Mobility support for the VOCAL SIP architecture

Juan F. Rodríguez Hervella
Univ. Carlos III de Madrid
Av. Universidad, 30, Edif. Torres Quevedo. E-28911 Leganés (Madrid)
Tel: (+34) 91-624-8859
E-mail: jrh@it.uc3m.es

Abstract

The SIP specification has support for mobility features at different levels. Terminal, session and service mobility can be offered seamlessly and they have already been described in the literature. In this paper, we examine the current support for mobility on the Vovida Open Communication Application Library (VOCAL). VOCAL is a distributed SIP architecture based on a set of systems that allow an incremental deployment of an open source SIP platform. We describe how the new mobility feature affects the whole system, and we implement an extension to the VOCAL program that is used to manage the User Agent (UA) in a graphical environment, called SIPSet. The new capabilities that have been added to the SIPSet tool allow the configuration of the mobility tags in a graphical and easy way.

1. Introduction to the SIP architecture

The Session Initiation Protocol (SIP) [4] allows two or more participants to establish one or more different media stream sessions [15]. SIP endpoints are addressed by SIP URLs that looks like email addresses, such as "sip:alice@example.com". SIP requests contain a source address and two destination addresses, one for identifying the original, logical destination of the request (the "To" header), and another one for identifying the current destination, which travels on the "Request URI" header. The protocol defines a set of logical entities, namely user agents, redirect servers, and proxy servers. User agents are the source and the destination of any request. Generally, user agents are the only elements where media and signalling converge. Redirect servers receive requests and return responses that indicate where the request should go next. A typical SIP architecture usually implements a proxy server, as well as a locator and a registrar server. The proxy server is the well known point of contact of every user agent. It is an intermediary program that acts as both a server and a client for the purpose of making requests on behalf of other clients. The locator server is used by a redirect server or a proxy server to obtain information about a called party's possible location and finally the protocol holds the concept of registrar server, which is a kind of server that accepts REGISTER request [4] and manages the registration process. The registrar saves the information about where a user agent can be found, using a database system or any other software.

Besides the previous components, some kind of provisioning must be set up in the network for purposes of Authentication, Authorization and Accounting (AAA). The SIP protocol does not define how this service should be provided.

1.1. SIP request and response messages

The following are the most common request messages. The format of the messages is described in [4]

- INVITE: Indicates that the user or service is being invited to participate in a sessions
- ACK: Confirms that the client has received a final response to an INVITE request.
- BYE: Indicates that the user wishes to terminate the session.
- CANCEL: Cancels a previous request.
- REGISTER: Registers the address listed in the "To" header field with a SIP server.

There are many different responses, arranged into six different types. The response messages resemble HTTP messages and they are identified by a three digit number. The first number groups the kind of message that is being transmitted.

The ACK messages are requests that complete a transaction after a final response, such as "302 Moved Temporarily" or "200 OK". ACK is a request because of its structure, not necessarily because of its behaviour or content. According to the RFC: "*SIP requests are distinguished by having a Request-Line for a start-line. Request-Line contains a method name, a Request-URI, and the protocol version separated by a single space character*". The reader can watch an example of this in Figure 2. As for responses the RFC states: "*Sip responses are distinguished from requests by having a Status-Line as their start-line. A Status-Line consists of the protocol version followed by a numeric Status-Code and its associated textual phrase, with each element separated by a single SP character.*"

1.2. Call setup

Depending on the servers involved in the establishment of the call, we can observe different kind of scenarios. We talk about "call" to refer to some communication between peers, generally set up for the purposes of a multimedia conversation. A "message" is the data sent between SIP elements as part of the protocol. SIP messages are either requests or responses. In the following examples, we identify a SIP "transaction" as the group of messages that are exchanged between a client and a server and it comprises all messages from the first request sent from the client to the server up to a final (non-1xx) response sent from the server to the client. For example, when a UA receives a response, other than a "1xx" response, to an INVITE, it sends an ACK message. The ACK message is considered a new transaction, as it is already been explained.

In Figure 1 we describe a flow of messages between two UAs. In the example, the INVITE message is addressed directly to the destination UA.

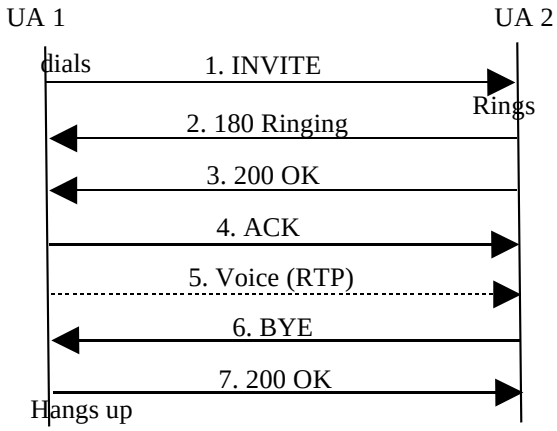

Figure 1. Basic call setup

The session starts with an INVITE message, which typically has the look-and-feel of Figure 2:

```
INVITE sip:6713@163.117.140.166:6060;user=phone SIP/2.0
Via: SIP/2.0/UDP 163.117.140.182:6060
From: UserAgent<sip:6710@163.117.140.44:6060;user=phone>
To: 6713<sip:6713@163.117.140.166:6060;user=phone>
Call-ID: 96561418925909@163.117.140.44
CSeq: 1 INVITE
Subject: VovidaINVITE
Conteact: sip@6710@163.117.140.44:6060;user=phone
Content-Type: application/sdp
Content-Length: 168

v=0
o=- 238540244 238540244 IN IP4 163.117.140.44
s=VOVIDA Session
c=IN IP4 163.117.140.44
t=3174844751 0
m=audio 23456 RTP/AVP 0
a=rtpmap:0 PCMU/8000
a=ptime:20
```

Figure 2. INVITE header

The INVITE message is divided into two sections. The top section contains the SIP headers, and the bottom section contains the body of the SIP message, in this case, it is Session Description Protocol (SDP) [11] information. The headers are separated from the body by a blank line. We can observe different fields that carry on different information. "From" field provides the identity of the request's initiator whereas the "To" field provides the identity of the intended recipient of the request. The "Call-ID" field gives a globally unique identifier to distinguish specific invitations or multiple registrations of the same user. "CSeq", which stands for "Command Sequence", is required in the request messages and in response messages. It provides the request method with a unique decimal sequence number. If new requests are sent by the user with the same Call-ID, but different methods or content, the "CSeq" value has to increase by 1 to let the other network entities know that they are receiving a new message. Otherwise, the entities will think they are receiving a retransmission. Note that SIP works with both UDP and TCP, so when working with UDP there are defined some timers to retransmit messages. A full description of every field in this and other messages can be found in [4].

The UA 2 answers with an informative message ("180" Ringing) to signal that the INVITE message has reached the destination but it has not still been accepted. When the destination user goes off hook, the OK message is sent. Upon reception of the OK message, the initiator issues an ACK message and establishes the voice communication channel. Once the UA 2 finishes the conversation, a BYE message is sent back to the caller, which hangs up and sends an OK message to indicate the called party that the session has been successfully closed.

Figure 3, on the other hand, uses proxy servers as well as a redirect server to route the call. The redirect server is shared between both proxy servers. The forward path is found by means of the "302 Move Temporarily" messages, which contain the URI of the server that should be contacted to reach the destination; in the first case it is the remote user's proxy server. The remote proxy server asks the same redirect server for the location of the callee, and finally the invite is forwarded to the real destination. The reason for passing through the remote proxy server is a matter of choice. If both users are local subscribers, the first time the redirect server is contacted, it could respond with the real IP address of the destination. Nevertheless, if the registration process involves the storage of the address of the proxy server that processed the called party's registration, the flow of messages occurs as it is shown in Figure 3.

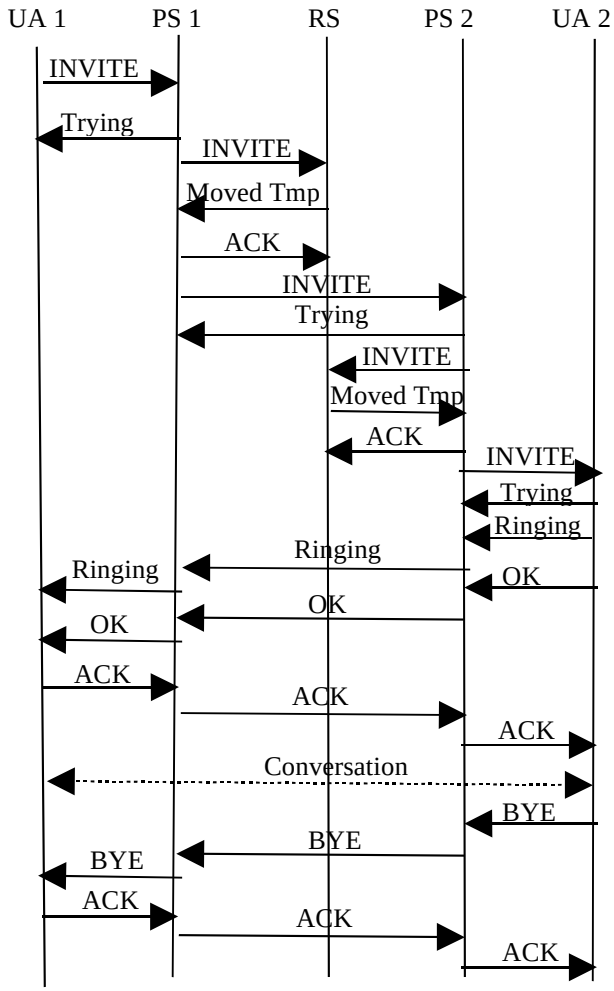

Figure 3. Proxying call flow

If a proxy server wants to be in the call path, it can insert a "Record Route" header. Moreover, loops are prevented using "Via" headers, which provide a history of the message's path through the network or networks. "Via" headers are used to determine the routing for SIP responses. If two, three, or more "Via"s are listed, their sequence is significant. As messages are sent through proxy servers, each server adds a "Via" line to indicate that the message has been through that proxy. The rest of the standard headers are fully described in [4], and we encourage the interested reader to dig in the references as well as any other good source of information to obtain a whole picture of the flow of messages that constitutes the SIP protocol.

1.3. Registration

The registration process allows UAs to be authenticated in the SIP network as well as being located by other SIP agents. It is also the main procedure to implement mobility features. The UA sends a REGISTER message to the proxy server, which will forward the message to the registrar. Upon correct registration, an OK message will come back to the UA. The REGISTER message can specify more than one address, so the OK response includes a "Contact" header with all current bindings approved. These bindings have an expiration timer, meaning that the registration process must be refreshed. More details about the registration process can be found in the SIP RFC [4].

2. SIP and mobility

The IETF has developed IP mobility support for IPv4 [13] and IPv6 [12], which provides for transparent mobility, in that they hide the change of IP address when the mobile host is moving between IP subnets. On the other hand, the application layer protocol SIP supports by default the ability of end users to originate and receive calls on any terminal in any location, which is often called "terminal mobility". Some studies have come up to analyze the pros and cons of both mobility systems [1][2] [3]. Mobility in IPv4 requires changes in the IP stack of non-mobile hosts to allow route optimization [17]. Another issue of the mobility protocols is that the mobile host needs a permanent home IP address, which obviously might be a problem due to the expected address exhaustion in IP version 4. Nevertheless, mobility at the IP layer provides transparent mobility which is needed to keep TCP connections alive as the user is moving. *H. Schulzrinne* and *E. Wedlung* [1] suggest to use mobile IP solutions for long-lived TCP connections but to use a more appropriate mobility support for real-time communication based on UDP.

2.1. Terminal mobility

When a user turns off his device, moves out to another place and turns it on again, the SIP device re-registers with its "home" registrar. This process is done every time the device obtains a new IP address and it is known as "pre-call mobility" [1]. If the user has a call session ongoing, and the device changes its IP address, the device issues a RE-INVITE message[1] which is routed accordingly to the destination UA. Upon reception of the RE-INVITE message, the correspondent UA starts sending its session packets to the new IP address.

2.2. Session mobility

Session mobility allows a user to maintain a media session even while changing terminals [1]. This is usually implemented by means of "third party call

[1] RE-INVITEs are simple INVITEs, there are no formatting differences. We talk about a RE-INVITE when the party issues another INVITE message.

control" (TPCC) agents [7] as well as "call transfers" [8].

Figure 4 shows how to move the session from an old device (the TPCC column) to a new device (NEW_DEV column), using TPCC methods. In the figure, the TPCC agent sends an INVITE message to the new device, and it transmits the session description parameters (SDP parameters that the NEW_DEV has transmitted with its OK message) to the correspondent host, using another (RE-)INVITE to carry on the information. As a result of this exchange, the new device can hold the previous session. One disadvantage of this method is that the TPCC agent (or the original session participant) has to remain involved in the session, as it will be contacted to change or terminate the session. The TPCC agent simply acts as proxy for the SIP signalling and the media sesion is redirected using INVITE messages.

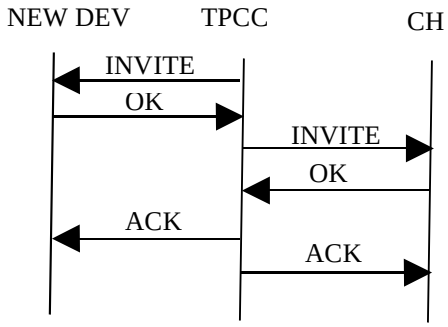

Figure 4. Session mobility with third-party call control.

In Figure 5, the session is moved out using a REFER request message [8]. The old device simply sends a REFER request to the correspondant host, indicating that it should contact the new device. The correspondant host then negotatiates a session using the regular INVITE exchange. The old device closes the old connection when the session is effectively transferred.

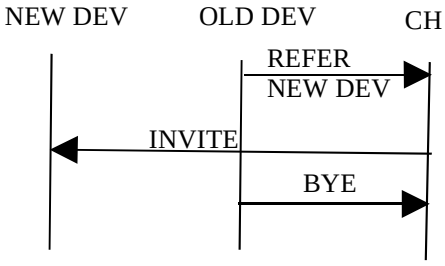

Figure 5. Session mobility using call transfer

2.3. Personal mobility

Personal mobility allows addressing a single user located at different terminals by the same logical address [19]. This can be achieved by using ENUM [20] or using multiple registrations. The registrar needs to be able to recognize different devices as belonging to the same person.

2.4. Service mobility

Service mobility refers to the capability of roaming around and at the same time making sure that the mobile host maintains the Quality of Service (QoS) of the ongoing sessions as well as minimizing the loss of transient data during handoffs, and satisfying the delay requirements of the multimedia application [1]. In a simple way, service mobility allows users to maintain access to their services even while moving or changing devices and network service providers. These services, besides QoS related ones, can be address books, call logs, media preferences, buddy lists and incoming and outgoing handling instructions (e.g. speed dials). One solution for service mobility is to have the user carry this information with him, as it is explained in [1].

3. Mobility support in VOCAL

The "Vovida Open Communication Application Library" (VOCAL) [21] is an open source project targeted at facilitating the adoption of VoIP in the marketplace. VOCAL provides the development community with software and tools needed to build new VoIP features, applications and services. The software in VOCAL includes a SIP based redirect server, feature server, provisioning server[2] and different marshal proxies, which are the typical proxy servers cited in the SIP protocol.

The VOCAL mobility support has been developed by *Rajarshi Chakraborty* and *Kishore Mundra*, at the Indian Institute of Technology, Kharagpur (India) (http://www.iitkgp.ernet.in/). The patch can be downloaded from [9]. The stable 1.5.0 version of VOCAL has been modified for this purpose. As the time of this writing, only patches are provided but it is expected that after a revision of the code, the VOCAL developers will merge it on the main source tree. Besides the modification of the UA, the proxy server ("marshall server" in VOCAL vocabulary) and the redirect server have also suffered modifications.

The mobility setup has only been tested in an IPv4 environment. For that purpose, the authors had to tweak the wireless driver of the mobile host to facilitate dynamic change of IPv4 addresses across access points in different subnets. Once the wireless driver reports that it has been associated with another access point, the mobile host should address

[2]Provisioning refers to the way VoIP users are managed in the network. VOCAL provides two ways of administering users. One based on web pages and the other based on Java.

a new DHCP [16] query to obtain a new IP address, provided a change of subnet had occurred.

It is worth noting that in any network, a SIP end system needs to establish two SIP-related configuration parameters, namely the local registrar and whether there is an outboud proxy. Usually both the local registrar and the outbound proxy are co-located so the only parameter that the UA typically needs to have configured is its domain name, which can be used to locate the SIP proxy.

The mobility design is based on the "outbound proxy interception" method defined in [24]. We define a "travelling user" or "visitor" as a SIP endpoint that is visiting a domain other than the domain indicated in its SIP URI. The concept surrounding the "outbound proxy interception" method can be explained as follows. The outbound proxy of the visitor UA intercepts the registration message as well as any other outbound request and it changes the "Contact" address field to its own address. Then it forwards the registration request to the home registrar server. In order to identify future income request, the foreign outbound proxy server also needs to create a new temporary user identifier for the visiting UA. This identifier should be unique between all the possible visiting nodes. Thus, [24] defines a "canonical visitor's name", though any other ramdom identifier could also be used. The canonical visitor's name is made of the concatenation of the visitor's address and the proxy's domain, such as "alice%40wonderland.com@visited.net", where the simbol "%40" stands for the URL-scaped "@" representation.

In a nutshell, the UA has made a registration on its home network with the IP address of the visiting outbout proxy, and it has also made a foreign registration with a canonical visitor's name in the foreign network, using the recently aquired IP address. As it is explained in [24], this approach has the advantage that it forces incoming requests to use the proxy server and thus solves the typical firewall problem. Besides, if the UA changes of IP address multiple times inside the same foreign network, the location update only has to travel locally, and the home proxy is not disturbed, improving the latency of the communication.

A rogue user can easily override the registration of the visiting UA. In order to make this registration process more secure, the visited proxy server can accept the registration of the mobile user only if the registration on its home network has finished successfully. Therefore, the VOCAL implementation waits for a "200 OK" message coming from the home registrar before executing the UA's temporary registration on its own SIP network. The architecture of VOCAL has been slightly modified to reflect this behaviour. Note that this delay does not solve the problem, but at least the rogue UA will have to implement a kind of proxy server behaviour to lie about the home registration process. At the time of this writing, there is not implemented any temporary authentication for foreign network registrations.

When the visitor needs to send a RE-INVITE message to its peer, it has to pass through the foreign proxy server. This is how the VOCAL architecture has been designed, as long as the UA has got a SIP proxy server configured. Hence, when the user is in the foreign network the foreign proxy server becomes the (temporary) local SIP proxy server for the mobile host.

Althought [1] explains the usage of the RE-INVITE method as if they could be sent directly to the correspondant host, *R. Chakraborty* and *K. Mundra*, the authors of the mobility implementation for VOCAL, have not taken that position. Sending the message directly would require the identification of the IP address of the other participant from the SIP headers which is not a standard behaviour. Trying to bypass the SIP server does not agree with the VOCAL philosophy (if you have a proxy server configured, you must use it). This could also bring problems with networks protected by firewalls.

The flow of SIP packets varies depending on the direction of the call setup. When the mobile UA sends a message to the other participant, this message is sent directly to the destination without travelling through the home network. The (foreign) outboud proxy server identifies that the user is a visitor and since the user is registered using the temporary canonical name, the proxy remembers that the "From" field has to be reset to the real identity of the UA. For example, if the UA has the canonical name "alice%40wonderland.com@visited.net", the foreign proxy server has to replace this identifier with "alice@wonderland.com".

From messages coming from a correspondant UA to the mobile UA, the messages take a triangular routing path. The correspondant UA sends the message to the home network of the mobile host, as the correspondant only knows the existence of the mobile UA as "alice@wonderland.com". Thus, the home network using the normal registration information forwards the message to the foreign proxy server where the mobile UA is currently located. Finally, the foreign outbound proxy server delivers the message to the mobile UA.

In short, foreign outbound proxies must be modified to forward the SIP REGISTER messages from a visiting mobile UA to its home network. The VOCAL marshall server does not verify the authentication for any SIP message coming from a "visitor". However, the home marshall server would verify the authentication of the REGISTER message, as the user is not a "visitor" to itself. Hence at least in the registration process it is ensured that the "authentic" visitor is sending the

SIP REGISTER message and not some masquerader. Another option could have been that the foreign marshall server returned back a temporary password to the "visitor" after temporarily registering it. The foreign marshall server would still have to be modified to be able to check a visitor's authentication, in which case the "visitor" would use this newly obtained temporary password. This would eliminate the need of the foreign marshall server to always forward the visitor's SIP REGISTER messages to its home VOCAL system. *R. Chakraborty* and *K. Mundra* have made a detailed description of all these alternatives in the implementation notes that come with the VOCAL mobility patch [9].

3.1. Activation of mobility features

To use the mobility implementation capabilities, the mobile flag of the graphical user agent (GUA) has to be set to 1. Moreover, there are two different and exclusive ways of looking for foreign proxy servers. Depending on the activation of the "DNS" flag, the system can use either DNS queries to locate SRV records [10] that inform about the actual IP address of a proxy server, or reading from a configuration file the mappings of IP address range to IP address of the (foreign) proxy server.

The usage of the DNS flag requires the presence of a SRV record for the SIP proxy servers in the DNS servers of the domains involved, where the user plans to use this feature. The details for this kind of location of SIP servers are given in [18]. Note also that the DNS SRV query requires the domain name to be set in the mobile host, which can be achieved using the –D option of the DHCP client.

In case the user switches on mobility and if the DNS flag is set to 0, the GUA will take the proxy server from the list of domains in another new configuration tag called "Domain". The "Domain" tag maps domains to proxy servers. The format of the "Domain" flag is shown in Figure 6:

```
Domain String [IP-Range]/Proxy1$DomainName/Proxy2$....
```

Figure 6. "Domain" format option

For example, if a mobile user had planned that he would roam between "it.uc3m.es" domain and "fis.uc3m.es" domain in advance, he or she could have configured the following tags in the GUA configuration file showed in Figure 7:

```
Mobile String 1
DNS String 0
Domain String
    it.uc3m.es/163.117.140.2$fis.uc3m.es/163.117.234.72
```

Figure 7: GUA mobile configuration options

Where "163.117.140.2" corresponds to the SIP proxy server of the "it.uc3m.es" domain and "163.117.234.72" corresponds to the same SIP server but located on the "fis.uc3m.es" domain. The "Mobile" tag switches on the mobility features.

4. SIPSet extensions to support mobility

SIPSet is a user agent with a graphical user interface front-end that works with the SIP stack of VOCAL. SIPSet can be used as a soft phone, to make and receives phone calls from a standard personal computer. It currently supports both IPv4 and IPv6 protocols.

The SIPset application consists of two pieces, the user interface, called "sipset", and the call control and media application, called "gua". in order to allow multiple user interfaces to be attached to the "gua", the "gua" communicates with the "sipset" via a protocol which runs over named pipes (also known as FIFOs) [22].

On start-up, the program reads a default configuration file from the standard installation path, later on it tries to read any local configuration file, and finally it writes the configuration values that have been read plus any changes that the user might do through the windowed interface to a local configuration file, usually located at "$HOME/.sipset/gua.conf". Every time the user modifies a value on the graphical interface, the new configuration is saved and if the change involves any action regarding the GUA, a corresponding message is sent through the communication pipe.

Our implementation has consisted in the addition of a new item under the "settings" menu entry, named "Mobile configuration", which opens a new window with all the flags that are needed to configure the mobility support of the GUA. This new interface has been developed using "glade" [23]. The new configuration windows are showed in figures 8 and 9.

Due to the fact that SIPset is only a wrapper for the command line GUA[3], the only behaviour that had to be implemented, apart for the event-driven programming, was the reading and writing functions that operates when a configuration value changes. The total number of lines of the patch that has been generated is over 1.100 lines.

Figure 8 shows the main SIPset window. You can observe the new item of the "settings" menu, which stands for enabling mobility configuration options. Figure 9 shows the mobility configuration window,

[3] The UA is called GUA because it is used by the SIPSet, which is the actual graphical interface. Hence the GUA is not a graphical interface on its own, though we recognize that the name is tricky.

where you can configure the previously explained tags.

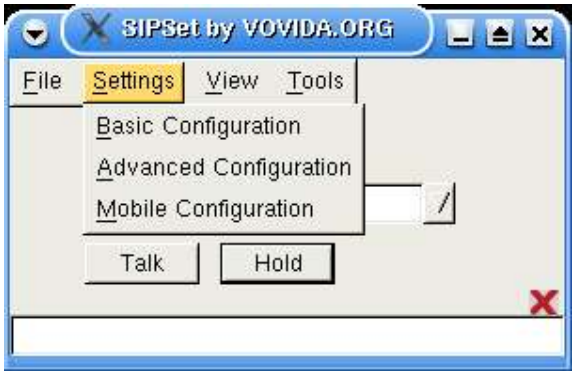

Figure 8. Main SIPset window

Figure 9. Mobility configuration window

5. Conclusions and future works

This paper shows the current status of the mobility support for the Vovida Open Communication Library (VOCAL) open source project. Mobility in SIP can be achieved efficiently at the application level without imposing requirements on external protocol elements [1][2][3]. This document shows the modifications that have been implemented in VOCAL in order to support mobility. These modifications range from the addition of new code to configure mobility in the UA, to structural code changes that must be done on the proxy server to forward RE-INVITE messages that come from un-authenticated sources. The author of the paper has also implemented an extension to the graphical UA of VOCAL, called SIPset, to allow an easy manner of configuring the different mobility options. These (exclusive) options are based on the utilization of DNS infrastructure to discover foreign proxy servers [10], as well as on statically pre-configured information about such servers for each foreign domain.

VOVIDA software is constantly improving. Newly defined SIP messages still have to be implemented [5][6] in VOCAL. New efforts on the mobility features should be directed to improve the capabilities of the third party call control agent, called "back to back UA" (B2BUA), as well as improving the way the UA detects IP changes, which is currently done by polling.

Long term work should be directed to the study of the interaction of SIP mobility with IP mobility. More detailed research on the delays and behaviour of such joined mechanisms, especially in 3GPP networks [3], are still needed.

Acknowledgments

I would like to name, firstly the authors of the mobility support for VOCAL, *Rajarshi Chakraborty* and *Kushore Hundra*, because their work has made possible that mine comes to light. Secondly, I would be evil if I did not cite the people who have been supporting my research all these years. Mainly, I would like to thank Alberto Garcia, Marcelo Bagnulo and Ignacio Soto. Special thanks to Manuel Urueña for reviewing the first version of this paper and making good suggestions.

References

[1] Henning Shulzrinne, Elin Wedlund, "Application-Layer Mobility Using SIP", Mobile Computing and Communications Review, Volumen 1, Number 2.

[2] Elin Wedlund, Henning Schulzrinne, "Mobility Support using SIP", Second ACM/IEEE International Conference on Wireless and Mobile Multimedia (WoWMoM '99)

[3] Ashutosh Dutta, Shinichi Baba, Hennin Schulzrinne et al., "Application Layer Mobility Management Scheme for Wireless Internet", 3Gwireless 2001,(San Francisco), pp. 7, May 2001

[4] M. Handley, H. Schulzrinne, E. Schooler and Rosenberg, "SIP: session initiation protocol", RFC 2543, IETF 1999.

[5] J. Rosenberg, "A Session Initiation Protocol (SIP) Event Package for Registrations", RFC 3680, IETF March 2004

[6] J. Rosenberg et al., "SIP extensions for instant messaging", RFC 3428, IETF December 2002.

[7] J. Rosenberg, J. Peterson, H. Schulzrinne and G. Camarillo, " Best Current Practices for Third Party Call Control (3pcc) in the Session Initiation Protocol (SIP) ", RFC 3725, April 2004

[8] R. Sparks, "The Session Initiated Protocol (SIP) Refer Method", RFC 3515, IETF April 2003.

[9] VOCAL Bugzilla entry number 765: http://bugzilla.vovida.org/bugzilla/show_bug.cgi?id=765

[10] A. Gulbrandsen, P. Vixie and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, IETF February 2000.

[11] M. Handley and V. Jacobson, “SDP: session description protocol”, RFC 2327, IETF April 1998.

[12] D. Johnson and C. Perkins, J. Arkko, “Mobility support in IPv6”, Internet Draft (-24.txt), IETF June 2003.

[13] C. Perkins, “IP mobility support”, RFC 2002, IETF October 1996.

[14] Luan Dang, Cullen Jennings and David Kelly, “Practical VoIP using VOCAL”, O’Reilly, July 2002, ISBN: 0-509-00078-2

[15] H. Schulzrinne, S. Casner et al., “RTP: a transport protocol for real-time applications”, RFC 1889, IETF January 1996.

[16] R. Droms, “Dynamic host configuration protocol”, RFC 2131, IETF March 1997.

[17] C. Perkins and D. Johnson, “Route optimization in mobile IP”, Internet Draft, IETF February 2000.

[18] J. Rosenberg, H. Schulzrinne, “Session Initiation Protocol (SIP): Locating SIP servers”, RFC 3263, June 2002

[19] H. Schulzrinne, “Personal mobility for multimedia services in the Internet”, in European Workshop on Interactive Distributed Multimedia Systems and Services (IDMS), Berlin - Germany, March 1996.

[20] J. Peterson, “Enumservice registration for Session Initiation Protocol (SIP) Address-of-record”, RFC 3764, IETF April 2004.

[21] Vovida Open Communication Library (VOCAL), http://www.vovida.org

[22] Bob Stearns, UCNS Senior Consultant, “Unix named pipes (FIFOs)”, http://www.eits.uga.edu/tti/Computer_Review/Winter95/UNIX.html

[23] Glade: the GTK+ user interface builder, http://glade.gnome.org/

[24] H. Schulzrinne, “SIP Registration”, Internet Draft, IETF October 2001.