SUMMER INTERNSHIP PROJECT REPORT

VoIP Communication Using SIP

SUBMITTED TO

ISSA, DRDO

DEFENCE RESEARCH & DEVELOPMENT ORGANIZATION

As a part of Summer Internship in the degree of

Bachelor of technology

SUBMITTED BY:

SITANSHU CHAUHAN
19/BIT/058
&
KUSHAGRA ANAND
05814802719

UNDER SUPERVISION OF:

Sh. Vivek Kumar Sharma
(Scientist 'E', ISSA)
DRDO

ACKNOWLEDGEMENT

We take this opportunity to express our profound sense of gratitude and respect to all those who helped us throughout our project.

This report acknowledges the intense driving and technical competence of the entire individual that have contributed to it. It would have been almost impossible to complete this project without the support of these people. We extend thanks and gratitude to **Sh. S.B.Taneja (Scientist 'H', Director ISSA,DRDO) & Sh. Vivek Kumar Sharma (Scientist 'E', ISSA, DRDO)** who has imparted us guidance in all aspects. They shared their valuable time from their busy schedule to guide us and provide their active and sincere support for our activities.

This report is an authentic record of our own work which is accomplished by the sincere and active support of our mentor. We have tried our best to summarize this report.

SITANSHU CHAUHAN
B. Tech IT, V Semester
Gautam Buddha University
&
KUSHAGRA ANAND
B. Tech CS, V Semester
Maharaja Agrasen Institute Of Technology

ABSTRACT

The **VoIP COMMUNICATION USING SIP** is a server based software that creates and manages a session, where a session is considered an exchange of data between an association of participants. It provides a user-friendly environment for users to communicate between themselves.

It uses SIP (Session Initiation Protocol) which is one of the most common protocols used in VoIP technology. VoIP is a technology that allows us to deliver voice and multimedia content over the internet. It is one of the cheapest ways to communicate anytime anywhere with internet availability.

This is a JAVA based server program which handles clients. The software makes excessive use of JAVA Socket API to receive requests and forward respective responses. It maintains details of registered clients i.e IP address and local port and their numbers. Upon receiving a call request by any registered user, it generates an appropriate SIP response and forwards it to its destination. This software can also act as a proxy server that can communicate as a proxy with another instance of the same software running on a different machine.

The VoIP COMMUNICATION USING SIP is accessible only by an administrator. The end- user only needs to know the server's IP address to initiate communication upon successful registration by the server. Admin's main function is to maintain the server. Admin will have complete access and control over the complete software.

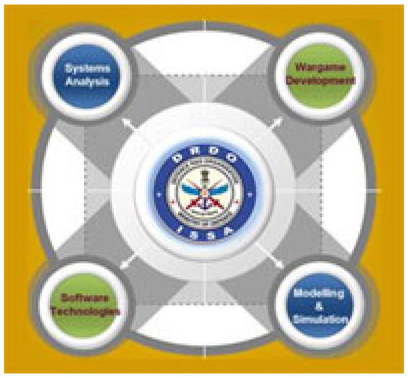
Thus, a major advantage of our software is that it can avoid the toll charge by ordinary telephone service by using fixed charge IP network services such as broadband.

TABLE OF CONTENTS

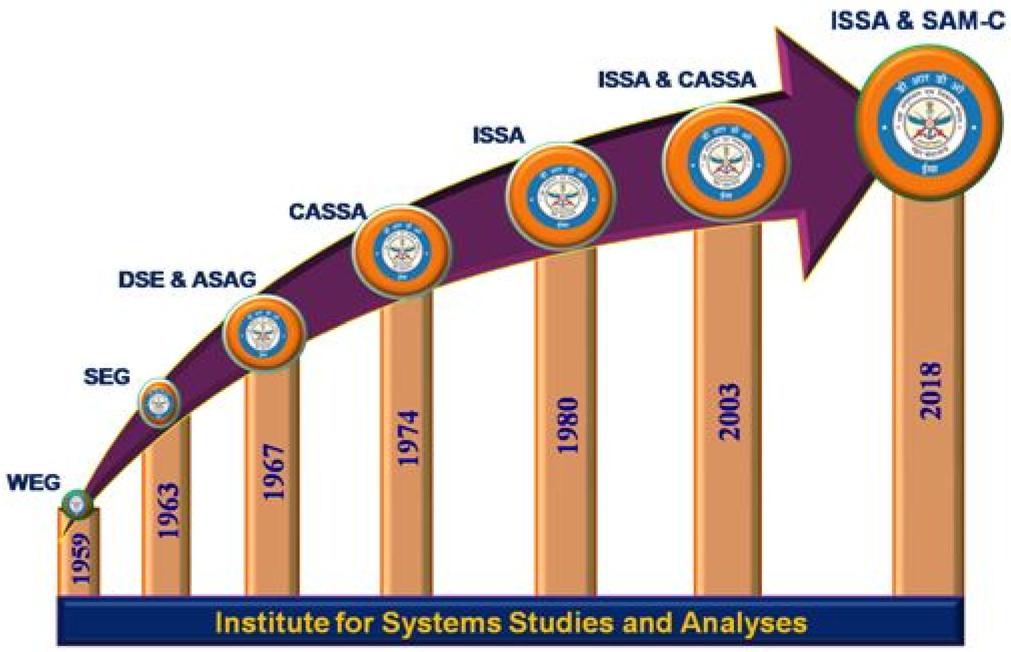
INTRODUCTION OF ISSA, DRDO

(INSTITUTE OF SYSTEMS STUDIES & ANALYSES)

ISSA is involved in System Analysis, Modeling & Simulation for various defence applications pertaining to employment/deployment, tactics & force potential evaluation, tactical/strategic & mission planning etc. We develop wargames for all the three Services.



HISTORY



The idea to apply Operational Research (OR) and Systems Analysis techniques in the Indian defence environment was conceived as early as the year 1959. A small group called Weapon Systems Analysis Group (WEG) was created to carry out Operational Research and Cost-effectiveness studies for weapons and equipment for the three Services, inter-service Organizations and the Ministry of Defence.

With the diversification of activities, WEG was renamed as Scientific Evaluation Group (SEG) in the year 1963 and later in the year 1968, SEG became a full-fledged Directorate named as Directorate of Scientific Evaluation (DSE). The Directorate continued to provide consultancy services to Services HQrs, SA to RM, DRDO HQrs and the Ministry of Defence in the fields of strategic defence planning and analysis, weapons evaluation, damage assessment, performance evaluation and OR applications, etc. During this period, a large number of systems analysis studies were carried out. The results and recommendations of these studies contributed to the top level decision making process.

Consequent to the reorganization of the system analysis activities within DRDO, the functions of DSE were redefined and was given the present name, i.e. Institute for Systems Studies & Analyses (ISSA) in the year 1980. In the year 1972, a small group named the Aeronautical Systems Analysis Group (ASAG) was created with an objective to carry out aeronautical systems studies and analyses. This group was functioning from National Aeronautics Laboratory, Bangalore as a detachment of the Directorate of Aeronautics. In the year 1974, ASAG was converted into a self-accounting full-fledged unit named as Centre for Aeronautical Systems Studies & Analyses (CASSA) and was shifted to the premises of Aeronautical Development Establishment, Bangalore. From 1974 to 2003 ,CASSA contributed significantly to a series of systems analysis studies attributed to DRDO HQrs and the Indian Air Force. It contributed to several design and development activities and policy level issues with its objective analyses.

In the year 2003, CASSA was merged with ISSA with an objective to synergies systems analysis activities and wargaming development processes under integrated combat environment. With this, ISSA has emerged as a nodal systems analyses lab and in 2013 ISSA is placed under SAM Cluster with the mandate in the field of Training & Planning Wargames, Integrated Air Defence & EW, Combat Modeling, Simulation System Analysis and computerised Wargame

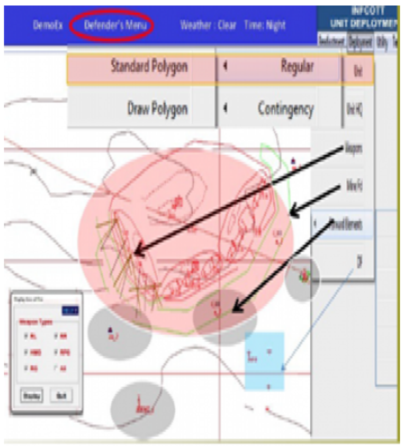
VISION

Transform ISSA into a centre of excellence in system analysis, modelling & simulation of defence systems to meet the challenges of the present and future requirements of the armed forces.

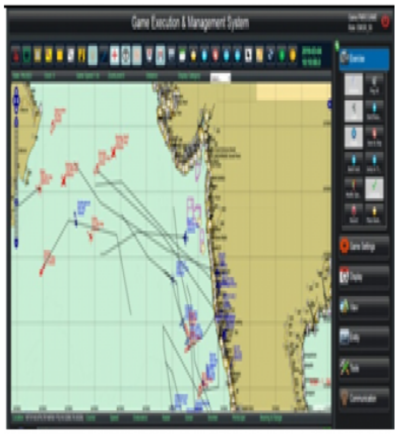
MISSION

Conduct system study and develop high quality integrated software for system analysis & decision support in application areas of sensors & weapons, Electronic Combat, Land & Naval Combat, Air-to-Air Combat and Air Design, Mission Planning, Tactics development and Training.

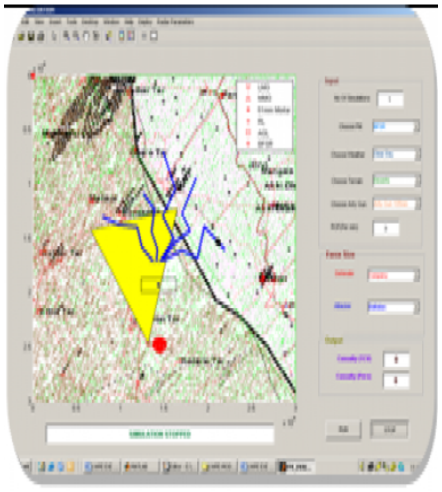
PRODUCTS



Land Wargaming Systems



Naval Wargaming System



Systems Analysis Tools

SOFTWARE AND TECHNOLOGIES USED

INTRODUCTION

The project aims at developing a communication server entitled as "**VoIP Communication Using SIP**". This application primarily allows the client to easily register his/her device and communicate among various registered clients.

The Software is for communication using UDP.
It maintains two levels of users:

Server Level

Client Level

The Software includes:

Maintaining client registering details.

Providing important information like transfer of data packets, invite messages, registered device messages etc.

Generating calls based on a certain request number.

PURPOSE

The goal of our system is to develop and implement the time effective, user friendly software.

OBJECTIVE

The main objective of VoIP communication using SIP is to increase communication between client and server which can help in providing a better platform for handling communication among clients. The scope of the project is very wide. This project is very flexible and can be easily expandable.

This application provides creation and management of a session, where a session is considered an exchange of data between an association of participants. The implementation of these applications is complicated by the practices of participants: users may move between endpoints, they may be addressable by multiple names, and they may communicate in several different media – sometimes simultaneously.

UDP (User Datagram Protocol) have been authored that carry various forms of real-time multimedia session data such as voice, video, or text messages. The Session Initiation Protocol (SIP) works in concert with these protocols by enabling Internet endpoints (called user agents) to discover one another and to agree on a characterization of a session they would like to share. For locating

prospective session participants, and for other functions, SIP enables the creation of an infrastructure of network hosts (called proxy servers) to which user agents can send registrations, invitations to sessions, and other requests.
SIP is an application-layer control protocol that can establish, modify, and terminate multimedia sessions (conferences) such as Internet telephony calls.
SIP can also invite participants to already existing sessions, such as multicast conferences. Media can be added to (and removed from) an existing session. SIP transparently supports name mapping and redirection services, which supports personal mobility - users can maintain a single externally visible identifier regardless of their network location.

SIP is an agile, general-purpose tool for creating, modifying, and terminating sessions that works independently of underlying transport protocols and without dependency on the type of session that is being established.

TECHNOLOGY USED

JAVA

JAVA Socket API

UDP

SIP

SOFTWARE USED

NetBeans 8.0.2 IDE

JDK 1.6 or higher

WireShark

Phoner (Virtual IP messenger)

Windows (client)

Linux (server)

HARDWARE REQUIRED

A server

ADVANTAGES

Low Cost

Portable

Reliable

Cost effective

No Extra Cables

SALIENT FEATURES

The major advantage of SIP is in its support for both IP and conventional telephone communication.

Highly flexible, scalable and customizable.

It is scalable, easy to implement, and requires less setup time.

Since SIP can be used to modify any session in progress, a normal telephone call session can be converted into a multi-party videoconference. Users can join in the session no matter what kind of terminal he is using or where he is located. The other person may be logged on to the Internet through a PC, or may be traveling with a cell phone.

OVERVIEW OF SIP

Given below are a few points to note about SIP:

SIP is a signalling protocol used to create, modify, and terminate a multimedia session over the Internet Protocol. A session is nothing but a simple call between two endpoints. An endpoint can be a Smartphone, a laptop, or any device that can receive and transmit multimedia content over the Internet.

SIP is an application layer protocol defined by IETF (Internet Engineering Task Force) standard. It is defined in **RFC 3261**.

SIP is incorporated with two widely used internet protocols: **HTTP** for web browser and **SMTP** used for email. From HTTP, SIP borrowed the client-server architecture and the use of URL and URI. From SMTP, it borrowed a text encoding scheme and a header style.

SIP takes the help of SDP (Session Description Protocol) which describes a session and RTP (Real Time Transport Protocol) used for delivering voice and video over IP network.

SIP can be used for two-party (uncast) or multiparty (multicast) sessions.

Other SIP applications include file transfer, instant messaging, video conferencing, online games, and steaming multimedia distribution.

ADVANTAGES OF VoIP

Low cost

Portability

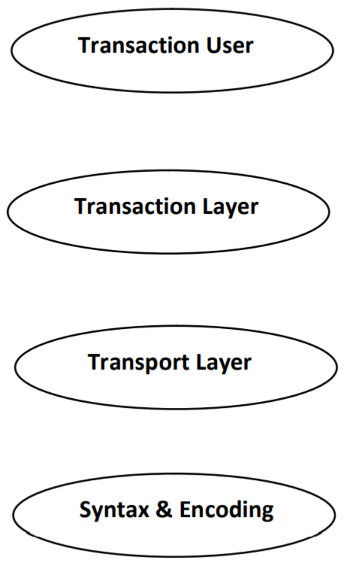
No extra cables

Flexibility

Video conferencing

SIP SYSTEM ARCHITECTURE

SIP is structured as a layered protocol, which means its behavior is described in terms of a set of fairly independent processing stages with only a loose coupling between each stage.



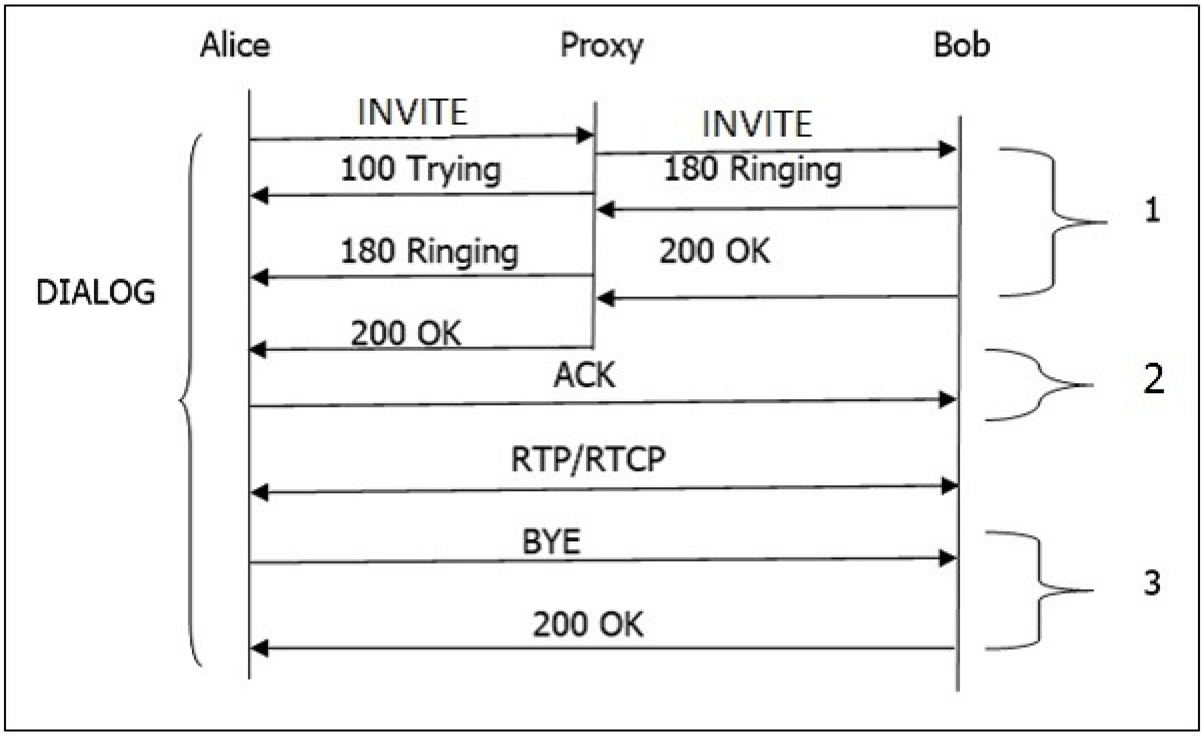
The lowest layer of SIP is its **syntax and encoding**. Its encoding is specified using an augmented **Backus-Naur Form grammar** (BNF).

At the second level is the **transport layer**. It defines how a Client sends requests and receives responses and how a Server receives requests and sends responses over the network. All SIP elements contain a transport layer.

Next comes the **transaction layer**. A transaction is a request sent by a Client transaction (using the transport layer) to a Server transaction, along with all responses to that request sent from the server transaction back to the client. Any task that a user agent client (UAC) accomplishes takes place using a series of transactions. **Stateless proxies** do not contain a transaction layer.

The layer above the transaction layer is called the **transaction user**. Each of the SIP entities, except the **stateless proxy**, is a transaction user.

BASIC CALL FLOW IN SIP



Given below is a step-by-step explanation of the above call flow:

An INVITE request that is sent to a proxy server is responsible for initiating a session.

The proxy server sends a **100 Trying** response immediately to the caller (Alice) to stop the re-transmissions of the INVITE request.

The proxy server searches the address of Bob in the location server. After getting the address, it forwards the INVITE request further.

Thereafter, **180 Ringing** (Provisional responses) generated by Bob is returned back to Alice.

A **200 OK** response is generated soon after Bob picks the phone up.

Bob receives an **ACK** from Alice, once it gets **200 OK**.

At the same time, the session gets established and RTP packets (conversations) start flowing from both ends.

After the conversation, any participant (Alice or Bob) can send a **BYE** request to terminate the session.

BYE reaches directly from Alice to Bob bypassing the proxy server.

Finally Bob sends a **200 OK** response to confirm the BYE and the session is terminated.

In the above basic call flow, three **transactions** are (marked as 1, 2, 3) available.

The complete call (from INVITE to 200 OK) is known as a **Dialog**.

INTRODUCTION TO SIP MESSAGES

SIP messages are of two types: **REQUESTS** and **RESPONSES**.

The opening line of a request contains a method that defines the request, and a Request-URI that defines where the request is to be sent.

Similarly the opening line of a response contains a response code.

REQUEST METHODS

SIP requests are the codes used to establish a communication. To complement them, there are **SIP responses** that generally indicate whether a request succeeded or failed.

There are commands known as METHODS that make a SIP message workable.

METHODS can be regarded as SIP requests, since they request a specific action to be taken by another user agent or server.

METHODS are distinguished into two types:

Core Methods

Extension Methods

CORE METHODS

There are six core methods as discussed below.

INVITE

INVITE is used to initiate a session with a user agent. In other words, an INVITE method is used to establish a media session between the user agents.

INVITE can contain the media information of the caller in the message body.

A session is considered established if an INVITE has received a success response (2xx) or an ACK has been sent.

A successful INVITE request establishes a **dialog** between the two user agents which continues until a BYE is sent to terminate the session.

An INVITE sent within an established dialog is known as a **re-INVITE**.

Re-INVITE is used to change the session characteristics or refresh the state of a dialog.

INVITE Example

The following code shows how INVITE is used.

```
INVITE sips:Bob@TMC.com SIP/2.0
Via: SIP/2.0/TLS client.ANC.com:5061;branch=z9hG4bK74bf9
Max-Forwards: 70
From: Alice<sips:Alice@atlanta.com>;tag=1234567
To: Bob<sips:Bob@TMC.com>
Call-ID: 12345601@ANC.com
CSeq: 1 INVITE
Contact: <sips:Alice@client.ANC.com>
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY
Supported: replaces
Content-Type: application/sdp
Content-Length: ...
v=0
o=Alice 2890844526 2890844526 IN IP4 client.ANC.com
s=Session SDP
c=IN IP4 client.ANC.com
t=3034423619 0
m=audio 49170 RTP/AVP 0
a=rtpmap:0 PCMU/8000
```

BYE

BYE is the method used to terminate an established session. This is a SIP request that can be sent by either the caller or the callee to end a session.

It cannot be sent by a proxy server.

BYE request normally route end to end, bypassing the proxy server.

BYE cannot be sent to a pending INVITE or an unestablished session.

REGISTER

REGISTER request performs the registration of a user agent. This request is sent by a user agent to a registrar server.

The REGISTER request may be forwarded or proxied until it reaches an authoritative registrar of the specified domain.

It carries the AOR (Address of Record) in the **To**header of the user that is being registered.

REGISTER request contains the time period (3600 sec).

One user agent can send a REGISTER request on behalf of another user agent. This is known as **third-party registration**. Here, the **From** tag contains the URI of the party submitting the registration on behalf of the party identified in the **To** header.

CANCEL

CANCEL is used to terminate an unestablished session. User agents use this request to cancel a pending call attempt initiated earlier.

It can be sent either by a user agent or a proxy server.

CANCEL is a **hop by hop** request, i.e., it goes through the elements between the user agent and receives the response generated by the next stateful element.

ACK

ACK is used to acknowledge the final responses to an INVITE method. An ACK always goes in the direction of INVITE. ACK may contain SDP body (media characteristics), if it is not available in INVITE.

ACK may not be used to modify the media description that has already been sent in the initial INVITE.

A stateful proxy receiving an ACK must determine whether or not the ACK should be forwarded downstream to another proxy or user agent.

For 2xx responses, ACK is end to end, but for all other final responses, it works on a hop by hop basis when stateful proxies are involved.

OPTIONS

OPTIONS method is used to query a user agent or a proxy server about its capabilities and discover its current availability. The response to a request lists the capabilities of the user agent or server. A proxy never generates an OPTIONS request.

EXTENSION METHODS

SUBSCRIBE

SUBSCRIBE is used by user agents to establish a subscription for the purpose of getting notification about a particular event.

It has a time period in the **Expires** header field that indicates the desired duration of existence of a subscription.

After the specified time period passes, the subscription is automatically terminated.

A successful subscription establishes a dialog between the user agents.

A subscription can be refreshed by sending another SUBSCRIBE within the dialog before the expiration time.

The server accepting a subscription returns a 200 OK.

Users can unsubscribe by sending another SUBSCRIBE method with Expires value 0 (zero).

NOTIFY

NOTIFY is used by user agents to convey the occurrence of a particular event. A NOTIFY is always sent within a dialog when a subscription exists between the subscriber and the notifier.

A 200 OK response is received for every NOTIFY to indicate that it has been received.

NOTIFY requests contain an **Event** header field indicating the package and a **subscription-state** header field indicating the current state of the subscription.

A NOTIFY is always sent at the start of a subscription and at the termination of a subscription.

PUBLISH

PUBLISH is used by a user agent to send event state information to a server known as an event state compositor.

PUBLISH is mostly useful when there are multiple sources of event information.

A PUBLISH request is similar to a NOTIFY, except that it is not sent in a dialog.

A PUBLISH request must contain an Expires header field and a Min-Expires header field.

INFO

INFO is used by a user agent to send call signalling information to another user agent with which it has established a media session. This is an end-to-end request and never generated by proxies. A proxy will always forward an INFO request.

UPDATE

UPDATE is used to modify the state of a session without changing the state of the dialog. UPDATE is used if a session is not established and the user wants to change the codec.

IF a session is established, a re-Invite is used to change/update the session.

MESSAGE

It is used to send an instant message or **IM** using SIP. An IM usually consists of short messages exchanged in real time by participants engaged in text conversation.

MESSAGE can be sent within a dialog or outside a dialog.

The contents of a MESSAGE are carried in the message body as a **MIME** attachment.

A **200 OK** response is normally received to indicate that the message has been delivered at its destination.

RESPONSE CODE IN SIP

A SIP response is a message generated by a user agent server (UAS) or SIP server to reply a request generated by a client. It could be a formal acknowledgement to prevent retransmission of requests by a UAC.

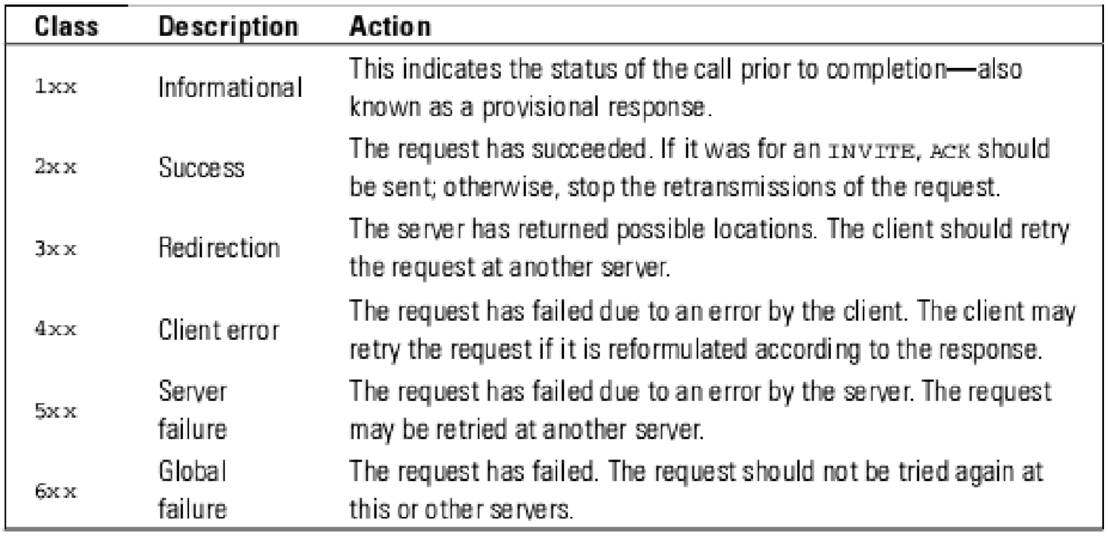
A response may contain some additional header fields of info needed by a UAC.

SIP has six responses.

1xx to 5xx has been borrowed from HTTP and 6xx is introduced in SIP.

1xx is considered as a **provisional** response and the rest are **final** responses.

Given below is the description of each response code :



Class	Description	Action
1xx	Informational	This indicates the status of the call prior to completion—also known as a provisional response.
2xx	Success	The request has succeeded. If it was for an INVITE, ACK should be sent; otherwise, stop the retransmissions of the request.
3xx	Redirection	The server has returned possible locations. The client should retry the request at another server.
4xx	Client error	The request has failed due to an error by the client. The client may retry the request if it is reformulated according to the response.
5xx	Server failure	The request has failed due to an error by the server. The request may be retried at another server.
6xx	Global failure	The request has failed. The request should not be tried again at this or other servers.

ABOUT SDP

SDP stands for Session Description Protocol. It is used to describe multimedia sessions in a format understood by the participants over a network. Depending on this description, a party decides whether to join a conference or when or how to join a conference.

The owner of a conference advertises it over the network by sending multicast messages which contain description of the session e.g. the name of the owner, the name of the session, the coding, the timing etc. Depending on these information the recipients of the advertisement take a decision about participation in the session.

SDP is generally contained in the body part of Session Initiation Protocol popularly called SIP.

SDP is defined in RFC 2327. An SDP message is composed of a series of lines, called fields, whose names are abbreviated by a single lower-case letter, and are in a required order to simplify parsing.

PURPOSE OF SDP

The purpose of SDP is to convey information about media streams in multimedia sessions to help participants join or gather info of a particular session.

SDP is a short structured textual description.

It conveys the name and purpose of the session, the media, protocols, codec formats, timing and transport information.

A tentative participant checks these information and decides whether to join a session and how and when to join a session if it decides to do so.

The format has entries in the form of <type>= <value>, where the <type>defines a unique session parameter and the <value>provides a specific value for that parameter.

The general form of a SDP message is:

```
x=parameter1 parameter2 ... parameterN
```

The line begins with a single lower-case letter, for example, x. There are never any spaces between the letter and the =, and there is exactly one space between each parameter. Each field has a defined number of parameters.

SESSION DESCRIPTION PARAMETERS

Session description (* denotes optional)

- v= (protocol version)
- o= (owner/creator and session identifier)
- s= (session name)
- i=* (session information)
- u=* (URI of description)
- e=* (email address)
- p=* (phone number)
- c=* (connection information -not required if included in all media)
- b=* (bandwidth information)
- z=* (time zone adjustments)
- k=* (encryption key)
- a=* (zero or more session attribute lines)

An SDP Example

Given below is an example session description, taken from RFC 2327 :

```
v=0
o=mhandley2890844526 2890842807 IN IP4 126.16.64.4
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.cs.ucl.ac.uk/staff/M.Handley/sdp.03.ps
e=mjh@isi.edu(Mark Handley)
c=IN IP4 224.2.17.12/127
t=2873397496 2873404696
a=recvonly
m=audio 49170 RTP/AVP 0
m=video 51372 RTP/AVP 31
m=application 32416udp wb
a=orient:portrait
```

NETBEANS IDE SOFTWARE

NetBeans is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called modules. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers.

The NetBeans IDE is primarily intended for development in Java, but also supports other languages , in particular PHP,C/C++ and HTML.

NetBeans is cross-platform and runs on Microsoft Windows, Mac OS X, Linux, Solaris and other platforms supporting a compatible JVM.

METHODOLOGY ADOPTED

I followed a 5-Step Development Methodology to develop the “**VoIP Communication using SIP**”. The 5-Step Development Methodology that I adopted is listed below:

Study and Learning : Learning java and studying all software to be used like server and communication software.

Layout : Scoping out the details and determining what type of design, programming, function, etc. is best suited for the particular solution for Communication between two clients that a development needs. Then refining and documenting of the design was carried out.

Development : Once the specs were detailed, I began the process to build a “VoIP using SIP”. During this phase the faculty/guide monitored the development through work in-progress.

Implementation : This is when the site went through tweaking and testing. I refined the codes and finalized the test of communication between two clients as per the project.

Demo : The final step of the methodology that I plan to follow will be to give a demo of the fully developed and tested “VoIP software using SIP”

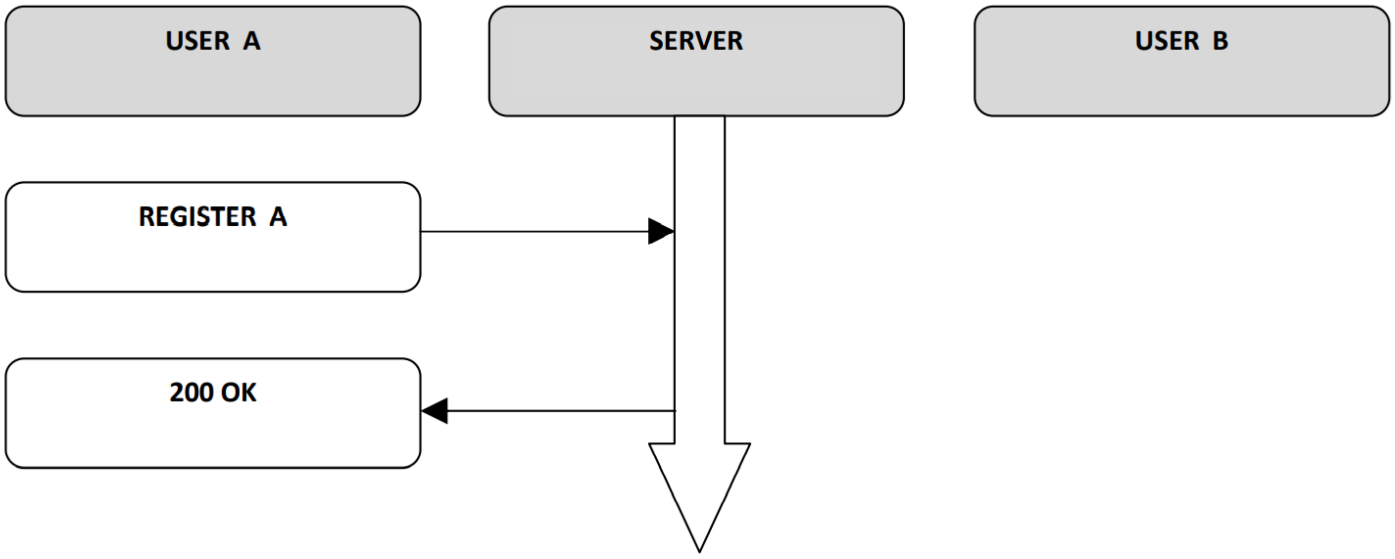
DESIGN AND IMPLEMENTATION

This chapter deals with the basic design and its implementation in creation of this software. This JAVA based server is designed using best OOM practices and takes full advantage of JAVA's Object Oriented Methodology.

BASIC FLOW CHARTS

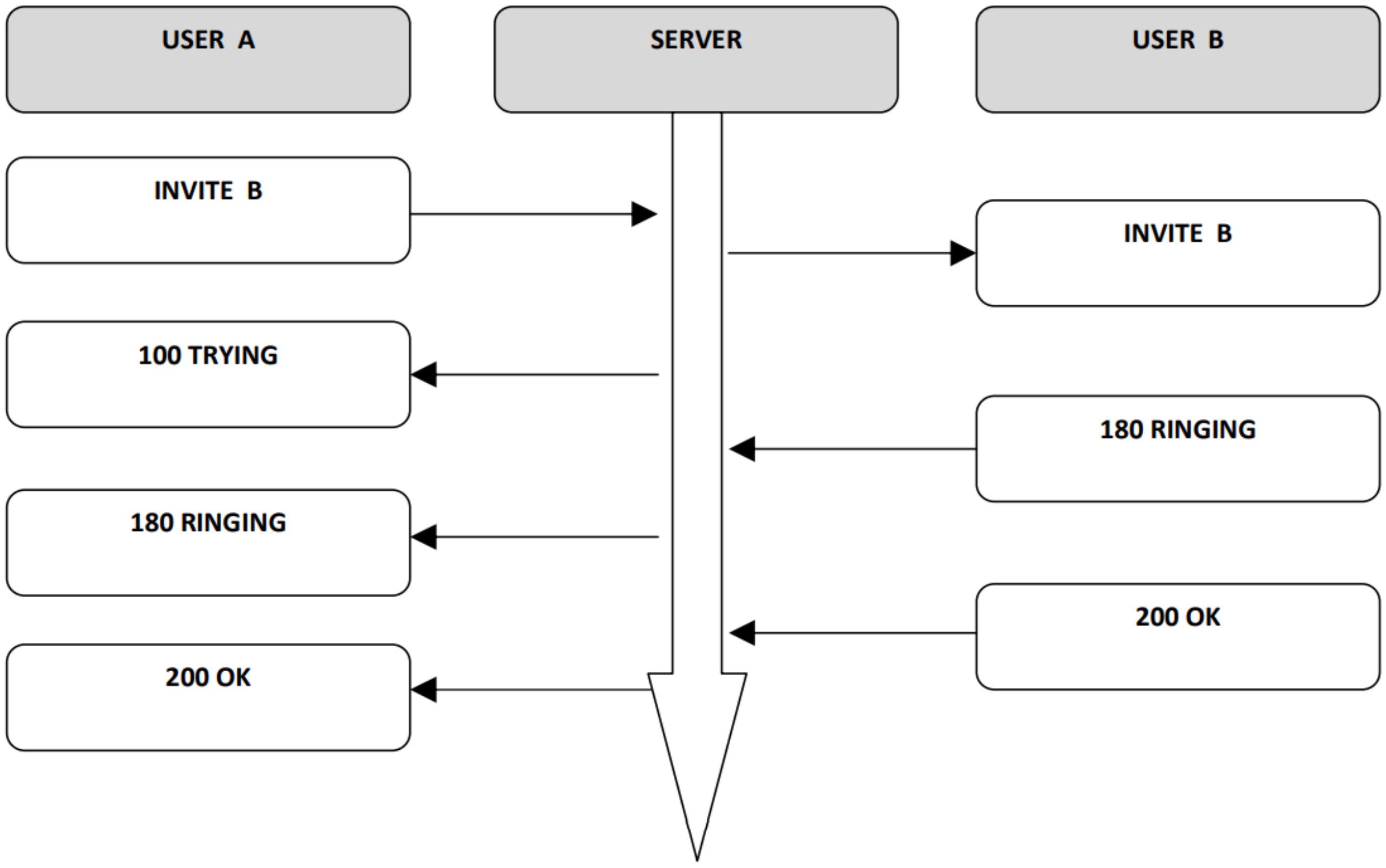
REGISTRATION OF USERS

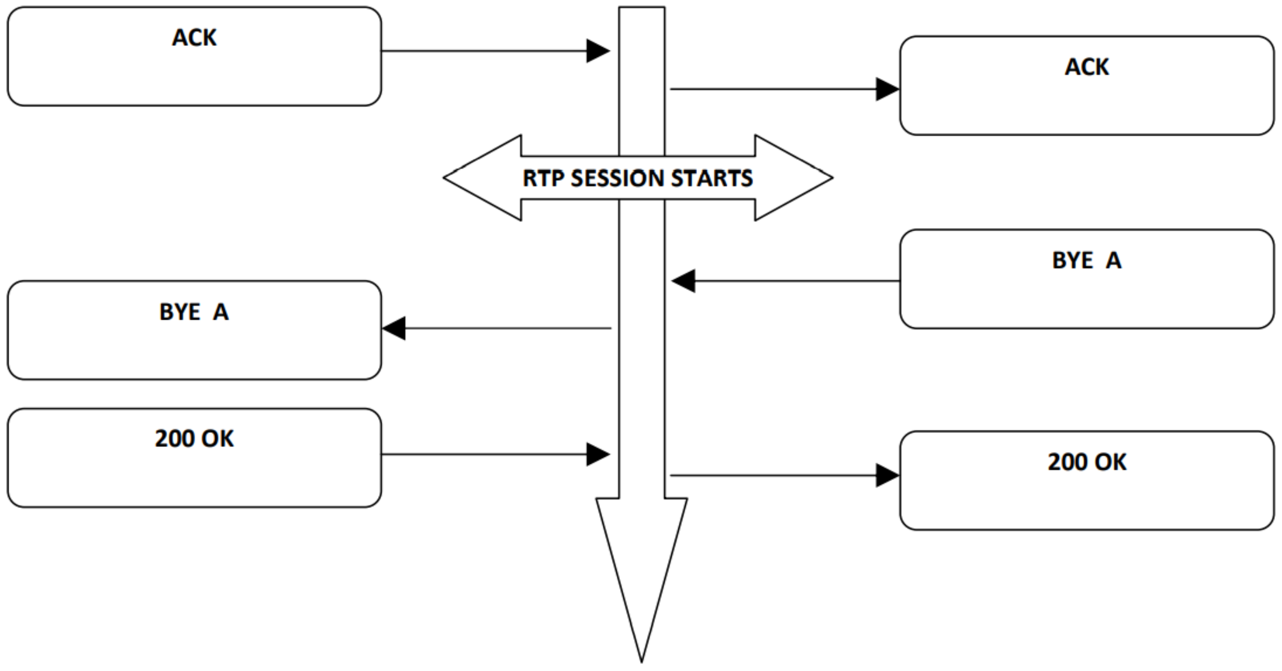
A user sends a REGISTER request to the server. The request includes the user's contact list. The SIP Server validates the user's credentials and if it succeeds then the user is successfully registered and the server replies with 200 OK response.



NORMAL FLOW

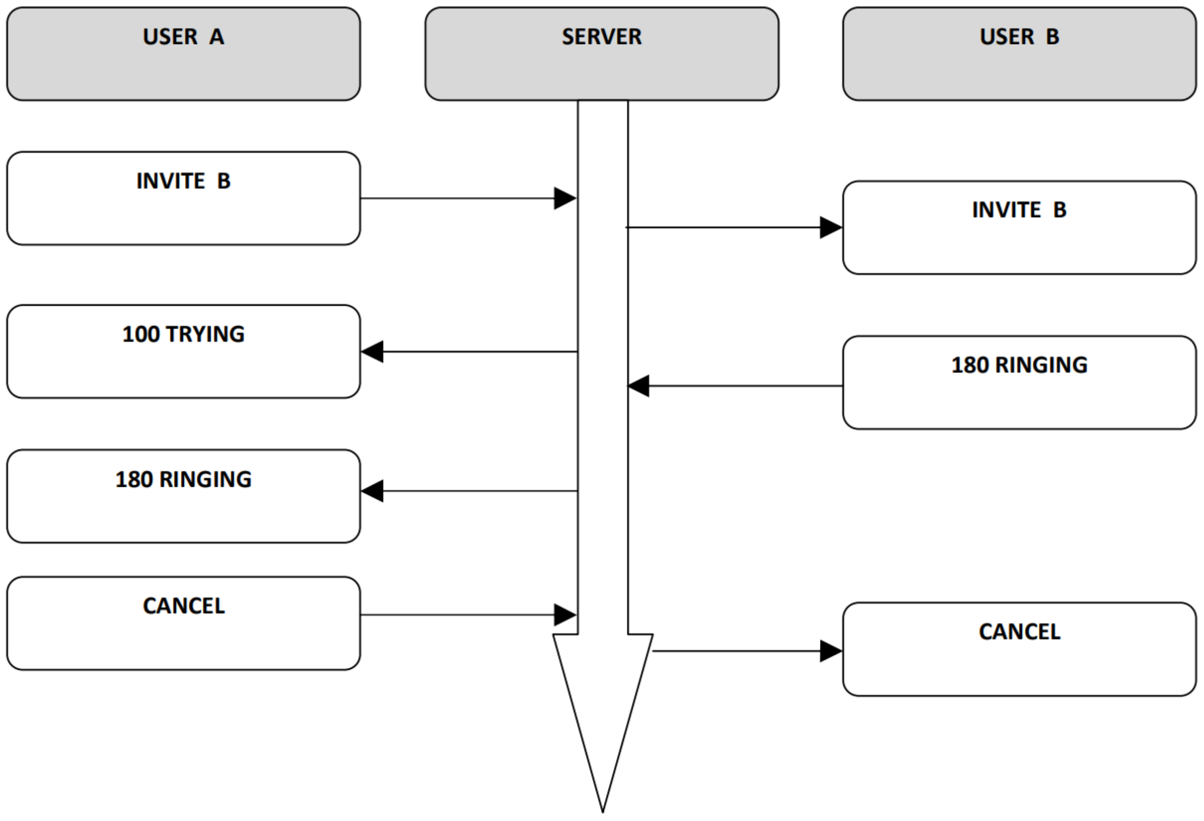
In this flow of control, the caller makes a call through an INVITE request which is handled by the server. The server figures out the correct recipient of the call and forwards it. It also sends back a TRYING 100 response back to the caller. The receiver recognizes the INVITE message and there is a ringing bell on the phone of the receiver. He picks up the call and a successful RTP connection is established between both the users. Any user can drop the call by sending a BYE request to the other user. The other user sends an appropriate ACK response and the session terminates.

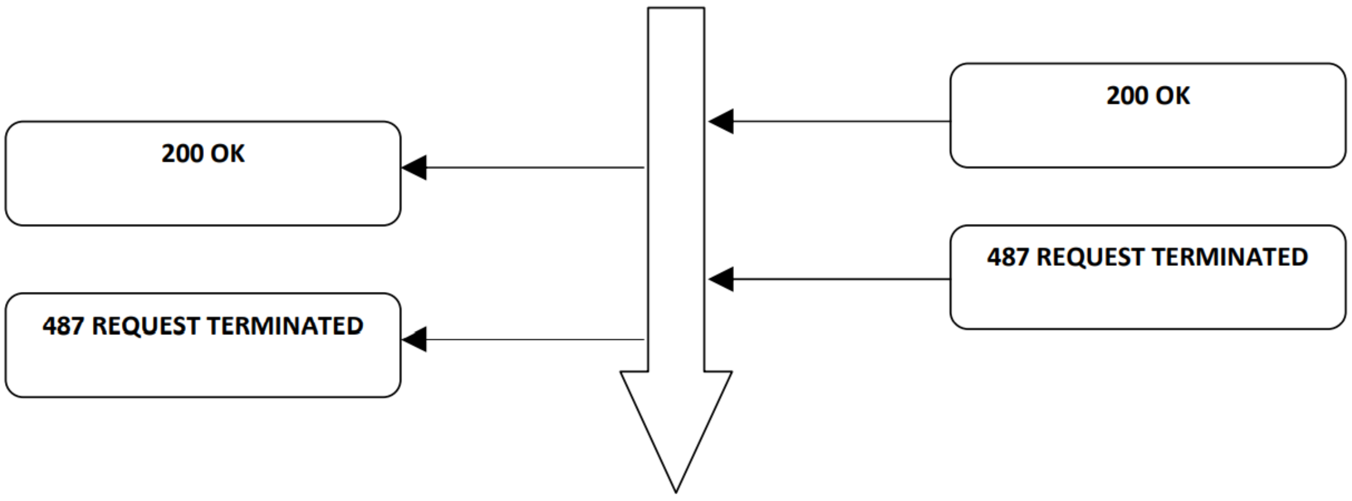




CANCELLATION PROCESS

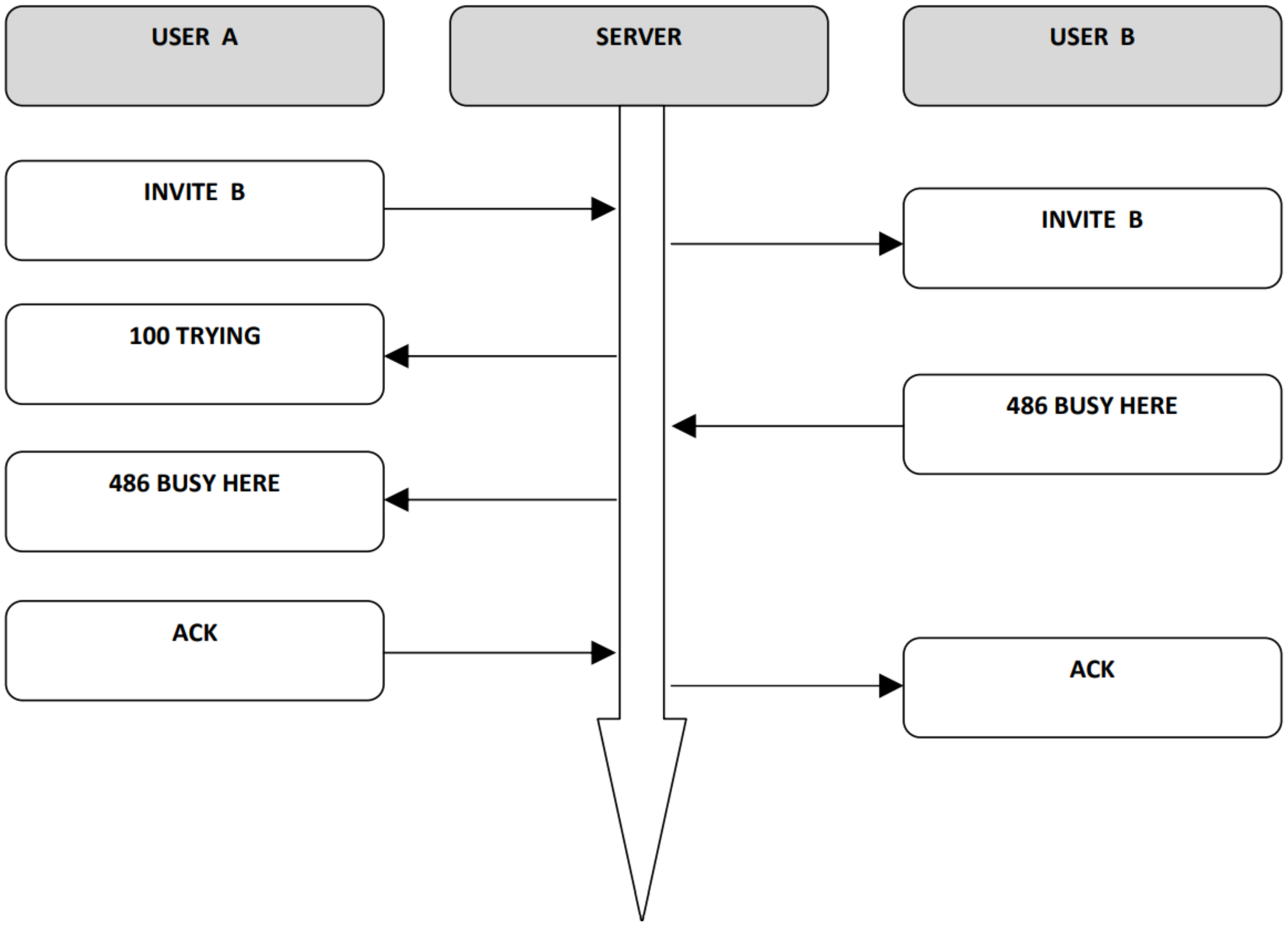
When the user who is calling, ends the call then a CANCEL request is sent to the server. The server forwards the CANCEL request to the receiver. Receiver sends an 200 OK and a 487 REQUEST TERMINATED response. The session is then cancelled successfully.





WHEN BUSY

When the receiver terminates the call during the ringing phase, then the caller receives 486 BUSY HERE response. In this scenario, the receiver is busy and sends a 486 BUSY HERE response to caller's INVITE. Note that the non 2xx response is acknowledged on a hop-by-hop basis instead of end-to-end. Also note that many SIP UAs will not return a 486 response, as they have multiple lines and other features.



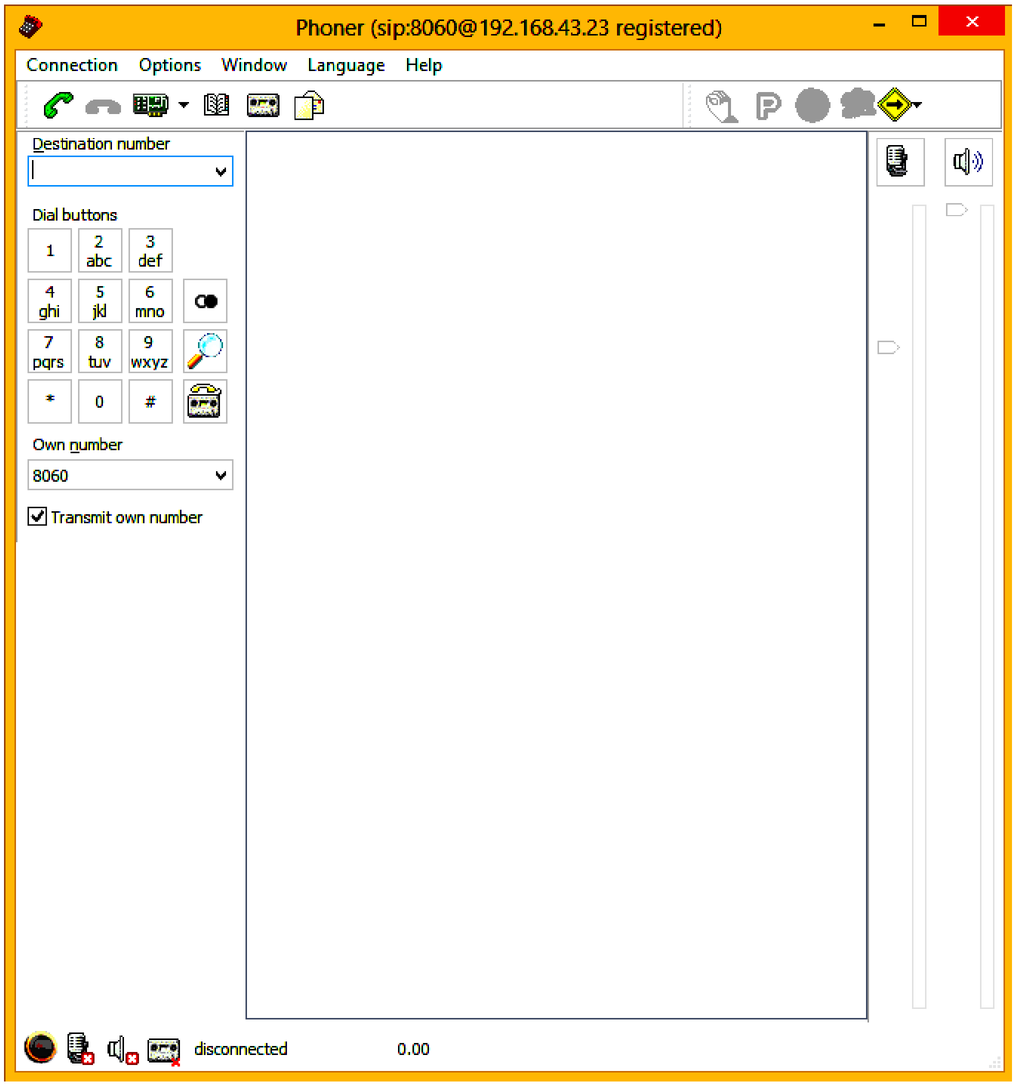
USAGE AND DISCUSSION

This chapter contains screenshots of the working of the software and basic guidelines of its usage. This is a step by step method that answers the question of how to actually use the software. Please note that this software is thoroughly tested under various conditions and found to be reliable for the user.

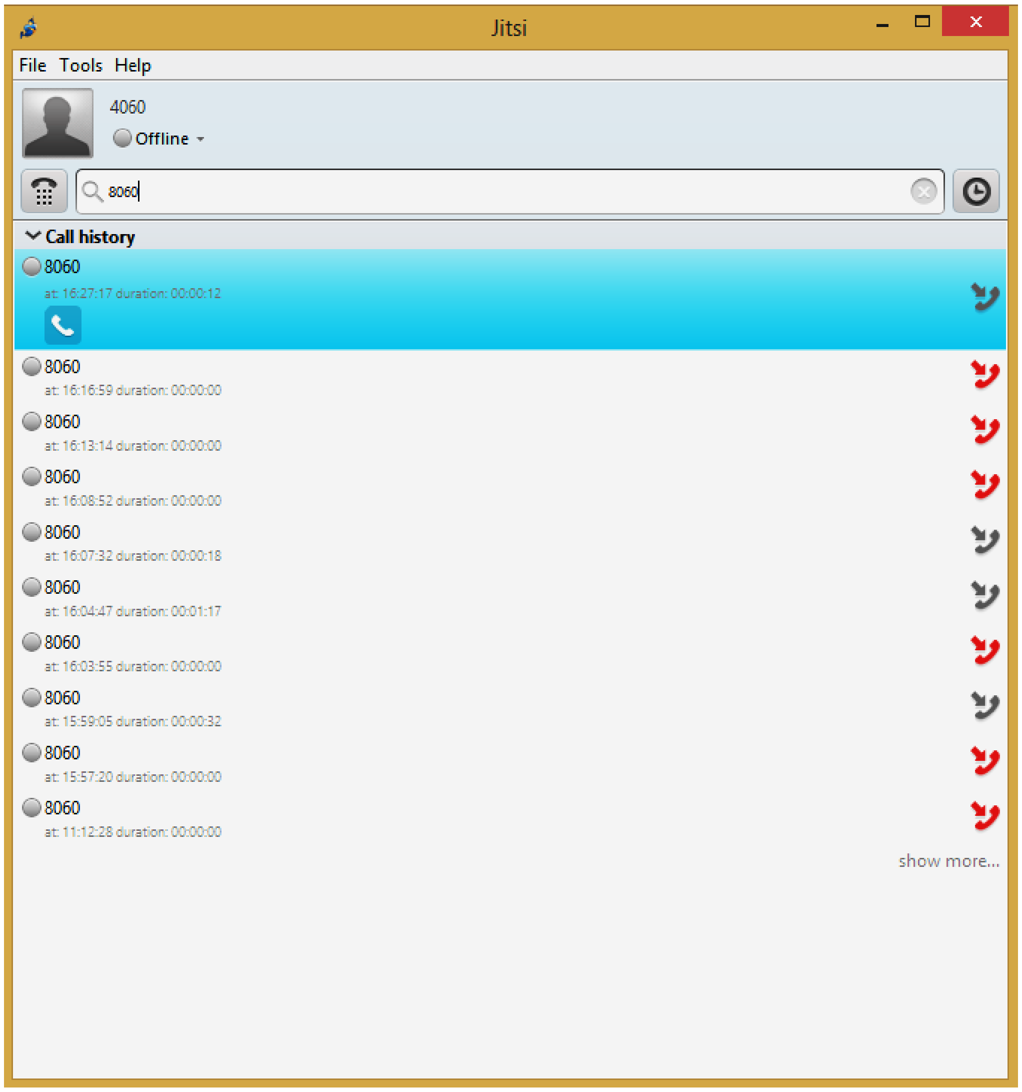
STEPS WITH SCREENSHOTS

- **DOWNLOAD PHONER (SIP COMMUNICATOR) ON CLIENT**

  Phoner is a freeware application for Microsoft Windows, from Win2k up to Win7. This application enables voice connections to landline, cellular network and VoIP. Phoner can be used as a softphone on stationary and mobile computer devices attached to analogue or digital (ISDN) subscriber lines and as SIP client.



Jitsi is also a popular SIP Communicator.



8060
at: 16:27:17 duration: 00:00:12

8060
at: 16:16:59 duration: 00:00:00

8060
at: 16:13:14 duration: 00:00:00

8060
at: 16:08:52 duration: 00:00:00

8060
at: 16:07:32 duration: 00:00:18

8060
at: 16:04:47 duration: 00:01:17

8060
at: 16:03:55 duration: 00:00:00

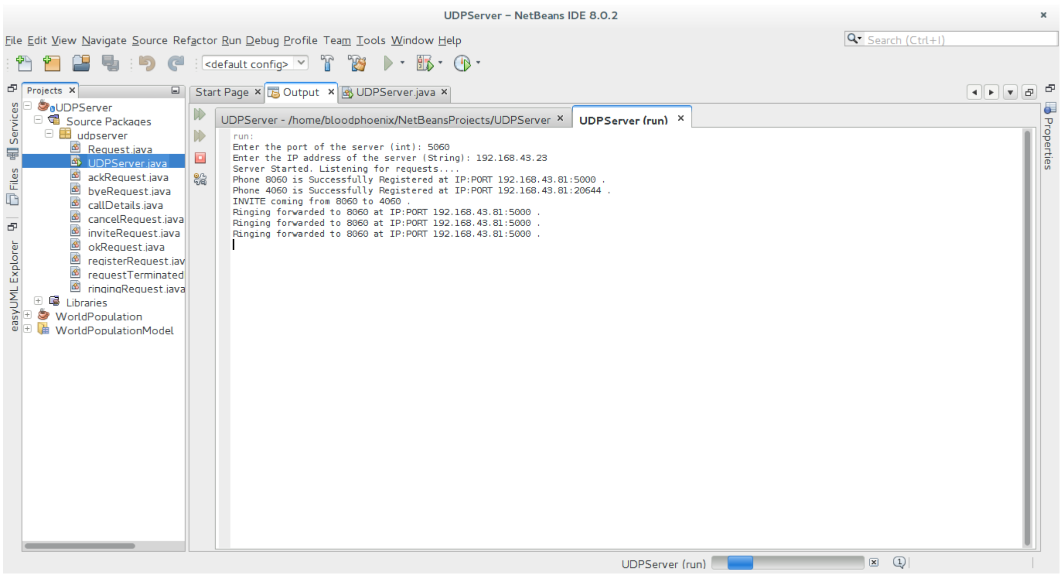
8060
at: 15:59:05 duration: 00:00:32

8060
at: 15:57:20 duration: 00:00:00

8060
at: 11:12:28 duration: 00:00:00

show more...

 <u>SET UP NETBEANS 8.0.2 or later with JDK 1.6 or later</u>



```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
Phone 8060 is Successfully Registered at IP:PORT 192.168.43.81:5000 .
Phone 4060 is Successfully Registered at IP:PORT 192.168.43.81:20644 .
INVITE coming from 8060 to 4060 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
```

- **RUN THE CODE ON A SERVER SPECIFYING THE PORT AND IP**

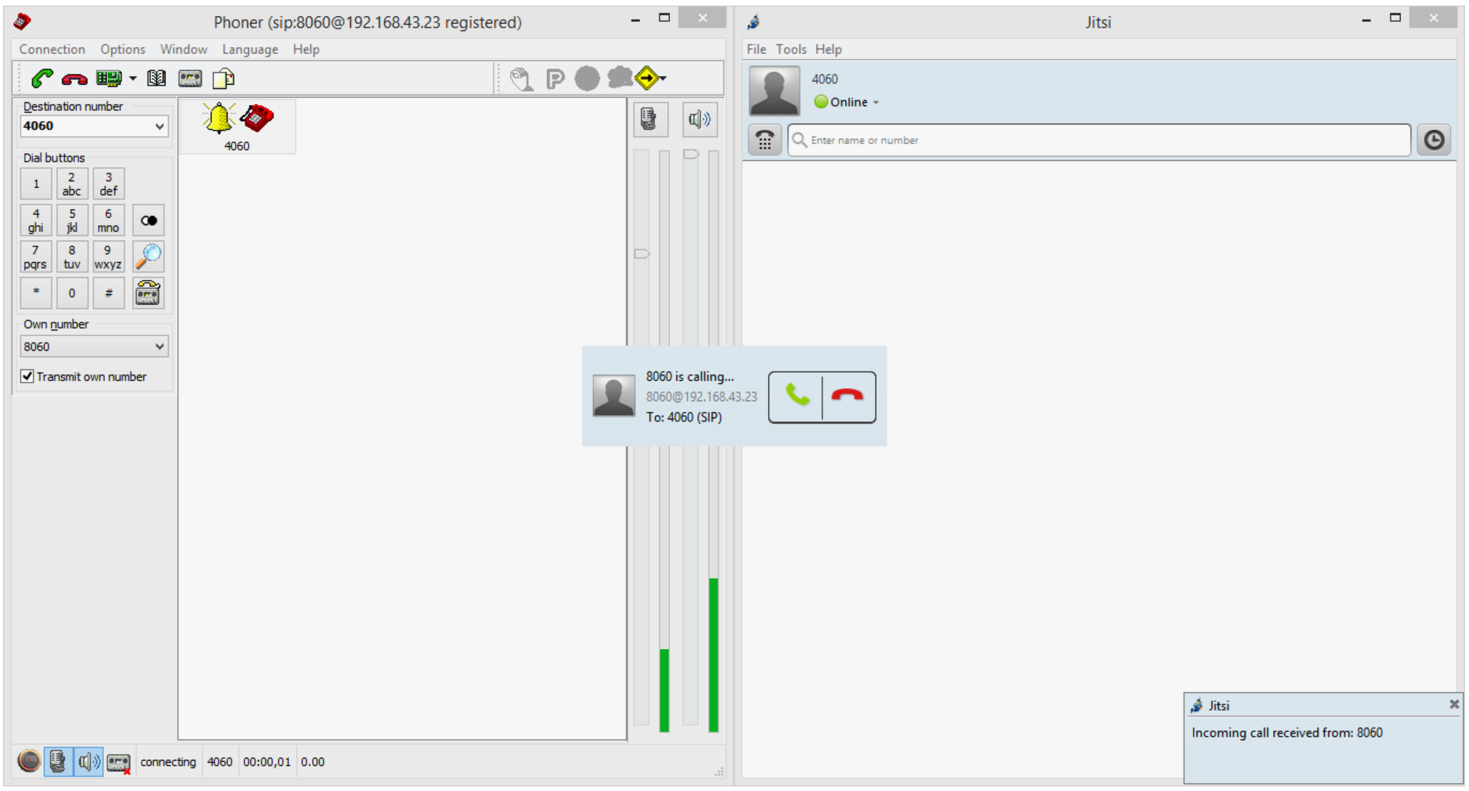
```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
```

Clients will register….

```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
Phone 8060 is Successfully Registered at IP:PORT 192.168.43.81:5000 .
Phone 4060 is Successfully Registered at IP:PORT 192.168.43.81:20644 .
```

- **RUN PHONER ON CLIENT COMPUTER AND SET SERVER IP AS REGISTRAR**

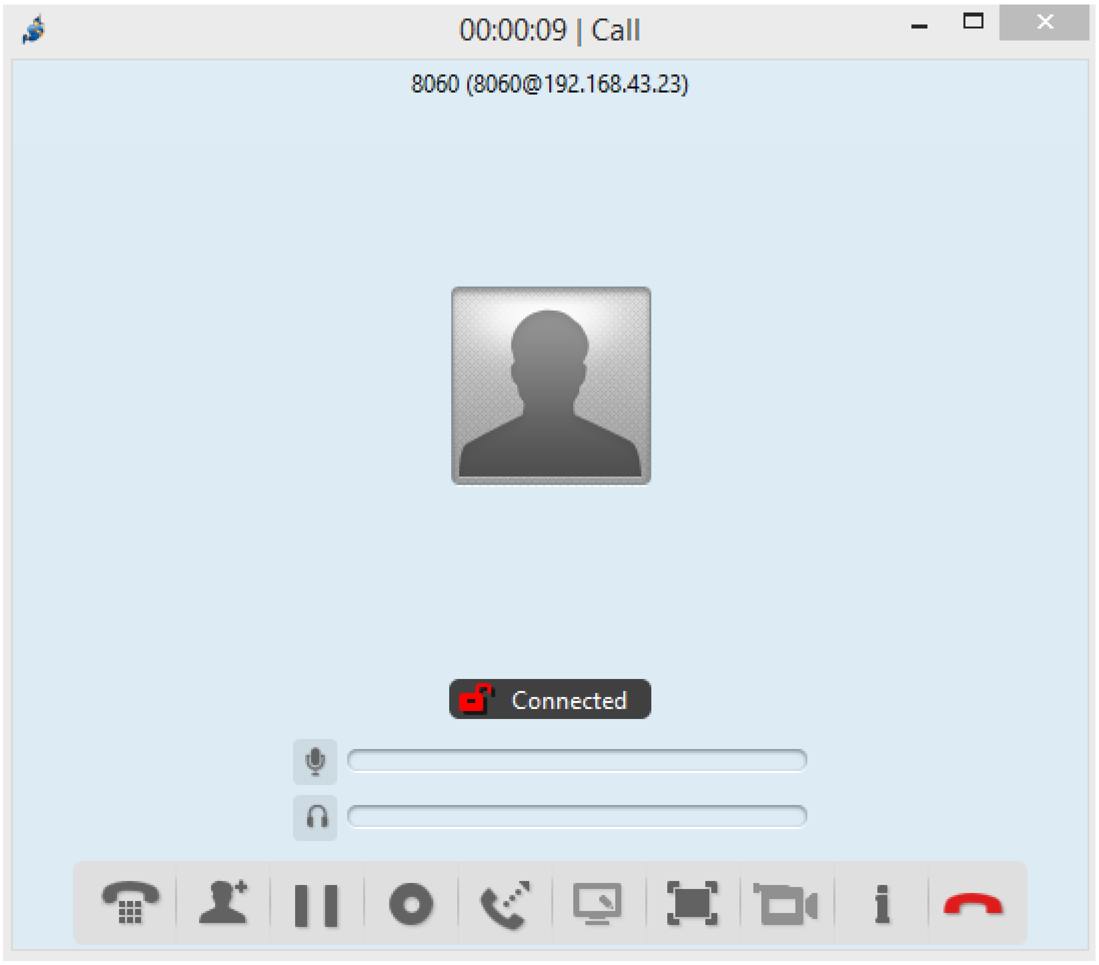
- **NOW DIAL THE NUMBER OF ANY REGISTERED USER AND CALL**



8060 is calling...
8060@192.168.43.23
To: 4060 (SIP)

Incoming call received from: 8060

- **RTP SESSION IS ESTABLISHED WHEN 'CONNECTED' IS DISPLAYED**

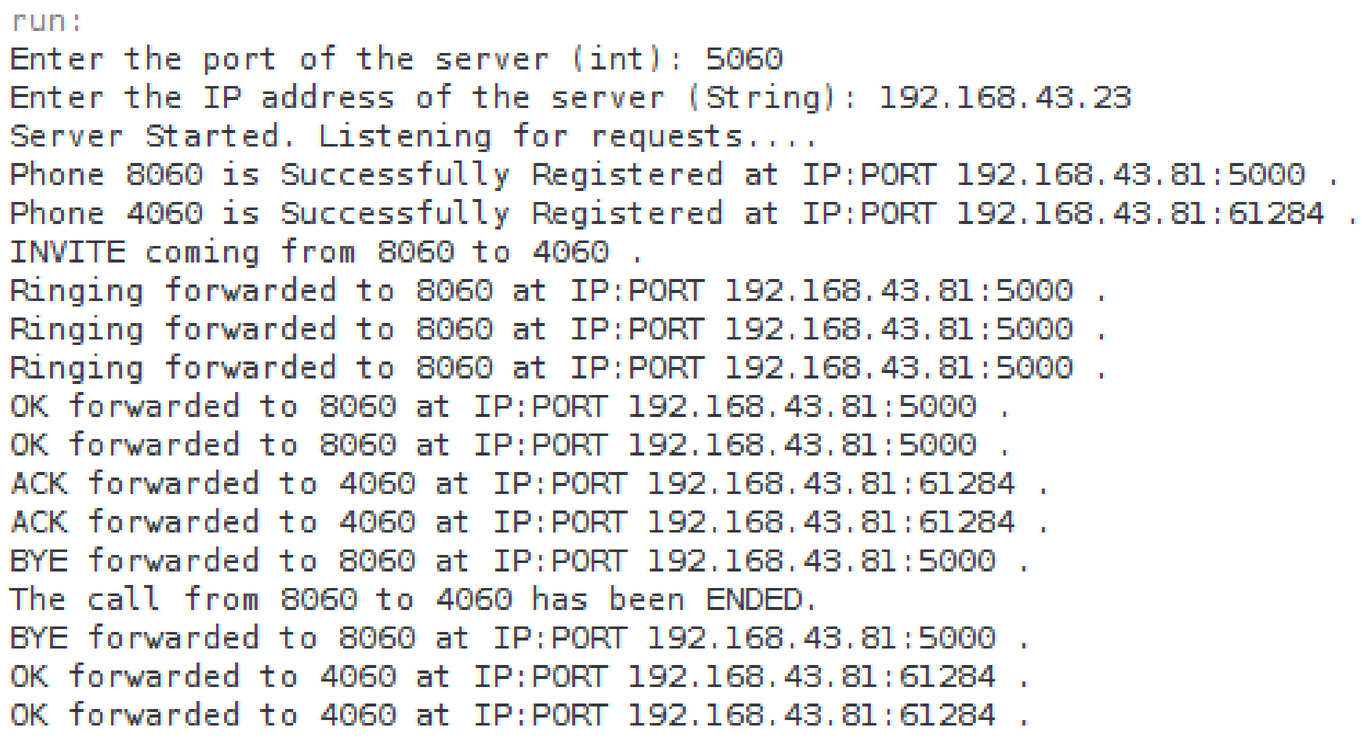


- **CLICK THE RED 'END CALL' WHEN FINISHED TALKING**

- **LOG SCREEN DURING 'BUSY HERE'**

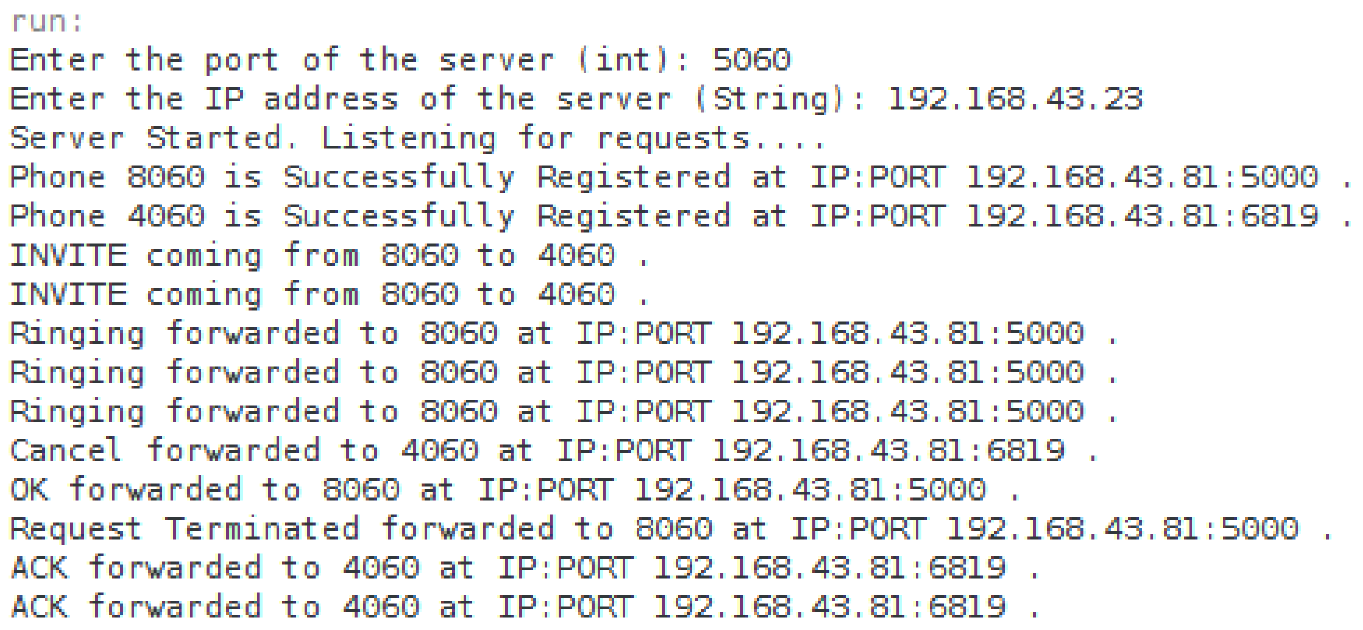
```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
Phone 8060 is Successfully Registered at IP:PORT 192.168.43.81:5000 .
Phone 4060 is Successfully Registered at IP:PORT 192.168.43.81:29034 .
INVITE coming from 8060 to 4060 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Busy Here forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Busy Here forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:29034 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:29034 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:29034 .
```

- **LOG SCREEN DURING NORMAL FLOW**



```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
Phone 8060 is Successfully Registered at IP:PORT 192.168.43.81:5000 .
Phone 4060 is Successfully Registered at IP:PORT 192.168.43.81:61284 .
INVITE coming from 8060 to 4060 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
OK forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
OK forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:61284 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:61284 .
BYE forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
The call from 8060 to 4060 has been ENDED.
BYE forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
OK forwarded to 4060 at IP:PORT 192.168.43.81:61284 .
OK forwarded to 4060 at IP:PORT 192.168.43.81:61284 .
```

- **LOG SCREEN DURING CANCELATION**



```
run:
Enter the port of the server (int): 5060
Enter the IP address of the server (String): 192.168.43.23
Server Started. Listening for requests....
Phone 8060 is Successfully Registered at IP:PORT 192.168.43.81:5000 .
Phone 4060 is Successfully Registered at IP:PORT 192.168.43.81:6819 .
INVITE coming from 8060 to 4060 .
INVITE coming from 8060 to 4060 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Ringing forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Cancel forwarded to 4060 at IP:PORT 192.168.43.81:6819 .
OK forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
Request Terminated forwarded to 8060 at IP:PORT 192.168.43.81:5000 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:6819 .
ACK forwarded to 4060 at IP:PORT 192.168.43.81:6819 .
```

CONCLUSION

VoIP Communication using SIP provides easy communication among users using only an internet connection or a local area network if available. The technologies used in developing this project are easily implemented and have vast support and user base worldwide.

REFERENCES

https://www.ietf.org/rfc/rfc3261.txt - RFC3261 Standard for SIP

https://tools.ietf.org/html/rfc3665 - RFC3665 Standard for SIP Call flow Examples

TCP/IP Sockets in JAVA, Second Edition : Practical Guide for programmers (The Practical Guides) 2nd Edition by Kenneth L. Calvert and Michael J. Donahoo

www.tutorialspoint.com/session_initiation_protocol - Understanding SIP

www.phoner.de - Phoner download

Voice over IP Fundamentals (2nd Edition) : Written by Jonathan Davidson (Author), James F. Peters (Author), Manoj Bhatia (Author), Satish Kalidindi (Author), Sudipto Mukherjee (Author)