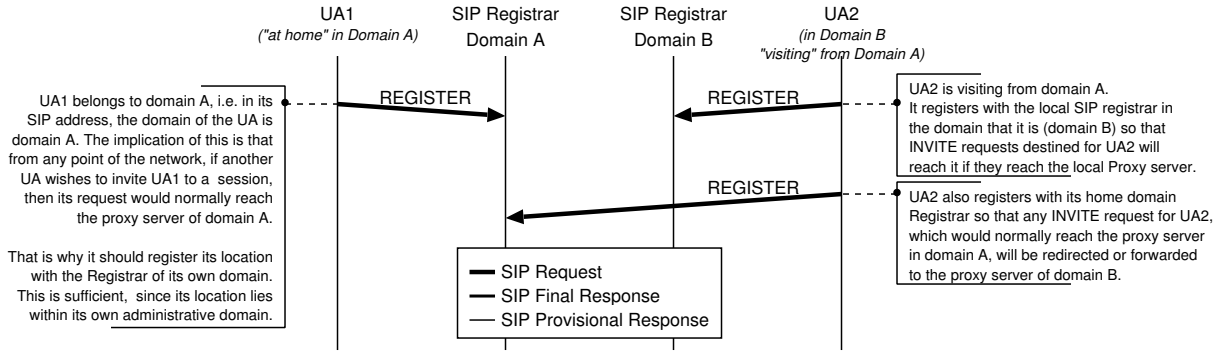

Figure 15:

Before receiving any calls through a proxy server, a user agent must register the address where it can be reached. This may be one or more addresses which are not restricted only to SIP addresses (for example it can point to a web page, email address, voicemail, etc). Furthermore, any call handling preferences may be specified during registration. Registration is required for the user agent to receive calls at its current location; not necessarily for placing calls. However, if the user agent needs to use the local proxy server for placing a call, the server may refuse to honor the request depending on its policy.

A user agent should always register its current location with its home registrar, regardless of the domain it is currently in. If the user agent is visiting another domain, it must, in addition to its home registrar, also register with the local registrar; otherwise it will not be able to receive any calls. Usually, a user agent should attempt to register periodically because registration requests expire after a specified amount of time.

A user agent needs to locate either the registrar or the proxy server of the domain in which it attempts to register. Most of the times the proxy server and the registrar are combined in the same application; otherwise, the proxy server knows how to forward the registration request to the registrar. So, it suffices for a user agent to locate the proxy server of the domain it wishes

to register. Two cases are distinguished, depending on which server the user agent attempts to register with:

1. *Local Server:* The user agent should attempt to register with the default outbound proxy server, if one is configured. If no server has been configured, then the user agent should attempt to send a registration request to the well known multicast address of all SIP servers ”sip.mcast.net” but scoped appropriately so it doesn't leave the local administrative domain.

2. *Home Server:* If the user agent is visiting a different domain, then this server is different from the local server. The user agent should attempt to register with its home server, deriving its address from its home address.

3.3.3 Call Session Establishment and Teardown

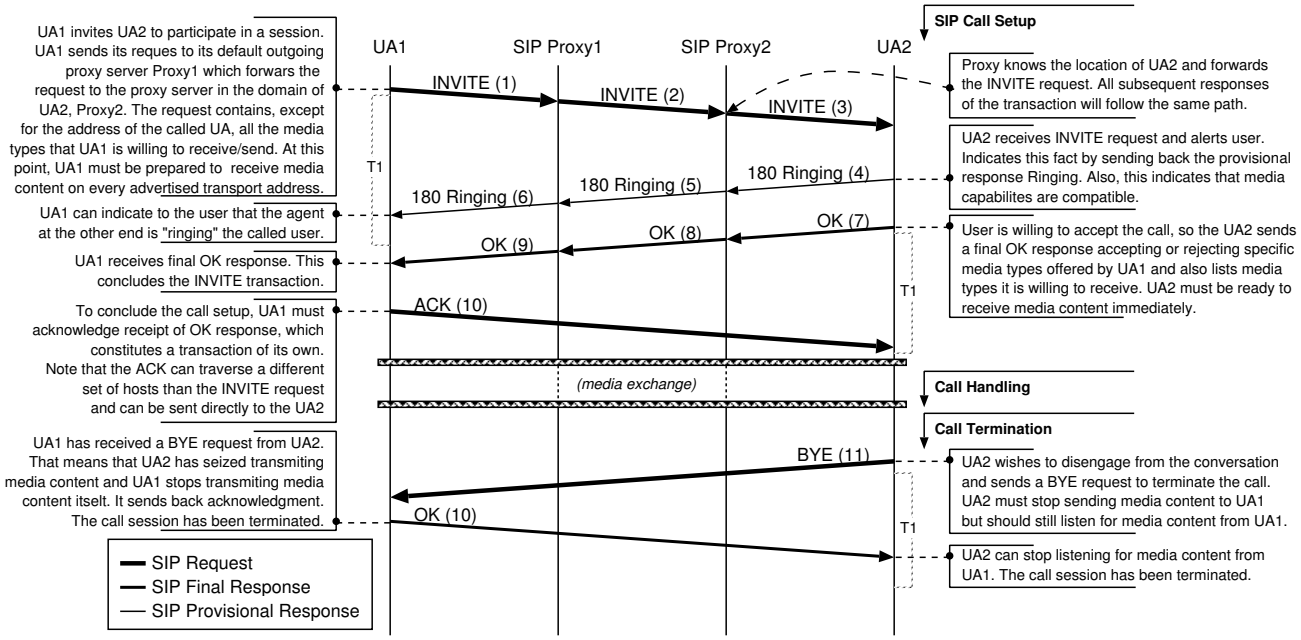

Figure 16:

In order to place a call, a user agent issues an INVITE request destined for the called user agent. This request is sent either through a default outbound proxy server, regardless of the destination address, or directly to the proxy server of the called user agent's home domain.

Each proxy may modify the INVITE request and then forward it or issue new concurrent or sequential multiple INVITE requests to locate a user agent at multiple locations. The INVITE request traverses one or more proxy servers before it reaches the destination user agent. Once this happens, the called user agent sends back (through the same route) zero or more provisional responses while it attempts to alert the user. Finally, it sends back a final response indicating the willingness of the user to accept the incoming call. This concludes the first SIP transaction of the session and has accomplished to locate the called user agent, alert the user and exchange media capabilities through session descriptors. The call is set up once the caller user agent acknowledges the acceptance of the call.

The two endpoints can engage now into conversation. If the call characteristics need to be modified at mid-session, then further INVITE transactions are needed to convey the new session description. When the conversation is over, either user agent initiates a BYE transaction which terminates the call session.

3.4 Advanced Functionality

3.4.1 Mutliple Redirect and Proxy Servers

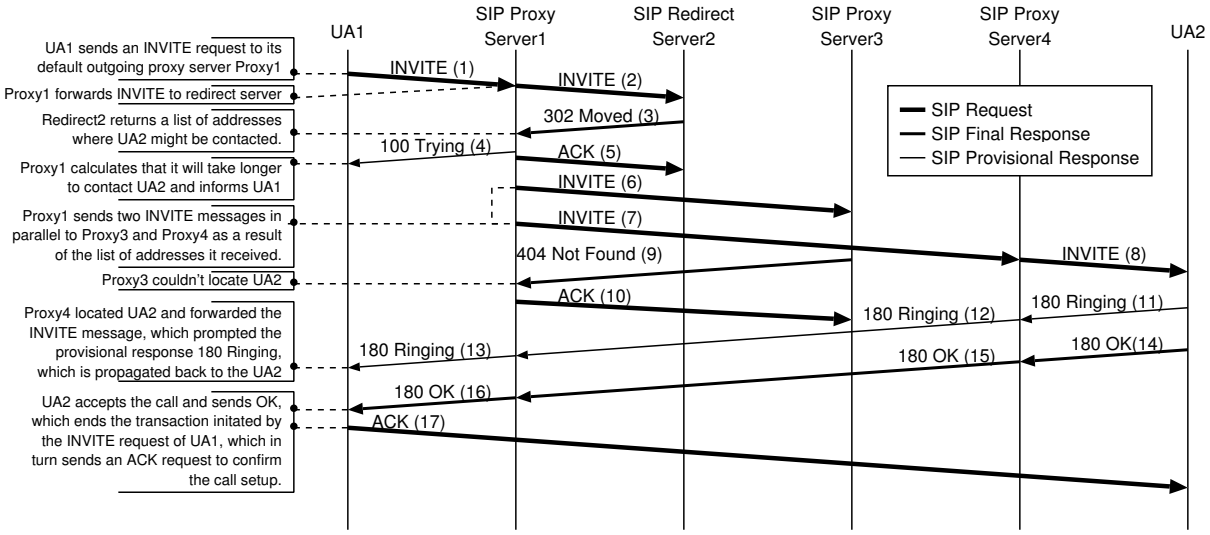

Figure 17:

So far we have seen the simple case where the intermediate servers always behave as proxy servers and forward the request. Another alternative is for the intermediate server to be a redirect server, in which case the INVITE request is not forwarded, but instead it is responded to with a list of alternative addresses for the initiating end to try. Whatever the case may be, the request reaches its destination user agent, if the latter has registered properly at its local and home server.

In figure 17 we present the case of two user agents attempting to set up a call session between them. UA1 sends the INVITE request to its default outbound proxy1. In turn, proxy1 forwards the request to redirect2 which instead of further forwarding the request, it returns a list of addresses where UA2 may be contacted. For example, this could be the home server for UA2 which is now visiting other domains. Proxy1, on receipt of the list of alternative addresses, launches a concurrent search for UA2 by sending INVITE requests to both proxy servers of the destination domains (proxy3, proxy4). Proxy3 denies the request because it cannot locate UA2 but proxy4 locates the called user agent and forwards the INVITE request. In the usual way, UA2 returns one provisional and the final response to UA1 through the same route, but UA1 chooses to acknowledge the call setup directly to UA2.

3.4.2 Multiparty Conferencing

SIP can be used to establish multipoint conferences, but for the time being it doesn't provide any floor control. *–Needs more work.*

4 Related Work

Since both H.323 and SIP have emerged as competing protocol standards, aiming at providing signaling and call control for IP telephony, it is expected that considerable attention should be drawn to broad and thorough comparative analysis of the two protocols.

In this section, we will review previous work comparing the two protocols by outlining their major points and examining their strengths and weaknesses. Then in the next section we will describe how we can supplement this work, by providing a more quantitave comparison of the two protocols using a combination of analysis, measurements and simulation of the protocol operations.

We will examine three major comparisons that can be found in the literature. The first was compiled by Nortel Networks in 2000 and compares the two protocols as candidates for inclusion in UMTS2000. The second is by Dalgic and Fang in 1999 which focus on features and characteristics most relevant to IP telephony and the third is by Schulzrinne and Rosenberg in 1998, two of the main contributors to the evolution of SIP.

With the exception of the first comparative work [9], most analyses restrict themselves to general conclusions on the various aspects of the two protocols, such as complexity, extensibility, scalability and services. Their conclusions are based largely on comparing, in a qualitative manner, the two protocols, such as contrasting the required number of round trips for each operation, speculating on the processing time of the fields in each protocols' headers, and stating observations that seem to be based more on the overall protocol behaviour rather than on measurements of the specific, real or simulated, protocol operation.

4.1 A Comparison of H.323v4 and SIP

By Nortel Networks, 2000 [9].

This technical report by Nortel Networks, is quite comprehensive and it aims at facilitating operators and vendors in selecting a control protocol for UMTS 2000. It compares SIP and H.323 version 4, based on complexity, extensibility, scalability, resource utilization, resource management, services and also considers how well each can perform in a wireless environment. Their main goal is to establish whether either call control protocol provides a significant advantage over the other in terms of the various categories mentioned above.

The report starts directly by comparing the two protocols on factors that, according to the authors, are important when choosing a protocol. The major criteria they use for their analysis are (i) time to market, (ii) estimated quantification of the work effort required and (iii) identification and quantification of the impact on the various network elements. They do not consider previous versions of H.323, and they assume that UDP is used in both protocols. Their results are quite comprehensive; they provide call flows of each protocol for many services, focusing on the case where a SIP or H.323 entity communicates with a UMTS 2000 entity and also, in some cases, they provide a few numerical results.

4.1.1 Complexity

- *Message Set.* The two protocols are similar but currently H.323 over UDP is not reliable and the mechanisms of Annex E must be used to provide reliability.
- *Encoding and Generation.* SIP compression/generation overhead is less (almost by a factor of two) than H.323 ASN.1 PER encoding/generation.

Criteria	H.323v1	SIP	Choice/Reason
Message Set	Complex, many messages for similar functionality	Logically numbered responses for extension, smaller set of messages for same functionality.	SIP - Time-to-market and extensibility
Debugging	Have to alter tools on each extension	Simple Tool developed once	SIP - Time-to-market and reduced complexity of development.
Re-use of code	H.323 and H.32x	SIP and Web	SIP - more modular
Service and Protocol	H.323 and H.32x	SIP and Web - more modular	SIP - more modular
Methods for implementing services	Can support all	Can support all	Equivalent
Distributed Call Signaling	Can Support	Can Support	SIP - Time-to-market / reduced complexity

Table 3: Complexity: Summary of results.

- *Decoding and Parsing.* The two protocols have the same overhead. If tokenized compression is used in SIP, then the overhead becomes less than that of H.323.
- *Debugging.* The authors favor SIP which in contrast to H.323 which requires special debugging tools, is simple to debug and has reduced complexity for development.
- *Implementing Services.* H.323 is less flexible than SIP, since H.323 has to add specific fields or parameter values in the signaling. In contrast, SIP combined with SDP for the session description provides this functionality transparently, due to its modular design.
- *Interworking with the PSTN.* The authors point out that the choice of signaling protocol is irrelevant to this issue, since most of the issues being debated concern inband streaming and QoS interactions.
- *Required Memory.* The stack size of SIP is smaller than the H.323 stack, thus lowering the memory requirements when using SIP.

4.1.2 Extensibility

- *Compatibility among versions.* SIP doesn't state explicit requirements for compatibility among versions, thus reducing code size and complexity. However it may have the adverse effect of newer versions not supporting features of older versions. H.323 on the other hand requires full backward compatibility, a fact that has resulted in very large code for H.323 implementations. The authors recommend that new implementations should not retain backwards compatibility for versions prior to H.323v4.
- *Feature evolution.* According to the authors SIP is much more flexible in defining new features and services, since it has built-in extensibility mechanisms, it is text based and is quite modular. On the other hand, H.323 is quite complex in defining new features and furthermore requires new vendor codes to be specified, a process that is quite time consuming.
- *Modularity.* SIP provides mainly user location, registration and basic session signaling. For further services and features other protocols can be used with SIP without making

Criteria	H.323v1	SIP	Choice/Reason
Version Compatibility	YES	YES - the Requires, Supported and ProxyRequire headers provide more flexibility than H.323.	SIP - more flexibility to support for multiple variants coexisiting.
Feature Evolution	Same as above	Same as above	Same as above
Operators in charge of own services	Less Ability - more complex ASN.1	Higher Ability - text formats and extension headers	SIP - Operators will be less dependent in vendors to add new services.
Modularity	Umbrella Standard - designed for limited feature set.	Modular, designed around other web technologies and can do GSTN services too.	SIP - built for web. H.323 originally derived from circuit world.
Codecs	Equivalent	Equivalent	Equivalent
Third party Call Control	Facility redirect	Also header	Equivalent

Table 4: Extensibility: Summary of results.

any changes to the basic protocol, thus providing large degree of modularity and flexibility, since even new headers can be added and pass through intermediate proxies and user agents. On the other hand, in H.323 there is no clean separation between its numerous subprotocols, which are closely intertwined to provide most of the built-in services.

- *Ability to work with existing an new multimedia codecs.* SIP requires the codec to be registered with IANA before it can be used. Any person or group may register such a codec with IANA. H.323 makes no such requirement and a new codec can be used with no modifications of the H.245 syntax.

- *Third-Party Call Control Mechanisms.* This can be supported easily and concisely in SIP. While it is also possible to support it in H.323, there is no standard comprehensive way to do it.

4.1.3 Scalability

- *Support of large number of domains.* Both protocols seem to be equivalent. In both SIP and H.323 the burden falls on the main network server (proxy server for SIP, gatekeeper for H.323), the underlying transport layer and the way peer entities communicate. They both accommodate different topologies (flat, hierarchical) and both can make use of various location and translation mechanisms suitable for global deployment.

- *Ability to handle large number of calls.* The authors note that this is primarily implementation/deployment specific. Both protocols can function in a stateless manner. However, most H.323 implementations are likely to be stateful. They also note that SIP takes less CPU cycles to generate signaling messages, thus they hypothesize that a server could handle more transactions.

- *Maintaining of call states effect on scalability.* Both protocols support stateless and stateful operation, but most H.323 carrier grade implementations are designed to be stateful and use hot-sparring techniques, which increases the complexity. It is generally

Criteria	H.323v1	SIP	Choice/Reason
Wide Area Support	YES	YES	Equivalent
Large Number of Calls	YES	YES	Equivalent
Call States	Can do both	Can do both	Equivalent
Elements that must maintain states	Clients, MC, MGCF-CSCF optional	UA, MC, MGCF-CSCF optional	Equivalent
Message Processing	More processor overhead, smaller messages	Less processor overhead, larger messages	Comparable - bandwidth vs. component complexity decision
Conferencing	All modes - H.224 floor control	All modes - GCCP, SCCP or even H.224 floor control	Comparable - no RFC exists indicating which to use for SIP.
DCS	Would have to be altered more than SIP	A closer original fit.	SIP - Time-to-market

Table 5: Scalability: Summary of results.

assumed that when a protocol entity retains states the scalability decreases as the same server can handle fewer transactions since each transaction requires more resources. Most SIP implementations on the other hand are designed to be stateless.

- *Conference sizes, conference control.* H.323 initially supported only centralized conferencing control mechanism but newer versions allows an application layer mutlicast conference concept which is better but still doesn't scale well for multipoint conferences. SIP on the other hand is based on distributed conference control, thus larger conferences can be supported.

4.1.4 Resource utilization and management

Criteria	H.323v1	SIP	Choice/Reason
Air-link bandwidth	Smaller Messages	Larger Messages	H.323 - smaller messages
CPU	More processing	Less processing	SIP - less processing
QOS/RRM Interactions	Same Issues	Same Issues	Equivalent

Table 6: Resource utlization and management: Summary of results.

- *Resources required during a call.* (i) Messages: Textual formats used in SIP are less space efficient than ASN.1 PER used in H.323 messages. However, the SIP parser is fairly simple and occupies much less space than a general ASN.1 decoder. (ii) Protocol Stack: SIP/SDP are less complicated than the H.323 protocol suite. (iii) Stateful operation: Both protocols require roughly equivalent resources for maintaining call states.

- *Compression gains.* The authors measure the compression gains on the messages of each protocol. For H.323, since the messages use ASN.1 PER aligned encoded rules, the compression gain is minimal. Using LZ77 compression, the gain ranges between 1-3

In the case of SIP, the authors first used tokenization of the header fields (replacing long header field strings with one character tokens). They observed a gain ranging between 13-19Then, they applied a common text compression technique and the additional gains ranged between 18-22SIP the usage of tokenization, since it does not incur additional CPU overhead. However, they recommend against compressing further, since they expect the compression/decompression overhead to impose additional burden on the system.

4.1.5 Services

Criteria	**H.323v1**	**SIP**	**Choice/Reason**
Supported Services	H.323 more explicitly defined	SIP defined in whitepapers and drafts	Equivalent but H.323 has better standardization
Delay Times	Equivalent - still issues with use of UDP and reliability	Equivalent	Equivalent
Billing	Needs work - to be embedded in protocol	Needs work - to use a separate protocol	Comparable
GSTN services	YES	YES	SIP - Time-to-market / less code
Capabilities Exchange	Better for media - worse for signaling extensibility	Worse for media - better for signaling extensibility	SIP - signaling is more of an issue
Personal Mobility	Added nomadicity in v3 - location based services still ongoing	Designed for nomadicity - location based services still ongoing	Comparable
Legacy interoperability	H.246	Draft status	H.323
IP telephony interoperability	Monolithic / OS bundled client	DCSGROUP / MGCP / SDP	SIP
Security	H.235 added later. Worse for firewall traversal using UDP.	Designed for it originally. Better for firewall traversal,	Comparable

Table 7: Services: Summary of results.

- *Services supported.* Both protocols provide almost the same services. In H.323, the services supported are standardized in the H.450 series of specifications, while in SIP services are not defined rigorously in the main RFC but are left to white papers and other informational RFCs.

- *Delay times to acquire services.* Using UDP, call setup delay is equivalent in SIP and H.323 if the FastConnect procedure is used in the latter. H.323 differs by setting up in parallel a backup TCP connection while SIP sets up the TCP connection sequentially, after the failure of UDP.

4.1.6 Conclusions

The authors then proceed with a list of comparison questions and answers aimed primarily at clarifying and supplementing the analysis in the first part. Many details are provided as well as quite a few message flows for many of the services. They mainly focus on issues regarding how each protocol interacts with the UMTS 2000 architecture.

Finally, the authors conclude by recommending SIP as their preference for a control protocol. They point out that even though H.323, unlike SIP, has currently more enterprise oriented and campus scale products deployed, SIP provides long term benefits which are related to and affect time to market, extensibility, multi-party service flexibility, ease of interoperability and complexity of development.

4.2 Comparison of H.323 and SIP for IP Telephony Signaling

By Dalgic and Fang, 1999 [10].

This paper by Dalgic and Fang is much less comprehensive than [9] but more detailed than [11]. It compares SIP and H.323 versions 1, 2 and 3, based on functionality, QoS, scalability, flexibility, interoperability and ease of implementation. However, it doesn't contain any quantitative results. Most conclusions rely on studying the behaviour of the protocols as described in their specifications; not as they behave in a real or simulated environment.

The paper gives a brief overview of the 2 protocols and then proceeds to compare the protocols on various categories. In most cases they present the detailed signaling exchanged for the two protocols for a specific service and then they juxtapose the two behaviours to draw their conclusions. Much work has been put into describing the call control services that each protocol provides. We summarize the main points of the analysis.

4.2.1 Functionality

- *Services.* Both protocols provide a rich set of services but each provides it with a different approach.
- *Call Control Services.* Most call control services (such as call hold, call transfer, call forwarding, call waiting etc.) are supported by both protocols.
- *Third Party Control.* Only available in SIP.
- *Capability Exchange.* The H.323 mechanism for capability exchange is much more precise and flexible than the corresponding mechanism in SIP.

4.2.2 QoS

- *Admission Control.* Provided by H.323v3 but not by SIP
- *Resource Reservation.* Not supported by any protocol, and both of them recommend using an external method such as DiffServ or IntServ.
- *Call Setup Delay.* Very large in H.323v1, shorter in H.323v2 with fast start and comparable with SIP in H.323v3. H.323v3 can use both UDP and TCP, making it as fast as SIP.

- *Error Detection and Correction.* Not an issue in H.323 v1 and v2 since TCP, a reliable transport protocol, is used. In H.323v3, where UDP can be used, a retransmission scheme is implemented to ensure reliability. In SIP a similar retransmission scheme is used.

- *Loop Detection.* SIP implements a loop detection algorithm similar to the one used in BGP (Border Gateway Protocol), which is much more efficient than the simplistic algorithm used in H.323v3. H.323 v1 and v2 didn't have any provision for detecting loop paths.

- *Fault Tolerance.* H.323 v3 provides better fault tolerance than SIP by redundant gatekeepers and endpoints.

4.2.3 Scalability

- *Complexity.* SIP is simpler to program and maintain than H.323 and therefore more scalable.

- *Server Processing.* H.323 v1 and v2 rely on TCP for reliability and in consequence must be stateful. H.323v3 on the other hand can be stateless, just as SIP, which lowers the server processing demands.

- *Endpoint Location.* H.323 uses alias mapped by the gatekeepers, while SIP makes use of a SIP URL.

4.2.4 Flexibility

- *Extensibility of Functionality.* SIP offers a more flexible extension mechanism through the use of a hierarchical namespace of feature names, while in H.323 extensibility is achieved through a vendor defined extension field.

- *Ease of Customization.* H.323 requires more interactions between its sub-protocols, and its size always increases since backward compatibility is required. On the other hand, SIP uses its header fields encoded in text, making customization much easier in this case.

4.2.5 Interoperability

- *Among Versions.* H.323 is fully backward compatible with all versions. In SIP newer versions tend to phase out older functionality if it is not used.

- *Among Implementations.* H.323 provides a "Implementers Guide" which clarifies the standard and smoothes interoperability problems among different implementations. SIP doesn't provide yet a similar implementation agreement.

- *With Other Signaling protocols.* The H.32x family of protocols fully specifies standards to interoperate with other protocols, namely circuit switched networks.

4.2.6 Ease of Implementation

- *Development time.* H.323 requires a special parser for ASN.1 syntax which complicates implementation and debugging. SIP text based message encoding allows easy implementation and debugging.

Functionality	H.323v1	H.323v2	H.323v3	SIP
Call Control Services				
Call Holding	No	Yes	Yes	Yes
Call Tranfer	No	Yes	Yes	Yes
Call Forwarding	No	Yes	Yes	Yes
Call Waiting	No	Yes	Yes	Yes
Advanced Features				
Third Party Control	No	No	No	Yes
Conference	Yes	Yes	Yes	Yes
Click-for-Dial	Yes	Yes	Yes	Yes
Capability Exchange	Yes-Better	Yes-Better	Yes-Better	Yes
Quality of Service				
Call Setup Delay	6-7 RT	3-4 RT	2-3 RT	2-3 RT
Packet Loss Recovery	Through TCP	Through TCP	Better	Better
Fault Detection	Yes	Yes	Yes	Yes
Fault Tolerance	N/A	N/A	Better	Good
Manageability				
Admission Control	Yes	Yes	Yes	No
Policy Control	Yes	Yes	Yes	No
Resource Reservation	No	No	No	No
Scalability				
Complexity	More	More	More	Less
Server Processing	Stateful	Stateful	Stateful or Stateless	Stateful or Stateless
Inter-Server Communication	No	No	Yes	Yes
Flexibility				
Transport Control Neutrality	TCP	TCP	TCP/UDP	TCP/UDP
Policy Control	Yes	Yes	Yes	No
Resource Reservation	No	No	No	No
Interoperability				
Version Compatibility	N/A	Yes	Yes	Unknown
SCN Signaling Interoperability	Better	Better	Better	Worse
Ease of Implementation				
Protocol Encoding	Binary	Binary	Binary	Text

Table 8: Summary of comparison results as presented in the Dalgic-Fang paper.

4.2.7 Conclusions

The paper's major conclusions are as follows. In terms of functionality and services that can be supported, H.323v3 and SIP are very similar. However, supplementary services in H.323 are more rigorously defined and therefore fewer interoperability issues are expected to arise. Furthermore, H.323 has better compatibility among its different versions and better interoperability with the PSTN. The two protocls are comparable in their QoS support (similar call setup delays, no support for resource reservation or class of service (QoS) setting), but H.323v3 will allow signaling of the requested QoS. On the other hand, according to the paper, SIP's primary advantages are its flexibility to add new features and its relative ease of implementation and debugging. Finally, the authors note that H.323 and SIP are improving themselves by learning

from each other, and the differences between them are diminishing with each new version.

4.3 A Comparison of SIP and H.323 for Internet Telephony

By Schulzrinne and Rosenberg, 1998 [11].

This is a less recent paper written in 1998 by Schulzrinne and Rosenberg, two of the major contributors to the SIP protocol [4]. It compares SIP and H.323 versions 1 and 2. It is mostly a qualitative and descriptive comparison of the two protocols regarding complexity, extensibility, scalability and services. The paper doesn't go in depth but provides a general insight on the differences between SIP and H.323.

According to the paper, H.323 embraces the more traditional circuit switched approach to signaling, based on the ISDN Q.931 protocol, which, as most traditional telecommunication protocols, suffers from complex signaling and call control. As an example, they present the H.323 call setup sequence of events, in which the H.225 protocol is used initially for connection establishment which in turn sets up the call control channel using the H.245 protocol which, once established, sets up and tears down the logical channels which carry the media. If even more complex functionality is required, additional protocols have to be used, such as H.332 for large conferences, H.450 for supplementary services, H.235 for security and H.246 for interoperability with the circuit switched services. This tactic of having multiple protocols coexist and interoperate for controlling one call is typical of most telecommunication protocols.

H.323	SIP
Quite Complex	Quite Simple
Long Documentation	Short Documentation
Hundreds of elements	Headers less than 40
Long implementation time	Short Implementation time
Binary representation ASN.1	Text, similar to HTTP, RTSP
Field parsing complicated	Field parsing and generation simple
Debugging hard	Debugging straightforward
Difficult to reuse code	High code reuse
Several protocol components	One simple protocol component
No clean separation, services are duplicated and require interactions between several of them	
Complicated firewall traversal	Single request, stateless, easy to pass through firewall
3 modes of operation	1 mode of operation

Table 9: Summary of comparison results of Schulzrinne paper.

On the other hand, according to the authors, SIP takes a much simpler approach by reusing many of the header fields, encoding rules, error codes and authentication mechanisms of HTTP, simplicity which is typical of a packet network approach.

Another point is made on the fact that, in both cases, the protocols restrict themselves into setting up, managing and tearing down the call. The actual media exchanged is handled through RTP, so that the choice of protocol doesn't influence Internet Telephony's quality of service.

The authors conclude that SIP provides a similar set of services to H.323, but provides far lower complexity, rich extensibility, and better scalability. They point out that future work is due to more fully evaluate the protocols, and examine quantitative performance metrics to characterize these differences. They also imply that a study measuring the processing overhead of SIP and H.323, would be quite useful.

5 Planned Work

By studying three major comparative analyses [9, 10, 11] of H.323 and SIP and an analysis of the call setup delay in IP telephony citeeyers00:delay, it became apparent that there is need for further measurements and quantitative comparison results. The two protocols have been compared in a fairly comprehensive manner regarding services supported, extensibility, complexity, functionality and interoperability issues. However, when it comes to Quality of Service, Scalability and Resource utilization and management, the results tend to be based on the intuitive and expected protocol behaviours.

It seems as a logical extension to supplement this work with a rigorous and thorough comparative analysis of H.323 and SIP on the aspects of quality of service, scalability and resource utilization and management. We have followed the guidelines in [12] to perform our performance analysis.

5.1 Assumptions

In this comparison, we will only focus on H.323v4, and we will use the most efficient way for call setup that this version provides. We will ignore any issues involving the actual transfer of media, since it is irrelevant in both protocols for the call setup. We will be referring to the whole duration of the multimedia call between two or more users as ”multimedia session” or plainly ”session”.

5.2 Goals

In order to compare H.323 and SIP, the following aspects of each protocol will be evaluated and then we will compare the corresponding metrics between two protocols. We always refer to per session metrics.

1. ***Call Setup Delay***
    - How long is it?
    - How is it affected by the transport protocol (TCP or UDP)?
    - How is it affected by the current load of the intervening protocol entities?
2. ***CPU and Memory Requirements***
    - Measure the CPU time for the execution of the protocol at the terminal.
    - Measure the aggregate CPU time for the execution of the protocol at all the intervening protocol entities.
    - Measure the aggregate CPU time for the execution of the protocol at all the intervening protocol entities and the participating terminals.
    - How is the CPU load affected by:
        - Tokenization
        - Compression after tokenization
        - Preservation of states
3. ***Network Resources***
    - How much bandwidth is consumed?

- How many transport addresses are required?
- How soon are the transport addresses available again after the termination of the session?

Every attempt will be made to compare the metrics that were measured under similar configurations. However, since the two protocols follow completely different approaches, there will be cases where a comparison of ”semantically” equivalent configurations would not provide much insight, in spite the fact that similar configurations exist among the two protocols. That may be the case when a certain configuration is described in the protocol specification but is not implemented in practical systems.

In this case we will proceed mainly with the comparison of the ”semantically” equivalent configurations, but we will also provide in some instances, as a special case, the comparison of corresponding ”usage” equivalent configurations. A notable example of this would be when comparing the two protocols for stateless operation. Even though, as of version 3, H.323 can operate in a stateless manner, this seems not to be reflected into the implementations which are almost in their entirety stateful.

5.3 System under study

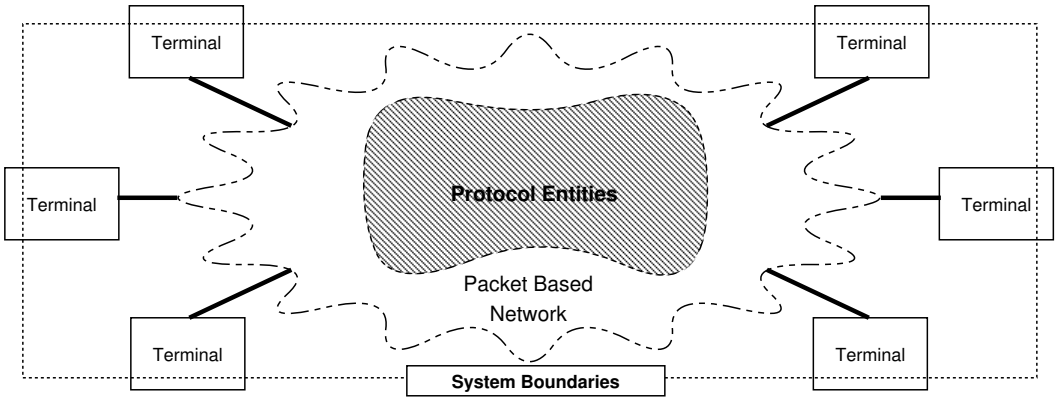

Figure 18: System boundaries

5.3.1 System Definition

This case study aims at comparing the performances of two application layer control protocols which provide signaling and call control services for multimedia sessions with one or more participants over a packet based network. The sessions may be as simple as subtoll quality voice calls or as complex as electronic whiteboard multimedia sessions with high quality voice, video and data streams.

The key component under study is the signaling protocol which is responsible for creating, maintaining, modifying and terminating the sessions. The signaling protocol in our case may be either the ITU recommendation H.323 or the Session Initiation Protocol of the IETF. The system consists of multimedia terminals and protocol network entities which are all connected via a packet based network.

The multimedia terminals can have varying capabilities, ranging from voice only to full audio/video/data capabilities. These terminals are usually applications that run on multi-purpose

computers or on specialized hardware and act as agents on behalf of the user by issuing and responding to protocol requests.

The protocol network entities are applications running on various machines scattered around the network, whose cooperation is essential for the implementation of the protocol between two or more terminal endpoints.,

The multimedia terminal endpoints, the protocol entities along with the underlying network that interconnects them constitute the system under study.

5.3.2 Services

The system provides signaling and call control services for multimedia sessions between endpoint terminals with varying capabilities. In other words, the system is responsible of establishing, managing and terminating multimedia calls between two or more terminal endpoints.

Specifically, we can summarize the services provided by the system as follows.

(a) *User Location Discovery.* Determination of the endpoint terminal where the user can be reached.

(b) *User Availability.* Determination of the user's willingness to engage in communication with the calling party.

(c) *Terminal Capabilities Exchange.* Determination of a minimum common set of capabilities required for the call to proceed.

(d) *Call Setup.* Establishment of a connection between two or more users and setup of the requested/required media channels.

(e) *Call Management.* Management of an ongoing call by adding/removing endpoints in the conversation and setting up additional media channels.

(f) *Call Termination.* Termination of the call session.

Among the services *not* provided by the system, we should mention the following.

- *Resource Reservation.* The system does not provide any means for reserving the required network resources for either the call setup or the media channel setup.
- *Media Transfer.* The system is responsible for setting up and controlling the multimedia call as well as preparing and negotiating the nature of the requested media channels. However, the actual exchange of the media is not handled by the system itself, as defined in our case study.

5.3.3 Metrics

For each signaling protocol, we will compare the response times, the throughputs and the resources consumed. The resources involved in our system are the computers on which the terminals and the protocol entities run and the network links that interconnect them. This leads to the following performance metrics:

I. **Response Time**

(a) *Call Setup Delay.* This delay is defined as the interval between the completion of the call request operation by the user (for a conventional telephone or ISDN network this would be the last dialed digit) and receiving ringback.

(b) *Dial-to-Ring Delay* This delay is informally defined in [13] as the interval between the completion of the call request operation by the user and the moment the callee's terminal rings. The definition is quite vague however, since the signaling protocol can start ringing multiple terminals where the user might be either sequentially or in parallel. – Needs further study

II. **Throughput**

(a) *CPU time at terminal.* This metric represents the CPU cycles spent, per session, for implementing the signaling protocol at the terminal.

(b) *CPU time at protocol entity.* This metric represents the CPU cycles spent, per session, for implementing the signaling protocol at a protocol entity.

(c) *Aggregate CPU time at protocol entities.* This metric represents the CPU cycles spent, per session, for implementing the signaling protocol across all the intervening protocol entities.

(d) *Total CPU time.* This metric represents the CPU cycles spent, per session, for implementing the signaling protocol across all terminal endpoints and the intervening protocol entities.

III. **Resources**

(a) *Bytes Exchanged.* This includes all packets used by the signaling protocol, per session.

(b) *Reserved Transport Addresses.* This pertains to the number of transport addresses reserved, per session at the terminals and the intervening protocol entities.

(c) *Reuse hysteresis of Transport Addresses.* Whenever a session terminates, the transport addresses it used are not available for reuse immediately. This metric represents the delay between the session termination and the moment the resource is available again.

This may not make much sense. I haven't found any reference discussing this issue, but it seems reasonable to me to look into it a little bit further.

5.3.4 System Parameters

The system parameters that affect the performance of a given signaling protocol, can be summarized as follows:

(a) Speed of terminal CPU and available memory

(b) Speed of protocol entity CPU and available memory

(c) Specific implementation of signaling protocol

(d) Operating system overhead for interfacing with the network

(e) Speed of the network

5.3.5 Workload Parameters

The workload parameters that affect the performance are the following:

I. **Signaling Protocol**

(a) *Type of signaling protocol.* The approach of each signaling protocol may result in increased or decreased workload for equivalent session signaling.

(b) *Type of transport protocol.* Since a reliable transport protocol, such as TCP, is designed to provide its services to a vast variety of applications, its parameters could not be possible fine tuned for every application. Thus, if an application uses instead, an unreliable transport protocol such as UDP and relies on its own customized retransmission scheme for reliability, then the delay performance is expected to be much better.

In our case, it would be interesting to study the performance of the two protocols in the case of UDP combined with a retransmission scheme to ensure reliability whenever it is explicitly required by the protocols.

(c) *Level of compression.* Signaling messages can be compressed before transmitted, thus decreasing the amount of traffic exchanged between the terminals and the protocol entities.

It would be interesting to study the tradeoff between smaller traffic and higher computation at the terminals and the protocol entities.

(d) *Number of protocol entities involved.* In all but the most basic session configurations there is need for at least one intervening protocol entity.

It would be desirable to study the effect of an increasing number of intervening entities in each protocol.

(e) *Signaling protocol routing scheme.* Both protocols provide a few alternatives for the routing of the signaling messages (directly between endpoints or through intervening entities). The actual scheme used may be dictated by the need for a specific feature or imposed by billing or political reasons.

Either way, it would be desirable to compare how each protocol performs with different routing schemes.

(f) *Signaling protocol mode of operation.* Both protocols support either stateless or stateful operation. By intuition we could argue that a stateless operation results in a higher call throughput for the intervening protocol entities. However, it is not clear how much overhead is added to the terminals, which in some cases may have computational and storage limitations. Furthermore, even though a protocol may have provisions for operating in a stateless manner, it may not enforce it in practice (H.323 for example).

For all those reasons, it would provide us with insight to study the behaviour of the system under different modes of operation.

(g) *Security measures.* Without any security provisions, the messages in both protocols face the risk of being intercepted, modified, dropped or duplicated. The obvious solution is to apply security mechanisms to ensure integrity, privacy and non-maleability of the messages. It would be interesting to compare how the overhead of such mechanisms affect the performance of each protocol.

II. **Call Session**

(a) *Time between calls.* Usually this metric follows a certain distribution that can be modeled after the behaviour of users in the telephone network, since in our study we are mostly concerned with calls made by human users.

(b) *Duration of call session.* This metric follows a certain distribution which can be speculated by comparing the corresponding distribution of the calls in a telephone network and taking into account the broader capabilities of the call (in contrast to the voice-only capability of the PSTN).

(c) *Session participation.* This metric represents the number of participants in the call session. We are interested in evaluating the effect of this number on the performance of each protocol.

(d) *Session characteristics.* This metric is used to convey the type of the call session. In both cases, the signaling protocol messages do not vary much, whether the call session comprises only of one voice channel or numerous multimedia streams. The protocol is not responsible for the transport of the media streams; it merely sets up and prepares the various channels for transport. However, it would be desirable to evaluate the effect of different session types in each protocol.

(e) *Characteristics modifications at mid-session.* This metric represents the modifications that may happen to the call session after it has been established.

III. **Other Loads**

(a) *Other CPU load at terminal and protocol entities* We must take into consideration that the CPU is not devoted entirely to implementing the signaling protocol.

(b) *Other load on the network* We must take into consideration, that along with the signaling protocol network traffic, we have also other unrelated traffic.

5.3.6 Factors

The key factors, selected among the system and workload parameters which we wish to vary, are the following:

1. *Type of signaling protocol.* Two types of signaling protocols, H.323 and SIP will be compared.

2. *Level of compression.* The messages exchanged between the terminals and protocol entities that implement the protocol can be exchanged in compressed form. We will study the effect of using this method.

   - For H.323 we will use two levels:
     (a) No compression
     (b) Compression
   - For SIP we will use three levels:
     (a) No compression
     (b) Compression
     (c) Compression after tokenization

3. *Number of protocol entities involved.* Both protocols can work without intervening protocol entities. However, most useful configurations require the cooperation of at least one protocol entity.

   Four levels will be used. One, two, four and six entities.

4. *Signaling protocol routing scheme.* Both protocols provide different alternatives for the routing of signaling messages through the protocol entities.

   - For H.323 we will use three levels:
     (a) *Gatekeeper routed* call signaling H.225 with *gatekeeper routed* control channel H.245.
     (b) *Gatekeeper routed* call signaling H.225 with *direct* control channel H.245.
     (c) *Direct* endpoint call signaling H.225 and *direct* control channel H.245
   - For SIP we will use two levels:
     (a) Signaling routed through SIP proxy server.
     (b) Direct endpoint signaling.

5. *Signaling protocol mode of operation.* We will use two levels, stateless and stateful mode of operation, which are supported by both protocols.

6. *Speed of the network.* Since the speed of the network is directly dependent on the distance between endpoints, three levels of distance will be used between endpoint terminals involved in the session.

   (a) Short distance (in campus)
   (b) National long distance (across countries)
   (c) International long distance (across continents).

7. *Time between calls* will follow an appropriate distribution – still under study.

8. *Duration of calls* will follow an appropriate distribution – still under study.

9. *Characteristics of calls.* The call sessions we are concerned with can involve any number of audio/video/data channels. Three levels will be used:

   (a) *Low complexity,* voice only sessions.
   (b) *Medium complexity,* voice and low quality video sessions.
   (c) *High complexity,* full fledged multimedia session.

10. *Session participation.* The participants in a call session can be two or more. We will be concerned with two levels of user participation. Call sessions between two users and sessions between four users.

5.4 Next Steps

1. Consider and document which combination of methods to use: analytical modelling, measurements or simulation. It seems very plausible to take measurements since real systems exist and are accessible. Perhaps it would be a good idea to investigate further the analytical part, just for gaining some experience in modelling a real system? It will probably not yield tangible results, but it might expose major inefficiencies of the protocols.

2. Create and document a measurement testbed. The objective is to set up limited real SIP and H.323 networks, decide on a traffic model for the number of incoming calls and their duration, supply the input to the real system and take measurements. The testbed would probably comprise of applications performing the tasks of the terminal endpoints and a few server applications residing on different machines for the intervening entities. We may want to run corresponding entities of the two protocols on the same machines and supply identical traffic to the two networks (SIP and H.323) simultaneously. For example we may have a server in Maryland, a server in Greece and have endpoints communicate between all sites (MD¡-¿GR, MD¡-¿MD, GR¡-¿GR, GR¡-¿MD). The measurements will be highly dependent on the time of day and also from possible downtimes of the interconnecting networks. But it is a starting point. –Needs more work

6 Measuring SIP and H.323 call setup delay

This section outlines the experiments needed to measure the call setup delay of the SIP and H.323 control protocols in the context of a real system in a WAN environment. These measurements will be subsequently used to compare the two control protocols.

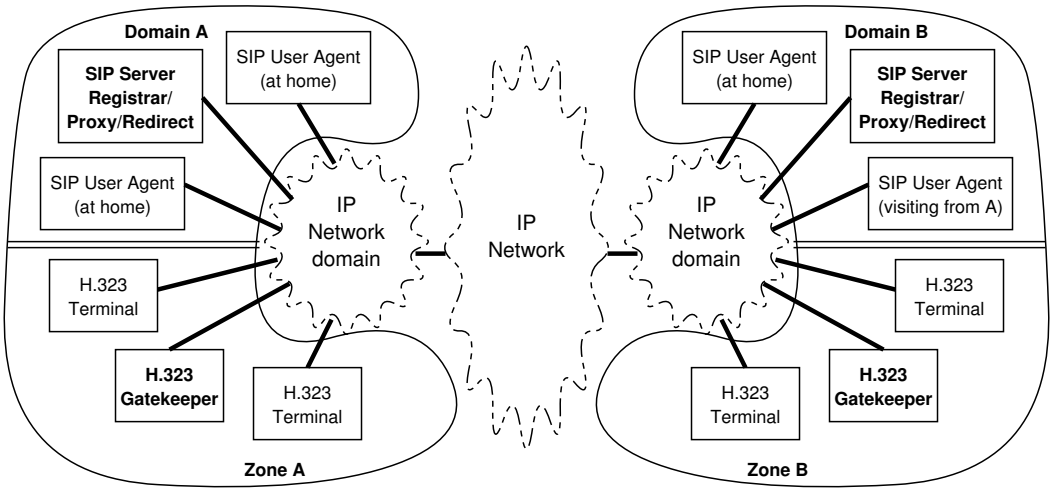

Figure 19: Testbed - General view

Our course of action will be to create initially two SIP domains and two H.323 zones, with almost equivalent entities and a pool of user agents/terminals in each case. Then, according to a traffic generation scheme, the user agents/terminals will be instructed to attempt to place calls to other agents/terminals and emulate a call session without actually transferring any media content.

In each case, we will be measuring the call setup delay, the distribution of the interarrival times, the sizes and types of requests to each of the intervening entities (proxy servers for SIP, gatekeepers for H.323) as well as the distribution of the UDP roundtrip delay and UDP losses between the two domains/zones.

6.1 Measuring SIP call setup delay

This experiment aims at measuring in an emulated real system the call setup delay incurred by the SIP control protocol. Our main goal is not to estimate an absolute value for the delay, but a relative value against which to compare SIP and H.323. This value can also be used to compare various modes of operation within SIP, such as stateful vs stateless operation, varying number of intermediate proxies, etc.

Since we will be taking measurements in a WAN environment, it is expected that the values measured will be statistically correlated to the behaviour of the UDP roundtrip delay and UDP losses. Therefore, any comparisons of different experiments should be made only when the UDP traffic patterns are highly correlated.

Another interesting point would be to determine if there is a point in the system's operation, beyond which the behaviour of the call setup delay deviates from the behaviour of the UDP traffic.

In this experiment we are solely concerned about the behaviour of the control protocol, i.e., we only generate the required traffic on the control channel used to setup a media call between two endpoints. Our system doesn't generate the actual media traffic exchanged between the terminals, since it doesn't affect the call setup delay in any way.

6.1.1 Testbed

The measurement testbed comprises of two SIP proxy/registrar servers connected through a WAN and two groups of user agents, each on the same LAN as the SIP servers. All calls should be routed through the SIP proxies which are stateful, i.e. they retain state information about the call requests they receive.

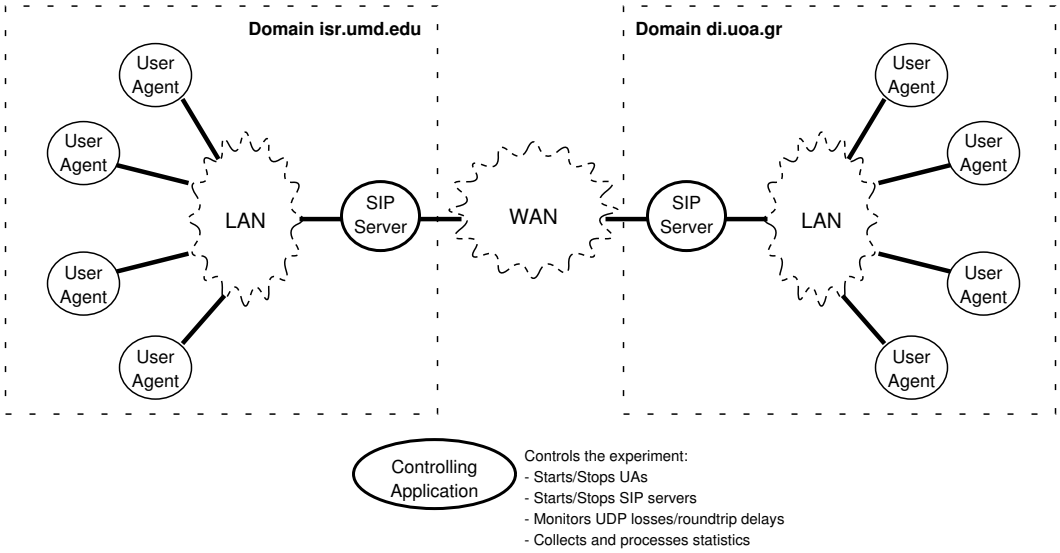

Figure 20: SIP Testbed

SIP Servers The SIP servers use location servers (in our case just an sql database) to determine how they should process an incoming call setup request. According to the SIP specification, the server can act either as a proxy or a redirect server. In the case of a proxy server, the request is forwarded to the suitable user agent if the agent is registered with the current proxy (regardless of whether it is in its home domain or just visiting). Otherwise, the request is forwarded to the agent's domain sip proxy server.

On the other hand, if the sip server acts as a redirect server, then instead of forwarding the request, it returns a list of locations (sip servers or user agents) where the user agent can be found. In this case, it is up to initiating client to forward requests to the new locations. A sip server determines whether to act as a proxy or a redirect server based on configuration settings and directions specified by each user agent during registration.

SIP User Agents Each user agent acts on behalf of a user by placing and receiving media calls to and from other user agents. There is no media content exchanged between the user agents, just control messages. Other than the absence of media content, the call will proceed

normally. A user agent initiating the call will issue through it's outbound proxy server an invitation to a call session, then the server will proxy the request the suitable user agent, forward it to another proxy server or return a list of new locations to try.

Once the call is established, the user agent essentially remains idle for the duration of the call, since in general only media content is exchanged during this time. However, control traffic may be generated if the call characteristics need to be modified in any way (inviting a third party, altering media channels) or the user agent is responding to new call session invitations (either by placing them in a queue or rejecting them).

In our testbed the users are emulated by a controlling application which will be instructing the user agents to place calls according to a certain distribution for the time between calls as well as the duration of the call.

6.1.2 Implementation

Since we are performing measurements of a real system, our results will rely heavily on the implementation. We will be using software from the University of Columbia, where SIP originated, as well as our own software to implement minimal user agents and the controlling application.

SIP Servers We are using the sipd SIP server from Columbia. This is a complete implementation of the specification and can be instructed to keep detailed log files which serves our purpose. The server uses a MySql database server for its location service. It has been installed successfully on the isr.umd.edu domain but not yet on the di.uoa.gr domain.

SIP User Agents The initial attempt to use sipc, the SIP user agent from columbia failed. The code is changing continuously and it uses too many other packages for the media support. It turned out that it was not worth the effort and a complete waste of time, since in our experiment we are not concerned at all with the media transfer.

Since the solution from Columbia was abandoned, it was necessary to write our own user agent client/server. These user agents should abide at least to the minimal requirements of the SIP standard. Because of the textual nature of the SIP standard the applications are not too hard to write. Basic support should be provided, i.e. support for the requests INVITE, ACK, SDP, BYE and the response classes 1xx, 2xx, 3xx, 4xx, 5xx, 6xx.

Furthermore, the agents contain the added functionality of continuously placing calls according to an exponential distribution for the time between calls as well as the duration of the calls. They also perform complete logging of the SIP activity. However, no support for handling media is provided.

Controlling application This is not related to the standard, but is used to control the whole experiment. It performs the following tasks:

- Starts and stops the user agents in all locations
- Starts and stops the SIP servers in all locations
- Collects statistics about UDP losses and roundtrip delay between the two domains.
- At the end of the experiment it processes the statistics from the various entities.

6.1.3 Measurements

We are interested in the following measurements:

- *Call setup delay* . The main metric we wish to evaluate and use for comparing SIP with H.323 as well as various modes of operations of SIP.
- *Distribution of UDP losses and roundtrip delay.* We need this in order to make fair comparisons. Furthermore, the call setup delay should be closely correlated and any deviation should be properly explained.
- *Distribution of interarrival times of call requests.*
- *Distribution of packet sizes of call requests.*

Assumptions In this experiment, we assume the following:

1. SIP control protocol.
2. UDP for packet transport.
3. Two SIP servers serving two domains connected through a WAN.
4. The SIP servers are stateful.
5. Calls will have only two participants
6. Call session characteristics will be typical, i.e. involving one audio and one low-bitrate video media channel. Even though we are not concerned with the media content, we must handle the negotiation of the session characteristics, since they are part of the call session control.

Scenarios The following scenarios will be used:

1. Only agents in domainA place calls to agents in domainB, no requests will be redirected and all agents in domainB are in their home domain.
2. Only agents in domainA place calls to agents in domainB, no requests will be redirected but some agents in domainA are visiting from domainB (i.e. Any calls to these agents from domainA, will be served locally by the proxy server of domainA).

In these scenarios, two levels of compression will be used:

- No compression
- Tokenization (substituting long field names with one letter codes)

The number of user agents on each side will be: 20, 40, 60, 80, 100

References

[1] I. T. Union, "Packet-based multimedia communication systems," Telecommunication Standardization Sector of ITU, Geneva, Switzerland, Recommendation H.323, Nov. 17 2000.

[2] ——, "Abstract Syntax Notation One (ASN.1): Specification of basic notation," Telecommunication Standardization Sector of ITU, Geneva, Switzerland, Recommendation X.680, Dec. 1997.

[3] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," Internet Engineering Task Force, Request For Comments 1889, Jan. 1996, proposed Standard.

[4] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: Session Initiation Protocol," Internet Engineering Task Force, Request For Comments 2543, Mar. 1999, proposed Standard.

[5] M. Handley and V. Jacobson, "SDP: Session Description Protocol," Internet Engineering Task Force, Request For Comments 2327, Apr. 1998, proposed Standard.

[6] A. B. Johnston, *SIP: Understanding the Session Initiation Protocol*. Artech House, 2001.

[7] J. Rosenberg and H. Schulzrinne, "The IETF internet telephony architecture and protocols," *IEEE Network*, vol. 13, no. 3, pp. 18–23, May / June 1999.

[8] T. J. Kostas, M. S. Borella, I. Sidhu, G. M. Schuster, J. Grabiec, and J. Mahler, "Real-Time Voice Over Packet-Switched Networks," *IEEE Network*, vol. 12, no. 1, pp. 18–27, Jan. 1998.

[9] N. Networks, "A Comparison of H.323v4 and SIP," 3GPP S2, Tokyo, Japan, Technical Report S2-000505, Jan. 5 2000.

[10] I. Dalgic and H. Fang, "Comparison of H.323 and SIP for IP Telephony Signaling," in *Proceedings of SPIE. Multimedia Systems and Applications II*, ser. Proceedings of Photonics East, Tescher, Vasudev, Bove, and Derryberry, Eds., vol. 3845. Boston, Massachusetts. USA: The Internationl Society for Optical Engineering (SPIE), Sept. 20-22 1999.

[11] H. Schulzrinne and J. Rosenberg, "A Comparison of SIP and H.323 for Internet Telephony," in *Proceedings ofThe 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 98)*, Cambridge, UK, July 8-10 1998, pp. 83–86.

[12] R. Jain, *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, Inc, Dec. 1991.

[13] T. Eyers and H. Schulzrinne, "Predicting Internet Telephony Call Setup Delay," in *Proceedings of the 1st IP-Telephony Workshop (IPtel 2000)*, Berlin, Germany, Apr. 12-13 2000.