

**KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY**

**KUMASI**

**COLLEGE OF SCIENCE**



**LEVERAGING VOIP ON LOCAL AREA NETWORK  
USING**

**JAVA MEDIA FRAMEWORK**

**(A CASE STUDY OF KUMASI TECHNICAL INSTITUTE)**

**By**

**GEORGE ASANTE**

**(B.Ed Information Technology)**

**A thesis submitted to the Department of Computer Science,  
Kwame Nkrumah University of Science and Technology,  
in partial fulfilment of the requirements for the degree of**

**MASTER OF PHILOSOPHY  
In  
INFORMATION TECHNOLOGY**

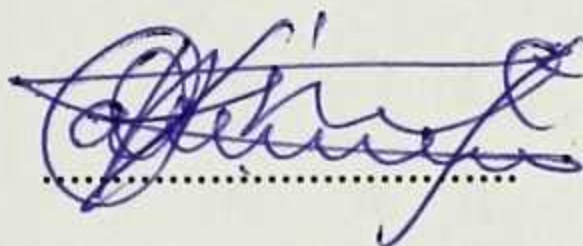
**June, 2012**



CERTIFICATION

I hereby declare that this submission is my own work towards the MPhil and that, to the best of my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgment has been made in the text.

George Asante



12/12/12

(PG 5092910)

Signature

Date

Certified by:

Dr. Michael Asante



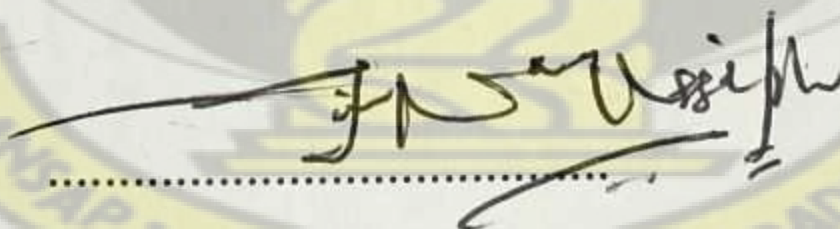
17/12/12

(Supervisor)

Signature

Date

Dr. Najim Ussiph



18/12/2012

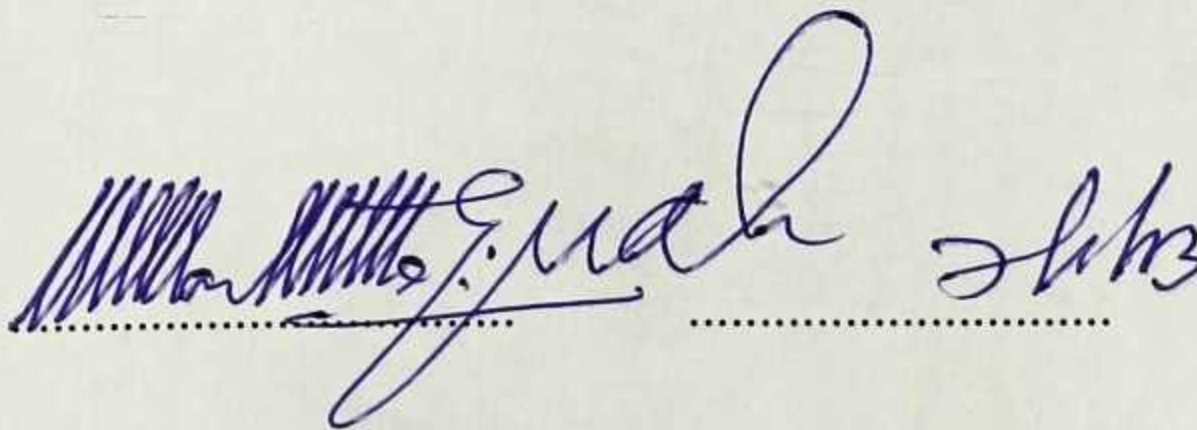
(Supervisor)

Signature

Date

Certified by:

Dr. J.B Hayfron-Acquah



(Head of Department)

Signature

Date

## DEDICATION

This work is dedicated to my wife, Joana Frempomaa, and my children,  
Rodaline Owusu Asante and Caleb Owusu Asante.

# KNUST





## ACKNOWLEDGMENT

I would like to express my sincere gratitude to the Almighty God for his favour and guidance throughout my study.

I also wish to express my profound gratitude to Dr. Michael Asante and Dr. Najim Ussiph, my supervisors, for the continuous support of my MPhil study and research, for their patience, motivation, enthusiasm, and immense knowledge. Their guidance helped me in the entire research and writing of this thesis. I could not have imagined having a better advisors and mentors for my MPhil study.

My next thanks goes to Mr. Stephen Agyepong, who helped me choose a topic for this study.

I would also like to extend my gratitude to Mr. Emmanuel Bassah of Saboney Technologies and Mr. Bright Ofosu, who gave me technical support during my research.

My sincere thanks also goes to the lectures at the Department of Computer Science, especially Dr. J.B Hayfron-Acquah, Mr. Dominic Asamoah and Mr. Emmanuel Oppong for their concern, encouragement, and insightful comments.

Last but not least, I would like to thank my family: my wife and children for their prayers and financial support.



## Table of Content

Title	Page
Certification.....	ii
Dedication.....	iii
Acknowledgment.....	iv
Table of Content.....	V
List of Figures.....	Viii
List of Tables.....	ix
Abstract.....	X
<b>CHAPTER ONE                      INTRODUCTION</b>	<b>1-7</b>
1.1 Background of the Study.....	1
1.2 Statement of the Problem.....	3
1.3 The Purpose of the Study.....	4
1.4 Research Questions.....	4
1.5 Hypothesis.....	5
1.6 Significance of the Study.....	5
1.7 Description of the Proposed System.....	6
1.8 Expected Results.....	6
1.9 Limitation of the Study.....	7



1.10 Organisation of the Study.....	7
<b>CHAPTER TWO</b>	<b>LITERATURE REVIEW</b>
	8-31
2.1 Analysis of the Existing System .....	8
2.2 Public Switched Telephone Network .....	11
2.3 Overview of VoIP.....	13
2.4 Different Forms of VoIP.....	14
2.5 How VoIP Works.....	15
2.6 Features of VoIP.....	17
2.7 Protocols.....	21
2.8 Signalling in VoIP.....	27
2.9 Socket Programming.....	28
2.10 Java Application Programming Interface (API ).....	29
<b>CHAPTER THREE</b>	<b>METHODOLOGY</b>
	33-56
3.1 System Analysis and Design.....	33
3.2 Text Chat.....	35
3.3 Audio Chat.....	35
3.4 Video Chat.....	38
3.5 File Transfer.....	39
3.6 System Use Cases.....	43



3.7 Goals.....	47
3.8 User Requirements.....	48
3.9 User Specification.....	50
3.10 Limitations and Restrictions.....	50
3.11 Functional Requirement.....	51
3.12 Non-Functional Requirement.....	52
3.13 Tools and Materials.....	53
 <b>CHAPTER FOUR IMPLEMENTATION, TESTING AND EVALUATION 57-70</b>	
4.1 User Interface.....	57
4.1.1 Server – Side GUI.....	57
4.1.2 Client – Side GUI.....	58
4.1.3 Database and Respective tables GUI.....	63
4.2 Testing.....	64
4.3 Evaluation.....	69
 <b>CHAPTER FIVE SUMMARY, CONCLUSION AND RECOMMENDATION</b>	
5.1 Summary.....	71
5.2 Conclusion.....	71
5.3 Recommendations.....	72



Bibliography..	73-75
Appendix A Codes for Text Chat .....	76
Appendix B Codes for Audio and Video streaming .....	77
Appendix C Codes for File transfer.....	82

## List of Figures

Figure	Page
2.1 Fully Interconnected Network.....	11
2.2 Centralised Switch.....	12
2.3 Two-Level Hierarchy.....	12
2.4 Basic VoIP Communication System.....	16
2.5 OSI Layer and VoIP- Used Protocols.....	22
2.6 OSI Model and TCP/IP Protocol Suite.....	23
2.7 IP/UDP/RTP Header Format.....	26
2.8 VOIP Signalling Procedure.....	27
2.9 Where Sockets Reside in the OSI Model.....	29
3.1 The 3-tier architecture.....	34
3.2 The Layered architecture.....	34
3.3 An Audio Call Process.....	35
3.4 Player States.....	37
3.5 A Simple Video Player Scenario.....	39
3.6 Sequence Diagram for File Transfer.....	39
3.7 Activity Diagram for File Transfer.....	41
3.8 Class Diagram for File Transfer Client.....	42
3.9 Class Diagram for File Transfer Server.....	42



3.10	State Transition Diagram.....	43
3.11	System Boundary Diagram.....	44
3.12	System Activity Diagram.....	46
4.1	Main Server Form.....	58
4.2	Login Form.....	58
4.3	Registration Form.....	59
4.4	Services Form.....	59
4.5	Making Voice Call.....	60
4.6	Making Video Call.....	61
4.7	Video Call Screens.....	61
4.8	Video Call Ended Form.....	62
4.9	File Transfer Form.....	62
4.10	File Aborted Dialog.....	63
4.11	Connection Error Form.....	65
4.12	User Exist Form.....	65
4.13	Wrong Login Credentials.....	66
4.14	User Account Creation Aborted Form .....	67

#### List of Tables

Table	Page
3.1 use case description .....	45
4.1 Call Charges for the various network operators in Ghana.....	70



## ABSTRACT

**Voice over Internet Protocol (VoIP)** is a standard for taking analogue audio signals, and turning them into digital data that can be transmitted over a network. VoIP has become an important factor in network communication. It has a lower operational cost, greater flexibility, and a variety of enhanced applications. VoIP is time – based. To ensure real-time transmission, Real-Time Transmission Protocol (RTP) is used on top of User Datagram Protocol (UDP). RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. Java Media Framework (JMF) is an Application Programming Interface (API) that uses RTP and therefore ideal for time-based media. Thus, the topic: *leveraging VOIP on LAN using Java Media Framework*. The research aims at designing a system that will allow users to communicate over a data network. That is to be able to send text, make voice and video call, and transfer file over a network. The system uses client/server architecture. The architecture is a 3-tier: the client, the main server and the database server. The system designed could be used on Android mobile phones and computers with Windows operating system. The outcome of the research will ~~allow~~ students, workers and teachers at Kumasi Technical Institute (KTI) to communicate at virtually no cost. The product will also put the network on KTI campus into full utilization.==



## CHAPTER ONE

### INTRODUCTION

This chapter is an introduction to the study. The chapter is discussed under the following sub-headings: Background of the study, Statement of the problem, the purpose of the study, Research questions, Hypothesis, Significance of the study, Description of the proposed system, Expected results, Limitation of the study, and Organization of the study.

#### 1.1 BACKGROUND OF THE STUDY

During one of my Information and Communication Technology (ICT) lessons at Kumasi Technical Institute (KTI), I was teaching on the topic, *uses of internet*. I tried to illustrate how the internet is used for communication. I visited [www.evaphone.com](http://www.evaphone.com) and tried to dial one of my students' phone numbers. Wow! It worked. I was able to communicate with my student over the internet. This is an application of Voice over Internet Protocol (VOIP) technology.

VOIP is a standard for converting analog audio signals into digital data that can be transmitted over the Internet (or simply a data network) rather than traditional Public Switched Telephone Network (PSTN). VoIP has become an important development in network communication. It is a rapidly growing Internet service. It has lower operational costs, greater flexibility, and a variety of enhanced applications. VoIP technology uses the Internet's packet-switching capabilities to provide phone service.

The internet is an Internet Protocol (IP)-based network. IP-based networks are connectionless, packets switched networks. Transmission by the connectionless



technology has no guarantee that the data is received by the destination. To avert this problem, VOIP communication should be connection oriented. In a connection-oriented communication, there is a guaranteed delivery of data: any data that is not received by the destination system is re-sent by the sending device.

Connection-oriented approach introduces a lot of delay in transmission. Since VoIP is time – based, using the connection –oriented approach is not the best. The known protocol that ensures connectionless-oriented transmission is User Datagram Protocol (UDP). Though UDP transmission is not secured, it is ideal protocol for transmitting time – based media.

To ensure real-time transmission, Real-Time Transmission Protocol (RTP) is used on top of UDP. RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP opens two ports for communication; one for the media stream and one for control.

Java has a lot of Application Programming Interface (APIs) that help programmers during coding. These APIs provide a lot of abstraction for the programmer. One of these APIs is the Java Media Framework (JMF). JMF uses RTP and it is therefore ideal for time-based media such as audio and video. JMF rich features make it prudent to integrate it in VOIP applications. Thus, the topic: *leveraging VOIP on LAN using Java Media Framework*.



## 1.2 STATEMENT OF THE PROBLEM

KTI has four main departments, namely Electrical Department, Mechanical Department, Building Department and Related Subjects Department. Staff and students within these departments communicate to each other and to the administration.

Some of the communication methods used at KTI campus includes electronic mailing, chatting, text messaging and voice calls using mobile phones. These methods are comparably costly and many a time unavailable to the users. When cellular phone operators' network is down, users cannot make calls. This at times affects the activities of students, teachers and workers.

When staff wants to share files, some of the options available to them are to use a Universal Serial Bus (USB) drive or external hard disk drive. The problem with these options is that both parties need to be at the source or destination of the file. Although one can use e-mail attachment technology to transfer files, the size of the file to be transmitted is always limited to say 25 MB.

When a staff wants to call other staff at different department, the staff needs to use his or her mobile phone which he or she will be charged by his or her network operator.

To avert this situation, the researcher decided to embark on a research that will allow students, teachers and workers to communicate (make voice call, make video call, and send text) over the institution's network without paying any fee. Users can also transfer files over the network.



### 1.3 THE PURPOSE OF THE STUDY

The purpose of the study is to design a system that will allow users to communicate over a data network.

The specific objectives are:

- I. To design a client software to be installed on participating mobile phones and computers.
- II. To make client software communicate (i.e. transfer file, send text, make video and voice call) with other clients over a network.
- III. To have a server application that will keep track of registered clients.
- IV. To make clients detect the availability of other clients.

### 1.4 RESEARCH QUESTIONS

In reference to the problem and objectives of the study, the following are the research questions:

- I. To what extent will the leveraging of the VOIP using JMF improve communication on campus?
- II. How will client software for VOIP using JMF be designed?
- III. How can a data network be used as a communication channel for mobile phones?
- IV. How can registered clients be tracked?
- V. How can one registered client detect the availability of the other?
- VI. Is it cost effective to leverage VOIP using JMF?
- VII. How could file, text, voice and video be sent over a network using mobile phones and computers?



## 1.5 HYPOTHESIS

- I. VoIP compresses data packets during transmission, and this causes more data to be handled over the carrier. As a result, more calls can be handled on one access line.
- II. The client software could be designed using Android Standard Development Kit (SDK) and Eclipse Integrated Development Environment (IDE).
- III. When a client is registered, the client ID is stored in the database server. This will help to track any client.
- IV. The server will broadcast all active clients. This will enable each client to automatically detect other active clients.
- V. Studies have shown that, compared to using a packed switched Telephone Network (PSTN) line, using VoIP can potentially make you save up to 40 % on local calls, and up to 90 % on international calls (<http://voip.about.com/od/voipbasics/a/ReasonsForVoIP.htm> 20/05/11).

## 1.6 SIGNIFICANCE OF THE STUDY

The study will be of great significance in these contexts:

- I. It will serve as a springboard for other researchers to research into leveraging VOIP using JMF.
- II. It will improve communication on KTI campus.
- III. It will reduce cost of making calls to friends on KTI campus.
- IV. It will put the network of KTI into full utilization.
- V. It will improve transfer of files from one location to the other.



## 1.7 DESCRIPTION OF THE PROPOSED SYSTEM

This thesis is basically aimed at creating an efficient and reliable communication system that make use of a Local Area Network. The system will allow clients to make calls, send text or transfer files from one location to the other using a data network. The system will be designed using the client/server architecture. For a client to send a text, the client will send the text to the server, with the receiving client ID and port number. The server upon receiving the message, will forward it to the receiving client, if available. Clients will be registered on the server. This will enable the server to broadcast all active clients. In view of the above, clients software will be installed on the communicating devices and server software on a dedicated computer. To make a call, the calling client needs to supply the called client ID. The call gets to the server and it is forwarded to the called client.

Android version of the application is also created to install on android mobile phones. Clients using the android version of the application can also connect to the server and communicate to the desktop clients. Users using the Android version can have all the features of the application except the video call which will not be available due to limited resources of the mobile phones such as low processing speed and storage.

## 1.8 EXPECTED RESULTS

- I. The outcome of the research will allow students, teachers and workers and at KTI to communicate at virtually no cost.
- II. The product will also put the network on campus into full utilization.
- III. The product will allow students and lecturers to transfer files with ease.



IV. The researcher expects that students, workers and lecturers will use the product as a communication tool to reduce call cost.

## 1.9 LIMITATION OF THE STUDY

Due to limited resources and time constraint, the study does not include voice and video conferencing. That is, the application is a one-on-one. Clients can only communicate to one client at a time.

The application for mobile devices was developed using the Android platform. The application will therefore run properly on devices with Android operating System.

The study was also restricted to KTI campus. It is likely that many other interesting findings could have been made if the above limitations were not there.

## 1.10 ORGANIZATION OF THE STUDY

This thesis is organized into five (5) chapters. The chapter one deals with the background of the study, statement of the problem, purpose of the study, research questions, hypothesis, significance of the study, expected results, Description of the proposed system, Limitation of the study, and organization of the study.

The chapter two is concerned with the review of the related literature. The chapter three describes the methodology used in the study. This includes the research design, Goals, User Requirement and User specification.

The chapter four is concerned with the implementation and testing of result. Finally, chapter Five deals with the summary, conclusion, recommendations and suggestions for further research.



## CHAPTER TWO

### LITERATURE REVIEW

This chapter reviews the literature relating to the problem: leveraging VOIP on LAN using JMF. The chapter discusses Existing systems, Public Switched Telephone Network (PSTN), VOIP, Protocols in VOIP, Socket programming and Java Media Framework API.

#### 2.1 ANALYSIS OF EXISTING SYSTEM

There are a lot of systems that enable one to make call, send text and transfer files over a network. Members of staff at KTI have been using various systems for communication. Such systems include Skype, Yahoo! messenger, Google Talk, Facebook, Twitter, and Inter-com system. But these systems have their own drawbacks.

Below are some of these systems, their features and major drawbacks.

##### 2.1.1 Skype

Skype is a peer-to-peer VoIP client developed in 2003 by the organization that created Kazaa. Skype claims that it can work almost seamlessly across Network Address Translations (NATs) and firewalls and has better voice quality than other VoIP clients. It encrypts calls end-to-end, and stores user information in a decentralized fashion. Skype also supports instant messaging and conferencing (Salman, 2004).

Though Skype has a lot of features such as text chat, voice and video call, calling landlines and sending files, the major problem with Skype is that it is purely internet-based. That is, if you are not connected to the internet you cannot use the application.



Even when connected to the internet, call quality from Skype may suffer if you have a slower-than-average Internet connection.

Users of Skype are provided with only the client application while the developer controls the server application. This architecture makes it unsuitable for organisations that want to communicate internally.

### 2.1.2 Yahoo! Messenger

Yahoo! Messenger is an all-in-one communication tool, and anyone, anywhere in the world, with a computer and Internet connection can use it. With Yahoo! Messenger, you can:

- i. Instantly message friends in real time.
- ii. Call and talk with your friends. You can call a friend's computer, mobile phone, and even the old-fashioned home phone.
- iii. Share your favorite photos.

Although Yahoo Messenger provides many cool features such as SMS messages and ~~video chat~~, there are severe limitations that will make you regret your choice if you go with Yahoo Messenger.

- i. Need a Yahoo Account

The first requirement you will notice when you sign up is that you need to sign up for a Yahoo account. That means a new username and a new password to remember. While this is perfectly reasonable, not everyone has a Yahoo account.



ii. Need Your Friends to Have a Yahoo Account

How many of your friends are using Yahoo Messenger themselves? Hopefully at least one or you will mostly be chatting with yourself.

iii. SMS Limit

Although Yahoo advertises that they let you communicate with your friends' cell phones via SMS, there appears to be a limitation on how many messages you can send. After you have sent 3 messages to a particular cell phone in a single session, you will begin receiving error messages.

iv. No Customer Support

If you are having trouble with the Yahoo Messenger service, their support team consists of an internet bot that you can chat with. If you have ever chatted with a bot, you probably know that they are almost never helpful. The only function that these bots seem to serve is to ask you to re-word your question because it cannot understand you.

([http://help.yahoo.com/tutorials/ms8/mess/im\\_setup1.html](http://help.yahoo.com/tutorials/ms8/mess/im_setup1.html) 9/7/12 8:40 pm)

### 2.1.3 Google Talk

Google Talk is an instant messaging service that provides both text and voice communication. Because the Google Talk servers communicate with clients using an open protocol, Extensible Messaging and Presence Protocol (XMPP), the service can also be accessed using any other client that supports XMPP ([http://en.wikipedia.org/wiki/Google\\_Talk](http://en.wikipedia.org/wiki/Google_Talk) Accessed 9/7/12 8:43 pm).

However, Google Talk requires a Gmail username and password before use. And it is also internet – based only.



All the existing systems are internet – based. This means they cannot be used without internet connectivity. It was based on this drawback that the researcher decided to embark on the research “*Leveraging VoIP on LAN using JMF*”.

## 2.2 PUBLIC SWITCHED TELEPHONE NETWORK (PSTN)

Traditional telephony was developed and constructed in 1878. It evolved in three main steps. Firstly, it existed in a form of a first *generic telephone network* (Kevin, 2005). At first the telephone network was fully interconnected, that is “*If a telephone owner wanted to talk to a number of other telephone owners, separate wires had to be strung to all those houses* (Tanenbaum, 2010)” (Knútur, 2011). Figure 2.1 (a) and (b) shows an interconnected network, where the dots represent telephone owners.

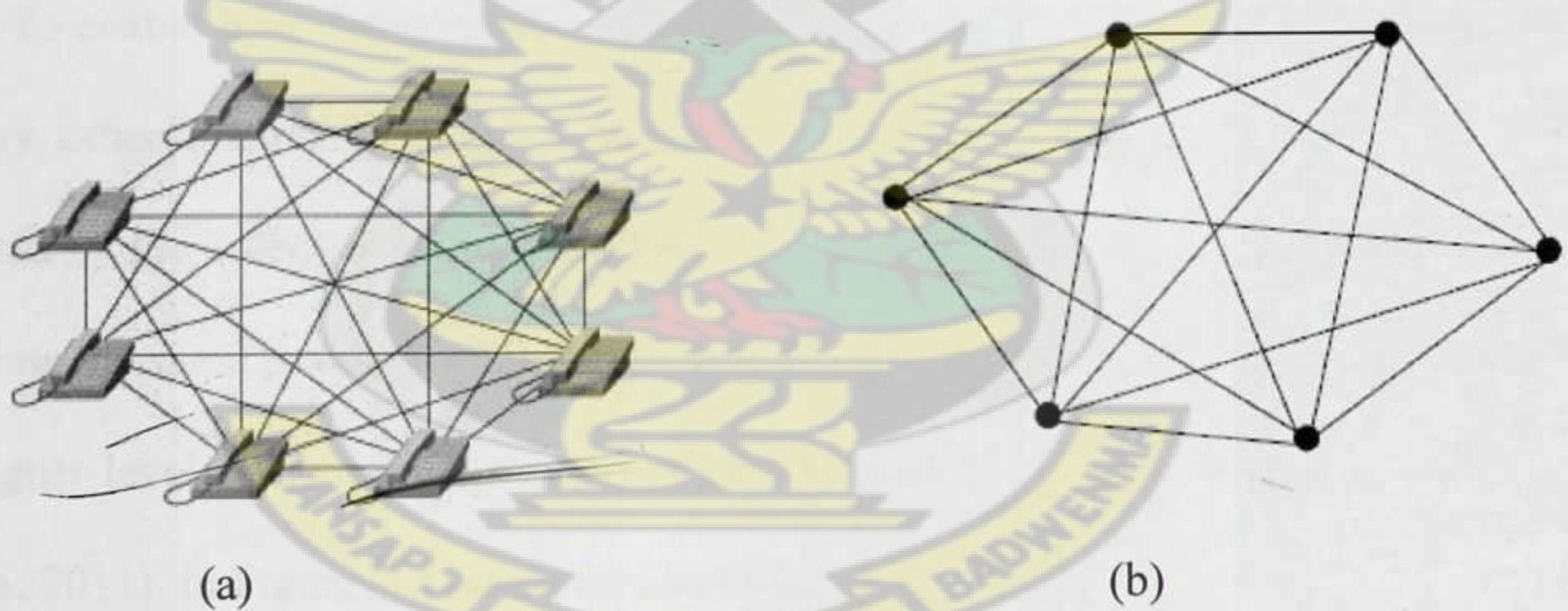


Figure 2.1 Fully Interconnected Network

It soon became obvious that connecting every telephone to every other telephone was not practical. Alexander Graham Bell, who is considered as the inventor of the telephone, acknowledged this problem early and formed the first switching office in 1878 where each phone was connected to a centralised switching office and human



presence was required to switch and set up telephone calls (Knútur, 2011), as shown in Figure 2.2.

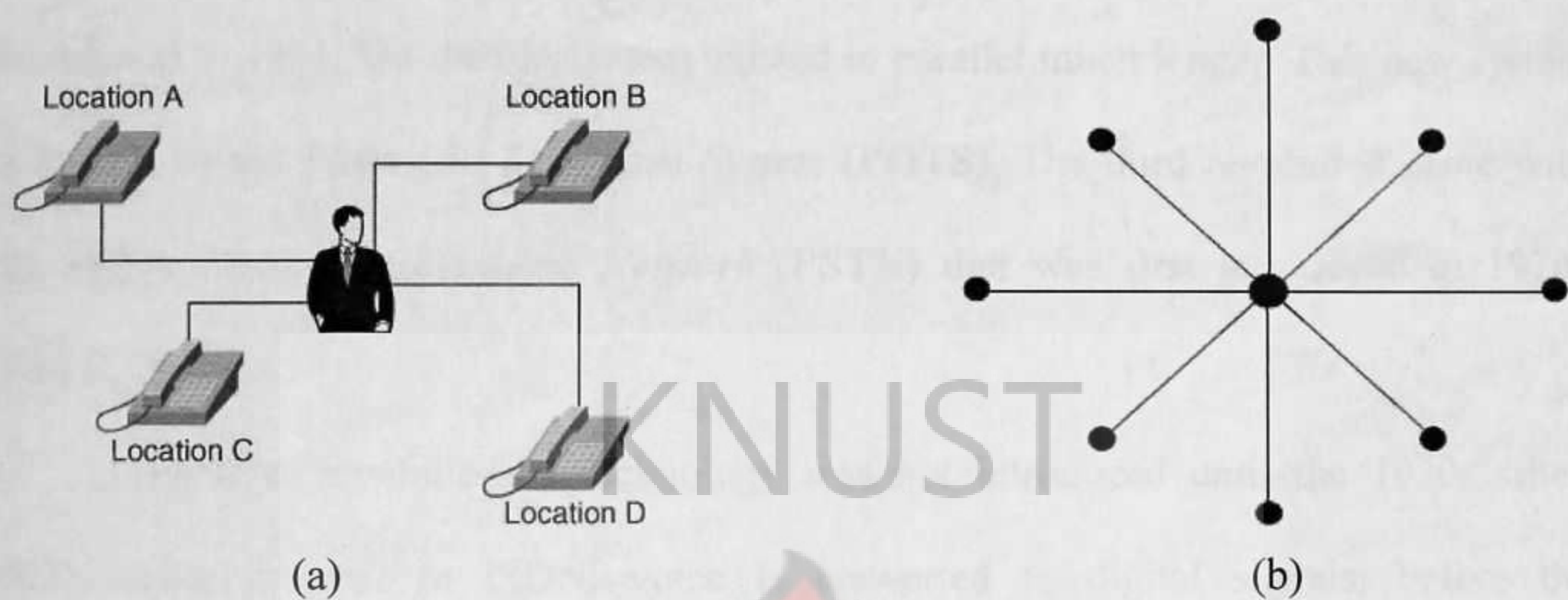


Figure 2.2 Centralized Switch

To enable long distance phone calls, every switching station had to be connected to every other switching station. Therefore the original problem returned but was countered by the invention of second-level switching offices. One can imagine that after some time the same problem would occur with connecting second-level stations and so it did. Higher level stations were needed and the hierarchy eventually grew to five levels (Knútur, 2011). In Figure 2.3 two-level hierarchy setup is depicted.

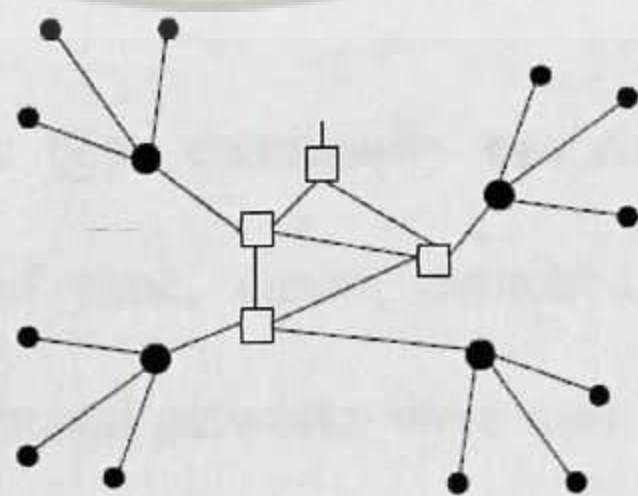


Figure 2.3 Two-Level Hierarchy



The biggest disadvantage of the first generic telephone network was that it required human intervention in order to switch and set up the telephone calls (Pawel, 2003). Indeed a new technology, which provided automated telephone switching, was introduced in 1891, but the old system existed in parallel much longer. This new system is known as the *Plain Old Telephone System (POTS)*. The third revolution came with the *Public Switched Telephone Network (PSTN)* that was first introduced in 1970s (Pawel, 2003).

The next revolutionary technology was not introduced until the 1970s when ISDN came around. In ISDN, voice is converted to digital signals; before the introduction of ISDN, voice has been travelling as analogue (Knútur, 2011).

Public Switched Telephone Networks (PSTN), uses circuit-switching. In *Circuit-Switching*, resources such as telephone lines and hubs are reserved along the entire communication channel for the duration of the call. In the PSTN, voice did not travel as an analogue signal, but as a digital one. Thanks to this transformation, new advanced services for example FAX and other data-based transmissions could be offered (Pawel, 2003).

## 2.3 OVERVIEW OF VOIP

Voice communication has been continually evolving since Alexander Bell's discovery. For a long period of time, circuit switched networks dominated the transmission of voice. Circuit switched networks were also used as a medium for data transmission. The picture today is totally different, with packet switched networks supporting both data and voice communications.



An emerging technology that uses packet switched networks for voice transmission is Voice over Internet Protocol (VoIP) telephony. VoIP is the family of technologies that allows IP networks to be used for voice applications, such as telephony, voice instant messaging, and teleconferencing. VoIP defines a way to carry voice calls over an IP network, including the digitization and packetization of the voice streams (Tiantioukas, 2007).

The advantages that make this technology preferable to traditional telephony are low cost, use of the existing infrastructure, and the ability to add new applications without additional cost.

## 2.4 DIFFERENT FORMS OF VOIP

VoIP is basically a clever "reinvention of the wheel." VoIP is a revolutionary technology that has the potential to completely rework the world's phone systems (<http://communication.howstuffworks.com/ip-telephony.htm>. Accessed 5/10/11 9:00 am).

The interesting thing about VoIP is that there is not just one way to place a call.

~~There~~ are three different "flavors" of VoIP service in common use today:

- i. Analog Telephone Adaptor (ATA): The simplest and most common way is through the use of a device called an ATA. The ATA allows you to connect a standard phone to your computer or your Internet connection for use with VoIP. The ATA is an analog-to-digital converter. It takes the analog signal from your traditional phone and converts it into digital data for transmission over the Internet. Providers like Vonage and AT&T Call Vantage are bundling ATAs free



with their service. You simply crack the ATA out of the box, plug the cable from your phone that would normally go in the wall socket into the ATA, and you're ready to make VoIP calls. Some ATAs may ship with additional software that is loaded onto the host computer to configure it; but in any case, it is a very straightforward setup

- ii. IP Phones: These specialized phones look just like normal phones with a handset, cradle and buttons. But instead of having the standard RJ-11 phone connectors, IP phones have an RJ-45 Ethernet connector. IP phones connect directly to your router and have all the hardware and software necessary right onboard to handle the IP call. Mobile phones with Wi-Fi allow subscribing callers to make VoIP calls from any Wi-Fi hotspot.
- iii. Computer-to-computer: This is certainly the easiest way to use VoIP. You do not even have to pay for long-distance calls. There are several companies offering free or very low-cost software that you can use for this type of VoIP. All you need is the software, a microphone, speakers, a sound card and an Internet connection; preferably a fast one like you would get through a cable or Digital Subscriber Line (DSL) modem. Except for your normal monthly Internet Service Provider (ISP) fee, there is usually no charge for computer-to-computer calls, no matter the distance (<http://communication.howstuffworks.com/ip-telephony.htm>. Accessed 5/10/11 9:00 am.).

## 2.5 HOW VOIP WORKS

Many years ago it was discovered that sending a signal to a remote destination could have been done also in a digital fashion: before sending it, the signal needs to be



digitalized with an analog to digital converter (ADC), transmit it, and at the end transform it again in analog format with digital to analog converter (DAC) to use it.

VoIP works like that, digitizing voice in data packets, sending them and reconverting them in voice at destination. Digital format can be better controlled: it can be compressed, routed, converted to a new better format, and so on; also digital signal is more noise tolerant than the analog one.

TCP/IP networks are made of IP packets containing a header (to control communication) and a payload to transport data: VoIP use it to go across the network to destination.

*Voice (source) -- ADC -- -- Network -- -- DAC -- -- Voice (destination)*

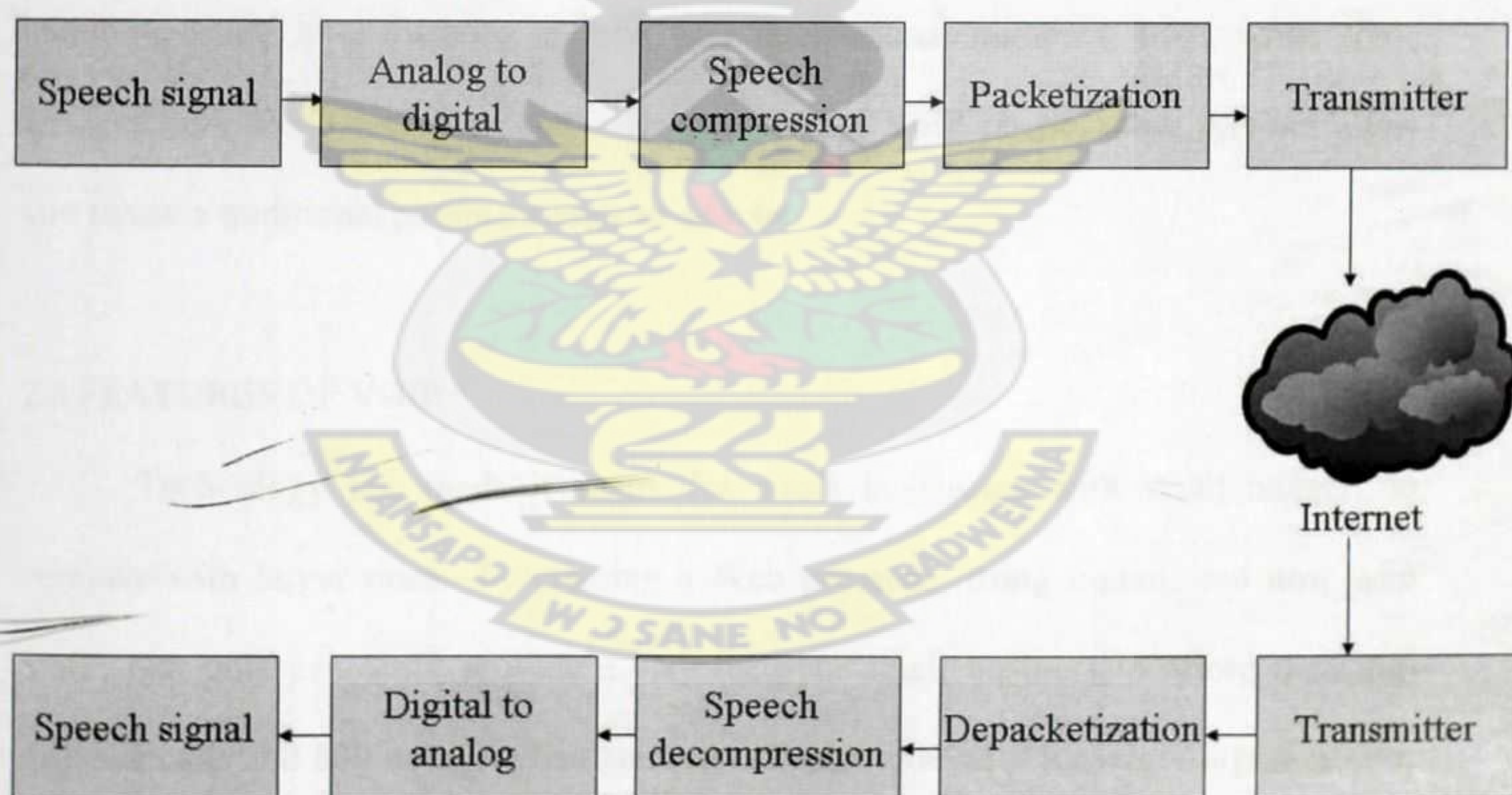


Figure 2.4 Basic VoIP Communication System

As shown in Figure 2.4, the voice of one user is digitized, compressed and then packetized before being sent through a wired or wireless communication channel to the other user. At the other end, the opposite procedure is followed: the received packet is



depacketized, decompressed, converted to analog form and then played back to the user. In order for the conversation to be natural, the same procedure must be followed in both directions so a full duplex communication is established. The Internet Voice can allow you to make a call directly from a computer, a special VoIP phone, or a traditional phone using an adapter. If you make a call using a phone with an adapter, you will be able to dial just as you always have, and the service provider may also provide a dial tone. If your service assigns you a regular phone number, then a person can call you from his or her regular phone without using special equipment.

Some services using VoIP may only allow you to call other people using the same service, but others may allow you to call anyone who has a telephone number - including local, long distance, mobile, and international numbers. Also, while some services only work over your computer or a special VoIP phone, other services allow you to use a traditional phone through an adapter.

## 2.6 FEATURES OF VOIP

Technology has made it easier for small businesses with small budgets to compete with larger ones—by creating a Web presence, using e-mail, and now with VoIP. Not only can VoIP provide a way for your small business to afford the long-distance calls and 800 numbers that were once reasonable only for larger organizations, but now you can use the advanced features of VoIP phones and Internet Protocol (IP) Private branch exchange (PBX) units to give callers the kind of experience once associated only with large companies.

VoIP services offer many features that you don't routinely get with traditional phone service, including simultaneous ring, do not disturb, virtual phone numbers, and



more. For example, Outlook integration allows you to "click to dial" Outlook contacts and automatically creates Outlook journal entries for the calls you make and/or receive. As VoIP becomes more ubiquitous in the workplace, it's likely that the differences between telephony features available to small and large businesses will fade even more (<http://www.techrepublic.com/article/the-top-five-advanced-voip-features-your-business-needs/6163858> Accessed 20/01/12 11:12 am).

Perhaps the biggest draws to VoIP for the home users and businesses that are making the switch are price, flexibility, Valued Added Services, Presence, and Conferencing.

#### 2.6.1 Flexibility

With VoIP, you can make a call from anywhere you have broadband connectivity. Since the IP phones or ATAs broadcast their info over the Internet, they can be administered by the provider anywhere there is a connection. So business travelers can take their phones or ATAs with them on trips and always have access to their home phone. Another alternative is the softphone. A softphone is client software that loads the VoIP service onto your desktop or laptop. The Vonage softphone has an interface on your screen that looks like a traditional telephone. As long as you have a headset/microphone, you can place calls from your laptop anywhere in the broadband-connected world (<http://communication.howstuffworks.com/ip-telephony.htm>. Accessed 5/10/11 9:00 am.).

This feature allows employees to move around, either within the organization or outside it, and still receive calls as if they're sitting at their desks. Workers



telecommuting from home, executives in hotel rooms on the road, technicians out on a job site—they can all get their calls no matter where they are.

You can configure the system so that when a call comes in for an employee, the desk phone rings first, then the employee's cell phone, then his or her home phone, until the system finds the employee. The employee can use the Follow Me functionality to define the phone number of the location where he or she will be and have all calls routed there.

With this feature, employees have more flexibility, and customers are less likely to end up talking to voice mail instead of the person they're trying to call. Different vendors may use different terms for this feature including *Find Me Follow Me (FMFM)*, *call hunting* or *advanced forwarding* (<http://www.techrepublic.com/article/the-top-five-advanced-voip-features-your-business-needs/6163858> Accessed 20/01/12 11:12 am).

#### 2.6.2 Price

Most VoIP companies are offering minute-rate plans structured like cell phone bills for as little as \$30 per month. On the higher end, some offer unlimited plans for \$79. With the elimination of unregulated charges and the suite of free features that are included with these plans, it can be quite a savings (<http://communication.howstuffworks.com/ip-telephony.htm>. Accessed 5/10/11 9:00 am.).



### 2.6.3 Value Added Services

There are many value added services offered by the various VOIP service providers. For example, there is voice mail facility, 3-way conference, speed dialing, call forwarding, simultaneous ring, call waiting, caller ID, call return, caller ID block, anonymous call rejection, an interesting feature of do not disturb, and last number dial. All these services are mostly free of cost in their basic service plan (<http://www.selfseo.com/story-19378.php> Accessed 20/01/12 9:43 am).

### 2.6.4 Presence

This is an extension of the FMFM functionality. Rather than passively depending on users to set up locations where they expect to be, the Presence feature can actually track them down. For example, the system can detect that a user logged onto his or her e-mail account from a computer in the accounting department or checked voice mail messages from a phone at the reception area, and it can then extrapolate from that where the user is.

A presence system also allows you to create rules about how to handle calls based on the user's location. For example, you can configure it not to ring your cell phone when you're in the CEO's office.

### 2.6.5 Conferencing

A good VoIP system will support audio and video conferencing. High-fidelity wideband VoIP conference phones allow multiple persons at a conference table to participate in a VoIP conversation from different distances, speaking at a normal volume.



You can also tie conferencing functionality into other IP collaboration applications, so that during the conference call, participants can exchange files, synchronize calendars, share presentations, and even see one another's computer desktops. Such functionality is called *unified communications*.

Many companies are offering enterprise-level unified communications solutions. Microsoft's entry into the market is its Office Communications Server 2007 (OCS), which provides for SIP-based calling, presence-based VoIP call management, instant messaging, and audio, video, and Web-based conferencing.

## 2.7 PROTOCOLS

Communication between two people or two devices needs to follow some protocol. A **protocol** is a set of rules that governs communication. For example, in a face-to-face communication between two persons, there is a set of implicit rules in each culture that define how two persons should start the communication, how to continue the communication, and how to end the communication. Similarly, in a telephone conversation, there are a set of rules that is needed to follow. There is a rule on how to make connection (dialing the telephone number), how to respond to the call (picking up the receiver), how to greet, how to let the communication flow smoothly by listening when the other party is talking, and finally how to end the communication (hanging up).

In computer networks, communication occurs between entities in different systems. An entity is anything capable of sending or receiving information. However, two entities cannot simply send bit streams to each other and expect to be understood. For communication to occur, the entities must agree on a protocol. A protocol defines what is communicated, how it is communicated, and when it is communicated (Forouzan, 2010).



The main protocols used on the Internet today are Transmission Control Protocol (TCP) and Internet Protocol (IP) and associated protocols. Since VoIP uses the Internet as the medium, it also uses the TCP/IP protocol suite. Considering the whole Open Systems Interconnection (OSI) layer system, VoIP is essentially an application running on top of the transport layer. Starting from the bottom to the top of the encapsulation process, any physical and data link layer can be used. IP is the choice of protocol for the network layer. In the transport layer, TCP introduces a large amount of setup delay, which makes it inefficient for voice transmission. The use of User Datagram Protocol (UDP) on the other hand provides for a much faster data exchange but without reliable data delivery. The Real-time Transport Protocol (RTP) and Reliable User Datagram Protocol (RUDP) are the protocols of choice on top of UDP since they are especially created for use in VoIP and media on demand (Tiantioukas, 2007).

A layered architecture of a VoIP network is shown in Figure 2.5b alongside the seven-layer OSI model for comparison.

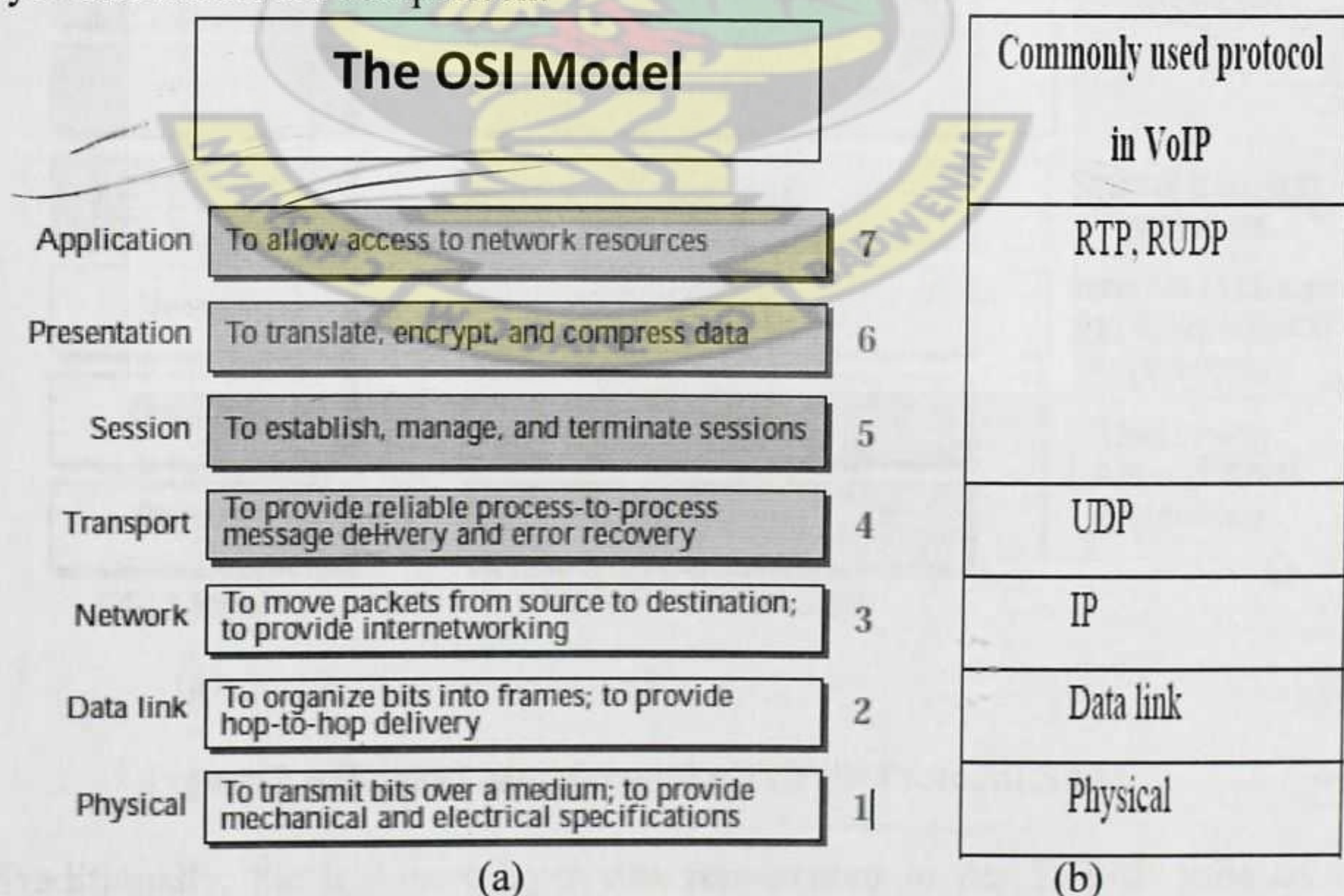


Figure 2.5 OSI Layer and VoIP-used protocols



The layered model that dominated data communication and networking literature before 1990 was the Open Systems Interconnection (OSI) model. Everyone believed that the OSI model would become the ultimate standard for data communications—but this did not happen. The TCP/IP protocol suite became the dominant commercial architecture because it was used and tested extensively in the Internet; the OSI model was never fully implemented (Forouzan, 2010).

The TCP/IP protocol suite was developed prior to the OSI model. Therefore, the layers in the TCP/IP protocol suite do not match exactly with those in the OSI model. Today, TCP/IP is thought of as a five-layer model with the layers named similarly to the ones in the OSI model. The application layer in the suite is usually considered to be the combination of three layers in the OSI model. Figure 2.6 shows both configurations.

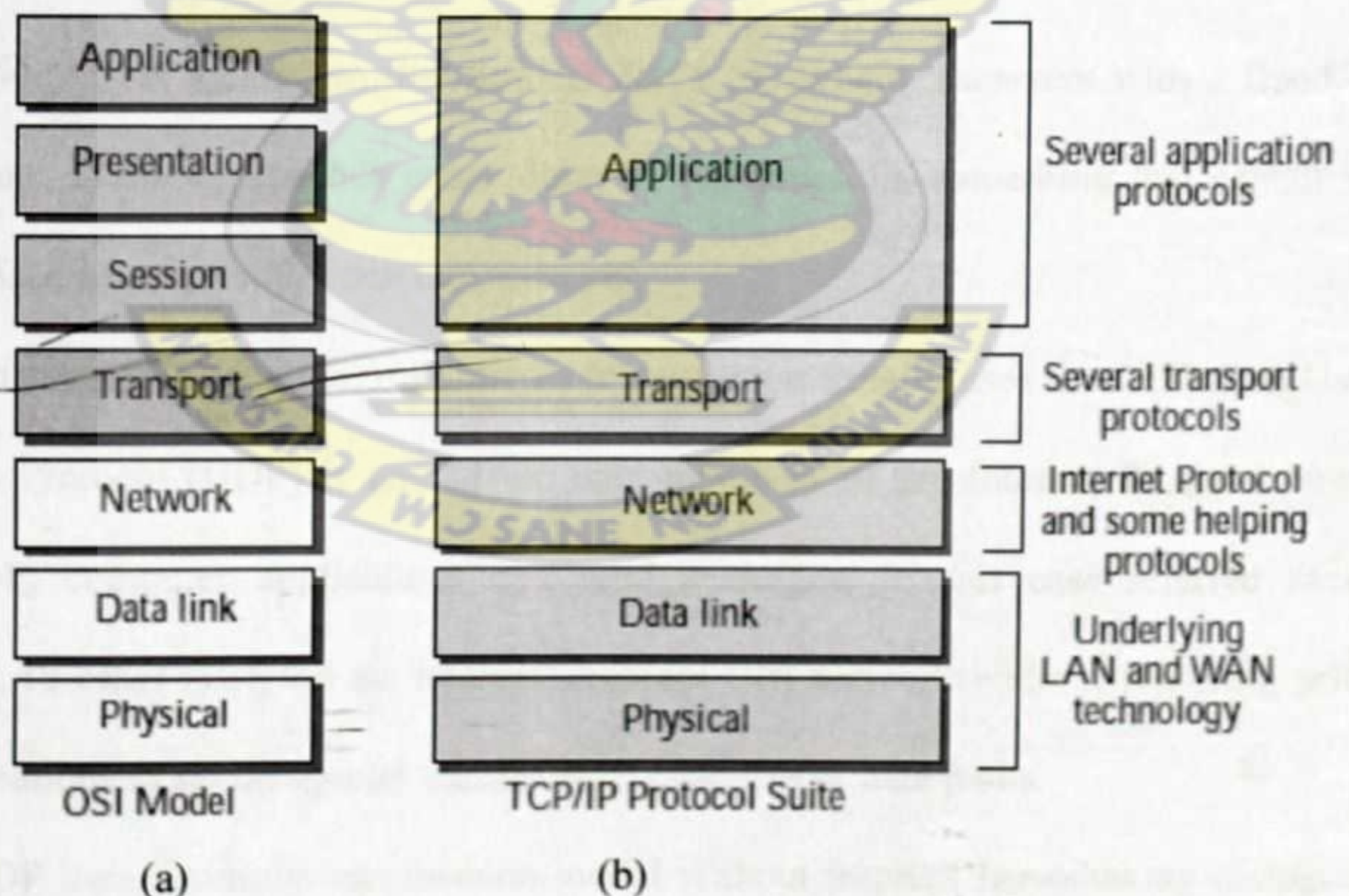


Figure 2.6 the OSI Model and the TCP/IP Protocol Suite

Traditionally, the transport layer was represented in the TCP/IP suite by two protocols:



User Datagram Protocol (UDP) and Transmission Control Protocol (TCP). A new protocol called Stream Control Transmission Protocol (SCTP) has been introduced in the last few years (Forouzan, 2010).

### 2.7.1 User Datagram Protocol (UDP)

Transmitting data using the TCP, though reliable, has its own limitations. Transmission Control Protocol (TCP) is a connection-oriented protocol that provides a reliable channel for communication. *Connection – oriented* means, the initiator (the client) first establishes a connection with the destination (the server), after which both of them can send and receive information. TCP implements the connection through an exchange of messages, called a *three-way handshake*, between initiator and destination. TCP achieves reliability by using receiver acknowledgments and retransmissions. TCP also provides flow control so that senders don't overwhelm receivers with a flood of information. These approaches cause delay in transmission, something that cannot be compromised in voice and video transmission.

To avert this situation, multimedia transmission usually uses the UDP. The User Datagram Protocol (UDP) is one of the core members of the Internet Protocol Suite. With UDP, computer applications can send messages, in this case referred to as *datagram*, to other hosts on an Internet Protocol (IP) network without requiring prior communications to set up special transmission channels or data paths.

UDP uses a simple transmission model without implicit handshaking dialogues for providing reliability, ordering, or data integrity. Thus, UDP provides an unreliable service and datagrams may arrive out of order, appear duplicated, or go missing without notice. UDP assumes that error checking and correction is either not necessary or



performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system ([http://en.wikipedia.org/wiki/User\\_Datagram\\_Protocol](http://en.wikipedia.org/wiki/User_Datagram_Protocol) 23/01/12 11:17 PM).

UDP is the main alternative to *TCP*. UDP is often used in videoconferencing applications or computer games specially tuned for real-time performance. To achieve higher performance, the protocol allows individual packets to be dropped (with no retries) and UDP packets to be received in a different order than they were sent as dictated by the application (<http://compnetworking.about.com/od/networkprotocolsip/g/udp-user-datagram-protocol.htm> Accessed 23/01/12 11:22 PM).

UDP network traffic is organized in the form of *datagrams*. A datagram comprises one message unit. The first eight (8) bytes of a datagram contain header information and the remaining bytes contain message data.

### 2.7.2 Real-Time Transport Protocol (RTP)

The Real-Time Transport Protocol is used on top of UDP to solve the problem of unreliable data transmission. RTP is the protocol often used to carry the audio data in a VoIP conversation; it is a standard developed by the Internet Engineering Task Force (IETF). It can also be used to carry video data and is designed specifically to handle this sort of real-time data. It attempts to guarantee that the data will be transmitted and received in a short period of time. Obviously latency in voice conversations can be a problem, so RTP avoids this latency as much as possible and concentrates on timely delivery of data (Gomillion et al, 2005). RTP also solves the problem of “out of order”



arrival of datagram (Perea, 2008). RTP utilizes the datagram service of UDP and provides two kinds of information. First, it provides a sequence number that helps the receiver reorder the packets. Second, it supplies a timestamp that helps the receiver deal with jitter estimation and synchronization (Tiantioukas, 2007). RTP provides the receiver with the tools to reproduce the content but does not provide any control functionality. This functionality is provided by Real Time Control Protocol (RTCP), which is a companion protocol of RTP and essential for its operation. RTCP provides additional information about the data exchange and the network performance. RTCP packets use a different port number than the RTP stream (Comer, 2006).

A drawback of the use of RTP is the additional overhead. The IP/UDP/RTP header presented in Figure 2.7 is 40-bytes long. The typical data payload carried by this packet is about one half of the header size (Tiantioukas, 2007).

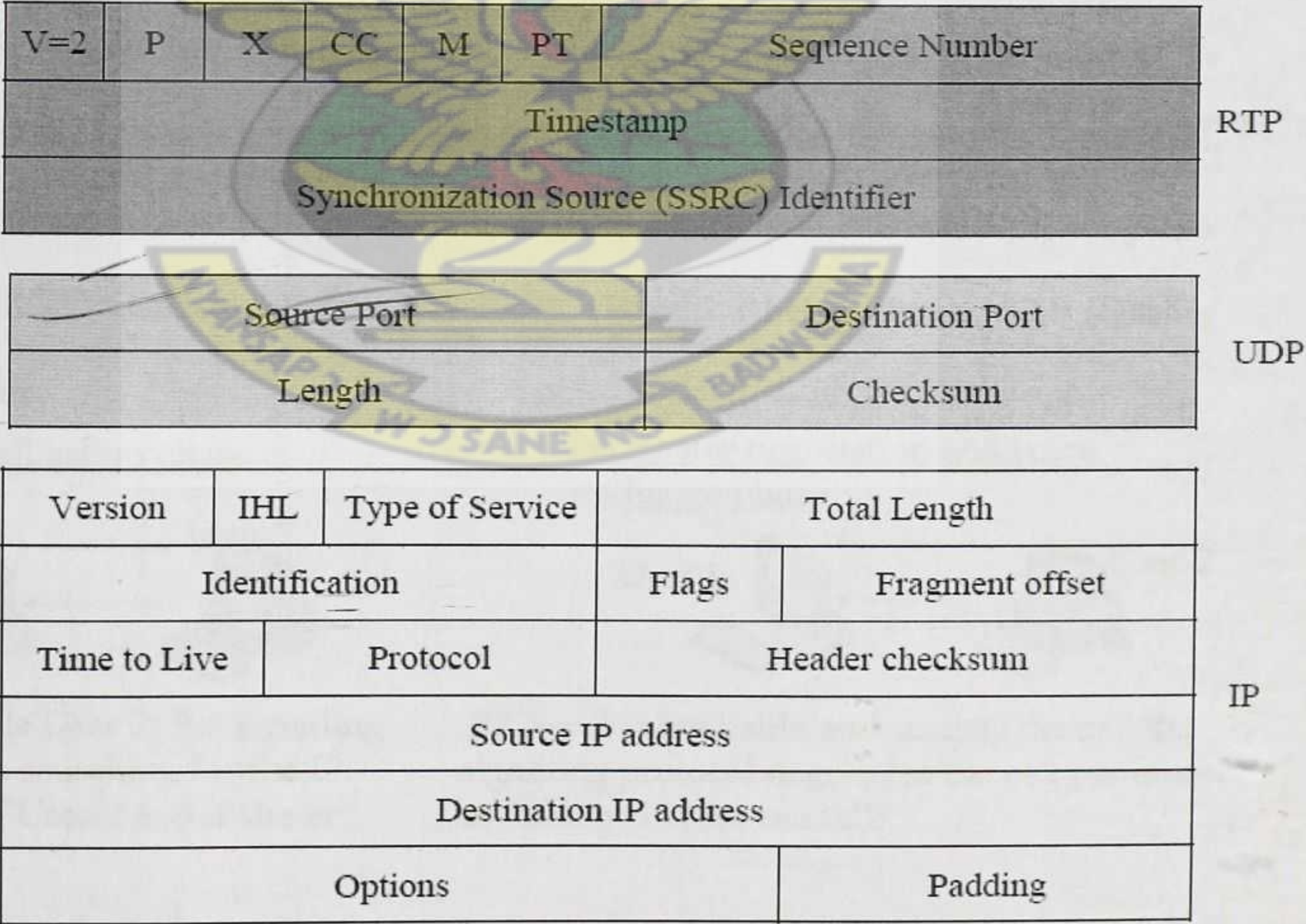


Figure 2.7 IP/UDP/RTP Header Format



## 2.8 SIGNALING IN VOIP

Before a communication takes place in conventional telephony, some necessary steps must be carried out. After the caller picks up the phone and dials the called party's number, a signaling protocol is activated in order to find if the called party is available. If the called party is available, it establishes a line of communication for the two parties to talk. The same procedure is followed in VoIP and can be seen in Figure 2.9. First, the signaling protocol looks for the IP address of the called person and if he or she is available and willing to participate a logical channel is established. The call is established and parameters like voice coding, session protocol and capability exchange are negotiated between the two users. After that, the two users can talk. The signaling protocol is still present and monitors the quality of the call and waits for the signal to terminate the call.

There are two signaling protocols widely used in the market. The older of the two is H.323, which is an International Telecommunication Union (ITU-T) standard, and the newer is Session Initiation Protocol (SIP), which is an Internet Engineering Task Force (IETF) standard (Tiantioukas, 2007). Figure 2.8 below shows VOIP signaling procedure.

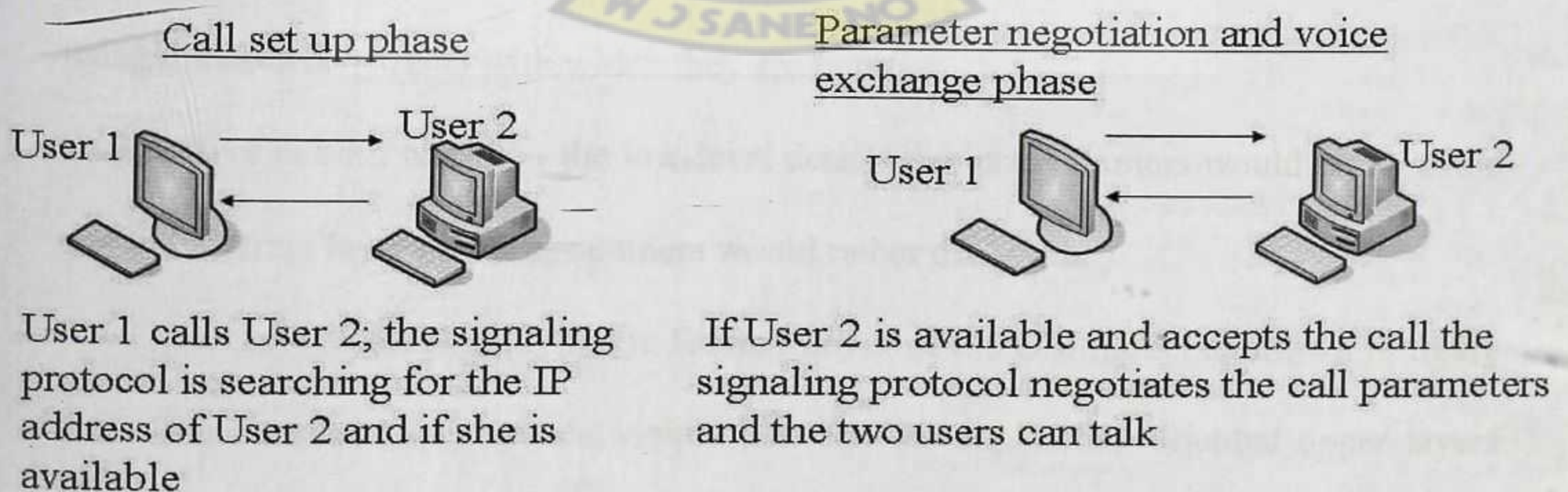


Figure 2.8 VoIP Signaling Procedure Including Call Setup and Voice Exchange Phase



## 2.9 SOCKET PROGRAMMING

Before you can talk to someone on the phone, you must supply a phone number to the telephone system. In a similar way, before a program can communicate with another program, it must tell the network something to identify the other program. In TCP/IP, it takes two pieces of information to identify a particular program: an *Internet address*, used by IP, and a *port number*, the additional address interpreted by the transport protocol (TCP or UDP) (Calvert, 2008). The IP address identifies a particular device on the network and the port number uniquely identifies a program running on that device. Programs communicate with each other on the network using a communication channel (wired or wireless).

The Java language was designed (from the start) for use over the Internet. It provides many useful abstractions for implementing programs that communicate via the application programming interface (API) known as *sockets*. Most programmers, whether they are coding in the Java language or not, do not want to know much about low-level details of how applications on different computers communicate with each other. Programmers want to deal with higher-level abstractions that are easier to understand. Java programmers want objects that they can interact with via an intuitive interface, using the Java constructs with which they are familiar.

Sockets live in both worlds -- the low-level details that programmers would rather avoid and the abstract layer that programmers would rather deal with.

Sockets reside roughly at the *Session Layer* of the OSI model as shown in figure 2.10. The Session Layer is sandwiched between the application-oriented upper layers and the real-time data communication lower layers. The Session Layer provides services for managing and controlling data flow between two computers. As part of this layer,



sockets provide an abstraction that hides the complexities of getting the bits and bytes on the wire for transmission. In other words, sockets allow us to transmit data by having our application indicate that it wants to send some bytes (ibm.com/developerWorks).

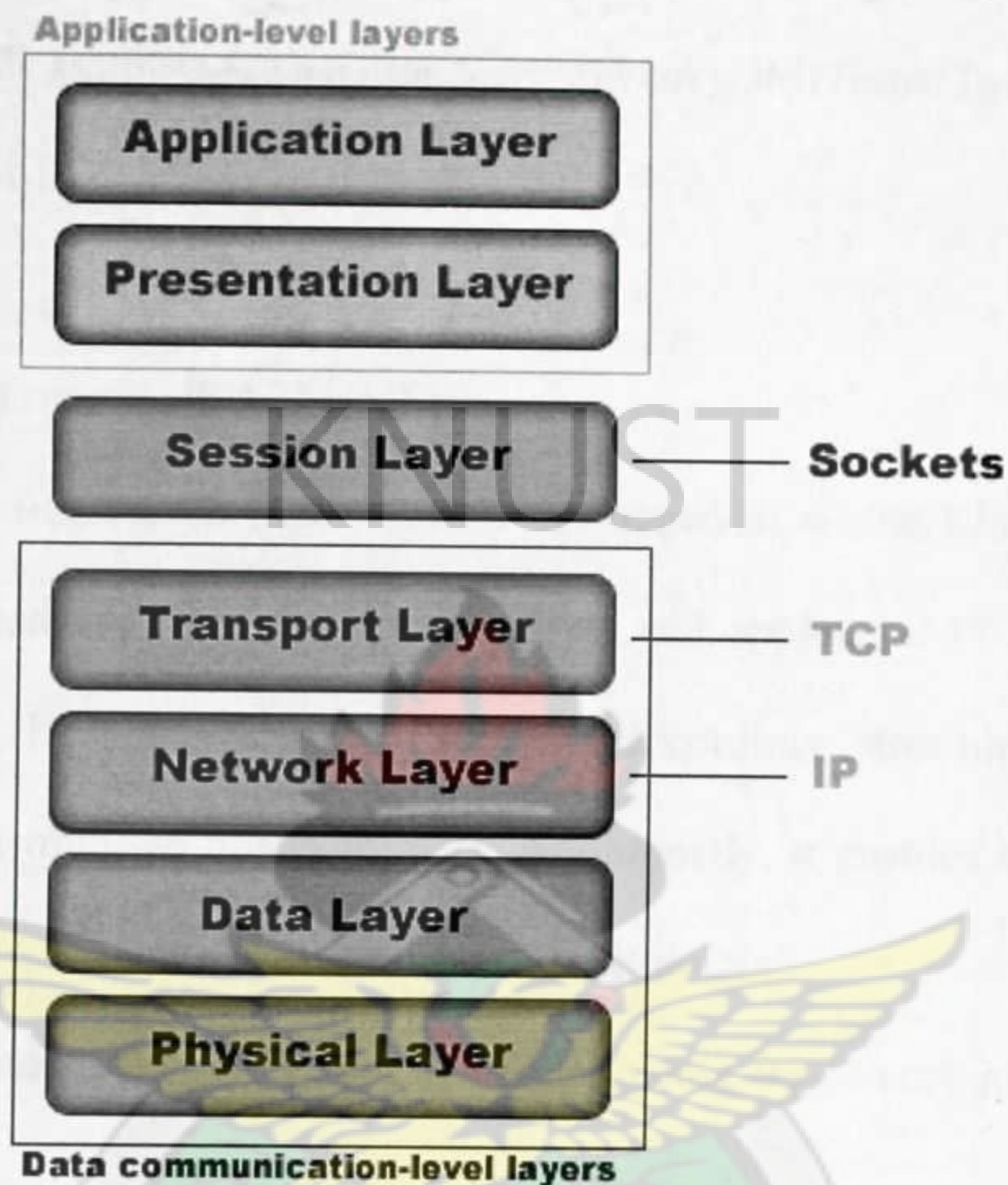


Figure 2.9 Where Sockets reside in the OSI Model

## 2.10 JAVA APPLICATION PROGRAMMING INTERFACE (API)

There are a lot of APIs available in Java that help programmers during coding. Java API is not just a set of classes and interfaces that comes with the Java Development Kit (JDK). Java API is actually a huge collection of library routines that perform basic programming tasks such as looping, and displaying GUI forms (<http://www.roseindia.net/java/javaapi/java-api.shtml> Accessed 28/01/12 10:53 PM).

JavaSoft is the business unit of Sun Microsystems that is responsible for Java technology. JavaSoft has developed a package of APIs called Java Media APIs. The Java Media APIs is a set of APIs that enables Java programmers to add multimedia to



their applications. The Java Media API's include the Java™ 2D API, Java™ 3D API, Java™ Advanced Imaging API, Java™ Image I/O , Java™ Media Framework (JMF) API, Java™ Image I/O , Java™ Shared Data Toolkit (JSDT) , Java™ Sound API, Java™ Speech API (<http://www.ee.iitm.ac.in/~tgvenky/JMFBook/Tutorial.pdf> 6/7/12 11:10 am.).

#### 2.10.1 Java Media Framework API (JMF)

Java media framework (JMF) is API developed to enable adding audio, video, and other time-based media to Java applications and applets. JMF extends the Java Platform Standard Edition (J2SE). It supports recording, streaming, playing, and converting between multiple media formats. Additionally, it enables developing cross-platform Java multimedia applications (<http://www.techopedia.com/definition/23996/java-media-framework-jmf> Accessed 5/7/2012 3:24 pm).

Sun Microsystems, Silicon Graphics, and Intel Corporation collaborated to release JMF 1.0 in 1997. However, the next release of JMF, that is 2.0, was developed by Sun and IBM and released in 1999. This version supported new features, including recording, streaming, and conversion among different media formats.

As already discussed, TCP dominates IP (Internet Protocol) traffic because of its reliability. This reliability is possible because TCP/IP contains numerous software layers that prevent packet loss and ensure that packets arrive in the order they were sent. TCP reliability has a devastating impact on multimedia streaming. This occurs because its numerous software layers steal critical processing power away from multimedia devices



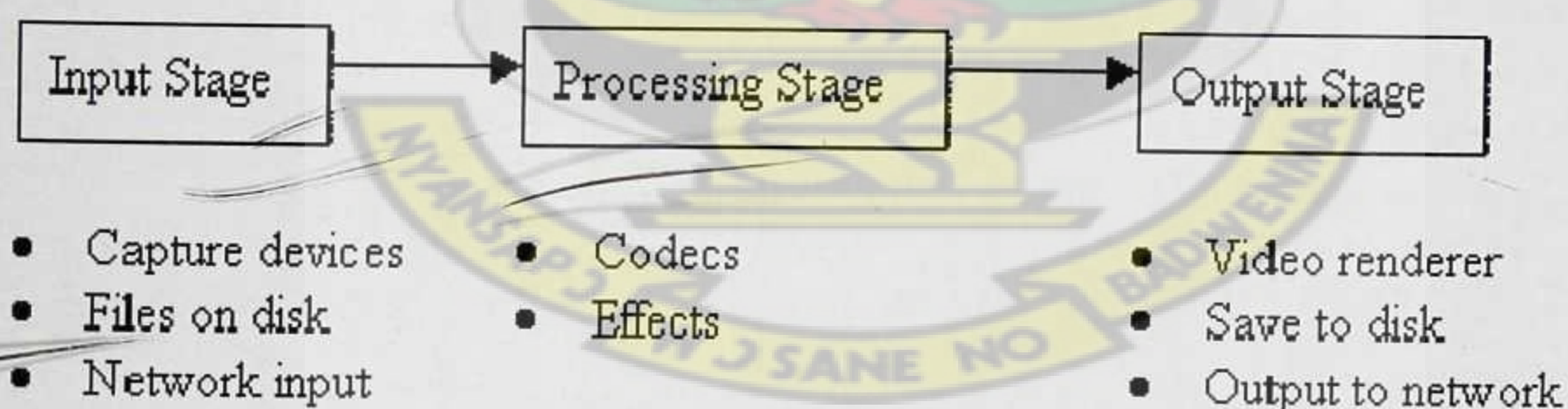
(<http://www2.sys-con.com/itsg/virtualcd/java/archives/0503/decarmo/index.html>

Accessed 5/7/2012 4:33 pm).

Although, UDP is a protocol that is ideal for multimedia transport, it does not guarantee that packets will arrive at their destination or that they will arrive in sequence. Should a packet be lost, multimedia CODECs (Compressors/ Decompressors), the software algorithms that process multimedia data, gracefully compensate to ensure smooth playback.

Most of the problems of UDP are solved by RTP. The RTP is built on top of UDP and it provides time stamps, synchronization, packet-loss detection and other multimedia features. Since JMF deals with real-time multimedia presentation and effects processing, it uses RTP. JMF handles time-based media, media which changes with respect to time. Examples of this are video from a television source, audio from a raw-audio format file and animations.

The JMF architecture is organized into three stages:



During the input stage, data is read from a source and passed in buffers to the processing stage. The input stage may consist of reading data from a local capture device (such as a webcam or TV capture card), a file on disk or stream from the network.

The processing stage consists of a number of codecs and effects designed to modify the data stream to one suitable for output. These codecs may perform functions



such as compressing or decompressing the audio to a different format, adding a watermark of some kind, cleaning up noise or applying an effect to the stream (such as echo to the audio).

Once the processing stage has applied its transformations to the stream, it passes the information to the output stage. The output stage may take the stream and pass it to a file on disk, output it to the local video display or transmit it over the network.

For example, a JMF system may read input from a TV capture card from the local system capturing input from a VCR in the input stage. It may then pass it to the processing stage to add a watermark in the corner of each frame and finally broadcast it over the local Intranet in the output stage (<http://www2.sys-con.com/itsg/virtualcd/java/archives/0503/decarmo/index.html> Accessed 5/7/2012 4:33 pm).





## CHAPTER THREE

### METHODOLOGY

This chapter discusses the approach and tools used in the design of the proposed system. The discussion dwells mainly on system design, user requirements and specification, and system goals.

#### 3.1 SYSTEM ANALYSIS AND DESIGN

The system uses client/server architecture. The architecture is a 3-tier: the client, the main server and the database server as shown in Figure 3.1. The clients consist of the various departments in KTI. These include the Building department, the Electrical department, the Mechanical department, the Related subjects department and the Administration. The main server contains the address-to-location mappings for each user in the system. The main server is located in the ICT sector of the Related subjects department. The database server contains information about users: First name, Surname, Other names, username, password, Gender, date of registration and date of modification. The database resides on the same computer that hosts the main server. This architecture enables the system to track all users and log any transaction that goes on in the system. Each user begins by establishing network links with their friends, now called network peers. Effectively, each user only accepts calls from people they trust.

Beside the 3-tier, the system can also be viewed as a three layer architecture. There are three major layers: Graphical User Interface Layer (GUI Layer), Control Layer and Data Layer as shown in Figure 3.2.



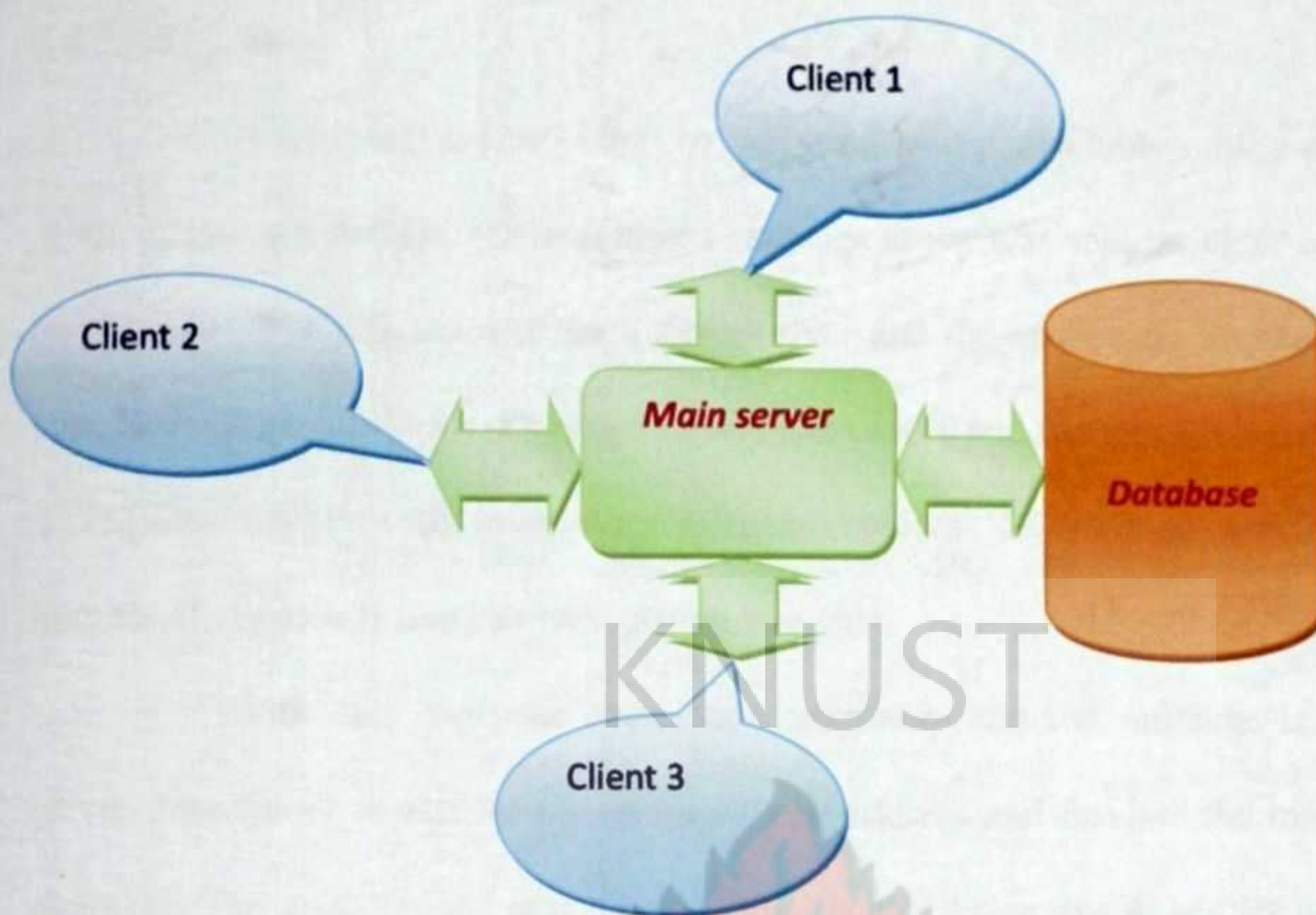


Figure 3.1 The 3-tier architecture

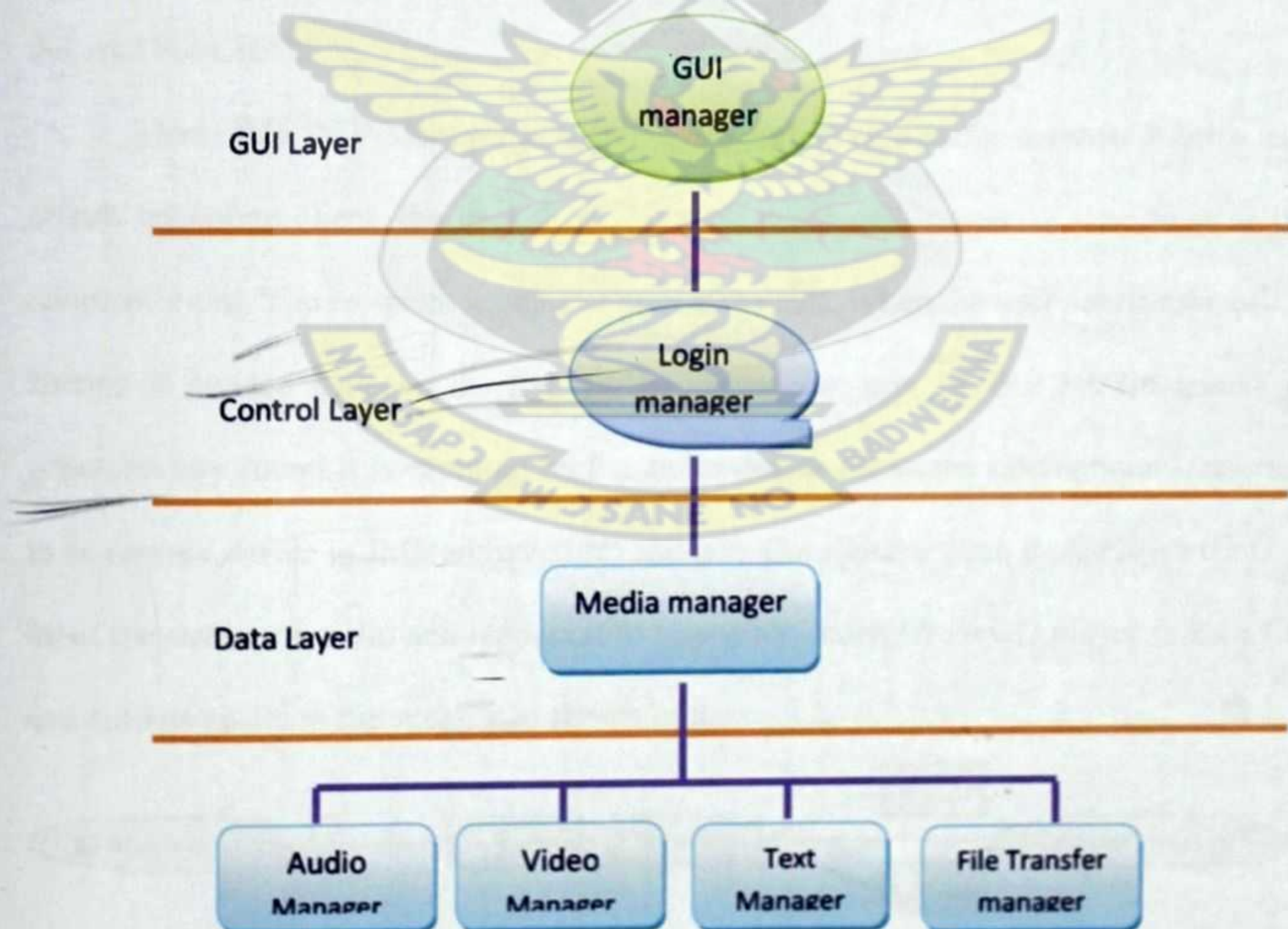


Figure 3.2 The layered architecture



### 3.2 TEXT CHAT

A user can use text chat by selecting a user available online to chat with. After selecting a partner, the user type a message at the text area provided and click the *send* button. The messages of both the receiver and the sender are displayed in a text area for both parties to see. During chatting, the sender and the receiver's user names are distinguished from each other with different colours. There is no restriction on the number of characters one can send during text chat.

With text chat, the recipient's username and the message is sent to the server. The server search for the recipient's IP address and forward the message to the recipient. The code for the text chat is found at the Appendix A and its actual source code is on the attached CD.

### 3.3 AUDIO CHAT

Here, JMF RTP Manager is used to send and receive audio streams. When a user selects an online client, the user then clicks on voice call button. A tone rings at the recipient's end. The recipient accepts or rejects the call. When the user accepts the call, a session is created between the two parties. When the user on one system speaks or generates any sound it is captured by the audio device (here, the microphone – referred to as *capture device* in JMF architecture) and gets *DataSource* from it. A *Player* takes as input the stream of audio and renders it to a speaker; much like a CD player reads a CD and outputs music to the speaker as shown in figure 3.3.

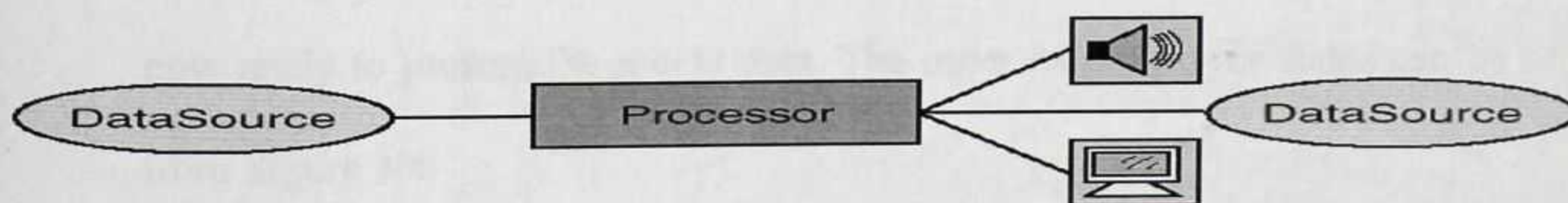


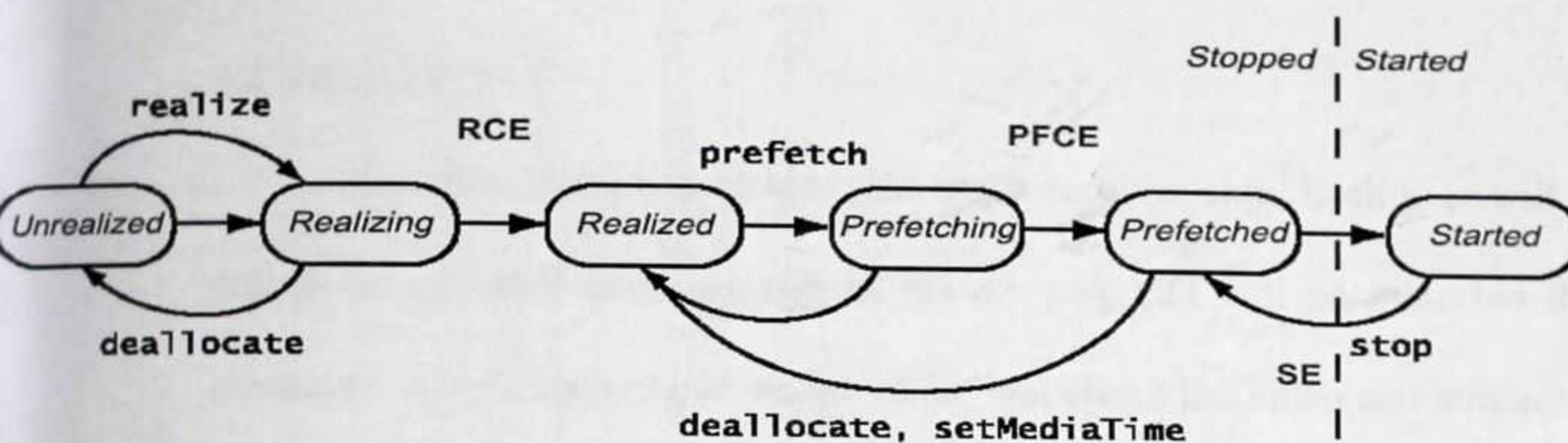
Figure 3.3 An Audio Call Process



A Player has states. A Player steps through each state until it reaches the final state. JMF defines six states in a Player:

- **Unrealized:** In this state, the Player object has been instantiated. Like a newborn baby who does not yet recognize its environment, a newly instantiated Player does not yet know anything about its media.
- **Realizing:** A Player moves from the unrealized state to the realizing state when you call the Player's `realize()` method. In the realizing state, the Player is in the process of determining its resource requirements.
- **Realized:** Transitioning from the realizing state, the Player comes into the realized state. In this state the Player knows what resources it needs and has information about the type of media it is to present. It can also provide visual components and controls, and its connections to other objects in the system are in place.
- **Prefetching:** When the `prefetch()` method is called, a Player moves from the realized state into the prefetching state. A prefetching Player is preparing to present its media. During this phase, the Player preloads its media data, obtains exclusive-use resources, and does whatever else is needed to play the media data.
- **Prefetched:** The state where the Player has finished prefetching media data – it is ready to start.
- **Started:** This state is entered when you call the `start ()` method. The Player is now ready to present the media data. The order of the player states can be seen from Figure 3.4.





Transition Events:

<b>RCE</b>	RealizeCompleteEvent
<b>PFCE</b>	PrefetchCompleteEvent
<b>SE</b>	StopEvent

Figure 3.4 Player States [Source: <http://www.javaworld.com/jw-04-2001/jw-0406-jmfl.html?page=2> Accessed 26/06/12 9:49 pm]

Before a Player plays a media, first an instance player is created through the media manager. Next the visual component and the control-panel components of the Player are displayed. Then the player object is prepared to start. The player can be started and stopped using `start ()` and `stop (-)` methods.

A *mediaProcessor* which is a variable processor is created which is responsible for converting our audio media from a given file format say mp3, to a format suitable for transmission over RTP protocol.

The *dataSink* variable is created which will be the output block. When a *dataSink* is created a *mediaLocator* is specified which is its actual destination. When our processed media is run through *dataSink*, it will be transmitted through *mediaTransmitter* instance to whatever location(s) specified in *mediaLocator*. In this way the desired audio stream is sent to desired destination and vice-versa (Radhika, 2012).

The codes for the audio chat could be found in Appendix B and its actual source code is on the attached CD.



### 3.4 VIDEO CHAT

Here too, there is a sender side and a receiver side. Each user will maintain an online list of other users present in the network and will be indicated through their username. A caller selects an online client and clicks the video call button. Same as the voice call, the recipient can accept or reject the call. When a recipient accepts a call, a session is created. This is followed by checking for supported capture devices and finding the media locator for them. A DataSource object for that media is then created. Next an instance processor is created through the media manager for transmitting. The processor goes through its phases –configuring, configured, unrealized, realizing, realized, prefetching, prefetched. This is followed by capturing the video tracks which will then be encoded, compressed and then passed onto RTP Manager. RTP session is then initialized by specifying the IP address and port number. At the receiver side a special thread will be listening to a special port for receiving purpose; if it detects new media streams it will create a player which goes through the stages mentioned in audio chat. The dataSink variable is created which is the output block. When a dataSink is created, a mediaLocator is specified which is its actual destination. When the processed media is run through dataSink, it will be transmitted through mediaTransmitter instance to whatever location(s) specified in the mediaLocator. In this way the desired video stream is sent to desired destination and vice-versa (Radhika, 2012).

This process is similar to a scenario where a video camera captures video, stores it on a magnetic tape or CD, plays the media with VCR Player and the VCR player outputs the media to a screen or speaker as shown in figure 3.5.



The codes for the video chat could be found at the Appendix B and the actual source code could be found on the attached CD.

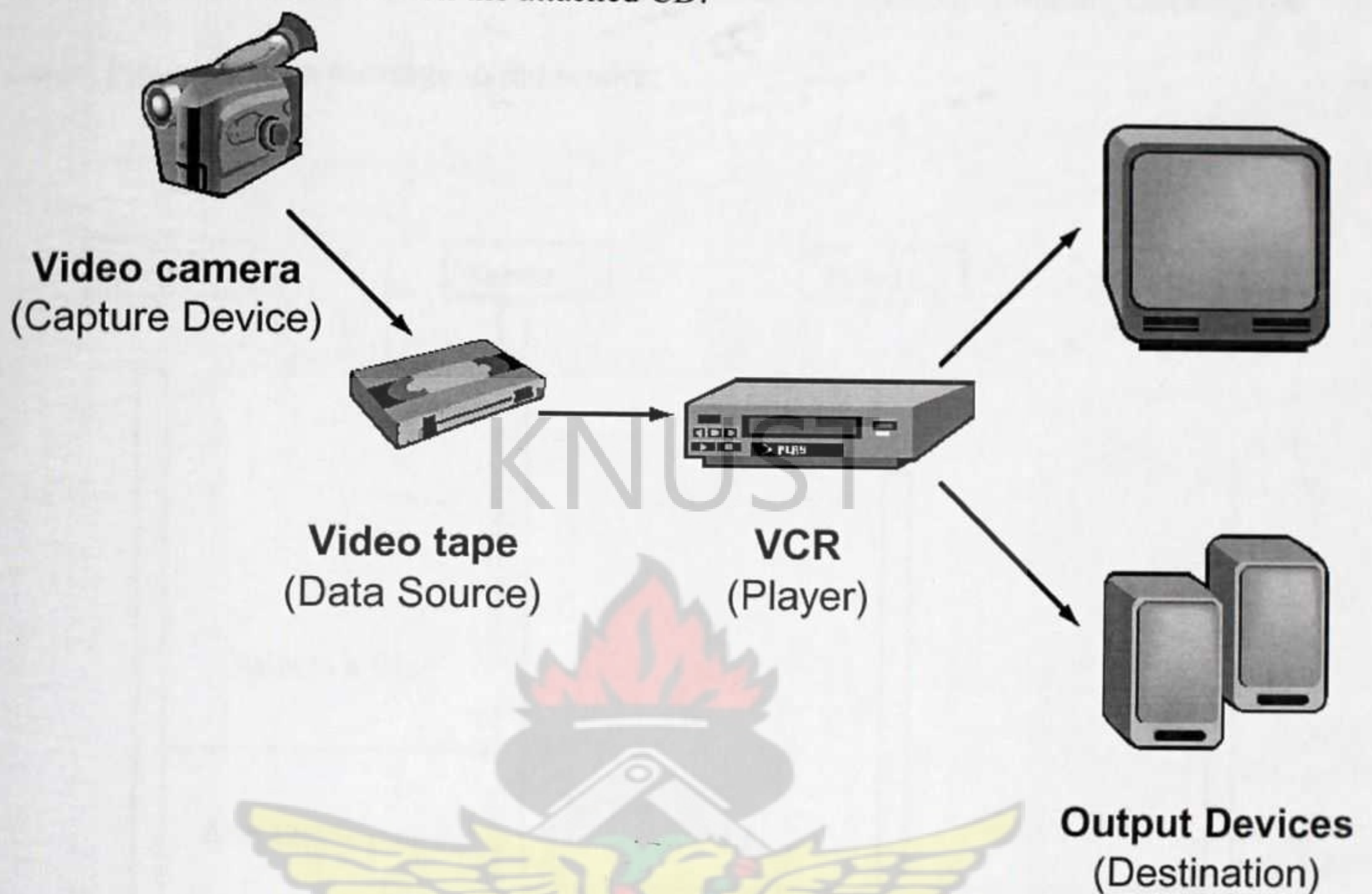


Figure 3.5 A simple Video Player Scenario

### 3.5 FILE TRANSFER

Here too there is a sender and receiver side. A sender will select a recipient from the online list. The sender clicks the *transfer file button* and this will display *choose file* dialog. The sender browses to select the file to be transferred and click *ok* to send the file name, file path and the recipient's username to the server. The Server forwards the file to the recipient. The recipient rename file if necessary and the recipient accept to save file.



The transmission of the file started when the recipient clicks the save button. The file will not be transferred when the recipient clicks the cancel button. Clicking the cancel button sends a message to the sender.

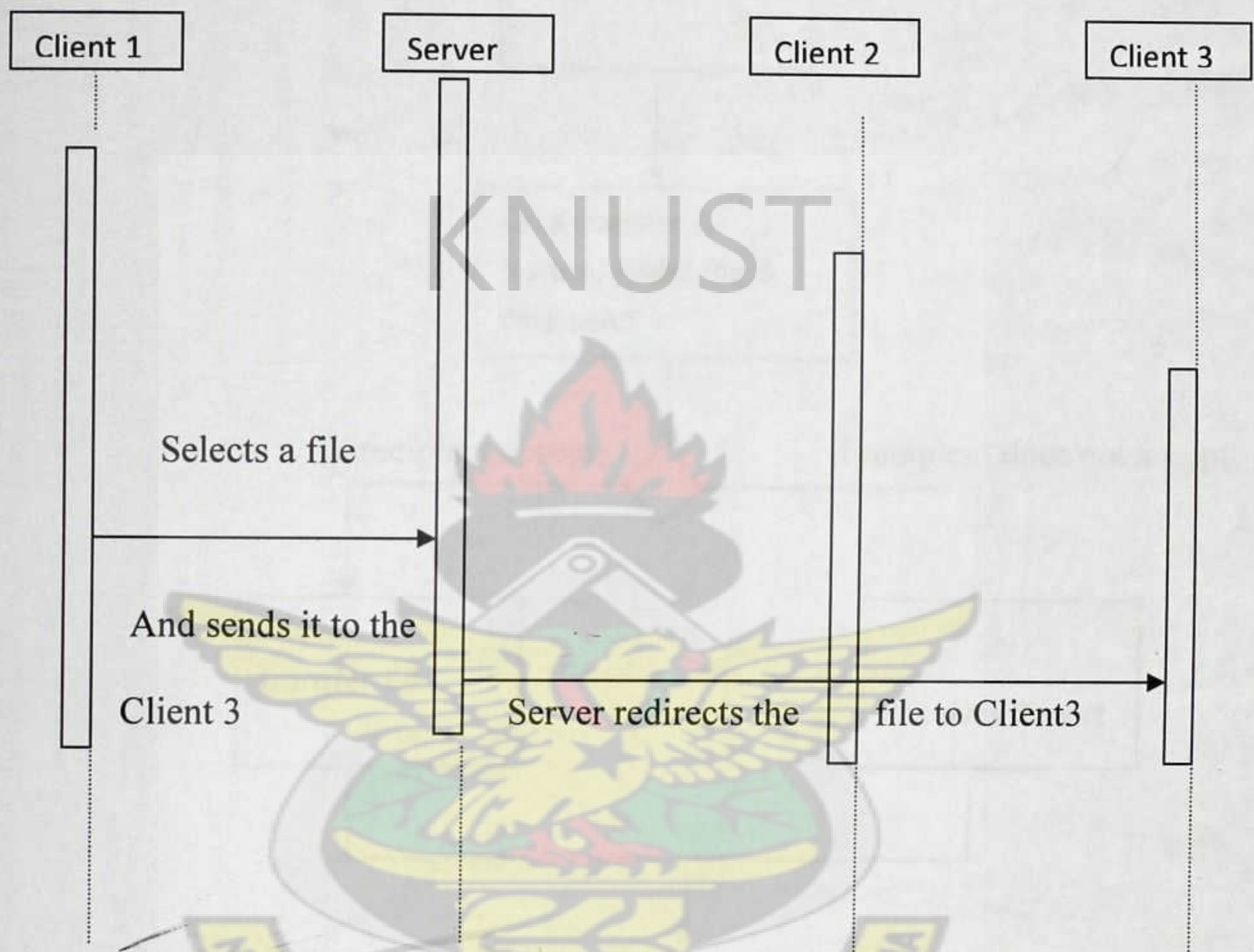


Figure 3.6 Sequence Diagram for transferring a file from Client 1 to Client 3.

Figure 3.6 illustrates how a client transfers file to other client. Client 1 selects a file and sends it to the client 3. The file request goes to the server and it is redirected to the client 3. Here client 3 is a passive mode client, which can only receive a file.

The code for the file transfer is located at the Appendix C and the actual source code could be found on the attached CD.



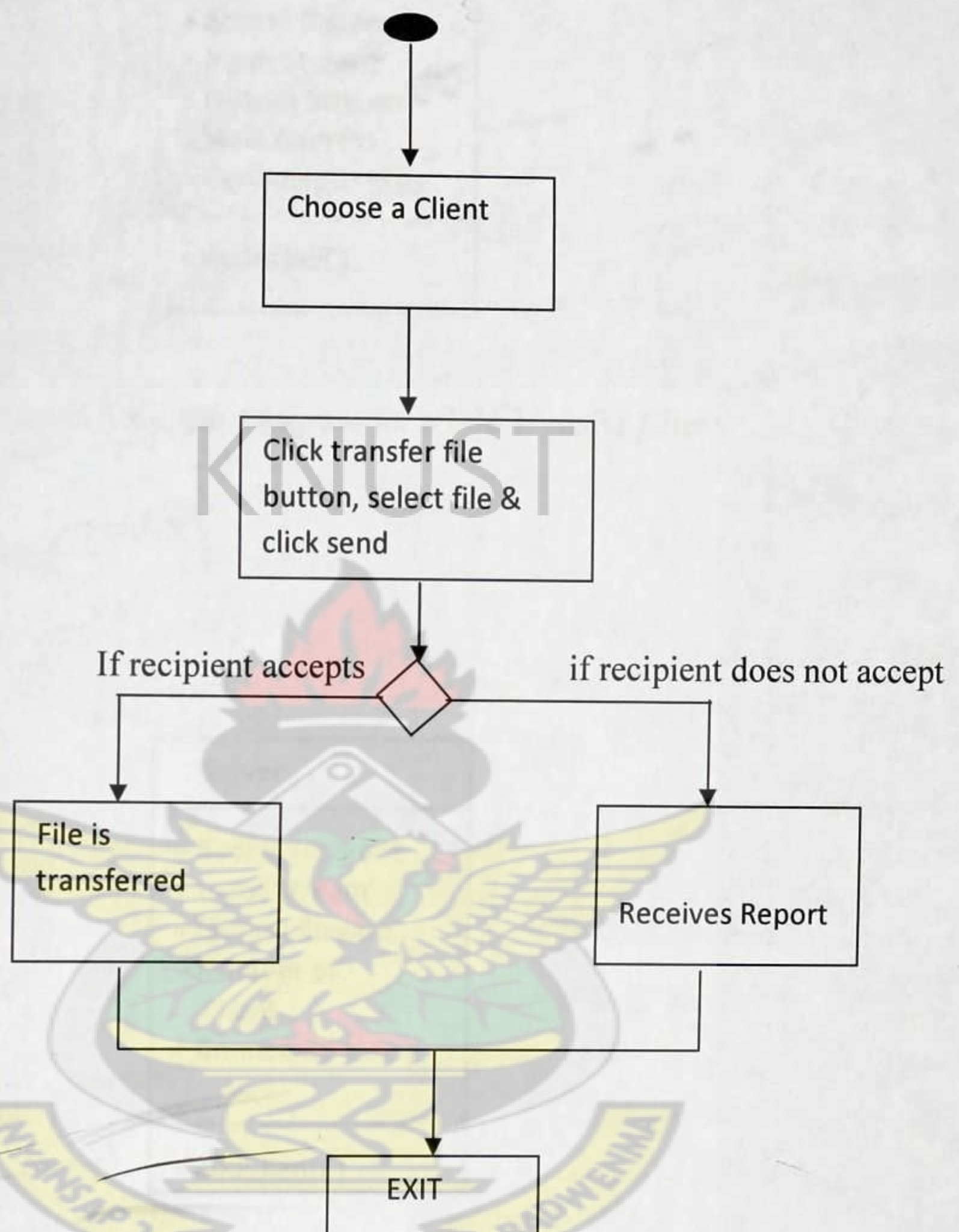


Figure 3.7 Activity Diagram for transferring file



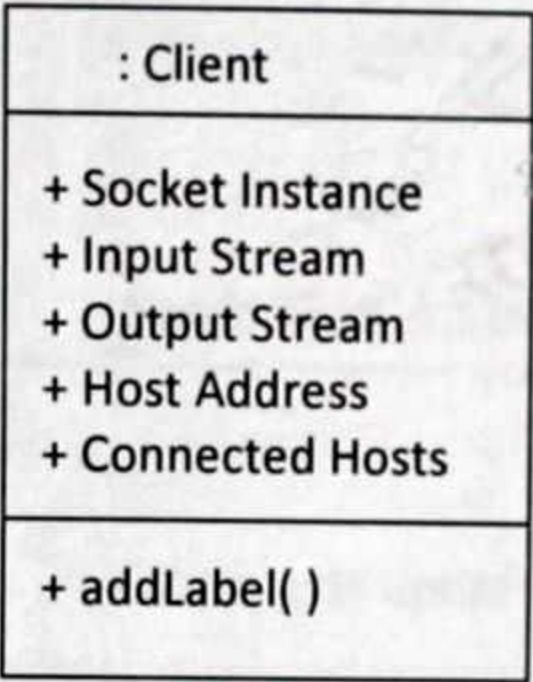


Figure 3.8 Class Diagram for a File Transfer Client

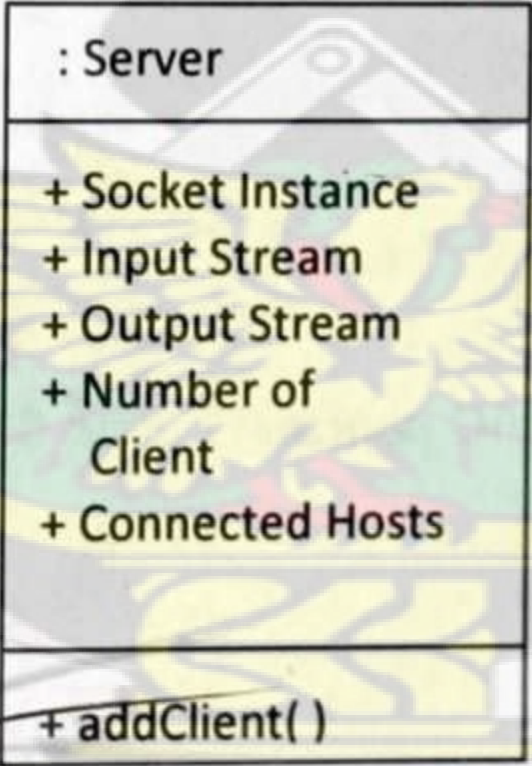


Figure 3.9 Class Diagram for File Transfer Server



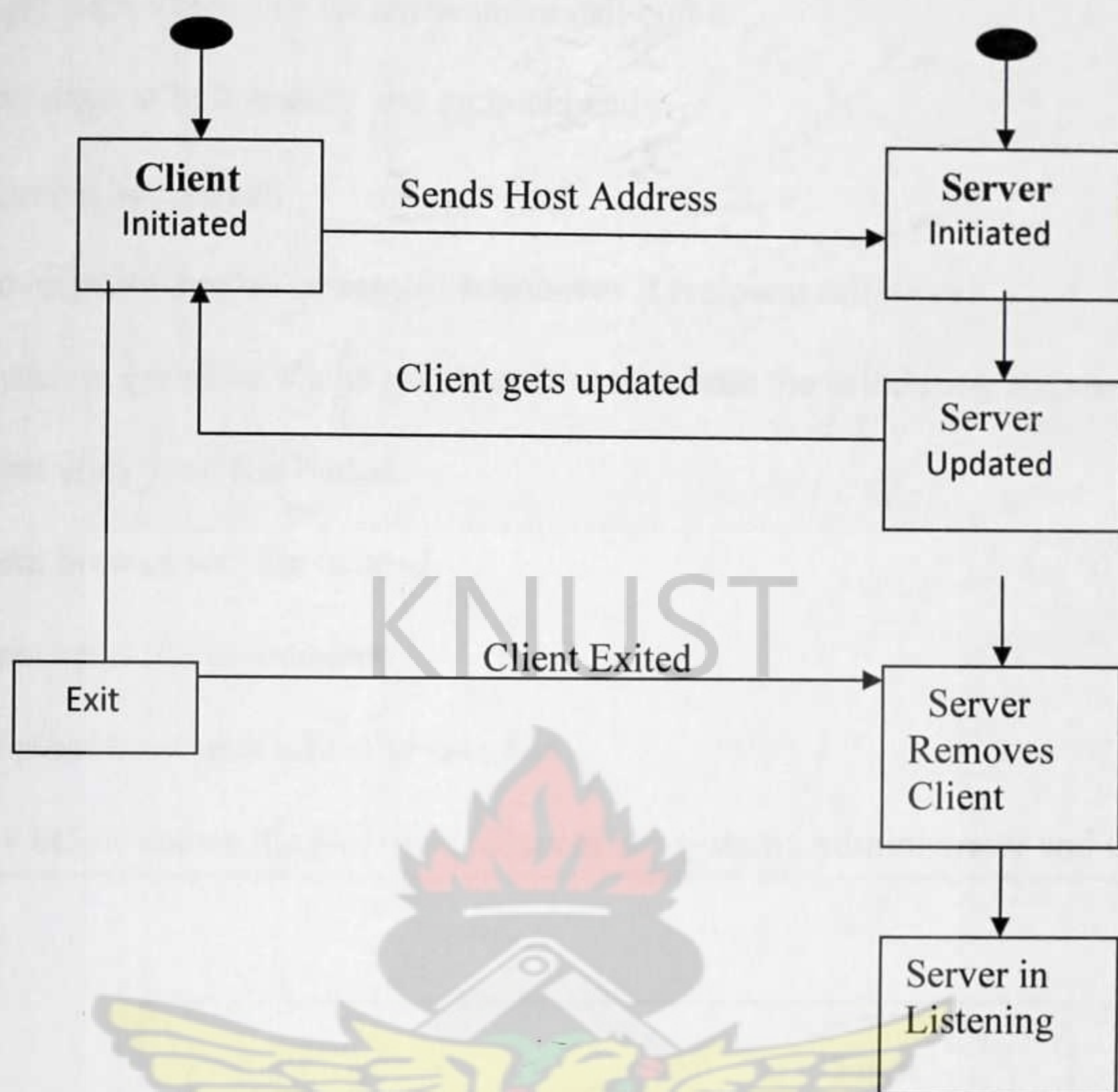


Figure.3.10 State Transition Diagram for Server

### 3.6 SYSTEM USE CASES

- i. Administrator starts the server.
- ii. Administrator configure settings
- iii. Client launches client software
- iv. Client login
- v. client selects other client to chat or send file to
- vi. client write text messages in the text box
- vii. client click send message button



- viii. client click voice call button or video call button
- ix. tone rings at both sender and recipient ends
- x. recipient accept call
- xi. conversation begins or session terminates if recipient rejects call
- xii. Sender or recipient clicks stop button to terminate the call during conversation.
- xiii. client click send file button
- xiv. client browse for file to send
- xv. client send file to recipient
- xvi. recipient browse to where to save file

Figure 3.11 below shows the two main actors of the system: Administrator and Client.

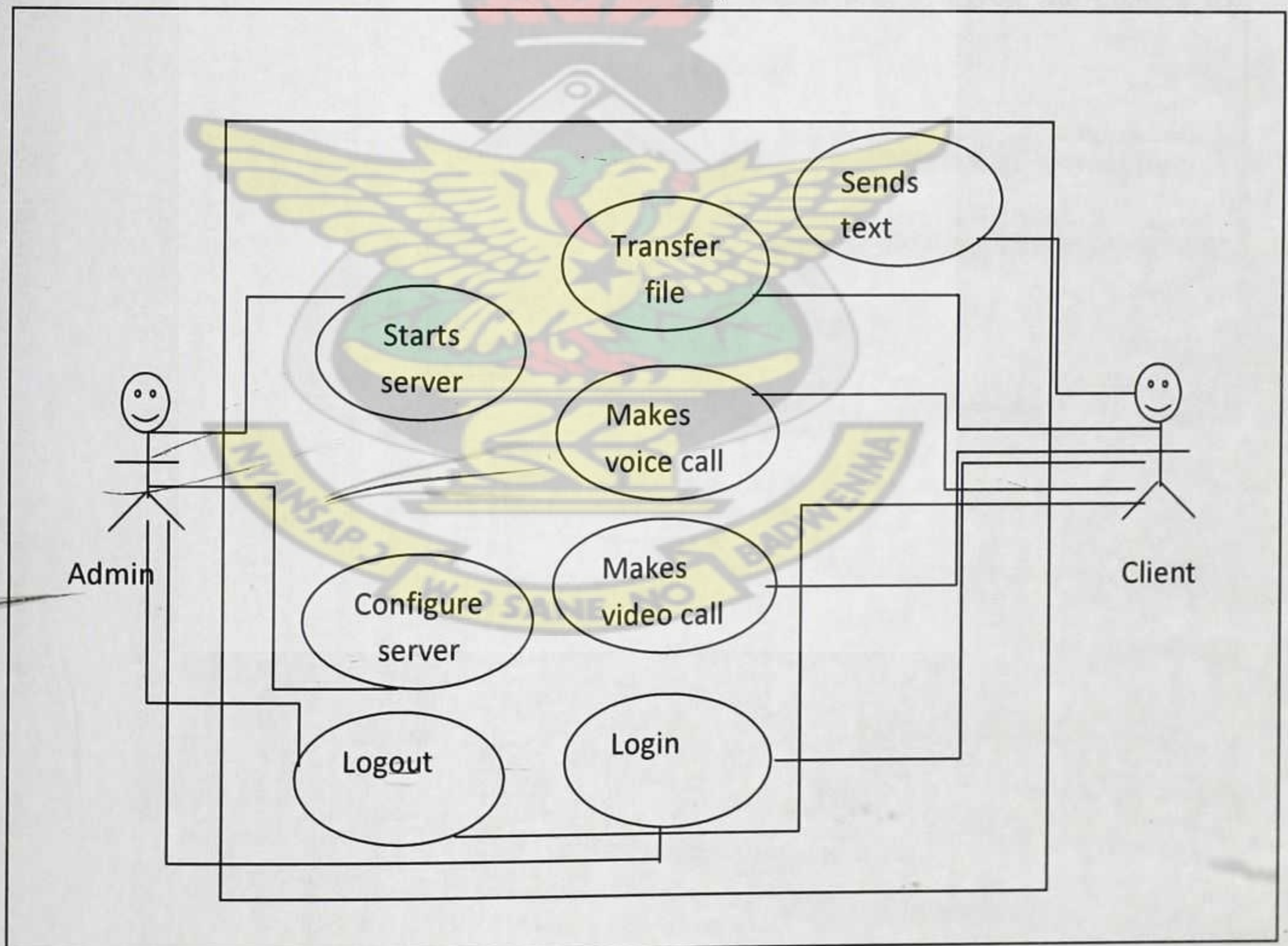


Figure.3.11 System boundary diagram



Table.3.1 Use Case Descriptions

This table describes the various activities that the actors of the system will be performing.

Use case	Description
Login	The user enters an assigned username and a correct password to have access to the system
Logout	A user who has logged in successfully exits the system.
Changed password	A user who has entered the system successfully re-enters his/her password and is given the chance to change the password.
Make Text Chat	A user selects a partner and sends text messages.
Make Voice Call	A user selects an online client and clicks voice call button to initiate conversation.
Accept Call	Recipient clicks Accept Call button to begin conversation with caller
Reject Call	Recipient clicks Reject Call button to terminate session.
Make Video Call	User clicks Video Call button to begin Video Call.
Transfer File	User selects an online client to send file. Browse to locate file. And click open to send file.



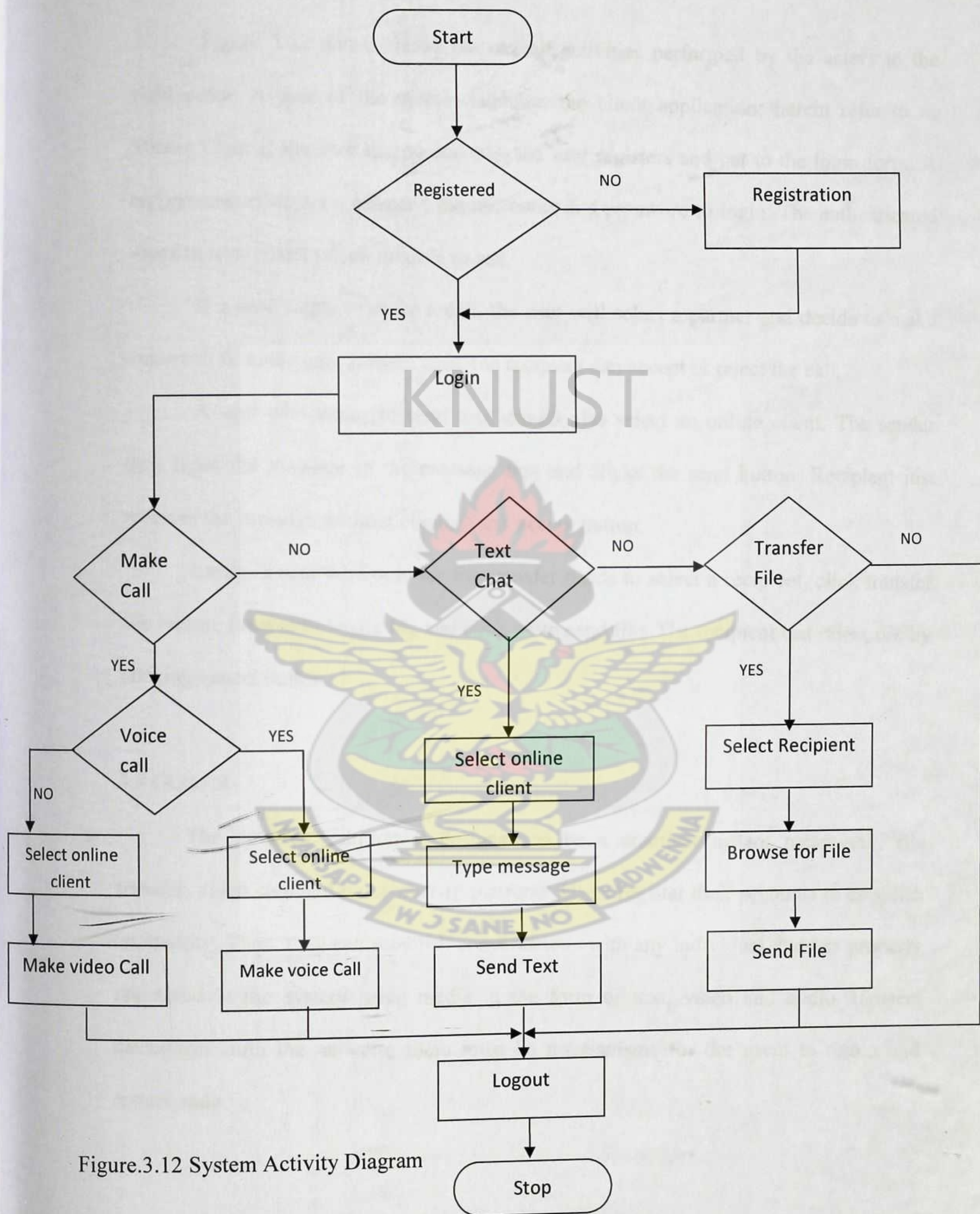


Figure.3.12 System Activity Diagram



Figure 3.12 above shows the overall activities performed by the actors in the right order. A user of the system launches the client application; herein refer to as Master Chat. If the user has no account, the user registers and get to the login form. A registered user enters a Master Chat username and password to login. The authenticated user can now select which module to use.

If a user wants to make a call, the user will select a partner and decide to make voice call or video call. In each case, the recipient can accept or reject the call.

A user who wants to send text should also select an online client. The sender then types the message in the message box and clicks the send button. Recipient just receives the message without clicking any accept button.

Lastly, a user who opts for file transfer needs to select a recipient, click transfer file button, browse to locate file and click ok to send file. The recipient can reject file by clicking cancel button.

### 3.7 GOALS

The research solution is designed to be a practical instant messaging, file transfer, video chat and voice-over-IP platform. Users register their accounts to establish an identity. Then, they can establish conversations with any individual that has properly registered in the system using media in the form of text, video and audio. If users disconnect from the network, there must be mechanisms for the users to rejoin and restore state.



### 3.7.1 Performance goals

The system should be able to scale properly such that even when the maximum number of clients is active in the system, lookups are still efficient. In order to be a practical system for use, session initiations must impose a hard-limit on call latency. That is when a user wants to chat, the time it takes to locate the other user and initialize a complete session must take less than 5 seconds. Also, during text chat, the time it takes for a message to move from a sender to a recipient should be less than 3 seconds. Any additional wait would cause users to quit in frustration. Ideally, this number should be as low as possible.

### 3.7.2 Usability Goals

In order to ease adoption, the user must interact with the software in the same way that users interact with existing popular client software. The software must provide Instant Messaging, voice call, video call and file transfer capabilities. Any additional effort the user must take to learn a new protocol may turn them off from using it. On a similar note, the service must bootstrap without user intervention.

## 3.8 USER REQUIREMENTS

The proposed system has four main capabilities: the VOIP, video call, Instant Messaging and Transfer of data.

A new user needs to create an account before using the system. Upon creating an account, the user will be provided with a unique username and a password. The user will be provided with an interface to log into the system. This will authenticate and



authorized the user. The user will then select which component of the system to use. After log in, available or active users will be displayed. A user needs to select a partner to chat with.

With the messaging component, a user can chat with only an individual. The Administrator can log all messages sent across the network. The Administrator too can allow for file transfer.

When using the file transfer application, the user will select the recipient and browse for the file to be transferred. When the sender sends the file, the recipient will be notified and the recipient can browse to a location where the file needs to be stored. The recipient can refuse to accept the file and this will terminate the process.

When using voice call (VOIP) component, the user will select the recipient and click on voice call. The recipient will be given the option to reject or accept call. Tone will continue to ring at the recipient's side until the recipient accept or reject the call.

The video component works same as the voice call. The sender has to send request to the recipient by clicking on video call button. The recipient is given the option to accept or reject the call.

During setup, the Administrator will have to select the port number and maximum number of client that can use the system at a time. This will be done on the server side of the architecture.



### 3.9 USER SPECIFICATION

- i. A user should provide a unique user name and a password (of not less than six characters) to register.
- ii. A maximum number of clients that can use the system at any particular point in time will be specified by the Administrator.
- iii. The time taken to connect to a partner should be less than 5 seconds.
- iv. When using the instant messaging facility, the time taken for a text to move to the recipient should be less than 3 seconds.
- v. When chatting, a user can send any number of characters.
- vi. With Transfer of data, the time taken to successfully transfer data will depend on the size of data to be transferred. Data less than 1Mb should take not more than one minute to be transferred.
- vii. The delay in voice transmission should not be more than two seconds.

### 3.10 LIMITATIONS AND RESTRICTIONS

- i. The system will not include voice and video conferencing
- ii. A user can send message while calling. However, when file is being transmitted, file transfer dialog box will be displayed and this will be active until the file is successfully transmitted. This will prevent other aspect of the system such as instant messaging and voice call to be used.
- iii. Since the client application for mobile devices was designed with Android SDK, the client application will run on only mobile phones with Android operating system.



### 3.11 FUNCTIONAL REQUIREMENT

Here, hardware and software requirements are discussed.

#### 3.11.1 Hardware Requirement

The computer hosting the server application should have the following basic requirements:

- i. At least 512 MB of memory size
- ii. A dual core processor of at least 1.2 GHZ each. The reason is that the rendering of the media should be fast enough to ensure a real time transmission.
- iii. A hard disk drive of at least 20 GB of free space.

The computers hosting the client application should have the above requirements together with the following:

- i. A webcam
- ii. A Microphone
- iii. A Speaker
- iv. In the absence of Microphone and Speaker, a headset could be used. In fact, to reduce noise, headset is preferred.
- v. The application uses client/server architecture. Therefore, for the application to be able to run, there must be a network connectivity.

The client application could also be installed on mobile phones. The mobile phone hosting the client application should have the following requirements:

- i. Must be a Smart Phone due to a high processing speed needed.
- ii. Must have an android operating system installed.



### 3.11.2 Software Requirement

The computers hosting the application should preferably have windows operating system installed. For windows operating system, it should be Windows XP or higher. The android operating system on the smart phones should be 2.2 version or higher.

### 3.12 NON-FUNCTIONAL REQUIREMENTS

#### i. Security

- Login requirements: A user must login with a unique user name and password. A user name and or password that are not in the database will not be accepted by the system. During registration, the system will not accept a user name that is already in used.
- Password requirements: The length of user password should not be less than six characters. The user is given an option to change password.
- Inactivity timeouts: The user is given three tries to enter user name or password. The system automatically exits after three unsuccessful attempts.

#### ii. Performance

- Response times - application loading, screen open and refresh times will be less than 5 seconds.
- Processing times: The time taken to render media and transmitting to the intended destination will be less than 5 seconds.

#### iii. Availability

- Hours of operation – The services of the system will be available as long as there is network connectivity.



iv. **Reliability**

If capturing devices are working well, the system will always perform its functions.

v. **Compatibility**

Compatibility on different operating systems: The application is developed with Java programming language. The application will therefore be compatible with any operating system. However, since the system was tested with Windows operating system, Windows operating system is preferred. With the android client, since the android client was developed with android SDK, it will only run on phones with android operating system.

vi. **Usability**

- Look and feel standards: The user interfaces were designed to make it richer and easy to use. Tool tips were provided where necessary. Keyboard shortcuts were also provided for most commonly used buttons such as *ok* and *exit*.

### 3.13 TOOLS AND MATERIALS

The following tools and materials were used in the design of the system:

1. Java programming language
2. Java Development Kit (JDK)
3. NETBEANS IDE
4. Android SDK
5. Eclipse IDE
6. Java Media Framework (JMF)
7. Mobile Phone with Android operating system 2.2 or higher



8. Wireless router (or Access Point)
9. Personal Computer with windows operating system

### 3.13.1 Why Java Programming language?

Proprietary hardware and platform specific video formats require video conferencing application to be tailored and rewritten for each computing platform. The emergence of Java technology provides tools to create platform independent applications. Java provides solutions to a number of problems—platform independence, security, and international character set being the most important—that are crucial to Internet applications.

One of the biggest secrets about Java is that it makes writing network programs easy. In fact, it is far easier to write network programs in Java than in almost any other language. This is the reason why the researcher chooses the java programming language over the rest.

### 3.13.2 Why Java Media Framework (JMF)?

The increase in the processing speed of the personal computers coupled with the increase in the bandwidth of the Internet has resulted in new and exciting multimedia applications. There has been a growth in the multimedia traffic over the Internet in recent years. Some of the promising multimedia applications that are deployable over the Internet include: Video conferencing, media-on-demand, Access to multimedia database over Internet, Interactive distance education, Collaborative computing, Virtual reality based scene navigation, Interactive games rich in 3D graphics, etc.



To provide a fully integrated network-based multimedia solution, the JavaSoft along with its industry partners have developed a family of APIs called the Java Media APIs. The Java Media APIs is a set of APIs that enables Java programmers to add multimedia to their applications. The Java Media API's include the Java™ Media Framework (JMF) API.

JMF is a framework for handling streaming media in Java programs. JMF is an optional package of Java 2 standard platform. JMF provides a unified architecture and messaging protocol for managing the acquisition, processing and delivery of time-based media. JMF enables Java programs to

- (i) Present (playback) multimedia contents,
- (ii) Capture audio through microphone and video through Camera,
- (iii) Do real-time streaming of media over the Internet,
- (iv) Process media (such as changing media format, adding special effects),
- (v) Store media into a file (<http://www.ee.iitm.ac.in/~tgvenky/JMFBook/Tutorial.pdf> 6/7/12 11:10 am.).

The researcher decided to use JMF since the research involves the capturing of audio and video, and transmitting it over a network.

### 3.13.3 WHY NETBEANS IDE?

NETBEANS is one of the visual development environments that improves and enhance the designing of Java applications. The Graphical User Interface (GUI) text editor in NETBEANS enables you to edit Java source code with colour context highlighting. This makes it much easier to read and understand code because keywords and other language constructs are coloured uniquely.



#### 3.13.4 WHY ECLIPSE IDE AND ANDROID SDK?

Eclipse is a robust, open-source IDE based on a general plug-in architecture. All components of Eclipse are plug-ins. With so many excellent, even free, IDEs out there, why Eclipse? There are three reasons. First, Eclipse is intuitive to learn and use, and just one IDE serves a wide variety of purposes--from Java, AOP, web, to C/C++ development. The investment pays good dividends. Second, Eclipse's general architecture has created a global community of plug-in developers from whom a wide variety of tools could be drawn. Some research groups release plug-ins unavailable elsewhere. Finally, Eclipse has extensive and reliable refactoring features.

There are many ways to develop Android applications on your PC. The easiest way is integrating the ADT (Android Developing Tools) with the Eclipse IDE.

#### 3.13.6 WHY ANDROID MOBILE PHONES?

Android has become one of the leading operating systems for modern mobile phones, and stands as serious competitor to the iPhone and Windows Mobile. The researcher chooses android mobile phones because they are faster and reliable. Android supports RTP/RTSP streaming and variety of media formats. While most Android applications are written in Java, there is no Java Virtual Machine in the platform and Java byte code is not executed. Java classes are compiled into Dalvik executables and run on Dalvik, a specialized virtual machine designed specifically for Android and optimized for battery-powered mobile devices with limited memory and CPU.



## CHAPTER FOUR

### IMPLEMENTATION AND TESTING

The research aimed at creating a system that will help users to communicate over a Local Area Network through text, audio (VOIP), video and transferring of file. This section deals with the implementation of the various methods and tools used in the research. Here, how the solution was implemented, tested and evaluated is discussed.

#### 4.1 THE USER INTERFACE

The Graphical User Interface is composed of three components: Server-Side GUI, Client-Side GUI, and Database and Table creation GUI.

##### 4.1.1 Server-side Graphical User Interface (GUI)

This GUI allows the administrator to configure the server side of the application. The server must start before any client can communicate to one another. The administrator can select the number of users to use the system at a time. If there is error in loading the application, the administrator can view the server log for more information. The main server form is shown in figure 4.1 below.



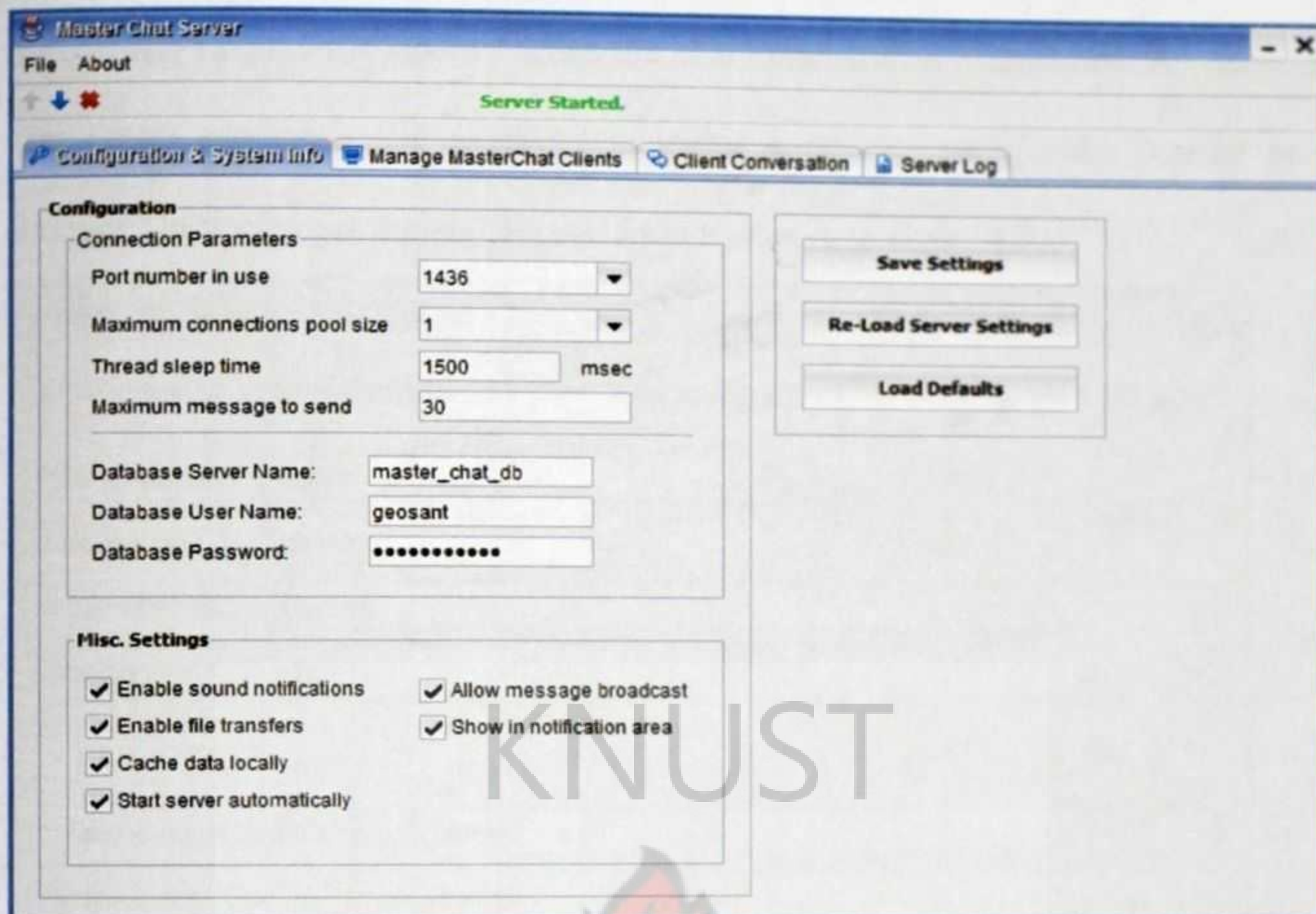


Figure 4.1 Main Server Form

#### 4.1.2 CLIENT-SIDE GUI

Client-side GUI has been created for the following:

- (1) Login form – for users to login into their account if existing.
- (2) Registration form – for users to create a new account if not existing.
- (3) Services form – gives access to authenticated users to use the services provided by the Application (options like for text chat, data transfer, audio and video call are provided here).



Figure 4.2 Login form

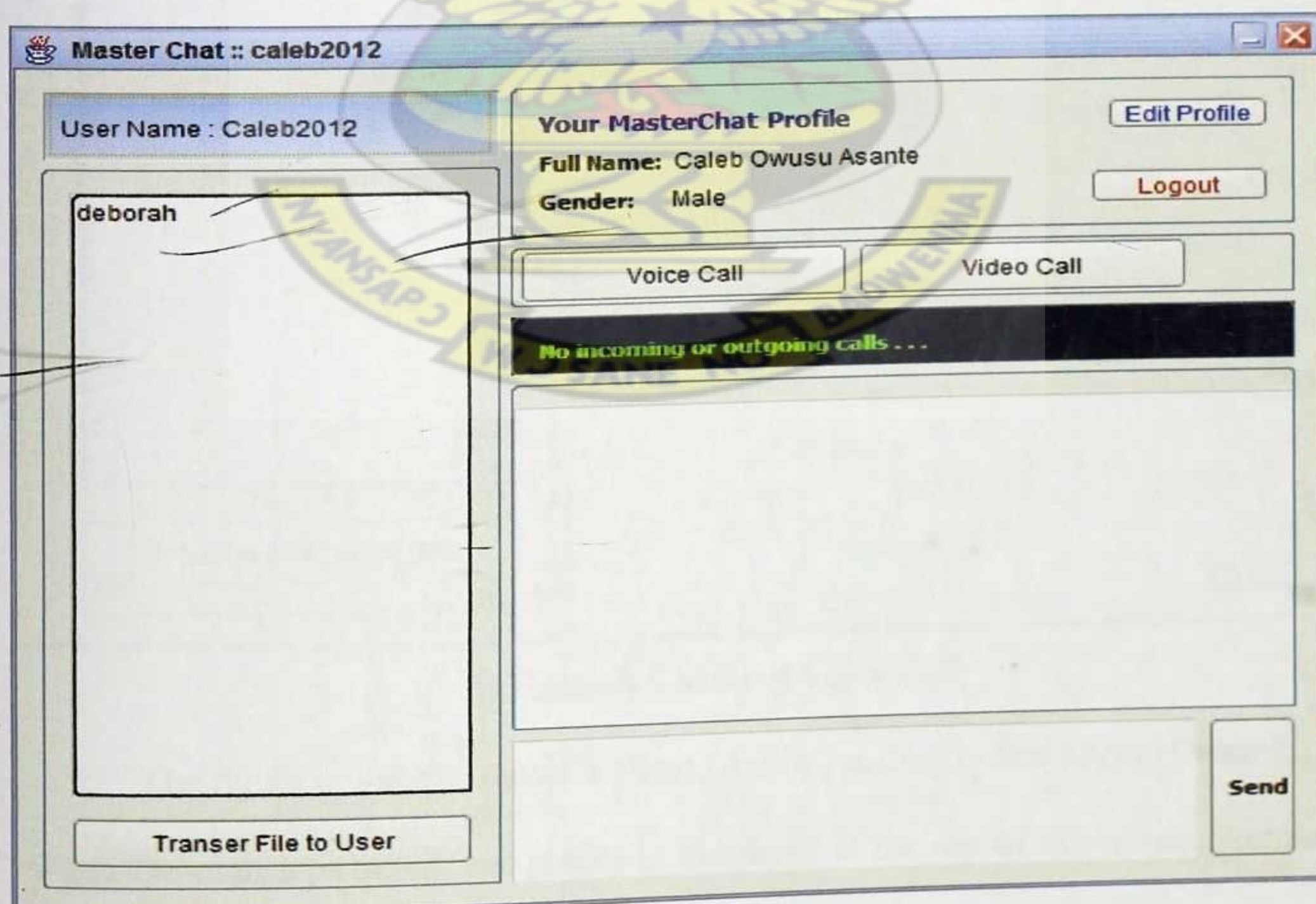


The Login form opened when the client application is launched. An existing user enters username and password to login. A new user clicks the “*create new account*” button to get the registration form as shown in figure 4.3.



The screenshot shows a window titled "Create your MasterChat User Account". Inside, there is a prompt: "Please enter your details to get your MasterChat user account created for you." The form contains several input fields: "First Name" with the value "Caleb", "Surname" with "Asante", "Other Names" with "Owusu", and "Gender" with a dropdown menu set to "Male". Below these are fields for "Your MasterChat Name" (Caleb), "Password" (masked with dots), and "Confirm Password" (masked with dots). At the bottom, there are two buttons: "Create User Account" and "Cancel". A large, faint watermark of the KNUST logo is visible in the background.

Figure 4.3 Registration form



The screenshot shows a window titled "Master Chat :: caleb2012". The interface is divided into two main sections. The left section, titled "User Name : Caleb2012", contains a large text area with the name "deborah" and a button labeled "Transer File to User". The right section, titled "Your MasterChat Profile", displays the user's details: "Full Name: Caleb Owusu Asante" and "Gender: Male". It includes buttons for "Edit Profile" and "Logout". Below the profile section are buttons for "Voice Call" and "Video Call". A status bar indicates "No incoming or outgoing calls...". At the bottom right, there is a "Send" button. The same KNUST watermark is present in the background.

Figure 4.4 Services Form



After login, the services form shown in figure 4.4 is displayed. Here available clients are listed at the left pane of the screen. A user who has login can edit his or her profile. To call an online client, the client needs to be selected and the voice call or video call button is clicked. Once an online client is selected, Transfer File to user button could be clicked to browse for a file and send to the recipient. User just need to enter a message in the message box provided and click send to send text message to selected client.

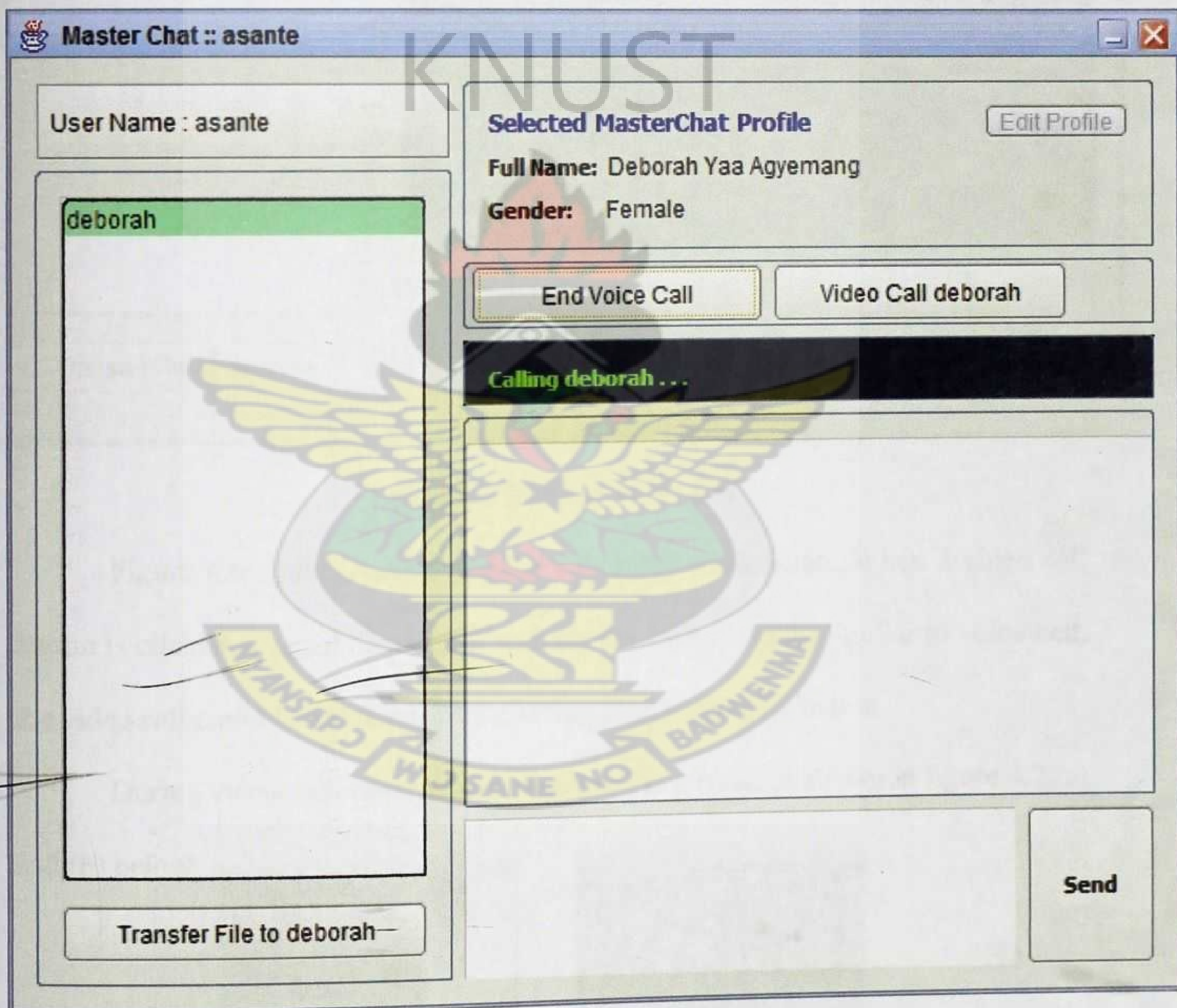


Figure 4.5 Making Voice Call

The figure 4.5 above shows a client (Asante) calling online client (Deborah). When Deborah is selected, her profile is displayed at the top of the screen. Asante can terminate the call by clicking *End Voice Call* button.



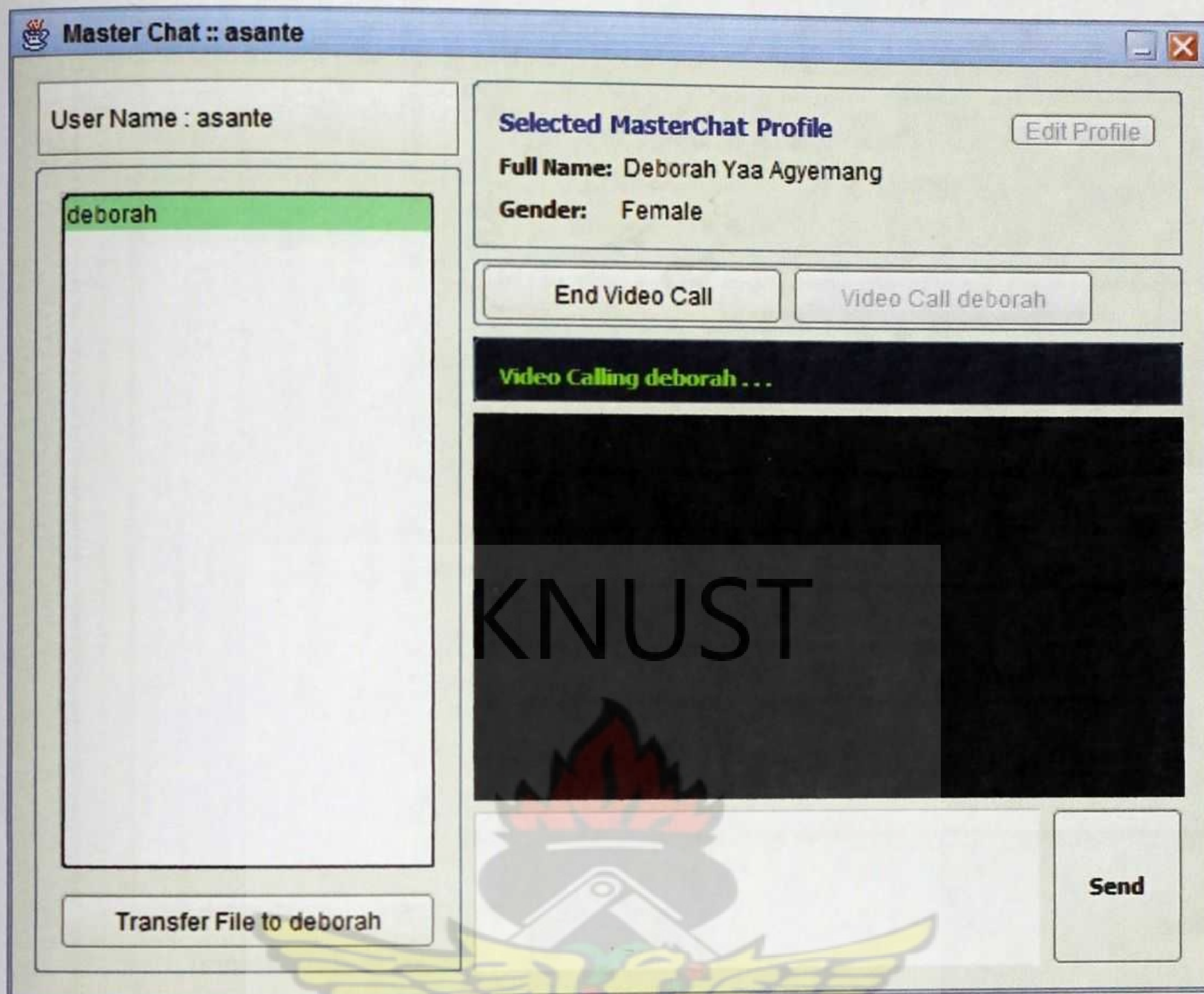


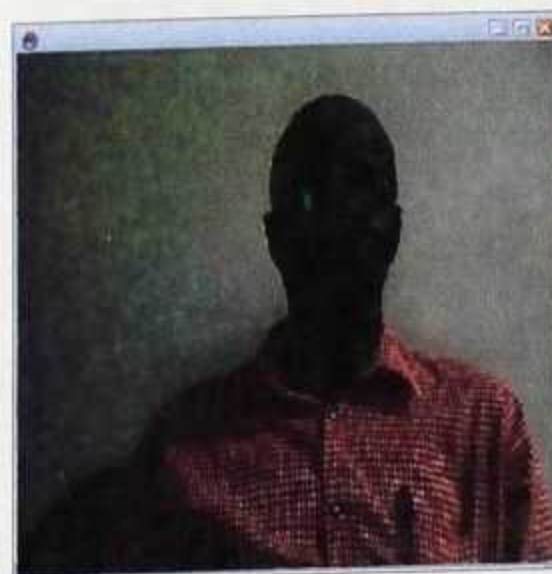
Figure 4.6 Making Video Call

Figure 4.6 shows Asante making video call to Deborah. When a video call button is clicked, a panel displays to wait for the video stream. Similar to voice call, the video call could be terminated by clicking *End Video Call* button.

During video call, the stream is displayed in a panel as shown in figure 4.7 (a) and (b) below



(a)



(b)

Figure 4.7 Video Call Screens



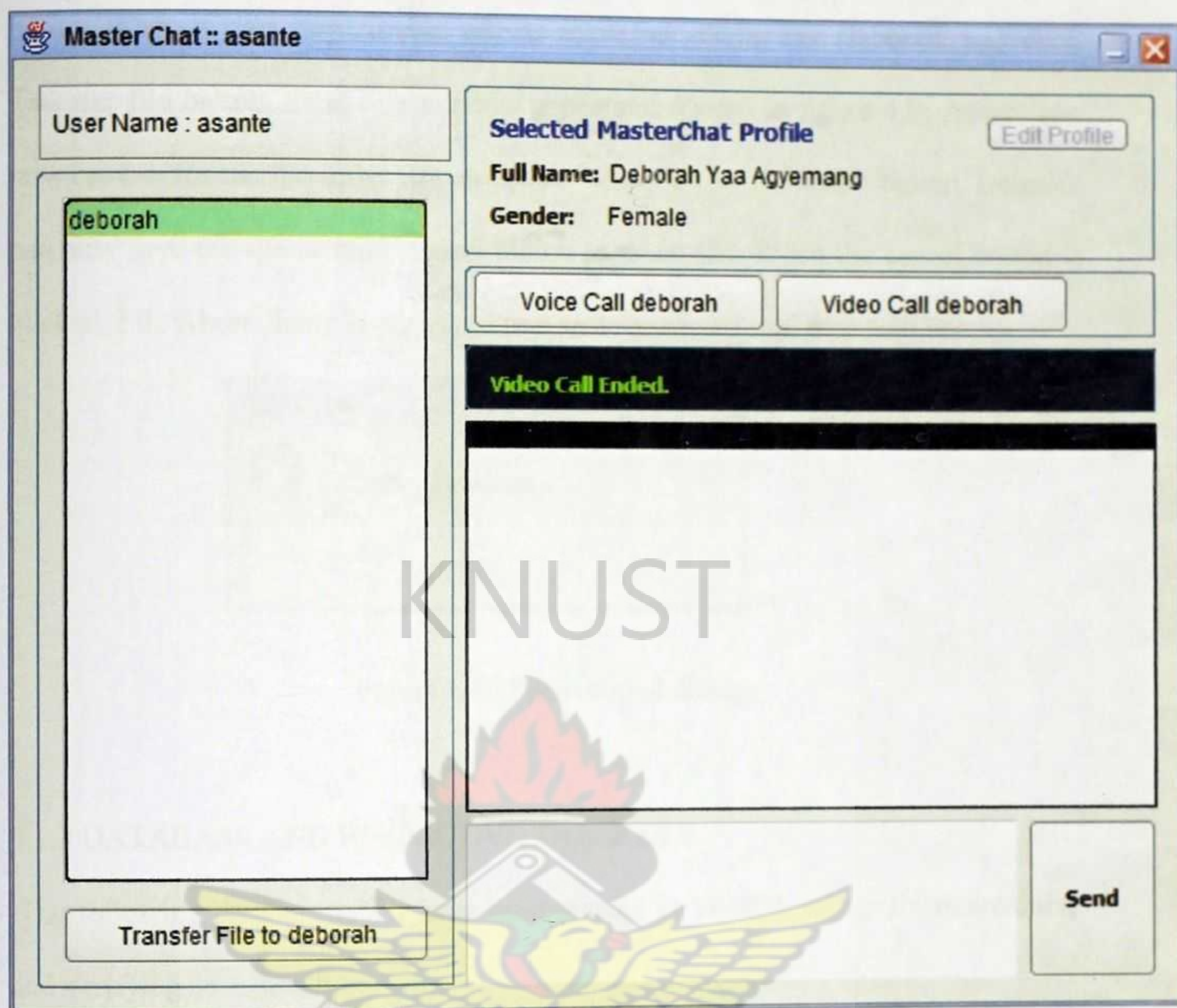


Figure 4.8 Video Call Ended

When End Video Call button in Figure 4.6 is clicked, the video call is terminated and Figure 4.8 is displayed.

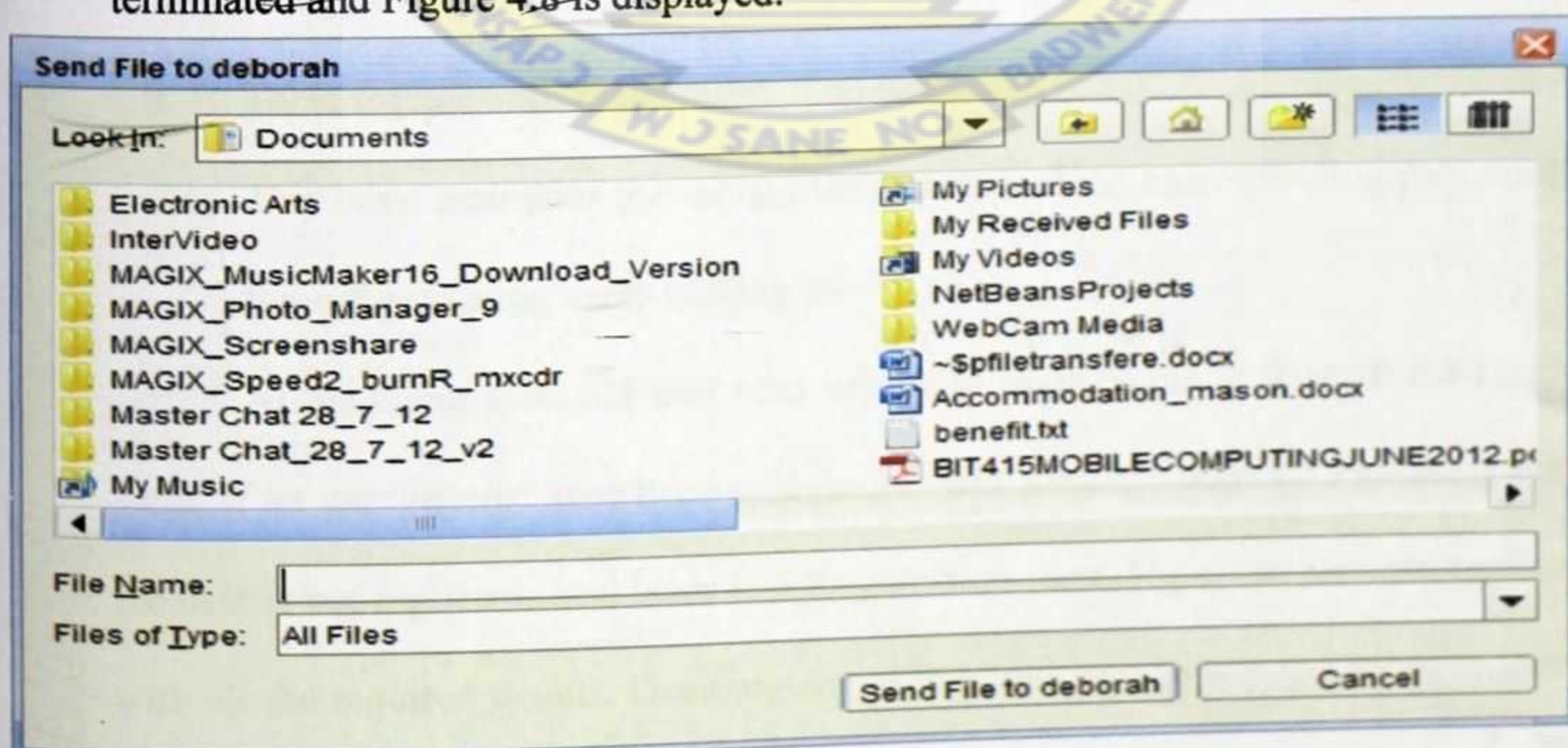


Figure 4.9 Transferring File



When a user, say Asante, selects an online client, say Deborah, and click Transfer file button, Send File window appears as shown in figure 4.9. Asante can now browse for the file to be sent and click “*send File to Deborah*” button. Deborah can now save the file or click cancel button to reject file. When the cancel button is clicked, File Abort dialog is displayed to Asante as shown in figure 4.10 below.



Figure 4.10 File Aborted dialog

#### 4.1.3 DATABASE AND RESPECTIVE TABLE GUI

The following database tables have been created in MySQL server for maintaining and verifying user details:

i. Table *registeredUsers* :

This table maintains the details (such as name, gender, etc.) of all the users using the application.

ii. Table *loginusers* :

This table maintains the unique usernames and the corresponding passwords for authenticating users logging in.

At the client side, the user who wishes to login, signs in through the Login form, if an unregistered user tries to login an error message pops up indicating that the user is not registered and leads to a Registration form. Here, the user can register with all the required details. Once registered, the user logs in and the services form opens which allow the user to indulge in the desired type of communication.



Once connected, the user name appears on the list of users available online at that particular moment. Now the user can enjoy chatting with any number of users/clients online. Whenever desired, the user can go offline by disconnecting from the network through the disconnect button.

## 4.2 TESTING

In a software development project, errors can be injected at any stage during development. In testing phase, the program to be tested is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as expected. Testing is the process of executing a program with the intent of finding errors.

Two main approaches for testing were considered: Functional (Black box testing) and Structural (White box testing). In Functional testing, the structure of the program is not considered. Test cases are decided solely on the basis of the user specifications of the program or module and internals are not considered for selection of test cases.

The test cases were designed in the following format:

Condition:

Input Criteria:

Input:

Result:

The **condition** describes the rule of the system. **Input criteria** specify the valid input boundaries. **Input** defines the sample input. **Result** gives the output of the sample input.



#### 4.2.1 Black Box Testing

- i. Test case for '*Server should run for a client when a client want to connect to the Server*'

**Condition:** "The Server must be under running state when a client get Connected".

**Input Criteria:** Any client can connect to Server, which is not running.

**Input:** Client 1 wants to connect.

**Result:** Error message as shown in Figure 4.11 is displayed.



Figure 4.11 Connection Error Form

- ii. Test case for '*A client wants to connect twice to the Server*'

**Condition:** "A Client should not be connected twice to the Server".

**Input Criteria:** Allow any client connect to the Server twice.

**Input:** Client 1 is already connected to the Server, and wants to connect again.

**Result:** Error message as shown in Figure 4.12 is displayed.



Figure 4.12 User Exists Form



- iii. Test case for '*A client should be disconnected from the Server, even when the client program is terminated abnormally*'

**Condition:** "A client has to be disconnected when it is terminated abnormally".

**Input Criteria:** Take any client.

**Input:** Terminate the client program in any way (normally or abnormally).

**Result:** The particular client is disconnected from the Server.

#### 4.2.2 White Box Testing

**Introduction:** -The Structural approach is also known as White Box testing.

In this approach test cases are generated based on the actual code of the program or module to be tested. This Structured approach is sometimes called "Glass Box testing".

Some of the test cases are listed below:

- i. Test case for "**Entering username and password accurately**".

**Condition:** "The username and password must be entered".

**Input Criteria:** Log in new user.

**Input:** do not enter correct username or password or both.

**Result:** Error message as shown in figure 4.13 is displayed.

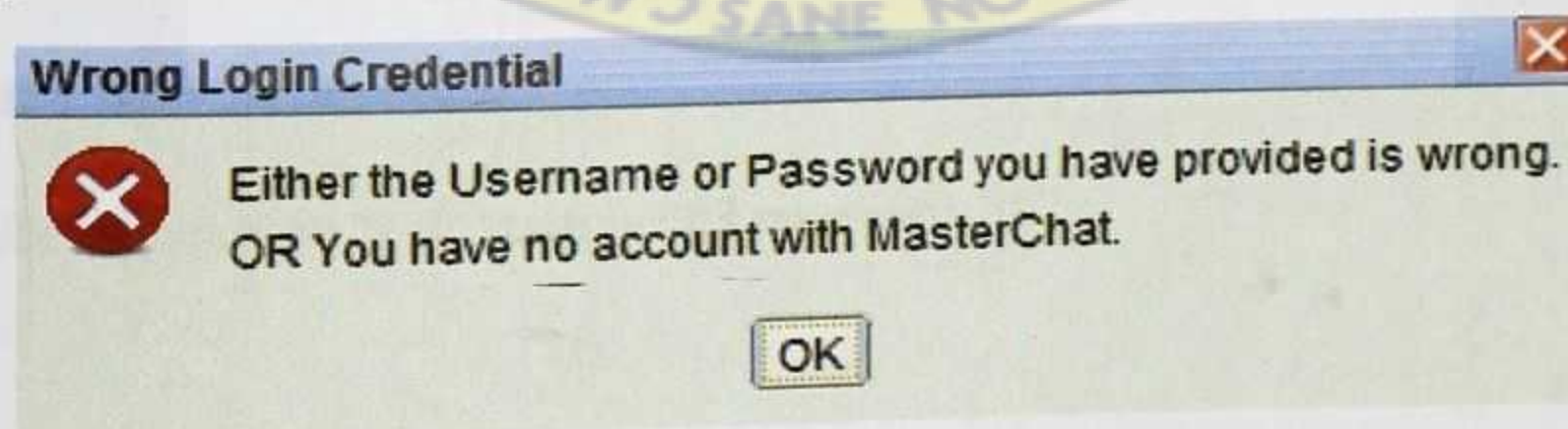


Figure 4.13 Wrong Login Credentials

- ii. Test case for "**Registering with existing username**".

**Condition:** "client can register only with a unique username".

**Input Criteria:** Register new client.



**Input:** use an existing username for the new client.

**Result:** Error message as shown in figure 4.14 is displayed



Figure 4.14 User Account Creation Aborted

- iii. Test case for “connecting more clients to the server than one specified by the administrator”

**Condition:** “A maximum number of clients that can use the system at any particular point in time will be specified by the Administrator”.

**Input Criteria:** Try to connect more clients than one specified by the administrator.

**Input:** The server is configured to allow only 3 connections. Client1, Client2, and Client3 is connected to the server. Afterwards, Client4 tries to connect to the server.

**Result:** Error message ‘Connection unsuccessful! Total clients exceeds limit!’

- iv. Test case for “Transferring a file to a single client”

**Condition:** “Clients can send a file to one client at a time”.

**Input Criteria:** Choose single.

**Input:** Client 1 is selected.

**Result:** A message “File transferred successfully”.



### 4.2.3 TESTING STRATEGY

The testing was also done using the following strategy:

- i. **Component testing:** Each component that makes up the Chat Application was tested. These include the Text Chat module, Audio Chat module, Video Chat module and File Transfer module.
- ii. **Integration testing:** The various modules of the application were integrated and tested to ensure the correct interworking of the components.
- iii. **Validation testing:** The integrated application was tested to ensure that it works correctly in a pseudo-live environment. This is shown in figures 4.1 to 4.14

### 4.2.4 TEST PRIORITIES

During testing of the application, the following qualities were tested in order of priority:

- i. **Functionality**—whether the required functions are available and working as expected. This is shown in figures 4.1-4.14
- ii. **Usability**—how user friendly and intuitive the Application is.
- iii. **Security**—how well-protected the application is. This is shown in figures 4.2,4.11-14
- iv. **Performance**—whether the response times are within acceptable limits.

### 4.2.5 TEST ENVIRONMENT

#### **Hardware and Software**

The test environment consists of:

- i. An Intel-Compatible 32bit computer running Windows operating system to host the server side of the application.



- ii. At least two personal computers equipped with webcam to host the client application.
- iii. At least two smart phones with Android operating system to host the android version of the client application.
- iv. A Local Area Network either wired or wireless

#### 4.3 EVALUATION

The proposed system was evaluated based on the research questions.

##### 4.3.1 Research Question I

*To what extent will the leveraging of VOIP using JMF improve communication on campus?*

The proposed system has really improved communication on KTI campus. Previously, if members of staff of the institution want to talk to any staff at the various departments, they have to use their mobile phones which they are eventually charged by their network operators. Now, with the new system in place, the members of staff can text or call any staff at the various departments without paying any fee.

Since members of staff do not pay for text chat and calls made, they can make any number of calls within a day. The table 4.1 below shows call and SMS charges for selected network operators in Ghana. Users of the various networks are charged for the number of minutes used but users who use the proposed system, *MasterChat*, will pay no fee.

From Table 4.1, the cost per minute for Tigo to Tigo is 3 pessewas, Tigo to other networks cost 9 pessewas. The standard cost for MTN to MTN call is 9 pessewas and MTN to other networks is 10 pessewas. Both GLO to GLO and GLO to other networks call cost 6 pessewas. Airtel charges are 6.5 pessewas for on-net call and 8 pessewas for off-net call.



SMS charges for the various networks are as follows: Tigo on-net SMS cost 4.1 pesewas and off-net SMS cost 4.7 pesewas. MTN on-net SMS cost 4 pesewas and off-net SMS cost 5 pesewas. Both on-net and off-net SMS for GLO cost 4 pesewas. On-net SMS cost for Airtel is 2 pesewas and off-net SMS is 5 pesewas.

From Table 4.1, SMS and call charges for MasterChat on-net is zero pesewa. This shows that using MasterChat is cost effective.

Table 4.1: Call charges for the various network operators in Ghana

(as at November, 2012)

OPERATOR	Cost per minute call (on-net)	Cost per minute Call (off-net)	Cost per sms (on-net)	Cost per sms (off-net)
<b>Tigo</b>	3p	8p	4.1p	4.7p
<b>MTN</b>	9p	10p	4p	5p
<b>GLO</b>	6p	6p	4p	4p
<b>Vodafone</b>	9p	14p	4p	4.2p
<b>Airtel</b>	6.5p	8p	2p	5p
<b>MasterChat</b>	0	-	0	-

#### 4.3.2 Research Question VI

*Is it cost effective to leverage VOIP using JMF?*

An alternate approach used by various institutions to communicate internally is the “inter-com”. According to home wyse, in 2012, the average cost to install an Intercom System ranges from \$743.56 to \$983.00 per each intercom system ([http://www.homewyse.com/services/cost\\_to\\_install\\_intercom\\_system.html](http://www.homewyse.com/services/cost_to_install_intercom_system.html).

Accessed 2/11/12 10:25 am). Since KTI has its own network with computers at each department, the installation of the proposed system, *MasterChat*, will cost no fee.



## CHAPTER FIVE

### SUMMARY, CONCLUSION, AND RECOMMENDATION

#### 5.1 SUMMARY

The purpose of this design research was to design a system that will allow people to communicate over a data network. Users using the proposed system will be able to send text, make voice and video call, and transfer file from one user to the other. Though, there are other existing systems that can perform the above mentioned tasks, those systems are internet-based which makes it impossible to use without internet connection. The proposed system runs on intranet and this makes it ideal application for organisations who want to communicate internally.

Java Programming language was used in the design of the application. JMF, a Java Media API, was used to abstract the internal implementation of media streaming. Client application for mobile devices was designed using Android SDK and eclipse IDE. Client/Server architecture was used in the design of the application. With this 3-tier architecture, users' information is stored in the database server. Address – to- location mapping is done by the main server. Many clients can connect to the main server application using required IP address and port number. The application could be used on Android mobile phones and computers with Windows operating system.

#### 5.2 CONCLUSION

The research aimed at the design of a system that allows users to send text, make voice call, make video call, and transfer file over a Local Area Network. Client software, server software and Database were created and integrated. Java Media Framework was the API used to handle media streaming. The system designed was



evaluated to be cost effective, faster, reliable and always available as compared to using inter-com system or cellular phone.

### 5.3 RECOMMENDATIONS

The targeted task was to design a system that allow users to send text, make voice call, make video call, and transfer file over a Local Area Network. The system designed has a number of advantages over the existing systems. These advantages include the following:

1. The system was evaluated to be cost effective as compared with the existing systems such as inter-com system and Cellular Phones.
2. Technical support could easily be provided locally unlike Yahoo! Messenger and Google Talk that the user needs to communicate with a bot.
3. The system is faster and always available. Unlike the internet-based systems that have intermittent connections, this system is purely intranet- based and will always be on when the intranet is available. Here, no bandwidth is needed. There is low traffic, high transmitted packets, low drop packets, high speed of transmission, and there is low packet collision.
4. The users of the proposed system have at their disposal both the client and Server applications. With systems such as Yahoo messenger, Google Talk, Facebook and Skype, users are provided with only the client application. Therefore if there is any problem with the Server application, users need to wait for it to be fixed from the developers end. With the MasterChat, the system could be easily monitored and troubleshoot is done locally.

It is therefore recommended that, within a local area network, **Master Chat** should be used instead of inter-com system, cellular phones or internet-based system such as Skype, Yahoo! Messenger and Google Talk.



## References

Calvert, K. L. and Donahoo, J. M. (2008). TCP/IP Sockets in Java: Practical Guide for Programmers, Second Edition, Morgan Kaufmann Publishers, USA.

Comer, D. (2006). Internetworking with TCP/IP: Principles, Protocols, and Architecture, Volume 1, Prentice Hall, Upper Saddle River.

DeveloperWorks, Java sockets 101: [ibm.com/developerWorks](http://ibm.com/developerWorks)

Forouzan, B. A. (2010). TCP/IP Protocol Suite, Fourth Edition, McGraw-Hill Companies, Inc., New York, USA.

Gomillion, D. and Dempster, B. (2005). Building Telephony Systems with Asterisk, Packt Publishing Ltd. Birmingham, UK.

<http://communication.howstuffworks.com/ip-telephony.htm>.

Accessed 5/10/11 9:00 am

<http://compnetworking.about.com/od/networkprotocolsip/g/udp-user-datagram-protocol.htm>. Accessed 23/01/12 11:22 PM

[http:// docs.oracle.com/javase/tutorial/sound/index.html](http://docs.oracle.com/javase/tutorial/sound/index.html)

Accessed 20/05/11 8:30 am

[http://en.wikipedia.org/wiki/Google\\_Talk](http://en.wikipedia.org/wiki/Google_Talk).

Accessed 9/7/12 8:43 pm

[http://en.wikipedia.org/wiki/User\\_Datagram\\_Protocol](http://en.wikipedia.org/wiki/User_Datagram_Protocol).

Accessed 23/01/12 11:17 pm



[http://grack.com/downloads/school/enel619.10/report/java\\_media\\_framework.html](http://grack.com/downloads/school/enel619.10/report/java_media_framework.html).

Accessed 5/7/2012 4:47 pm

[http://help.yahoo.com/tutorials/ms8/mess/im\\_setup1.html](http://help.yahoo.com/tutorials/ms8/mess/im_setup1.html)

Accessed 9/7/12 8:40 pm

<http://voip.about.com/od/voipbasics/a/ReasonsForVoIP.htm>.

Accessed 20/05/11 8:20 am

<http://www.ee.iitm.ac.in/~tgvenky/JMFBook/Tutorial.pdf>.

Accessed 6/7/12 11:10 am.

[http://www.homewyse.com/services/cost\\_to\\_install\\_intercom\\_system.html](http://www.homewyse.com/services/cost_to_install_intercom_system.html).

Accessed 2/11/12 10:25 am.

<http://www.javaworld.com/jw-04-2001/jw-0406-jmfl.html?page=2>.

Accessed 26/06/12 9:49 pm

<http://www.roseindia.net/java/javaapi/java-api.shtml>.

Accessed 28/01/12 10:53 PM

<http://www.selfseo.com/story-19378.php>.

Accessed 20/01/12 9:43 am

<http://www.techopedia.com/definition/23996/java-media-framework-jmf>.

Accessed 5/7/2012 3:24 pm

<http://www.techrepublic.com/article/the-top-five-advanced-voip-features-your-business-needs/6163858>. Accessed 20/01/12 11:12 am

<http://www2.sys-con.com/itsg/virtualcd/java/archives/0503/decarmo/index.html>.

Accessed 5/7/2012 4:33 pm

Kevin, W. (2005). Voice over IP First-Step, Cisco Press.



Knútur, B. O. (2011). Risk analysis on VoIP systems, Master's thesis.

Pawel, L. (2007). VoIP Security in Public Networks, master's thesis.

Perea, R. M. (2008). Internet Multimedia Communications Using SIP: A Modern Approach Including Java Practice, Elsevier's Science & Technology Rights Department in Oxford, UK.

Radhika, P.(2012) Multimedia conferencing using java media framework, International Conference on Computing and Control Engineering (ICCCE 2012), 12 & 13 April, 2012 [www.iccce.co.in/Papers/ICCCECS696.pdf](http://www.iccce.co.in/Papers/ICCCECS696.pdf)

Salman, A. B. and Schulzrinne, H. G. (2004). An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol, Columbia University, New York NY 10027

Tiantioukas, N. (2007). Effects of the Wireless Channel, Signal Compression and Network Architecture on Speech Quality in VoIP Networks, Master's thesis, Naval Postgraduate School, Monterey, California, USA.



## Appendix A

### CODES FOR TEXT CHAT

```

SendMessageToServer("HELO " + getUsername() + "~" +
    getPassword());
    }

    dataInputStream = new DataInputStream(is);
    /*******Send HELO To Server *****/
    StartFlag = true;
    thread = new Thread(this);
    thread.start();
    // EnableAll();
    } catch (IOException ioe) {
        System.out.println("" + ioe);
        QuitConnection(RuntimeConfig.QUIT_TYPE_NULL);
        return ioe.getMessage();
    }
    return null;
    }

    private void SendMessageToServer(String Message) {
        try {
            System.out.println("Message..." + Message);
            dataOutputStream.writeBytes(Message + "\r\n");
        } catch (IOException _IoExc) {
            System.out.println("Quiting..." + _IoExc);
            QuitConnection(RuntimeConfig.QUIT_TYPE_DEFAULT);
        }
    }

    /******* Function To Send MESS Rfc to Server *****/
    public void SendMessage(JTextArea txtMessage) {
        /*******Sending a Message To Server *****/
        SendMessageToServer("MESS " + UserRoom + "~" + getUsername() + ":" +
            txtMessage.getText());
        ClientUI.getClientUI().AddMessageToMessageObject(getUsername() + ":" +
            txtMessage.getText(), RuntimeConfig.MESSAGE_FROM_SELF);
        //SendMessageToServer("PRIV " + UserRoom + "~" + getUsername() + ":" +
            txtMessage.getText());
        //messagecanvas.AddMessageToMessageObject(getUsername() + ":" +
            txtMessage.getText(), MESSAGE_TYPE_DEFAULT);
        return serverConnectionStatus;
    }

    public void setServerConnectionStatus(String serverConnectionStatus) {
        this.serverConnectionStatus = serverConnectionStatus;
    }
}

```



## Appendix B

### CODES FOR AUDIO AND VIDEO STREAMING

```
package edu.knust.cos.mphil.masterchat.common;

import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.util.Vector;
import javax.media.CaptureDeviceInfo;
import javax.media.CaptureDeviceManager;
import javax.media.DataSink;
import javax.media.Manager;
import javax.media.MediaLocator;
import javax.media.Player;
import javax.media.Processor;
import javax.media.protocol.ContentDescriptor;
import javax.swing.JFrame;
import javax.swing.JPanel;
import org.geosant.ui.commons.Application;

/**
 *
 * @author George
 */
public class DataStreamer extends Thread {

    private String streamType;
    private Vector<CaptureDeviceInfo> deviceList;
    private CaptureDeviceInfo audioDevice;
    private CaptureDeviceInfo videoDevice;
    private Processor audioProcessor;
    private Processor videoProcessor;
    private String[] rtpUrl; //rtpUrl[0]=audio rtpUrl[1]=video
    private DataSink audioSource;
    private DataSink videoSource;
    private Player videoPlayer;
    private JPanel videoPanel;

    public DataStreamer(String streamType, String[] rtpUrl) {
        this.streamType = streamType;
        this.rtpUrl = rtpUrl;

        deviceList = CaptureDeviceManager.getDeviceList(null);
```



```

        if (deviceList.size() > 0) {
            audioDevice = deviceList.get(0);
            try {
                audioProcessor = Manager.createProcessor(audioDevice.getLocator());

                if (configureAndRealizeProcessor(audioProcessor)) {
                    System.out.println("Configuring audio...");
                    audioSource =
Manager.createDataSink(audioProcessor.getDataOutput(), new
MediaLocator(rtpUrl[0]));
                    audioSource.open();
                    audioSource.start();

                    audioProcessor.start();
                }
            } catch (Exception e) {
                System.out.println("audio device capture unsuccessful");
            }

            if (streamType.equals(ClientConstant.VIDEO_TYPE)) {
                for (CaptureDeviceInfo device : deviceList) {
                    if (device.getName().contains("vfw")) {
                        videoDevice = device;
                        break;
                    }
                }

                try {
                    videoProcessor = Manager.createProcessor(videoDevice.getLocator());
                    if (configureAndRealizeProcessor(videoProcessor)) {
                        videoSource =
Manager.createDataSink(videoProcessor.getDataOutput(), new
MediaLocator(rtpUrl[1]));
                        videoSource.open();
                        videoSource.start();
                        videoProcessor.start();

                        System.out.println("Video Processor Starting...");
                    }
                } catch (Exception e) {
                    System.out.println("video device capture unsuccessful");
                }
            }
        }
    }
}

```



```

private boolean configureAndRealizeProcessor(Processor processor) {
    try {
        processor.configure();

        while (processor.getState() != Processor.Configured) {
            System.out.println("still configuring...");
            Thread.sleep(1000);
        }

        processor.setContentDescriptor(new
ContentDescriptor(ContentDescriptor.RAW_RTP));

        processor.realize();

        while (processor.getState() != Processor.Realized) {
            System.out.println("still realizing...");
            Thread.sleep(1000);
        }

        return true;
    } catch (Exception e) {
        System.out.println("error occured realizing or configuring processor");
    }

    return false;
}

private JPanel videoComponent(Player player) {
    return createMainPanel(player);
}

@Override
public void run() {
    Player audioPlayer;

    try {
        //receive audio by default
        audioPlayer = Manager.createRealizedPlayer(new MediaLocator(rtpUrl[0]));

        //receive both audio and video
        System.out.println("Audio and Video Streaming...");
        if (streamType.equals(ClientConstant.VIDEO_TYPE)) {

```



```

        videoPlayer = Manager.createRealizedPlayer(new
            MediaLocator(rtpUrl[1]));
        videoPlayer.start();

    }

    audioPlayer.start();

    System.out.println("receiving stream");
    videoPanel = videoComponent(videoPlayer);
    } catch (Exception e) {
        System.out.println("unable to receive stream");
    }
}

private JPanel createMainPanel(Player player) {
    JPanel mainPanel = new JPanel();
    GridBagLayout gbl = new GridBagLayout();
    GridBagConstraints gbc = new GridBagConstraints();

    mainPanel.setLayout(gbl);

    boolean visualComponentExists = false;

    // if the visual component exists, add it to the newly created panel.
    if (player.getVisualComponent() != null) {
        visualComponentExists = true;
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.weightx = 1;
        gbc.weighty = 1;
        gbc.fill = GridBagConstraints.BOTH;
        mainPanel.add(player.getVisualComponent(), gbc);
    }

    // if the gain control component exists, add it to the new panel.
    if ((player.getGainControl() != null)
        && (player.getGainControl().getControlComponent() != null)) {
        gbc.gridx = 1;
        gbc.gridy = 0;
        gbc.weightx = 0;
        gbc.weighty = 1;
        gbc.gridheight = 2;
        gbc.fill = GridBagConstraints.VERTICAL;
    }
}

```



```

mainPanel.add(player.getGainControl().getControlComponent(), gbc);
    }

    // Add the control panel component if it exists (it should exists in
    // all cases.)
    if (player.getControlPanelComponent() != null) {
        gbc.gridx = 0;
        gbc.gridy = 1;
        gbc.weightx = 1;
        gbc.gridheight = 1;

        if (visualComponentExists) {
            gbc.fill = GridBagConstraints.HORIZONTAL;
            gbc.weighty = 0;
        } else {
            gbc.fill = GridBagConstraints.BOTH;
            gbc.weighty = 1;
        }

        mainPanel.add(player.getControlPanelComponent(), gbc);
    }

    JFrame jf = new JFrame();
    jf.setSize(500,500);
    Application.centerFrame(jf);
    jf.add(mainPanel);

    jf.show();

    return mainPanel;
}

public JPanel getVideoPanel() {
    return videoPanel;
}

public void setVideoPanel(JPanel videoPanel) {
    this.videoPanel = videoPanel;
}
}

```



## Appendix C

### CODES FOR FILE TRANSFER

```
private void SendFileToServer(String filePath, String Message) {  
  
    try {  
  
        Properties fileProperties = new Properties();  
  
        dataOutputStream.writeBytes(Message + "@" + filePath + "\r\n");  
  
        File myFile = new File(filePath);  
  
        // send the properties  
  
        StringWriter writer = new StringWriter();  
        fileProperties.store(writer, "");  
        writer.close();  
  
        dataOutputStream.writeUTF(writer.toString());  
  
        // send the length of the file  
  
        dataOutputStream.writeLong(myFile.length());  
  
        System.out.println("Preparing to send file to server...");  
  
        byte[] bytes = new byte[8 * 1024];  
  
        FileInputStream fis = new FileInputStream(myFile);  
  
        int len;  
  
        while ((len = fis.read(bytes)) > 0) {  
            dataOutputStream.write(bytes, 0, len);  
        }  
  
        fis.close();  
  
        dataOutputStream.flush();  
  
        System.out.println("File sent to server...");  
  
    } catch (IOException _IoExc) {  
        QuitConnection(RuntimeConfig.QUIT_TYPE_DEFAULT);  
    }  
}
```