

MASTER'S THESIS

SIP (RFC 2543), an Implementation for Marratech Pro

Petter Tiilikainen

Civilingenjörsprogrammet

Institutionen för Systemteknik
Avdelningen för Programvaruteknik

**SIP (RFC 2543),
an Implementation for Marratech Pro**

Petter Tiilikainen

April 2000

Division of Software Engineering
Department of Computer Science and Electrical Engineering
Luleå University of Technology
S-971 87 Luleå, Sweden

Petter.Tiilikainen@marratech.com

Abstract

This Master Thesis discusses SIP, Session Initiation Protocol, which provides services for user location, determination of user availability and media negotiation for the setup of subsequent multimedia sessions.

Also discussed is the author's implementations of a few SIP components, and how they could be used together with Marratech Pro, an application for multimedia conferencing developed by Marratech AB.

SIP is then examined in terms of its relation to H.323, another protocol for setup and management of multimedia sessions, and the possibilities for SIP to coexist with this protocol.

Preface

This work has been done at the Marratech AB office in Luleå, Sweden between November 1999 and April 2000.

I would like to thank Claes Ågren, my supervisor, and everyone else at Marratech AB for ideas and support, Peter Parnes, my examiner at Luleå University of Technology, and last but not least, my family, for guidance and support throughout life.

I wouldn't be without you.

Luleå, April 2000
Petter Tiilikainen

Contents

Abstract	iii
Preface	v
1. Introduction	1
1.1 Goals	1
1.2 Thesis Outline	1
1.3 Technical background	1
1.3.1 SIP.....	1
1.3.2 H.323.....	1
1.3.3 Java.....	2
1.3.4 Marratech Pro.....	2
2. SIP	3
2.1 UserAgent	3
2.2 Proxy Server	3
2.3 Redirect Server	4
2.4 SIP Message Syntax	5
2.4.1 SIP URL.....	6
2.4.2 Methods and Response Codes.....	6
2.4.3 Headers.....	7
2.5 Finding the Server Location	7
3. SIP Implementation	8
3.1 Common SIP	8
3.1.1 TransactionHandler.....	9
3.1.2 Parsing a SIP Message and its Components.....	9
3.1.3 Test Results.....	9
3.1.4 Future Work.....	10
3.2 SIP UserAgent	11
3.2.1 UserAgent.....	11
3.2.2 UserAgentFrame.....	12
3.2.3 CallingDialog.....	12
3.2.4 IncomingCallDialog.....	12
3.2.5 Test Results.....	13
3.2.6 Future Work.....	13
3.3 SIP Redirect Server	13
3.3.1 Redirectory.....	13
3.3.2 LocationDatabase.....	14
3.3.3 Testing.....	14
3.3.4 Future Work.....	15
3.4 Conclusions	15

4. SIP and H.323	16
4.1 Interoperability interworking	16
4.2 The Shape of Things to Come	17
5. Conclusions	18
6. Bibliography	19
Appendix	20

1. Introduction

In the setup and management of multimedia conferences or calls, the ability to invite a single person, or a selected group of people, is desirable. SIP, Session Initiation Protocol [1], provides such mechanisms.

1.1 Goals

The goals of this thesis were to follow the standardization work on SIP and to produce an implementation of the current version of SIP that could be used together with Marratech Pro.

So that it would be easy to incorporate the results with Marratech AB's¹ products, the implementation was to be done in Java [2]. The implementation should also be verified against independent implementations, in order to determine its interoperability and verify the core functionality.

Finally, the relation to H.323 and the possibilities for SIP to coexist with this protocol were also to be investigated.

1.2 Thesis Outline

Chapter 2 describes the basics of SIP, **chapter 3** describes the implementation of the core SIP functionality, the SIP UserAgent and the SIP Redirect Server and **chapter 4** discusses SIP in terms of its relation to H.323.

1.3 Technical background

1.3.1 SIP

SIP is developed by IETF, and is an application-layer protocol operating on top of UDP or TCP. For transport with unreliable protocols such as UDP, SIP provides a retransmission scheme of its own in order to provide reliability. SIP must support SDP, Session Description Protocol [3], for the description of the multimedia sessions. RTP [4] is typically used for media transport.

1.3.2 H.323

H.323 [5] is an ITU-T recommendation, and acts as an umbrella for several other protocols, which together provide services similar to SIP. H.323 was initially designed for use with reliable transport protocols. Like SIP, H.323 uses RTP for media transport.

¹ <URL:<http://www.marratech.com>>

1.3.3 Java

Java is a platform independent, object orientated programming language developed by Sun Microsystems.

1.3.4 Marratech Pro

Marratech Pro² is an application for multimedia conferencing, offering real-time audio and video, together with shared application windows, whiteboard, web browser and chat.

² <URL:<http://www.marratech.com/products.html>>

2. SIP

The idea with SIP is that it should provide mechanisms that makes it just as easy to establish a multimedia call, as placing a normal single media telephone call.

The backbone of SIP consists of:

- **User Location:** Finding where to send the requests for a user specified in a SIP Uniform Resource Locator, SIP URL.
- **User capability exchange:** The determination of what media types and parameters to be used for the call, using Session Description Protocol, SDP.
- **User availability:** If the user is busy, not available, does not want to participate in the session.
- **Call setup:** Establishment of the call parameters at the different ends of a call.
- **Call handling:** Transfer and termination of ongoing calls.

To achieve this, there are a few different, distinct components defined within SIP, such as the UserAgent, Proxy and Redirect Server.

Other features of SIP are schemes for authentication and encryption.

2.1 UserAgent

The UserAgent, UA, consists of two distinct parts, the UserAgent Client, UAC, which sends requests and receives responses to those requests and the UserAgent Server, UAS, which receives requests and sends responses to those requests.

A UserAgent is usually involved in some kind of interaction with the user, but can also be used for voice mail, call forwarding and similar services without involving the user.

Fig. 2.1 illustrates an example of a typical call signal flow for a successful INVITE request between two UAs.

2.2 Proxy Server

A user wanting to contact a person based upon a SIP URL might be sending the request to a proxy, which then forwards the request as showed in Fig. 2.2.

A proxy can also be outbound, so that all outgoing requests has to go through it, which makes it possible for calls to be established between users behind firewalls.

The proxy usually offers some kind of user lookup procedure, such as forking requests to different destinations, or by simply doing an address lookup in a location database.

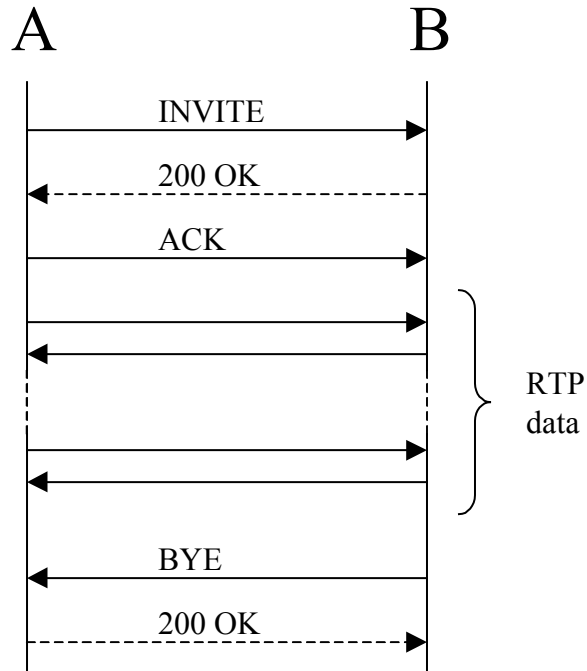


Figure 2.1: Call signal flow, UserAgent A sends INVITE request to UserAgent B

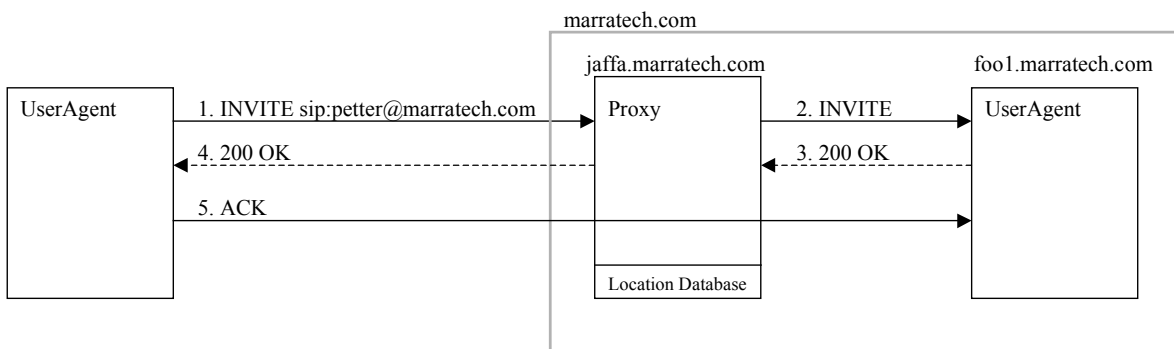


Figure 2.2: Call flow for an INVITE request through proxy

2.3 Redirect Server

The redirect server might be a part of the proxy server, for those cases where the user prefers requests to be redirected rather than forwarded. Fig. 2.3 shows an example of the call flow for a redirected INVITE request.

A redirect server can also be a stand alone component, and since it only responds to requests received, and never sends any requests of its own, it is simpler to implement than the proxy, yet it still offers similar services. However, it can only be taken into consideration if there is no firewall that the incoming requests have to get through.

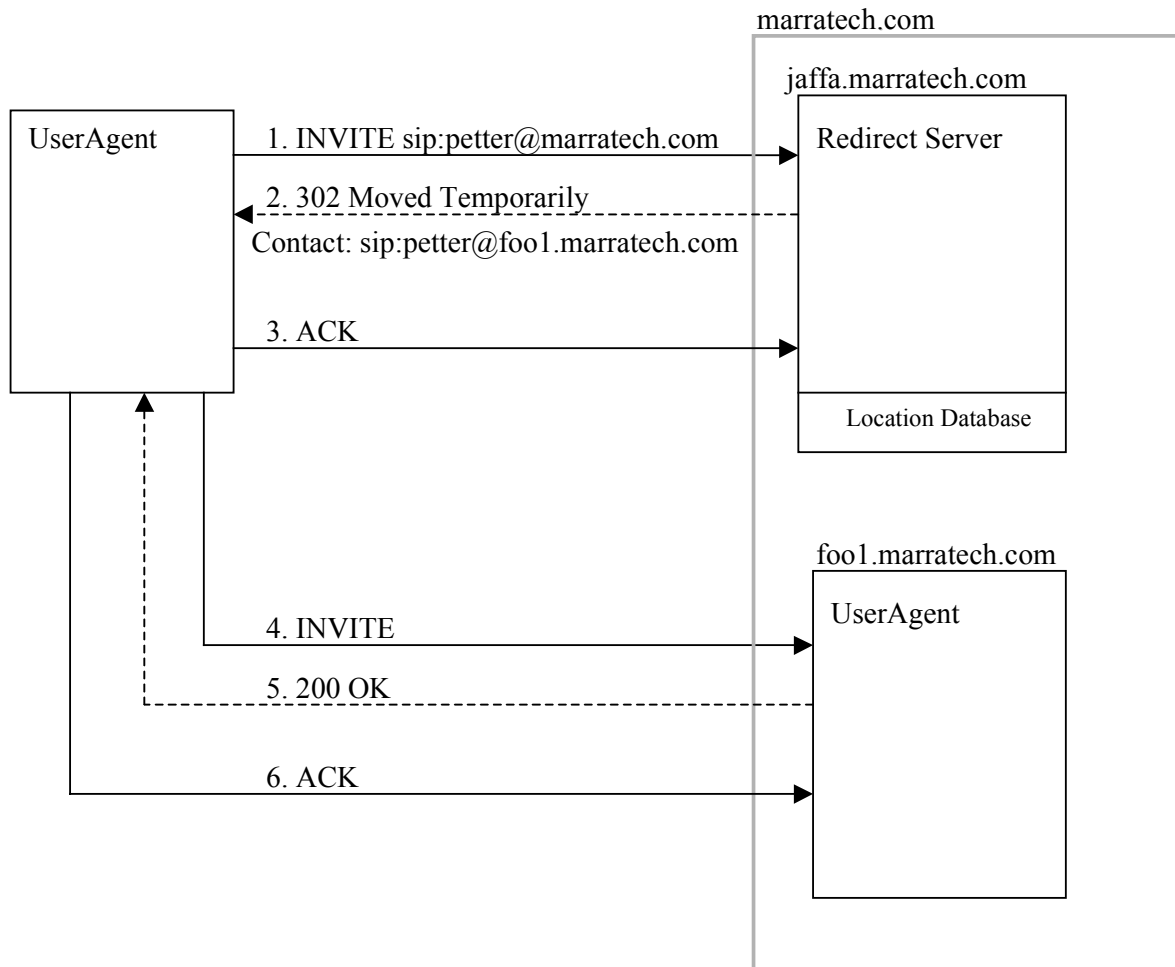


Figure 2.3: An INVITE request via a Redirect Server

2.4 SIP Message Syntax

SIP is a text-based protocol, and has syntax similar to HTTP. There are two different types of SIP Messages, requests and responses. The main difference between requests and responses are that the former has a method, defining the nature of the request, and a Request-URI, usually a SIP URL, indicating where the request should be sent, in its first line, while the latter has a response code instead.

All messages have headers of different types, sometimes associated with the type or subtype of the message. A message can also contain a body, where the length, type and encoding specified in different headers. Support of SDP in the message body is mandatory. There is an example of a basic INVITE request in Fig. 2.4, and an example of how the response to that request might look like in Fig. 2.5, if the request passed through a proxy before reaching its final recipient.

```
INVITE sip:classe@marratech.com SIP/2.0
Via: SIP/2.0/UDP jota12.sm.luth.se
From: Petter Tiilikainen <sip:pettii-6@student.luth.se>
To: Claes Ågren <sip:classe@marratech.com>
Call-ID: 2ef-1219425@jota12.sm.luth.se
CSeq: 1 INVITE
```

Figure 2.4: Simple INVITE Request Message

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP sip.marratech.com
Via: SIP/2.0/UDP jota12.sm.luth.se
From: Petter Tiilikainen <sip:pettii-6@student.luth.se>
To: Claes Ågren <sip:classe@marratech.com>;tag=3494-43a-344
Call-ID: 2ef-1219425@jota12.sm.luth.se
CSeq: 1 INVITE
```

Figure 2.5: 200 OK Response to INVITE

2.4.1 SIP URL

In many cases the Request-URI, From, To and Contact headers contain a SIP URL, which follows the guidelines for a generic URI [6]. For example, the SIP URL can consist of an e-mail address, such as `sip:petter@marratech.com`, or a host name, like `sip:sip.marratech.com`.

2.4.2 Methods and Response Codes

The different methods, defining the types of requests are:

- **INVITE:** Invite a user to participate in a session
- **OPTIONS:** Check for user capabilities and/or availability.
- **REGISTER:** Register a location with a registrar.
- **CANCEL:** Terminate a pending call
- **BYE:** Terminate an ongoing call

There are also several types of responses, defined by the different classes of status codes, which are:

- **Informational:** Such as 100 Trying, indicating that the server is trying, 180 Ringing, indicating that the UAS is alerting the user.
- **Success:** Only consists of 200 OK, which is a positive response to the corresponding request.
- **Redirection:** Such as 302 Moved Temporarily, used by redirect servers.
- **Client-Error:** 404 Not Found, 401 Bad Request and other types indicating a client side error.
- **Server-Error:** 500 Internal Error, 501 Not Implemented or some other response type indicating a server side error.
- **Global-Failure:** Indicating that the user could not be reached here or elsewhere, with 600 Busy Everywhere and 603 Decline as examples for status codes.

2.4.3 Headers

The five headers Via, From, To, Call-ID and CSeq are present in all messages. Others are only defined for use with requests or responses or subtypes thereof. Headers can indicate when a request expires, if the message body is encrypted, what language will be spoken during the session, among many other things.

2.5 Finding the Server Location

Where the request is sent depends on the contents of the SIP URL in the Request-URI, if it has the typical format of an e-mail address, SIP uses the DNS SRV RR [7] to find out where the server to send the request to is located.

An example of this would be if the SIP URL is sip:petter@marratech.com and the transport protocol the caller wishes to use is UDP, the DNS would then be queried for the SRV record for sip.udp.marratech.com. If everything goes well, the DNS provides an answer containing information about which port and what address the request should be sent to, in this case, port 5060 on sip.marratech.com.

3. SIP Implementation

The implementation work was an evolutionary process, although some of the major parts were identified beforehand, such as lower-level transport (UDP, TCP), application layer transport (SIP retransmission scheme, message cache etc.), and message handler, the interpreter of messages, and in the case of the UserAgent, a user interface used by the interpreter.

Note that the class diagrams in this chapter only describes the design in principle, and not every exact detail. The notation used to describe the relationships between different classes distinguishes between "uses" and "has instance of". The former indicates that the class only uses the class the arrow points to, while the latter means that it contains a specified number of instances of the class connected to.

3.1 Common SIP

In order to provide reusable code, it was kept in mind to be on the look out for overlap in functionality between the different SIP components implemented. As a result of this, there are many common classes.

Fig. 3.1 describes how the application layer transport in the TransactionHandler class, implemented by the author, fits in with the classes for sending and receiving data with UDP and TCP, provided by Marratech.

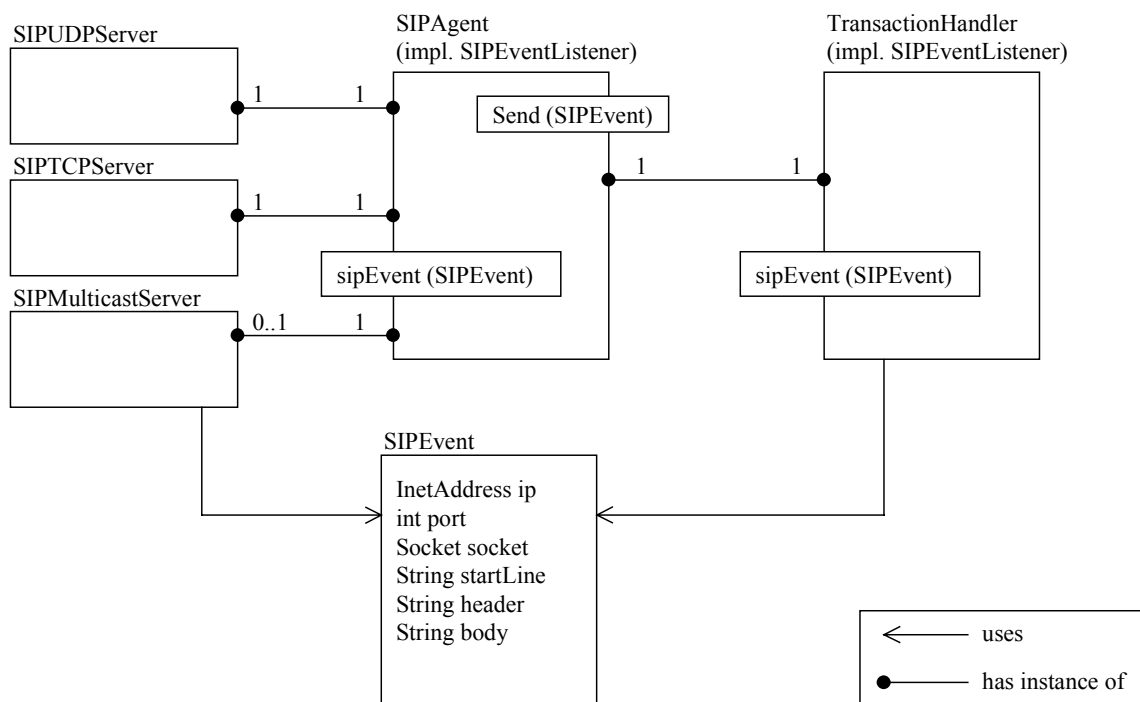


Figure 3.1: Provided functionality for lower level transport, marratech.app.sip

3.1.1 TransactionHandler

The TransactionHandler contains a scheme for providing reliability when using unreliable transport protocols, which retransmits requests for which no response was received. Also, responses for repeated requests are retransmitted. Fig. 3.2. shows how the SIP Message parser, and MessageHandler class fits in with these parts.

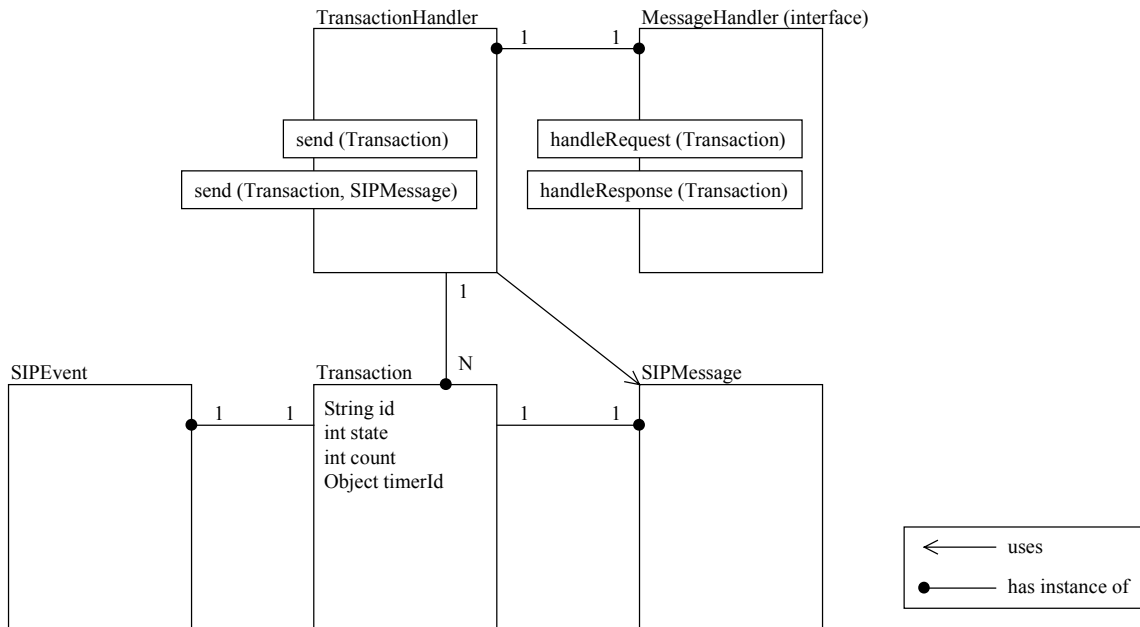


Figure 3.2: TransactionHandler and involved classes, marratech.app.sip

3.1.2 Parsing a SIP Message and its Components

The parser consists of a general message class, SIPMessage, which uses special classes for the more complex headers, such as Via and Contact as well as a class for SIP URLs, as illustrated in Fig. 3.3.

3.1.3 Test Results

The parser does not work with most of the SIP Bake-Off³ test messages and torture tests, as expected. Most "nice" messages parse, but one faulty, yet unimportant header may cause an error resulting in that the message is discarded, even though there might have been enough information in the message for it to be processed further.

³ <URL: <http://www.cs.columbia.edu/sip/bakeoff2/bakeoff2.html>>

The reliability scheme was tested with the UserAgent and Redirect Server in terms of the performing the retransmissions required. However, no testing was done where packet loss was provoked in order to measure the actual performance of the reliability scheme in such situations.

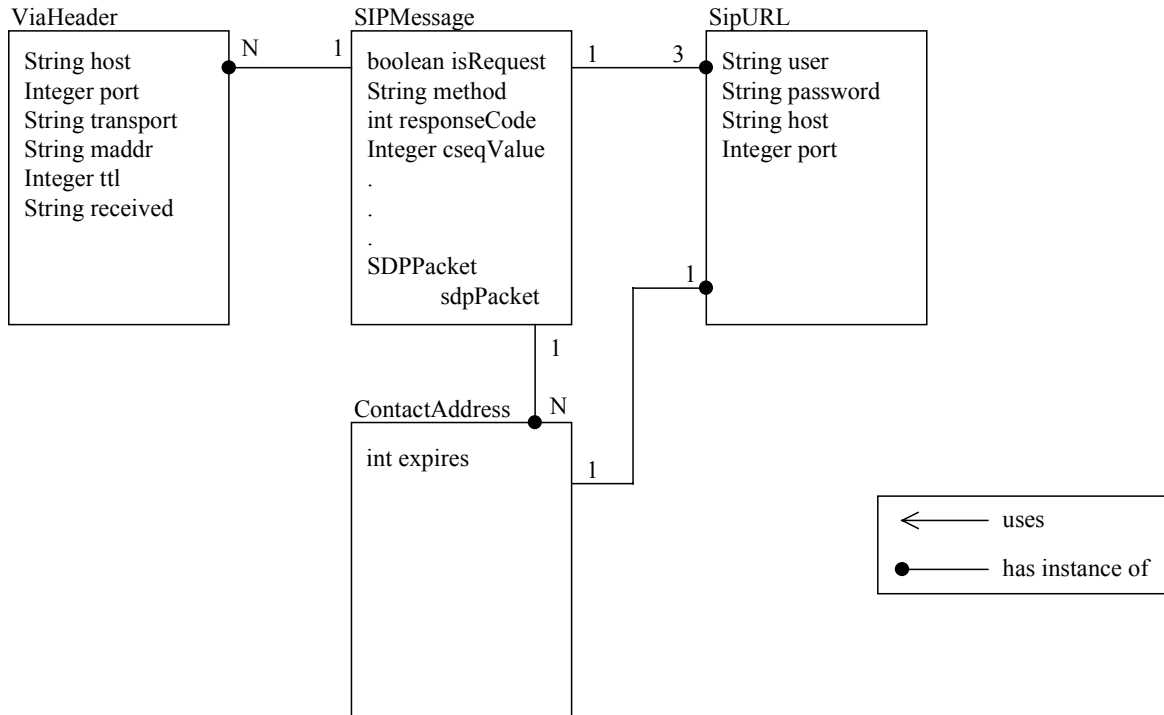


Figure 3.3: SIP Message Parser Classes, marratech.app.sip

3.1.4 Future Work

Messages that are no longer likely to be of any use should be removed from the cache-like structure in the TransactionHandler. Some implementation details such a more general method for matching CANCEL and ACK requests with the correct request or response, independent of if the To-header has been tagged, something that is done in a slightly awkward manner at this time being.

At the moment the only transport protocol supported is UDP, TCP should also be implemented.

Messages with syntax errors should also be handled better, as well as a few minimal implementation requirements of SIP that wasn't fulfilled.

3.2 SIP UserAgent

Fig. 3.4 presents an overview of the classes unique to the UserAgent implementation, UserAgent, the message interpreter, and the three GUI components, UserAgentFrame, IncomingCallDialog and CallingDialog.

3.2.1 UserAgent

The UserAgent implements the MessageHandler, through which it communicates with the TransactionHandler. It interprets the messages it receives and interacts with the user if necessary through the different GUI components.

The procedures for querying the DNS for SRV records comes from an earlier project of the author's, and was reused with slight modifications.

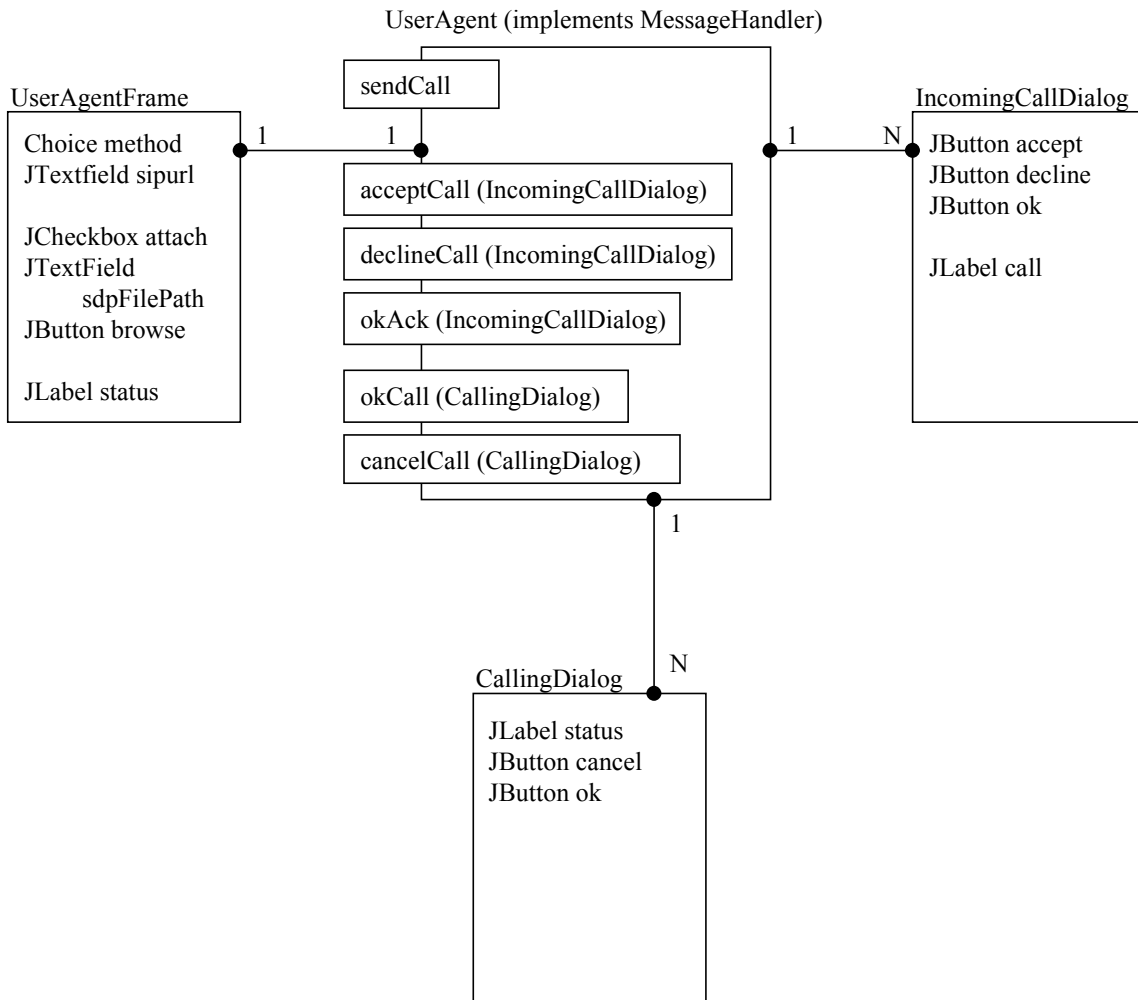


Figure 3.4: UserAgent classes, marratech.app.sip.ua

3.2.2 UserAgentFrame

Shown in Fig. 3.5 is the main GUI window, which opens up when the UserAgent is started. There is a button for sending requests, choices for different types of requests, a button to open a file selector to pick an SDP-file to attach to the request and text fields for specifying the path and the SIP URL for the Request-URI field.

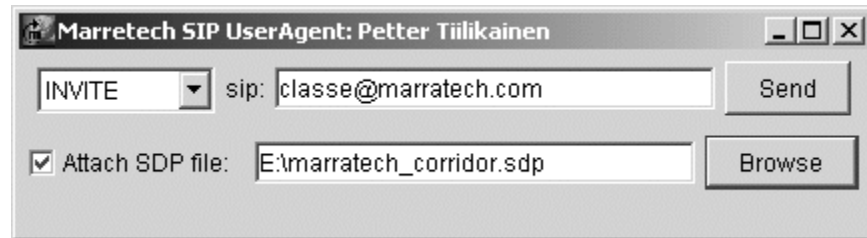


Figure 3.5: UserAgentFrame

3.2.3 CallingDialog

When pressing the send button of the UserAgentFrame, the dialog in Fig. 3.6 is displayed, with status for the ongoing, outgoing call. If the call has been accepted or rejected, the OK button is activated, and the tools for the multimedia session can then be launched if it was an invitation to participate in a session.

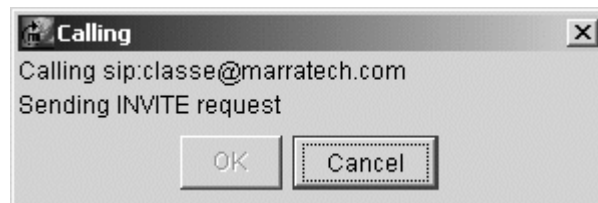


Figure 3.6: CallingDialog

3.2.4 IncomingCallDialog

When an incoming call is received, the dialog in Fig. 3.7 is shown, which contains information about who's issued the request and buttons for accepting or declining to participate in the session. If the accept button is pressed, it and the decline button are deactivated, and once the acceptance of the call is acknowledged, a new button is displayed, that when pressed, launches the tools for the multimedia session.



Figure 3.7: IncomingCallDialog

3.2.5 Test Results

The UserAgent has been tested in terms of interoperability with a few public SIP servers⁴ with satisfactory results, REGISTER requests are issued successfully and the responses to those requests are handled in a civilized manner.

A performance issue was raised when there wasn't any answer to the SRV query from the DNS, as it took much too long for the UserAgent to go to the next step in the procedure of finding the server location.

3.2.6 Future Work

There should be interfaces defined in order to allow different types of user interfaces to be used with the UserAgent implementation. When that has been done, a new, user-oriented GUI should be designed, to replace the current one that was designed to make it possible to test the SIP implementation.

The UserAgent needs to support for media negotiation, including the exchange of unicast SDP packets.

Finally, the issues with the DNS query should be resolved.

3.3 SIP Redirect Server

The time restraints were the reason for implementing a redirect server, instead of the more versatile and advanced proxy. The redirect server consists of the message interpreter, Redirectory, and a simple LocationDatabase implementation, MyLocationDatabase, as can be seen in Fig. 3.8

3.3.1 Redirectory

Redirectory communicates with TransactionHandler through the MessageHandler interface, and interprets the messages received.

⁴ <URL:<http://www.cs.columbia.edu/~hgs/sip/servers.html>>

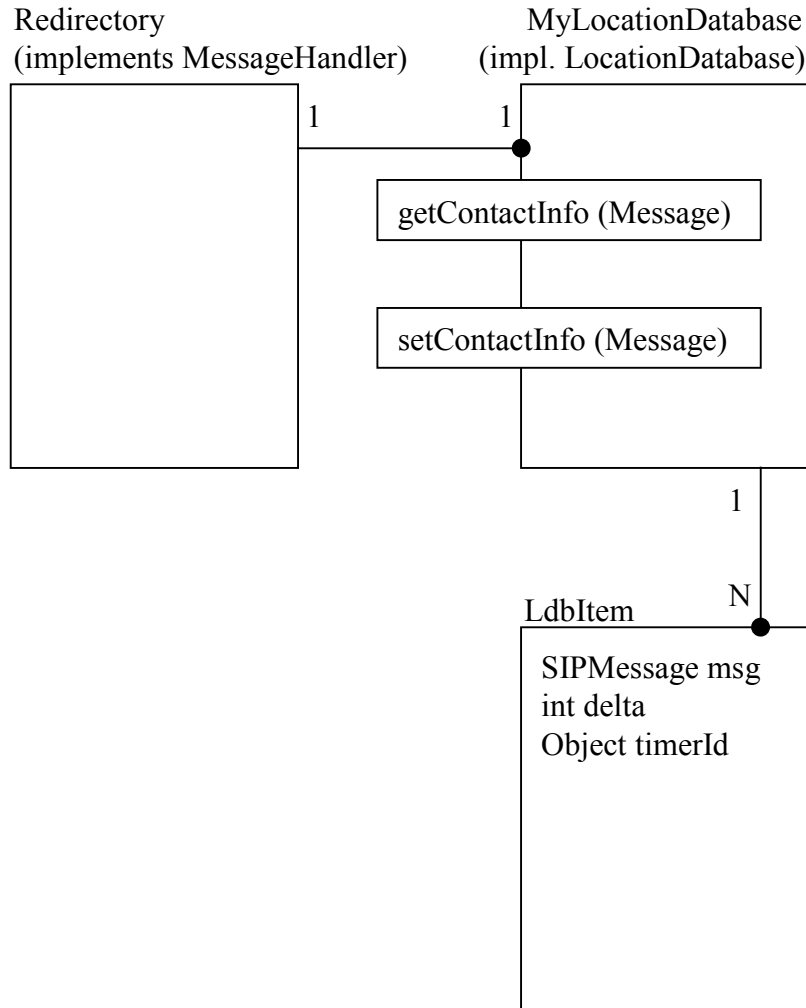


Figure 3.8: Redirect Server class diagram, marratech.app.sip.ldb

3.3.2 LocationDatabase

The LocationDatabase interface was defined so that it would be easy to implement it with any suitable database structure, depending on the applicable needs. In this case, it was implemented with MyLocationDatabase, which only contains a simple hash table, with no real database functionality, since it was mainly needed for testing and no real deployment.

3.3.3 Testing

The redirect server was tested with the UserAgent as well as with DynamicSoft's UserAgent and call simulator⁵ with satisfactory results in terms of interoperability and functionality.

⁵ <URL:http://www.dynamicsoft.com/jsip_uac.html>

3.3.4 Future Work

When a CANCEL request is received in for an earlier registration request, it should be unregistered. If only a few of the SIP URLs where a specific user can be contacted expires, or is cancelled there is no support for partial removal of the database entry.

The implementation of the LocationDatabase should contain a real database structure, and finally, the redirect server should evolve into a proxy.

3.4 Conclusions

Java provides no methods to locate the locally configured DNS, complicating it to implement a good, platform independent DNS SRV lookup. A simple solution was settled for, which allows the user to configure the local DNS in a settings file.

As mentioned in Section 3.1.4, there were some problems with matching tagged responses with non-tagged requests, as well as matching ACK requests to 200 OK responses. Solving this problem in a more elegant manner, which requires some redesign of the TransactionHandler, would also ease the full implementation of CANCEL and BYE methods.

Interpreting the SIP specifications and determine exactly how the different SIP components should behave was a time-consuming, and sometimes frustrating task. This applies mainly to the behaviour of the TransactionHandler and to some extent, the SIP Message parser. For the parser, the String and StringTokenizer classes in Java were very useful.

To simply launch Marratech Pro from the UserAgent, once a call has been established, was decided upon because extensive integration seemed like it would take time that wasn't available. Further integration of SIP with Marratech Pro would benefit from redesigning the UserAgent, so that it can be used with any kind of user interface implementing a predefined set of methods.

On a whole, keeping the overlap of functionality between the different SIP components in mind turned out to be worth the effort, as when the common SIP classes had been implemented, the SIP redirect server was implemented in very short time.

4. SIP and H.323

H.323 has evolved from H.320 and other earlier protocols, which were designed to provide multimedia conferencing over switched circuit networks. The H.323 specification consists of numerous different ITU-T recommendations, such as:

- H.225.0: Call setup and conference control.
- H.450: Supplementary telephony-like call control services like call forwarding.
- H.245: Capability exchange.
- H.235: Authentication and encryption.

In comparison to SIP, H.323 offers a diverse range of services for multimedia conferencing, which at the same time means it is rather complex [8]. It is based on reliable transport for the call signaling, and uses a binary message representation, while SIP uses simple, text-based messages that can be sent using unreliable transport protocols.

There are many similarities in terms of the services provided by SIP and H.323, and both use RTP for media transport and run over IP. With H.323v3, UDP and TCP can be used for call setup messages, as SIP does. SIP does not offer any of the conference control provided by H.225.0, instead it relies on other protocols to provide such facilities.

SIP has a loop detection algorithm, while H.323 solves that problem by defining a maximum number of gatekeepers a message is allowed to traverse.

SIP uses a more user oriented scheme for endpoint location, the e-mail-like SIP URL, while H.323 uses H.323ID and E.164, which are more telephony oriented.

The capability negotiation for deciding what media types to use for the multimedia session is more flexible with H.323, which is provided by H.245, than with SIP. [9]

4.1 Interoperability interworking

It is clear that SIP can coexist with H.323 [10], especially since H.323 and SIP seem to be transcending, meaning that they are learning from each other in terms of call management, control and so on. This is also illustrated by the existence of a SIP-H.323 work group⁶.

There are two distinct alternatives for interworking:

- Find out with SIP that user can be contacted with H.323 and then use H.323 to establish the call. This is a simple solution, but needs a full, separate H.323 implementation, so it might not be so easy after all.

⁶ <<http://www.softarmor.com/sipwg/teams/sip-h323>>

- Translation of requests and responses, via SIP-H.323 gateways, interworking in progress. Harder to do for full H.323 compliance, but translation between H.323v2 FastConnect and SIP is easy. That raises the question of what availability of SIP-H.323 gateways are, and how well they work.

4.2 The Shape of Things to Come

To quote the SIP 2000⁷ conference and exhibition program:

"SIP is indisputably the most singular tendency in the telephony over IP world. Unknown and even rejected by many vendors and operators only a few months ago, the IETF's signaling protocol is on the brink of making a definitive name for itself."

This view is confirmed by all the ongoing work in the world of IP Telephony and multimedia conferencing, evident as there are many major companies involved in SIP development and SIP-H.323 interworking.

A few examples of resulting products are PalmOS telephony applications from 3Com⁸, SIP compliant routers from Cisco and a SIP server from Lucent⁹ as part of a software platform for integrated IP telecommunications services.

Apart from IETF's standardization work on SIP¹⁰, ETSI has announced in a press release¹¹ that their TIPHON project will include SIP-H.323 interworking. The aim of TIPHON is to merge the telecommunications and internet world, more specifically, to ensure that users on IP and various switched circuit networks can communicate with each other.

Also, there are several IETF internet-drafts involving SIP, on topics such as PSTN interworking, call control services and more. An overview of the work in progress can be acquired through the SIP Homepage.¹²

⁷ <URL:<http://www.upperside.fr/basip.htm>>

⁸ <URL:<http://www.3com.com/technology/sip>>

⁹ <URL:<http://www.lucent.com/press/0999/990928.coc.html>>

¹⁰ <URL:<http://www.ietf.org/html.charters/sip-charter.html>>

¹¹ <URL:<http://www.etsi.org/press/tiphonh323sip.htm>>

¹² <URL:<http://www.cs.columbia.edu/sip>>

5. Conclusions

It is possible to use SIP together with Marratech Pro in two distinct ways. The complicated solution is to use SIP as an integrated part of Marratech Pro, where SIP provides the session management throughout the entire session. The easy solution is to use SIP as a stand-alone part, where it is only used to initiate the session, and when the session has been established, launches Marratech Pro, but has no management of the session once it has started.

The future for SIP looks bright, as there is a lot of work being done involving it, as well as for its use together with H.323 since SIP-H.323 interworking is a work very much in progress.

The complexity of H.323 makes SIP a more viable alternative to a small company. H.323 seems to be favoured mainly by large telecom industry type companies who can set aside a large number of developers on a project like that, and when there will be SIP-H.323 interworking solutions, the reasons for choosing H.323 over SIP will be few.

The SIP approach to call signaling provides you with more functionality in relation to the time invested.

6. Bibliography

- [1] M. Handley, H. Schulzrinne, E. Schooler and J. Rosenberg, "SIP: Session Initiation Protocol", IETF RFC2543, March 1999.
- [2] JavaSoft Inc. The Java Language <URL:<http://www.javasoft.com>>
- [3] M. Handley and V. Jacobson, "SDP: Session Description Protocol", IETF RFC 2327, April 1999.
- [4] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications", IETF RFC1889, January 1996.
- [5] International Telecommunication Union, "Packet based multimedia communications systems", ITU-T Recommendation H.323, February 1998.
- [6] T. Berners-Leem, R. Fielding, and L. Masinter, "Uniform resource identifiers (URI): generic syntax", IETF RFC2396, August 1998.
- [7] A. Gullbrandsen and P. Vixie, "A DNS RR for specifying the location of services (DNS SRV)", IETF RFC2052, October 1996.
- [8] H. Schulzrinne, J. Rosenberg, "A comparison of SIP and H.323 for Internet Telephony", Network and Operating System Support for Digital Audio and Video (NOSSDAV), Cambridge, England, July 1998.
- [9] I. Dalgic, H. Fang, "Comparison of H.323 and SIP for IP Telephony Signaling", In *Proceedings of Photonics East*, Boston, Massachusetts, USA, September 1999.
- [10] K. Singh, H. Schulzrinne, "Interworking between SIP/SDP and H.323", Internet Draft, January 2000.

Appendix

A. Abbreviations

DNS	Domain Name System
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
IETF	Internet Engineering Task Force
IESG	Internet Engineering Standardization Group
ITU	International Telecommunication Union
IP	Internet Protocol
PSTN	Public Switched Telephone Network
RFC	Request For Comments
RR	Resource Record
RTP	Real-time Transport Protocol
RTCP	Real-time Transport Control Protocol
SAP	Session Announcement Protocol
SIP	Session Initiation Protocol
SDP	Session Description Protocol
TCP	Transmission Control Protocol
TIPHON	Telecommunications and Internet Protocol Harmonization Over Networks
UAC	UserAgent Client
UAS	UserAgent Server
UDP	User Datagram Protocol
UI	User Interface
URL	Uniform Resource Locator
URI	Uniform Resource Indicator