MULTICLIENT CHATTING APPLICATION SERVER AND CLIENT USING CAsyncSOCKET

PROJECT REPORT

P-1098

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

MSc. (APPLIED SCIENCE -SOFTWARE ENGINEERING)
OF BHARATHIAR UNIVERSITY

Submitted by

DEEPA-P
Reg. no- 0037S0086

Guided by

Mr. M. Nageshwara Guptha

Department of Computer Science and Engineering
Kumaraguru College Of Technology

(Affiliated to Bharathiar University)

Coimbatore-641006

SEPTEMBER 2003

CERTIFICATE

Department of Computer Science and Engineering

Kumaraguru College of Technology

Coimbatore – 641 006

This is to certify that the project work entitled

"Multiclient chat application server and client using CAsyncsocket"

has been submitted by

Miss. DEEPA.P

(Reg.No: 0037S0086)

In partial fulfillment of the award of the degree of
Master of Science in Applied Science – Software Engineering of
Bharathiar University, Coimbatore
During the academic year 2003-2004

[illegible] [illegible]

Guide **Head of the Department**

Certified that the candidate was examined by us in the Project Work Viva Voce Examination held on 29.9.03.

[illegible] [illegible]

Internal Examiner **External Examiner**

DECLARATION

I here by declare that the project work entitled

"MULTICLIENT CHATTING APPLICATION SERVER AND CLIENT USING CAsyncSocket"

done at

ELECTRONICS RESEARCH AND DEVELOPMENT CENTRE OF INDIA TRIVANDRUM

and submitted to

KUMARAGURU COLLEGE OF TECHNOLOGY
(Affiliated to Bharathiar University)

in partial fulfillment of the requirement for the degree of MSc(Applied Science-Software Engineering)

is a report done by me during the period of study in

Kumaraguru College of Technology, Coimbatore-641006.

Under the supervision of

Mr. M. NageshwaraGuptha , B.E, Dept of CSE

Name of the Candidate	**Register Number**	**Signature of Candidate**
DEEPA.P	**0037S0086**	Deepa

Date: 29.9.03

Software Training and Development Centre
3rd Floor, Chennankara Building
Vellayambalam, Thiruvananthapuram 695 010
Tel. 0471 - 2726531, 2726586, 2726012. Fax: 0471 - 2721209
E-mail: stdc@erdcitvm.org, Website: www.erdcitvm.org

A Scientific Society of the Ministry of Communications and Information Technology, Government of India

CDAC
The Supercomputing People
CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING

(Formerly ER&DCI)
Vellayamblam,
Thiruvananthapuram 695 033, India
Tel: +91- 471-272 3333
Fax: +91- 471-272 3456. 2722230
http://www.cdacindia.com

BONAFIDE CERTIFICATE

This is to certify that the project entitled **"Multiclient Chat Application"** using **VC++** is a bonafide record of the work done at the Centre by **Ms. Deepa P,** Kumaraguru College of Technology, Coimbatore, in partial fulfillment of the requirements for the award of the **MSc (SE)** degree from **Bharathiar University.** She has been working on the project in the Centre during the period June 2003 to September 2003.

Certified further that to the best of my knowledge, the work reported herein does not form part of any other thesis on the basis of which a degree or award was conferred on an earlier occasion to this or any other candidate.

[illegible]
Prof. R Sahadevan
Head, STDC

SOFTWARE TRAINING AND DEVELOPMENT CENTRE
A SCIENTIFIC SOCIETY OF THE MINISTRY OF
CDAC
GOVT OF IND
TC 15/162-164
TRIVANDRUM
Pin-695 010

[illegible]
Vinnie John Thottan
Scientific Officer

(Project Guide)

Thiruvananthapuram
26th September 2003

● Pune : (Head Office) - Tel: +91--20- 569 4000 / 1 / 2, Fax : +91-20-569 4059 ● Bangalore : (Knowledge Park) - Tel: +91-80-534 1874, 534 1909, Fax:+91-80-524 7724, (Formerly NCST) - Tel : +91-80-852 3300, Fax : +91-80-852 2590 ● Delhi : (C-DAC) - Tel / Fax: +91-11-668 1156 / 668 1209, (Formerly ER&DCI) - Tel : +91-11-243 63815, Fax: +91-11-243 60187 ● Hyderabad : (C-DAC) - Tel: +91-40-340 1331 / 32, Fax:+91-40-340 1531 ● Mumbai : (Formerly NCST) - Tel : +91-22-262 01606, Fax : 262 32195 ● Chennai : (C-DAC) - +91-44-371 9326/27
kata: (C-DAC) - Tel / Fax: +91-33-357 3581, Fax:+91-33-357 5141, (Formerly ER&DCI) - Tel : +91-33-235 76309, Fax :+91-33-235 75141 ● Mohali: (Formerly CEDTI) - Tel: +91-172-257 050, Fax : +91-172-257
● Noida: (Formerly ER&DCI) - Tel: +91-120- 240 2551 / 60, Fax: 240 2569

ACKNOWLEDGEMENT

I express my sincere gratitude to our Principal **Dr .K. K.Padmanabhan,Ph.D., Kumaraguru College of Technology**, Coimbatore , for his constant encouragement throughout my course.

I extend my gratitude to **Dr. Thangaswamy,Ph.D.**, our beloved **H.O.D** for his constant support , encouragement and valuable internal guidance.

I express my sincere thanks to internal guide **Mr. M. Nageswara Guptha , B.E,** and **Ms .S. Devaki ,B.E.,M.S,** for the overall support that they provided during the project work.

I am greatly indebted to **Mrs. Saramma Chacko , Joint Director, ER&DCI, Trivandrum** for providing me the golden opportunity to do my project at their institution , and sparing her time with me to give all details about the institution.

I am grateful to **Mr.Vinnie John Thottan, Scientific Officer** and project guide for sparing his valuable time for the successful completion of this project.

I also take this opportunity to thank **my parents** for their constant support and encouragement.

Last but not the least I thank **GOD ALMIGHTY** for his blessings for the success of my project.

SYNOPSIS

Multiclient chatting application client and server side using CAsyncSocket is based on *Client-Server architecture*. Modern day offices are quite large and in most cases are multistoried. Employees and managers have offices at different locations in the same building. Communication among them becomes a problem.

This software is developed in ***VC++*** with ***MS-ACCESS*** support. It is simple and easy to use and avoides brain wrecking protocols or ambiguos working methods. The project is divided into four modules .The modules together serve all the communication needs of the office.

The services offered include:

- **Multiclient Chatting (Private Chat)**

  Discussions about project , recruitment of employees, purchase plans etc. needs the interaction of various executives from different departments. Multiclient chatting serves this purpose by allowing users to interact with one another and share their ideas and plan.

- **NetMeeting**

  It removes all the privacy norms. When a client sends a message to someone it is displayed on all client programs.It is an online service providing a common Interactive Platform. Files of small size can also be transffered by double clicking it.

- **Text to voice**

  Here text messages sent during private or netmeeting is converted to voice. So this is very convenient as the users need not read the message; instead he could hear the message .

- **Voice Chat**

  An *Online full duplex voice communication*. Voice Chatting is the most natural way of interaction and is a powerful means of communication.

  Collection , storage ,distribution and processing of information are a day to day activity in any enterprise. The successful assimilation and dissemination of information is the major factor deciding the success or failure of a business institution. In such a scenario the need for a cheap , reliable, simple communication software is generally felt. Though not a pioneer in the communication field, Intranet software has proved itself to be a leader.

CONTENTS PAGE No

1.1 INTRODUCTION

1.1 CURRENT STATUS OF THE PROBLEM TAKEN UP:
1.2 ABOUT THE PROJECT:
1.3 COMPANY PROFILE:

1.1.1) Existing System: -

The major problem is how to communicate efficiently. The existing system consists of only private chat .

1.1.2) Limitations: -

a) No facility of Net Meeting

b) No Text to voice conversion

c) No voice chat facility which is the most effective tool of communication.

1.1.3) Proposed System:-

Computer Networks provide a cheap and efficient means of communication.

Though the computers connected to each other through cables or wires can view each others files, directories etc. but some software has to be installed that would provide a means for the users to communicate with one another.

The proposed system is the software christened as "***Multiclient Chatting Application of client and server***".

It seeks to provide all forms of communication to cater to the needs of all executives and managers in the Office. Implemented in *VC++* with ***MS-ACCESS*** support .The system is divided into four modules:

MODULE:1

MultiClient chatting(Private Chat)

Employees require exchanging views with their colleagues on important aspects of business .Going all the way to the office or phoning them is not feasible at all times. Further another major need of corporate office is to transfer files from one office to another. Carrying files manually is not a good option. Multiclient chatting systems solve all problems. It is an online communication system. A server program always runs in the central system. The server has the list of all clients currently logged into chatroom. A user can join chat room by running the client program with the specified servers IP address and logging in by a valid user name and password.

Module:2

Netmeeting

Top business decisions require the interaction of a group of people. Each person may be a specialist in his field. Convening a meeting of the group may not prove economical in all situations .An online interactive system is the need.

NetMeeting system provides an online interactive communication method by which users can exchange views with one another in a common chat room. All views are displayed in each of client program. A server program always runs in the central system and users connect to it as and when required by running client program on their system.

Files of small size can be transsffered, by choosing the file name and double clicking it.

Module :3

Text To Voice

This is a very special feature of this software. Here messages sent could be heard. So the person need not take the trouble to read it.

Module: 4

VoiceChatting

In some situations talking to one another is more effective than sending textual messages.Without using phone or mobile cell phones which can be easily tapped,the personnals require to transfer sensitive information. Voice chatting system caters to this need. It is an online communication system that allows users to talk to each other using a headset. A server program runs in the çentral system and users can communicate with each other by running client programs on their system.

The proposed system with its four modules meets all the communication needs of a Business firm. It removes the communication barrier, which had been a major deterrent to the growth of the organization.

Another feature is we can choose the colour of the font, size and style while sending messages.

1.2.1 About the Project

Modern day offices are quite large and in most cases are multi storeyed. Employees and managers have offices at different locations in same building. Communication among them becomes a problem. Conventional communication leads to wastage of time and is quite unreliable. Corporate offices need some efficient method of communication. Intranet communication software provides the answers to all these problems. This software is developed in **VC++** with **MS-Access** support. It allows for general purpose communication among computers in a network.. The project is divided into four modules.

MODULE :1

MULTICLIENT CHATTING(PRIVATE CHAT)

Online communication is a primary need in any modern office .A VC++ program with socket . It needs a server program, which runs full time, and various client programs that may or may not run according to the needs of the user. The server program first goes online with its port address. It is always in the 'Listen' mode. Listening to the network for client connection is the major function of the server. Client programs are executed by users at different terminals at different location in a building or in an office. The client programs gives the IP address of the server and goes into the connect mode. When successful connection is established the user can log in by a valid password and user name and see all people who are logged into chatting. The user selects any one of the people in the chat room and sends messages to him. Multiclient chatting ensures privacy by allowing a member to select another member and transmit private messages.

Another important utility of multiclient chatting is its ability to transfer files from one client to another. A client can select another client from the displayed list of current users and send files to him. This eliminates the need to copy a file from one system and transfer it to another system..

MODULE:2

NetMeeting

NetMeeting requires a central server program running in a 'listen' mode. Client programs run in a connect mode.. All Messages are directed from the server to the clients. A list of all the clients currently logged in is displayed in the server as well as in all client programs.

MODULE: 3

Text to voice:

In this module whatever typed and sent will be converted to voice when we press the speak button.

MODULE :4

Voice Chatting

Voice chatting is an online program which connect clients into a common server .A server program runs in 'Listen mode'. Client programs run in 'Connect Mode'. When successful connections are established, the name of clients automatically appears in the list of users . A client can select another client and talk to him using headphones and mike sets. A successful connection would involve client *A* sending a message to client *B* requesting for voice chat. On acceptance, client *B* sends an acknowledgement. Then client *A* comes to know that no other client is currently chatting to *B* and connects to *B*. Voice chatting is the most natural way of interaction and is a powerful means of communication.

1.3 ORGANIZATION PROFILE

INTRODUCTION TO CDAC (FORMER ER&DCI)

Electronic Research and Development Centre (ER&DCI) is a scientific society which was originally started in 1974 under the Department of Electronics, Govt. of India. It is a pioneer organization engaged in application oriented research and development in Electronics and Information Technology, providing services to practically all government departments as well as private organizations.

The Centre has developed strong expertise over the years in the area of computer communication, instrumentation and control, artificial intelligence, Power Electronics etc. The resources available in ER&DCI include an IBM- ES-9000 MAINFRAME SYSTEM, VLSI DESIGN CENTRE and advanced CAD facilities.

The Centre's latest venture is the Software Training and Development Centre (STDC) ,addressing to the needs of software development – market on mainframe platform. This Centre is equipped with the advanced IBM- ES-9000 MAINFRAME COMPUTER SYSTEM and related softwares. It provides training on MVS, DB2, CICS and JCL position to-address the major commercial segment of mainframe users.

STDC undertakes software development on the IBM mainframe platform. It has a team of dedicated and well trained software professionals.

ORGANIZATIONAL SET UP

There are six ER&DCI's located in the country. They are at

1.CALCUTTA

2.MOHALI

3.ERNAKULAM

4.LUCKNOW

5.THIRUVNANTHAPURAM

6.DELHI

2) SOFTWARE REQUIREMENT SPECIFICATION

2.1 AIM OF THE PROJECT:

The aim of the project is to develop a software for chatting with facilities **Voice chat** and **Text to voice** .

2.2 REQUIREMENT ANALYSIS:

It is the initial point of software development activity. The first step is to understand the user requirements within the frame work of the objective and the environment in which the system is being installed. Consideration is given to the user resources and finance. After requirement analysis a clear picture of the chatting system became visible. Different issues were considered:

The major requirements that became visible are:

a) There has to be a server part

b) There has to be a client part.

c) Server must be able to accept connection request from the clients.

d) Processing of connection request has to be done simultaneously as processing other client's request.

e) Each and every connection has to be uniquely identified.

2.3 FUNCTIONAL REQUIREMENTS

All the details required to design the software and eventually implement the software are to be specified. One of the major design goals is to develop software that is easily modifiable.

2.4 INPUT AND OUTPUT REQUIREMENTS

The software design is to be performed using real time requirements like better functionality, a much better functionality , a much better interface, easily accessible features.

2.5 SYSTEM REQUIREMENTS

HARDWARE SPECIFICATION

- Processor -Intel Pentium 266,Celeron300 or above
- Memory - 16MB SDRAM
- Hard disk -2.2 GB or above
- Monitor-14 inch VGA Monochrome
- Pointing device-Logitech Mouse
- Microphone
- Speaker

SOFTWARE SPECIFICATION

- Programming Language-Visual C++ 6.0
- Operating System- Windows 98,Windows NT Server
- Back End Tool – Ms-Access

2.7 SOFTWARE DESCRIPTION

FEATURES OF MICROSOFT VC++

The core of Visual C++ environment is built around three elements. The C/C++ compiler and linker, the developer studio and the Microsoft Foundation Class Library.

Desired programming language features are:

Strong Typing:

This property greatly supports the software development effort. Programming errors related to number of parameters, parameters types, and module interfaces are detected during the design and implementation phases rather than at runtime.

Encapsulation:

It is highly desirable that the programming language supports information hiding with separate specification and implementation parts. This can provide a loosely coupled design.

Incremental compilation:

This is a great benefit in the development of large systems where portions of the system are created and implemented in a piecemeal fashion.

Generality:

Parameter typed templates for classes and operation support a high degree of reusability .The generic elements are presorted templates with formal parameters that are instantiated with actual parameters to create instance of modules to be compiled, and later linked and executed.

2.8 SOFTWARE DESCRIPTION

Message Passing:

It is highly desirable that the language supports bi-directional message between modules. This provides for loosely coupled modules and flexible designs. It is also desirable to be able to pass signals between modules without actually passing any data elements.

PolyMorphism:

This is usally used along with inheritance. Program elements should be able to refer to an object name that may belong to different classes that are derived from a super class Operations can be used to manipulate objects of different types at execution time without reprogramming these methods.

Support for Internet Development:

Visual 6.0 provides strong support for client-side development. In fact MFC App Wizard now includes support for Internet Explorer 4.0 HTML/DHTML functions. MFC supports new CHtml View class together with extensive support for Dynamic HTML development and web based development.

Dialog Boxes:

VC++ supports dialog based application dialog box is a special type of window designed to obtain input from the user. VC++ makes creation and use of controls in a dialog box much simpler because it requires no calculation of position and size of controls in a window.

ABOUT BACKEND:ACCESS

Access is a Relational Database Management System that can be used to store and Manipulate large amount of information. Because its tools are user friendly and since it is a powerful development environment, Access is equally appropriate for novices and Professionals.

Access is the best tool for building an application due to the following reasons: -

1.Windows Interface
2.Mouse Operators
3.Color and Graphics
4.forms
5.one file
6.wizards
7.Easy testing
8.SQL

Access stores all the various parts: tables, indices, reports, macros and Access Basic program modules into single disk file with extension mdb. When mdb file is opened all parts contained in the database are listed in a window called database window. The Database Window is a collection of six units: tables, queries, forms, reports, macro sheet and basic modules. In Access a collection includes all the parts of a specific type. The individual parts in a collection are called objects. Using objects and collection ,Access creates a single frame work that can be used to define the relationship between any of the parts of the system.

- DBENGINE: It is a part of Acess that controls all basic database operations in moving the storage and retrieval of data.
- Workspace: The DBEngine controls a collection of one or more workspaces.

3.SYSTEM DESIGN

3.1) Input design

3.2) Output design

The input design is the link between the information system and the user. It comprises the developing specification and those steps which are necessary to put transaction data into a usable form for processing data entry. The design of input focuses on controlling the amount of input required,controlled errors,avoiding delay, avoiding extra steps and keeping the process simple.

The system needs the data regarding the users for file transfer chatting etc.The following things are considered .

- What data should be given as input
- How the data should be arranged or coded.
- The dialogue to guide the operating personnal in providing input.
- Methods for preparing input validation and steps to follow when error occur.

MODULE :1

Multiclient Chatting:

IP address of the server is entered in edit box. Then a dialog box appears where the client has to enter a valid username and password. If he forgets password, there is provision for password lookup.

Messages are sent through edit box. File Transfer is done by clicking the menu bar `send file'. Immediately the open dialog box opens. Files can be selected and sent.

MODULE :2

NetMeeting:

IP address of server is entered in edit box to get connected and user should enter a valid password and username to log in. .List of current users are shown in the list box.

Messages are sent through edit box. Here Provision to change font , color, size, style is also included.

MODULE :3

Text to voice

On button press the text messages sent will be converted to voice.Microsoft Speech SDK helps in this conversion. So it should be installed to run this application.

MODULE 4:

Voice chatting :

IP address of the server is entered in the edit box. When the voice is received blinking blue dashes in progress shows it.

3.2 OUTPUT DESIGN

Computer output is the most important and direct information source to the user. Output design is a process that involves designing necessary output in the form of report that should be given to the users according to the requirements. Efficient, intelligible output design should improve the systems relationship with the user and help in decision making. Since the reports are directing reffered by the management for taking decision and to draw conclusions they must be designed with utmost care and details in the report must be simple ,descriptive and clear to the user.So while designing output the following things are to be considered.

- Determine what information to present.
- Arrange the presentation of information in an acceptable format.
- Decide how to distribute the output to intended receipts.

Depending on the nature and future use of output required, they can be displayed on the monitor for immediate need and for obtaining the hardcopy.

DATABASE DESIGN

Database design is a crucial factor in the performance of the system in terms of system Timings and in case with which the system can be maintained and modified. The organization of data in the database aims to achieve three major activities- data integration, data integrity and data independence.

TABLES

Each client will have a table of its own..

1.AUTHENTICATION Varchar(2)

Field Name	Data Type	Description
Username	Varchar(2)	User name
Password	Varchar(2)	Pass word

2.CLIENTS

Field Name	DataType	Description
Username	Varchar(2)	User name
Password	Varchar(2)	Pass word
FName	Varchar(2)	First Name
LName	Varchar(2)	Last Name
Desig	Varchar(2)	Designation
Phno	Varchar(2)	Phone no
Gender	Varchar(2)	Gender
Question	Varchar(2)	Question
Answer	Varchar(2)	Answer

4. SYSTEM STUDY

System Analysis is for finding out what happens in the existing system, deciding on what changes and new features are required and defining excatly what the proposed system must be. The process of system analysis is largely concerned with determining, developing and agreeing to the user's requirements. It provides prime oppurtunity to communicate well with the user's requirements. It provides prime oppurtunity to Communicate well with the user and conceive a joint understanding of what a system should be doing, together with a view of the relative importance of the system facilities using interactive techniques.

4.1 INTIAL INVESTIGATION

Initial investigation is the activity that determines whether the user's requisition is valid and feasible. The first step in initial investigation is the problem definition. It includes the identification of the problem to be solved for the task to be accomplished and the system goals to be achieved.

4.2 DETAILED STUDY AND ANALYSIS

Detailed system study is a very critical activity while developing application software. This phase involves detailed-study of the existing system and interacting with users to determine their requirements and specification using certain techniques. After the Feasibility study, a detailed analysis of the exsisting system and new system is to be done. It involves deep investigation into the existing system for collecting all data carriers, forms, and records are thoroughly studied and limitations and insufficiencies are stated. For overcoming these limitations and insufficiencies a new system is proposed so that nature of existing data manipulation may increase.

These features are incorporated into a candidate system to produce the necessary improvement. The improvement must include faster information retrieval and processing elimination of human errors; provide better friendly services with the operator etc.

A cost/benefit analysis is also put forward to justify the system change.

4.3 FEASIBILITY STUDY

During system analysis a feasibility study of the proposed system is carried out to see whether it is beneficial to the organization.

The integration unit is currently manuel; The results of the feasibility study are given below:

4.4 TECHNICAL FEASIBILITY STUDY

It is a study of resource availabilty that may affect the availability to achieve an acceptable system. It is essential that the process of analysis and definition be conducted in parallel with an assessment of technical feasibility.

Intranet communication requires a network of computers. Computers must have VC++, MS-Access, Windows 98 installed in it. Already a network exists in the firm and Computers have the required software. Hence the project is ***technically feasible.***

4.5 ECONOMIC FEASIBILITY STUDY

The computers with VC++,Ms-Access ,windows 98 required by project is already present in the firm. So only a small investment is required compared to the large returns that the project would bring. Hence the project is ***economically feasible***.

4.6 FUNCTIONAL FEASIBILITY

The project does not involve complex protocols; It has a simple working style. The extensive use of GUI proves its user –friendly nature. A basic knowledge of computers is all that is required. Though knowledge of VC++, MS-Access, is required for the system maintenance personnal but it is not mandatory for the user. The project ***is functionally Feasible.***

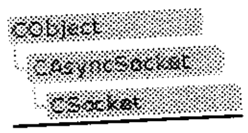
5 DETAILED DESIGN

5.1) BUILDING THE SOFTWARE

5.2)SOCKET PROGRAMMING

The major classes and functions of VC++ used by this project are as follows:

CSocket Class:



At run time the server application usually starts first in order to be ready and "listening" when the client application seeks a connection. If the server is not ready when the client tries to connect, you typically require the user application to try connecting again later. To set up communication between a server and client socket for a CSocket client object, you should normally use the default parameters to Create, unless you need a datagram socket .For a CSocket you must specify a port in the Create call.

CSocket ::Create()

Call this member function after constructing a socket object to create the Window socket and attach it. Create then binds the socket to specified address.

Carchieve doesn't work with datagram sockets. Because they are unreliable they aren't Compatible with serialization via an archive. You expect a serialization operation to complete reliably in sequence. If you try to use CSocket with a CArchive object for a datagram ,an MFC assertion fails.

CAsyncSocket

- **CAsyncSocket::Connect()**

If the socket is a client ,call Connect to connect the socket object to a server socket.

BOOL Connect (LPCTSTR lpszHostAddress, **UINT** nHostPort);

Return Value

Nonzero if the function is successful; otherwise 0,and a specific error code can be retrieved by calling GetLastError. If this indicates an error code of WSAEWOULDBLOCK, and your application is using the over-ridable callbacks, your application will receive an 'On Connect' message when the connect operation is complete.

- **CAsyncSocket::Listen()**

If the socket is a server, call CAsyncSocket:: Listen to begin listening for connects attempts from a client.

BOOL Listen(int nConnectionBacklog=5);

Return Value

Nonzero if the function is successful; otherwise 0,and a specific error code can be retrieved by calling Get Last Error.

- **CAsyncSocket::Accept()**

Upon receiving a connection request accept it by calling CAsyncSocket::Accept.The Accept member function takes a reference to a new empty CSocket object as its parameter. You must construct this object before you call Accept. Keep in mind that if this socket object goes out of scope ,the connection closes.

Virtual BOOL Accept(CAsyncSocket&rConnectedSocket,SOCKADDR* LpSockAddr=NULL,int *lpSockAddrLen=NULL);

Return Value

Nonzero if the function is successful; otherwise 0,and a specific error code can be retrieved by calling GetLastError.

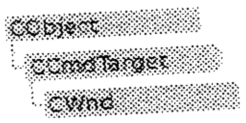
CArchive object moves data in one direction only; either for receiving or sending .In some cases we will use two CArchive objects, one for sending data the other for receiving acknowledgements.

After accepting a connection and setting up the archive you can perform such tasks as validating passwords.

Class Cstring

Cstring does not have a base class. A Cstring object consists of a variable-length Sequence of characters. Cstring provides functions and operators using syntax similar to that of Basic.Concantenation and comparison operators together with simplified memory management, make Cstring objects easier to use than ordinary character arrays.

Class CWnd



The CWnd class provides the base functionality of all window classes in the Microsoft Foundation Class Library.A CWnd object is created or destroyed by the CWnd constructor and destructor.The Windows window on the other hand,is a data structure internal to Windows that is created by a Create member function and destroyed by the CWnd virtual destructor.The CWnd class also lets you create a Windows child window for your application.

- **CWnd::OnCreate()**

Return Value

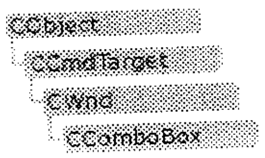
On Create must return 0 to continue the creation of the CWnd object. If the application returns –1 the window will be destroyed. The framework calls this member function when an application requests that the Windows window be created by calling the Create or CreateEx member function.The CWnd object receives this call after the window is created but before it becomes visible.

- **CWnd::Show Window()**

Return Value

Nonzero if the winbdow was previously visible;0if the CWnd was previously hidden.It sets the visibility state of the window.

Class CComboBox



The CComboBox class provides the functionality of a Windows combo box. A combo box consists of a listbox combined with either a static control or edit control.The list-box portion of the control may be displayed at all times or may only drop when the user selects the drop-down arrow next to control.

- **CComboBox::GetLBText()**

Return Value: The length of string excluding the terminating null character .Get a string from the listbox of a combo box.The second form of this member function fills a Cstring object with the items text.

- **CComboBox::GetItemDAta()**

Return Value: The 32-bit value associated with the item, or CB_ERR if an error occurs.

5.2 SOCKET PROGRAMMING

MAKING A CONNECTION

Once we create a socket we are ready to open a connection with it.Three steps go along with a single connection.Two of these steps take place on the server ,the application listening for the connection.and the third step take place on the client, the one making the call.

For the client opening the connection is a simple method of calling the connect method,Computer nameor networkaddress and the port of application are the parameters to be passed.

Once the connection is made an event is triggered to let the application know that the connection is made. For the server the application must tell the socket to listen for incoming connections by calling the listen method.The listen method takes only a single argument. This parameter specifies the no:of pending connections that can be queued.waiting for the connection to be completed.

When another application is trying to connect to the listening application, an event is triggered to let the application know that connection request is there.. The listening application must accept the connection request by calling the accept method.This method uses second Csocket variable which is connected to the other application.Once a socket is placed into the listen mode it stays in listen mode.Whenever connection requests are received the listening socket creates another socket which is connected to the other application.This second socket should not have the Create method called for it because the Accept method creates socket.

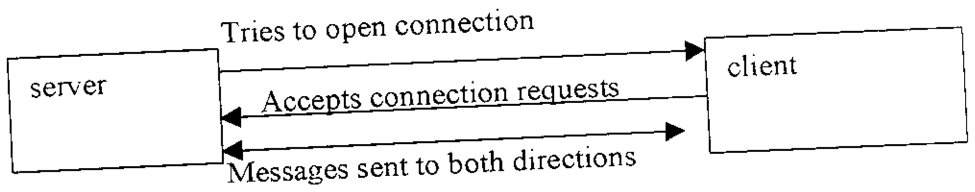
SENDING AND RECEIVING MESSAGES

To send a message to a socket connection we use the Send method. This method requires two parameters and has a third ,optional parameter that can be used to control how the message is sent.

If the message is in a Cstring variable we can use the LPSTR operation to pass the Cstring as the buffer.The second parameter is a length of the buffer.

When the data is available to be received from the other application the event is triggered on receiving application.To get the message ,the Receive method must be called.This method takes the same parameter as the send method. The first parameter is a pointer to a buffer into which the messages may be copied .The second parameter is the size of the buffer.

The basic socket connection:



CLOSING THE CONNECTION

Once all the application have finished communicating with each other the communication can be closed by calling the close method.

SOCKET EVENTS

The primary reason that we create our own descendant class of CasyncSocket is to capture the events that are triggered when the messages are received.All the functions of CASynSocket use same definitions . They are declared as protected.The functions have a single parameter.

Some of the **functions** are:

OnAccept->This function is called on a listening socket to signal that a connection request from another application is waiting to be accepted.

OnClose-> This function is called on a socket to signal that the application on the other end of the connection has closed its socket.

OnConnect-> This function is called on a socket to signal that the connection with another application has been completed and that the application can now send and receive messages through the socket.

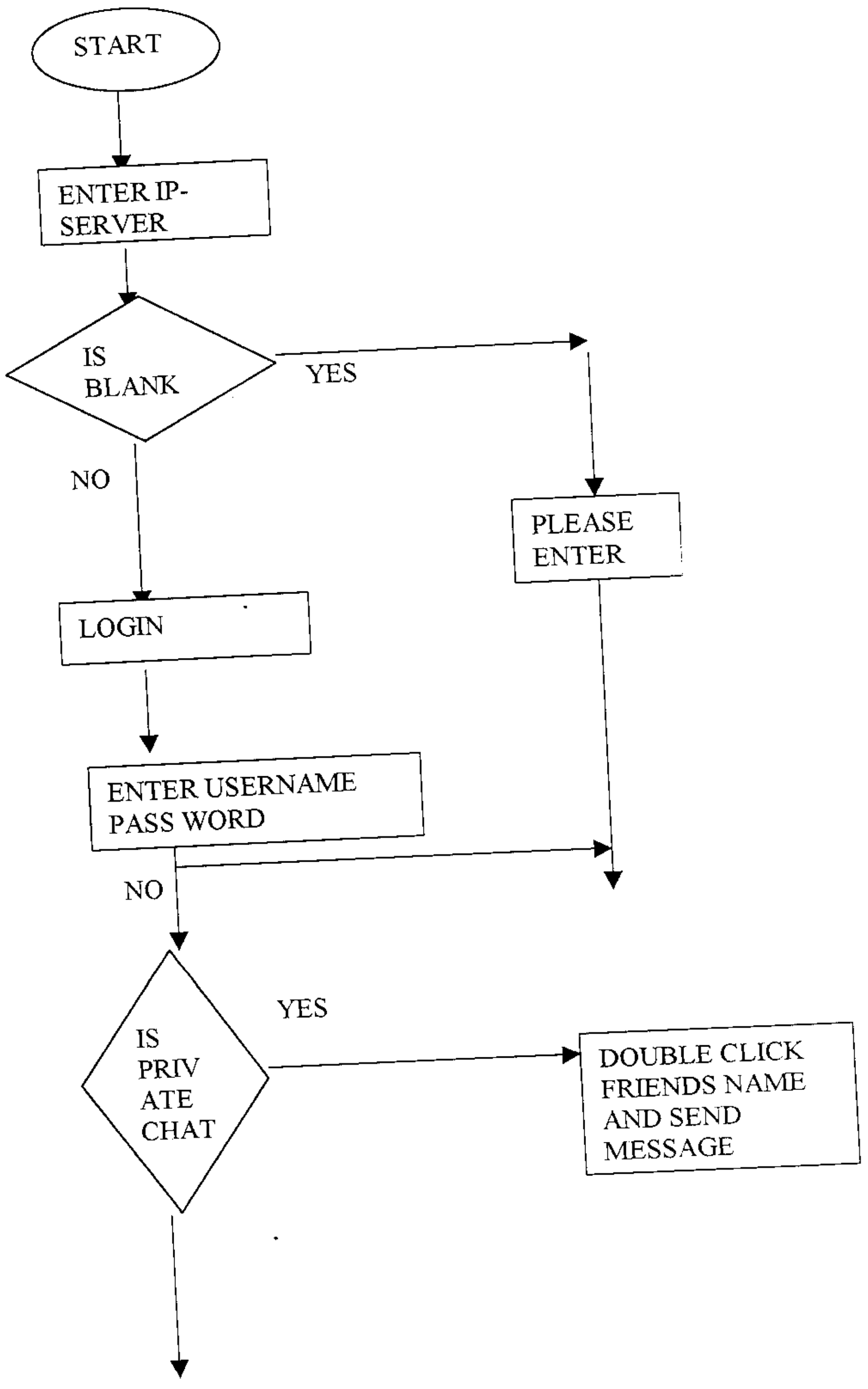
OnReceive->This function is called to signal that the data has been received through the socket connection and that the data is ready to be retrieved by calling the receive function.

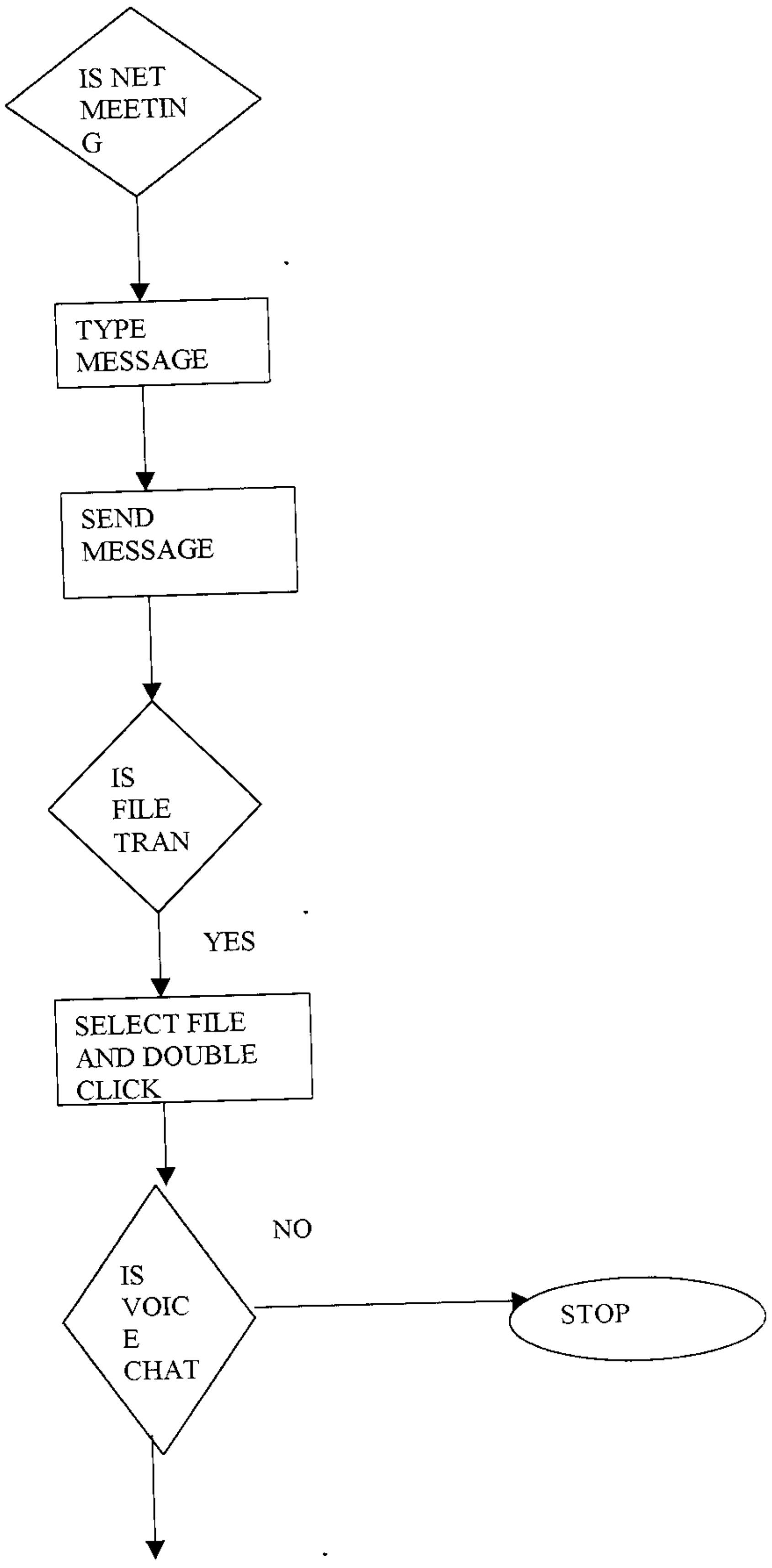
OnSend->This function is called to signal that the socket is ready and available for sending data.This function is called right after the connection hs been completed.

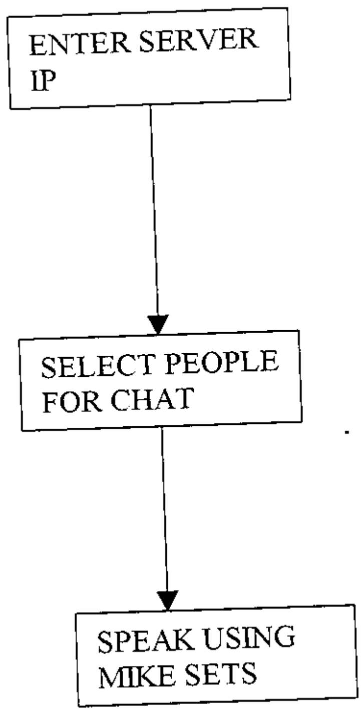
DETECTING ERRORS

Whenever any of theCAsyncSocket member function return an error,either FALSE for the most functions or SOCKET_ERROR on the Send and Receive function,the GetLastError method can be used to get the error code.This function returns only error codes.All the WinSock error codes are defined with constants,so that the use of the constants ,so that the constants in the code determines error message to display for the user.

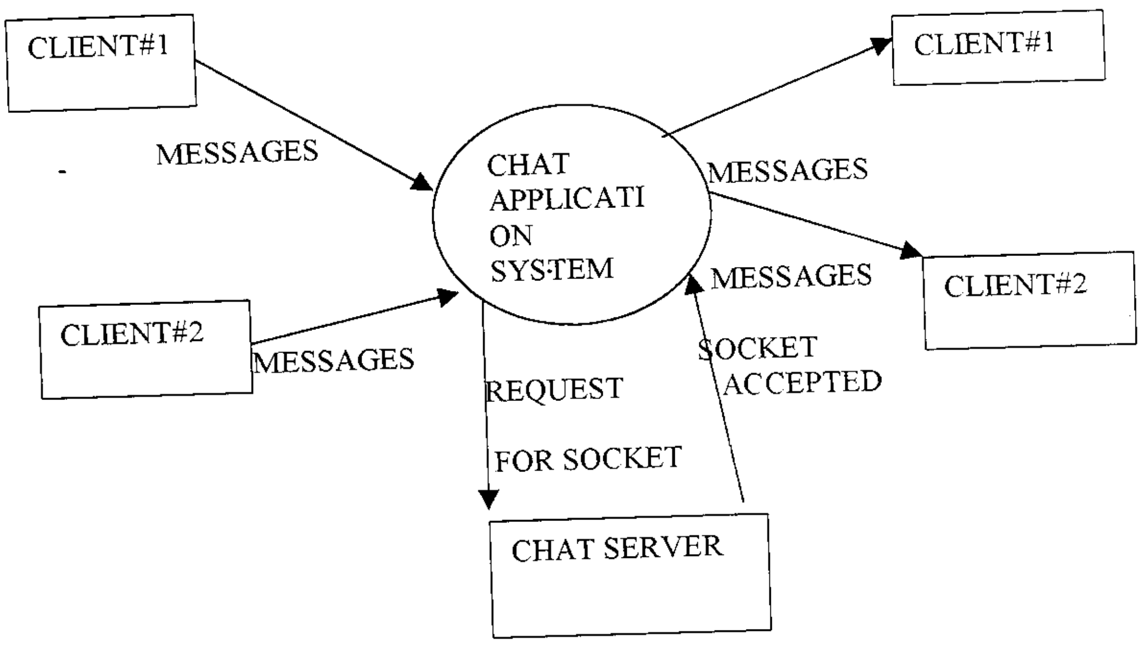
6.FLOW CHART


START
ENTER IP-SERVER
IS BLANK
YES
NO
PLEASE ENTER
LOGIN
ENTER USERNAME PASS WORD
NO
YES
IS PRIV ATE CHAT
DOUBLE CLICK FRIENDS NAME AND SEND MESSAGE


IS NET MEETIN G
TYPE MESSAGE
SEND MESSAGE
IS FILE TRAN
YES
SELECT FILE AND DOUBLE CLICK
NO
IS VOIC E CHAT
STOP

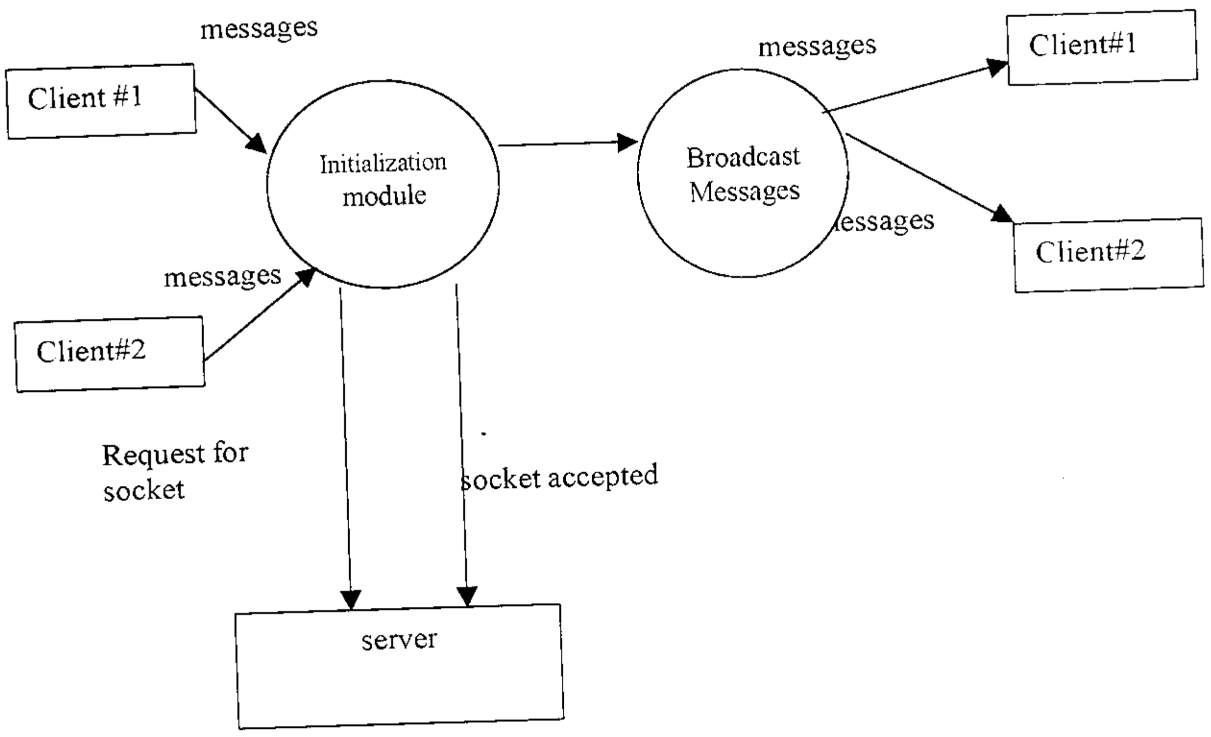

ENTER SERVER IP
SELECT PEOPLE FOR CHAT
SPEAK USING MIKE SETS

7 CONTEXT DIAGRAM & DFD (DATAFLOW DIAGRAM)



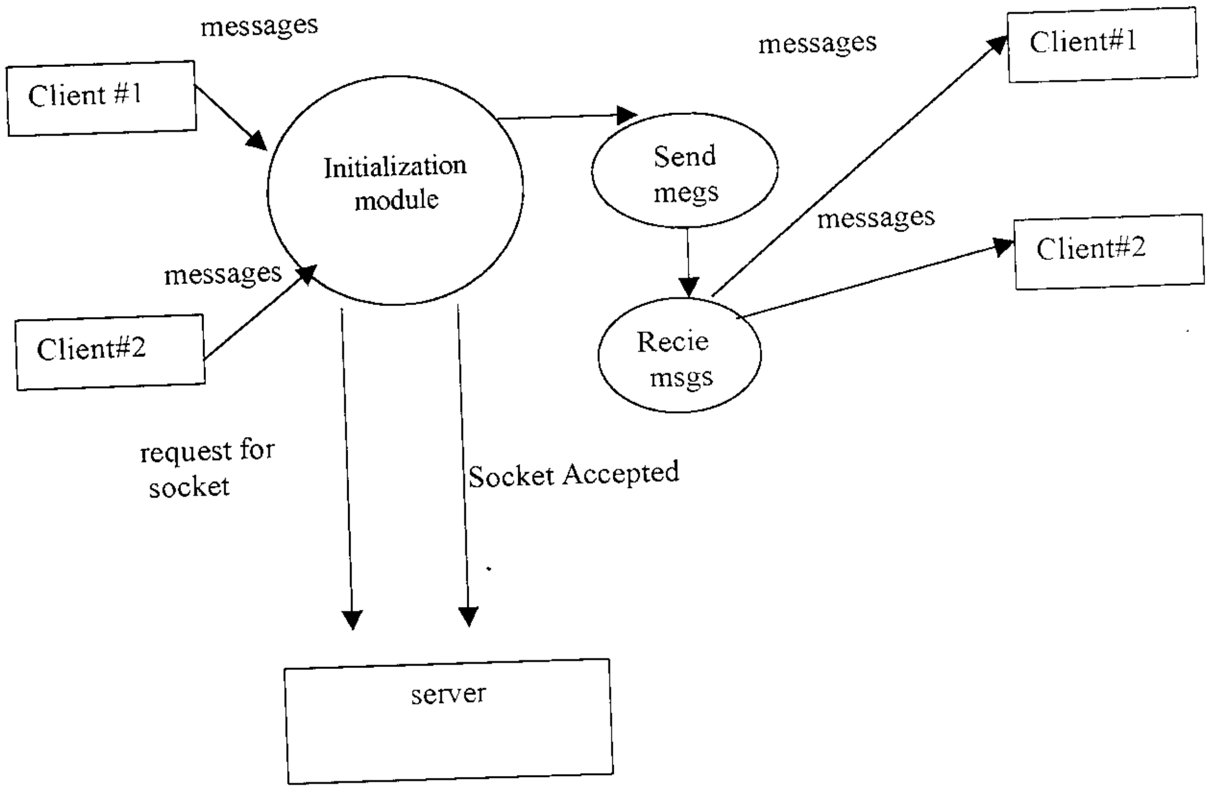
FIRST LEVEL DFD

PRIVATE CHAT



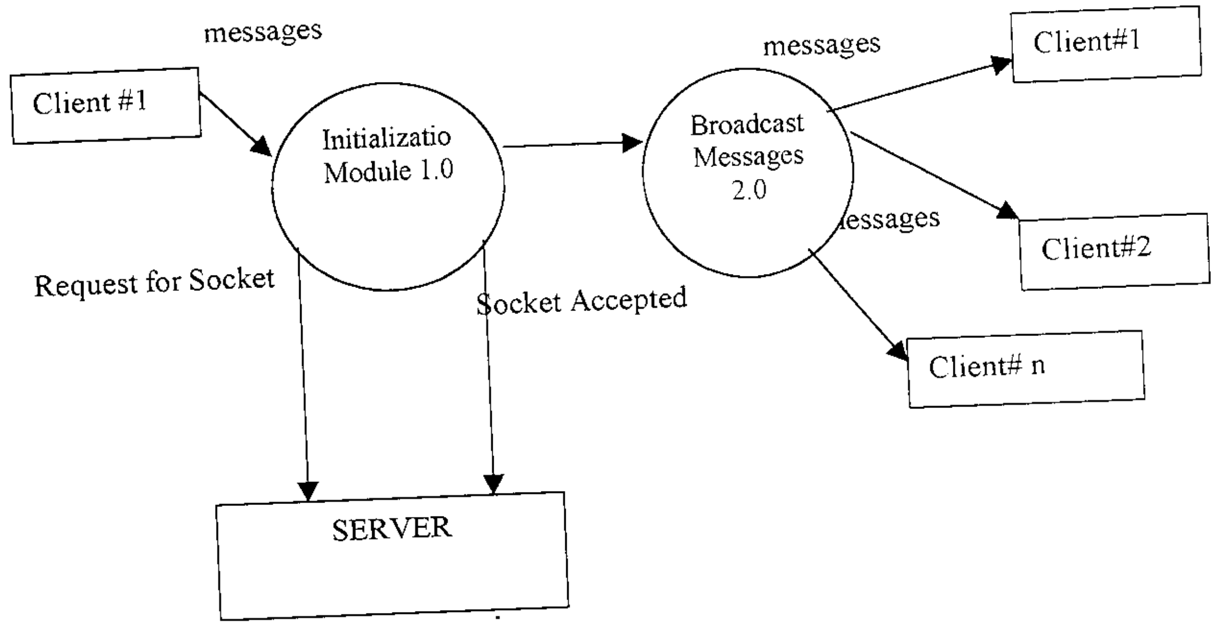
SECOND LEVEL DFD

PRIVATE CHAT



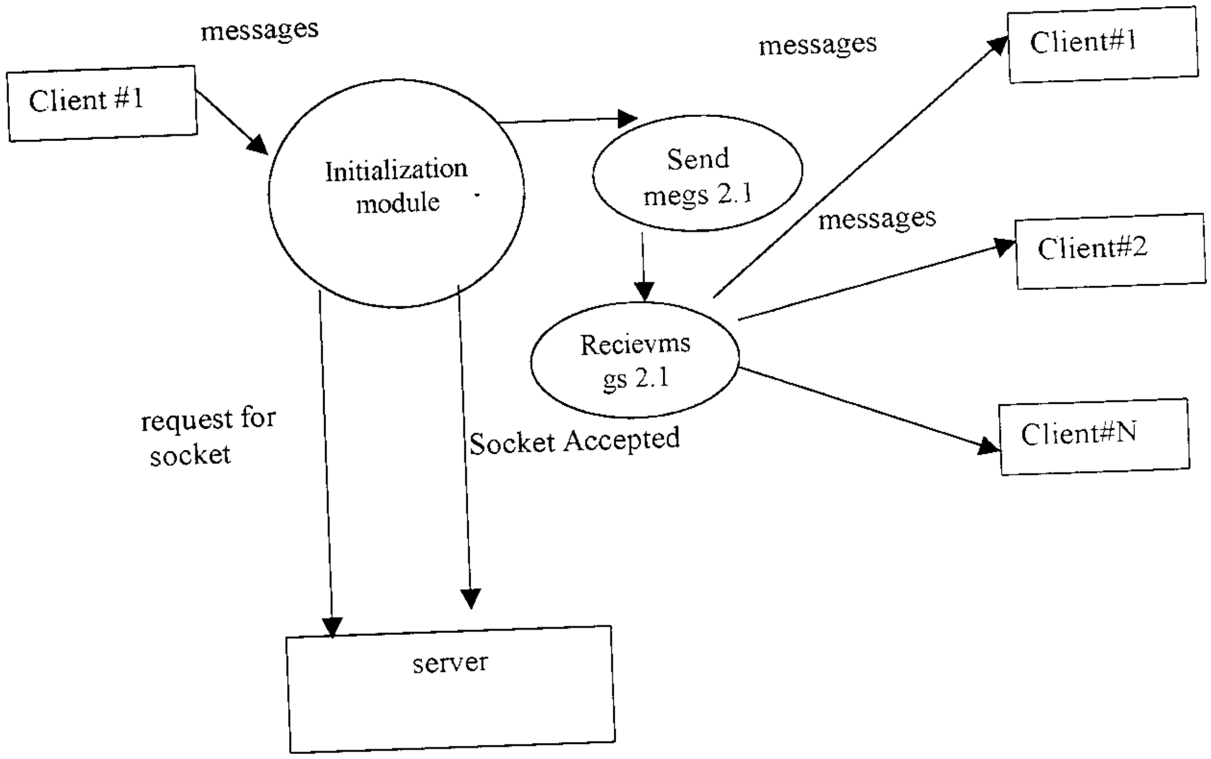
FIRST LEVEL DFD

NET MEETING



SECOND LEVEL DFD

NETMEETING



8. IMPLEMENTATION PLAN

The implentation plan include a description of all the activities that must occur to implement the new system and to put it into operation. It identifies the personnal responsible for the activities and prepares a time chart for implementing the system. The implementation plan consists of the following steps.

- List all files required for implementation
- Identify all data required to build new files during the implementation.
- List all new documents and procedures that go into the new system.

The implementation plan should anticipate possible problems and must be able to deal with them. The usual problems may be missing documents, mixed datd formats between current files, error in data translation, missing data etc.

6.1 EDUCATION AND TRAINING

Implementation of the proposed system includes the training of system operators. Training the system operators include not only instruction in how to use the equipment,but also in how to diagnose malfunctions and in what steps to take when they occur. So proper training should be provided to the system operators. No training is complete without familiarizing users with simple system maintenance. Since the proposed system is developed in GUI, training will be comparatively easy than systems developed in a non-GUI. There are different types of training. We can select off-site to give depth knowledge to the system operators.

Training must ensure that the person can handle all possible operations. Training must also include data entry personnal. They must also be given training for the Installation of new hardware, terminals, how to power the system, how to power it down, how to detect malfunctions, how to solve the problems etc. The operators must also be provided with the knowledge of trouble shooting which involves the determination of the cause of the problem.

6.2 POST IMPLEMENTATION REVIEW

After the system is implemented a review should be conducted to determine whether the system is meeting expectation and where improvements are needed.System quality ,user confidence and operating systems statistics are accessed through such technique event logging, impact evaluation and attitude surveys. The review not only assesses how well the proposed system is designed and implemented, but also is valuable source of information that can be applied to a critical evaluation of the system.

The reviews are conducted by the operating personnals as well as the software Developers in order to determine how well the system is working, How it has been accepted and whether adjustments are required. The system can be considered successfully only if information system has met its objectivies.

The following are the issues to be considered in the evaluation of the system:

- The change in the cost of operation after the installation of the computerised system.
- The basic change that has been effected after the introduction of the system.
- The improvement in the accuracy of the computation.
- The acceptance of the new system by the staff and the convenience it brought to them.
- The change in the effectiveness caused by the implementation of the new system.

9)Testing & Validation

System testing is a critical aspect of Software Quality Assurance and represents the Ultimate review of specification, design and coding. Testing is a process of executing a program with the intent of finding an error. A good test is one that has a probability of finding an as yet undiscovered error. The purpose of testing is to identify and correct bugs in the developed system. Nothing is complete without testing. Testing is the vital to the system.

In the **code testing** the logic of the developed system is tested. For this every module of the program is executed to find an error. To perform specification test, the examination of the specification starting what the program should do and how it should perform under various conditions.

Unit testing focuses first on the modules in the proposed system to locate errors. This enables to detect errors in the coding and logic that are contained within that module alone. Those resulting from the interaction between modules are intially avoided. In unit testing each module has to be checked seperately.

System testing doesnot test the software as a whole, but rather than integration of each module in the system. The primary concern is the compatibility of individual modules. One has to find areas where modules have been designed with different specification of data lenghts, type, and data element name.

Testing and validation are the most important steps after the implementation of the developed system. The system testing is performed to ensure that there are no errors in the implemented system. The software must be executed several times in order to find out errors in the different modules of the system.

Validation refers to the process of using the new software for the developed system in live environment i.e. new software inside in order to find out the error. The Validation phase reveals the failure and the bugs in the developed system. It will become to know about the practical difficulties the system faces when operated in the new environment. By testing the code of the implemented software, the logic of the program can be examined. A Specification test is conducted to check whether the specifications stating the program is performing under various conditions. Apart from these tests, there are some special tests conducted which are given below:

1) **Peak Load Tests**: - This determines whether the new system will handle the volume of activities when the system is at the peak of its processing demand. The test has revealed that the new software for the agency is capable of handling the demands at the peak time.

2) **Storage Testing**: - This determines the capacity of the new system to store transaction data on a disk or on other files. The proposed software has the required

storage space available, because of the use of a number of hard disks.

3) **Performance Time Testing**: - This test determines the length of the time used by the system to process the data.

10) CONCLUSION& FUTURE ENHANCEMENTS

'MULTICLIENT CHATTING APPLICATION CLIENT AND SERVER USING SOCKET CLASS' is developed for a homogeneous network connecting a large number of computers. The designers developed it keeping the idea of corporate office in mind, where a number of machines are connected. It thus provides all the communication required in an office environment. However the software would not work in an environment having a large number of heterogeneous networks. Protocols and code can be added as future enhancement, so that computers on different networks can communicate with each other. Multiclient chatting provides transfer of textual files (small) only. Transfer of bitmaps, mpegs, jpegs and other files can be included to make it versatile. The simplicity and easy coding allows for quick modification. Extending the functionality of the communication software from Intranet to Wide Area Network, Metropolitian Area Network and even Internet cannot be ruled out.

The software has the potentiality to grow and develop into a major communication tool.

BIBLIOGRAPHY

1. David J. Kruglinski, George Shepherd, Scott Wingo
   "Programming Microsoft Visual C++"
   Microsoft Press; Fifth Edition, 1997

2. Richard C. Leinecker and Archer Tom
   "Visual C++ Programming Bible"
   IDG Books India (P) Ltd; Fifth Edition, 1998

3. Roger S. Pressman
   "Software Engineering and Application"
   Tata Mc-GrawHill; Fourth Edition, 1998

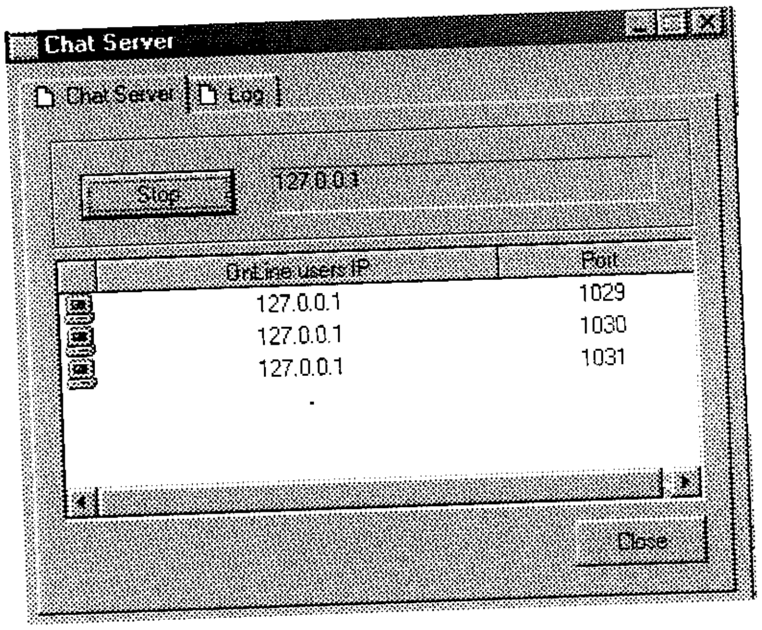
WEBSITES

www.msdn.microsoft.com

www.programmersheaven.com

www.codeguru.com

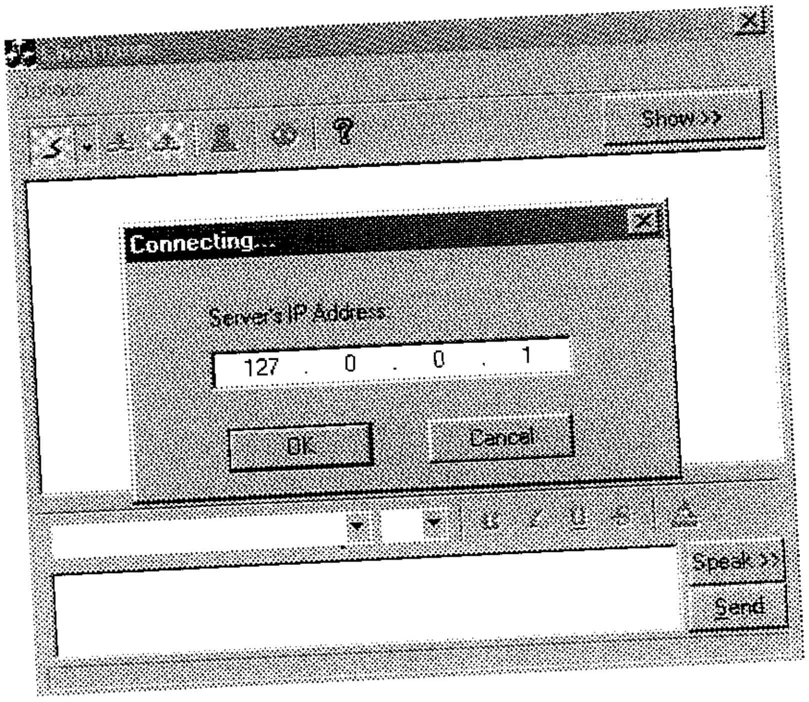
Chat server



127.0.0.1

	OnLine users IP	Port
	127.0.0.1	1029
	127.0.0.1	1030
	127.0.0.1	1031

Client Connecting



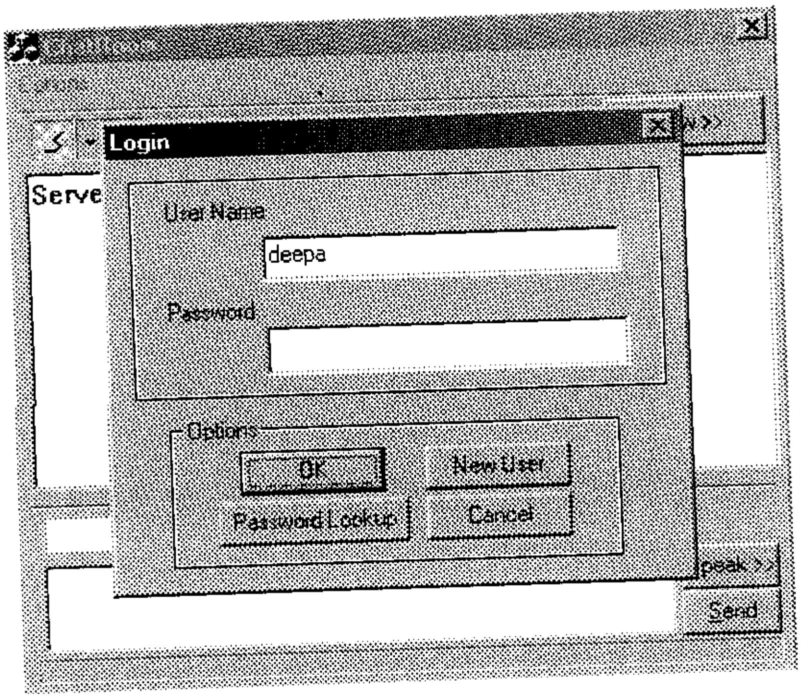
Connecting...

Server's IP Address

127 . 0 . 0 . 1

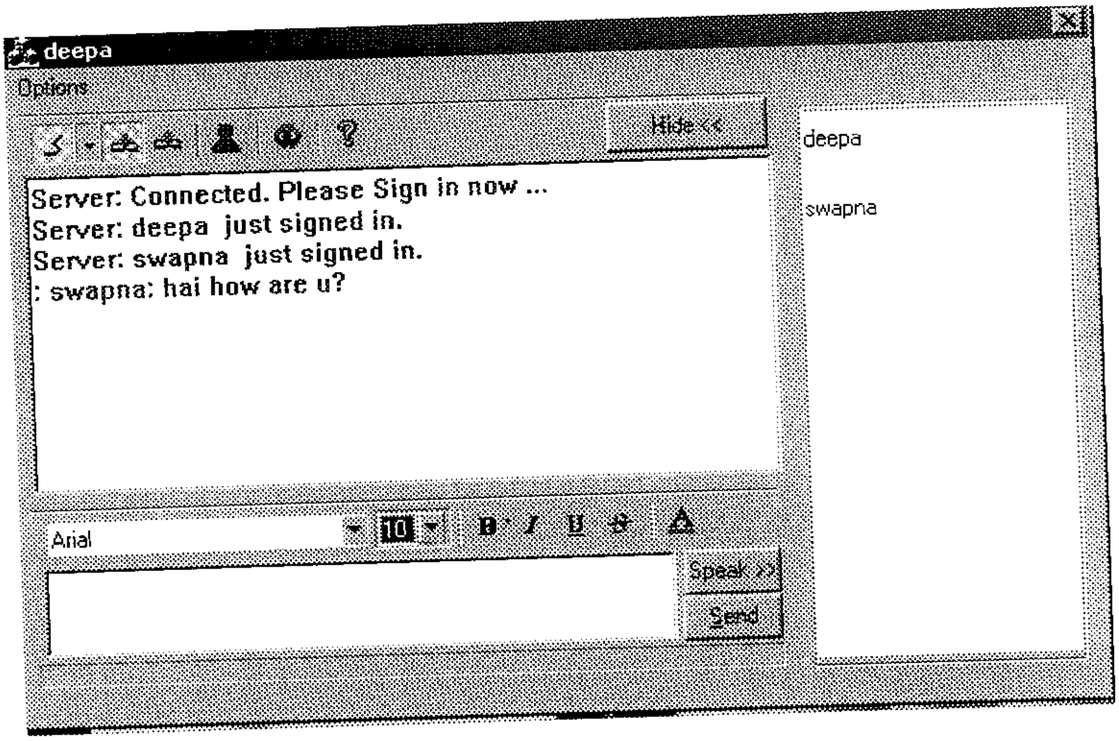
OK Cancel

Client Login Form



User Name: deepa

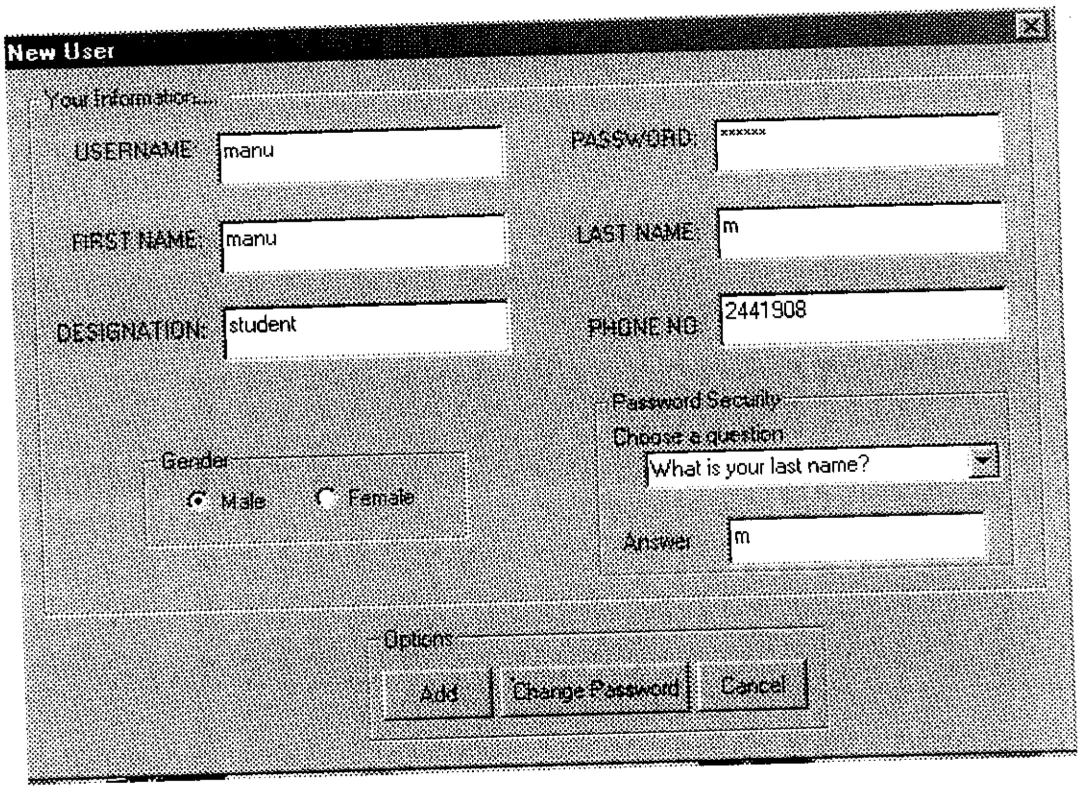
Chatting Session After Clients logged in



```
Server: Connected. Please Sign in now ...
Server: deepa  just signed in.
Server: swapna  just signed in.
: swapna: hai how are u?
```

deepa

swapna


New User

Your Information...

USERNAME:	manu	PASSWORD:	******
FIRST NAME:	manu	LAST NAME:	m
DESIGNATION:	student	PHONE NO:	2441908

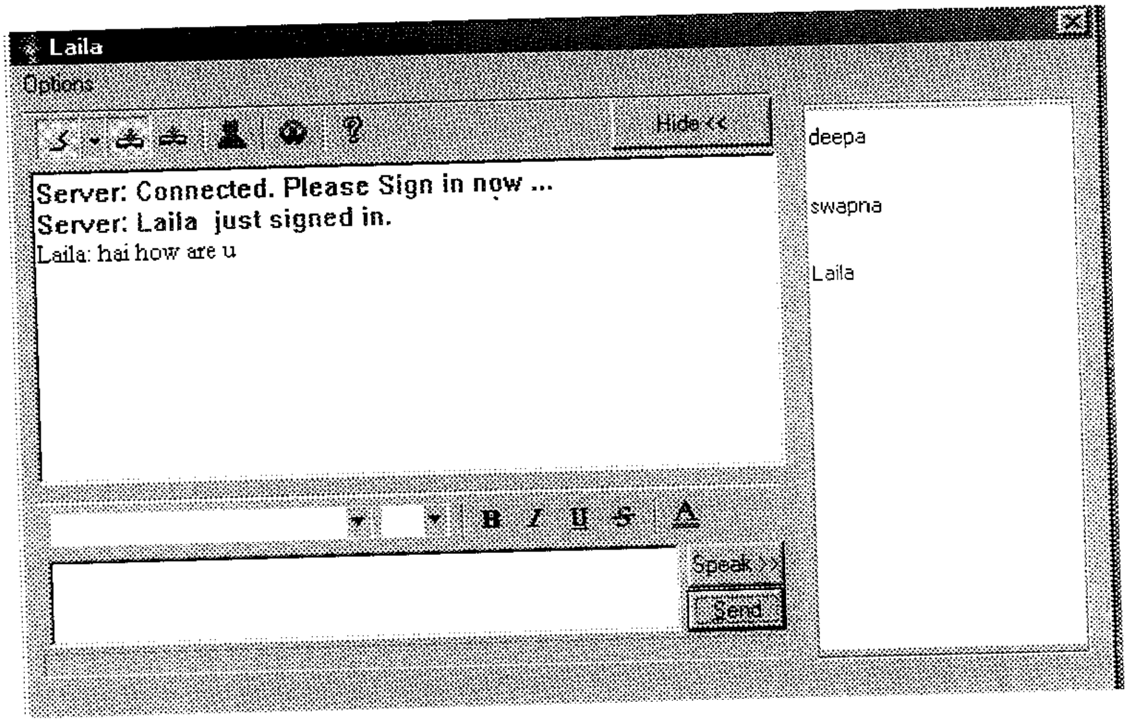
Gender: [✓ Male] Female

Password Security

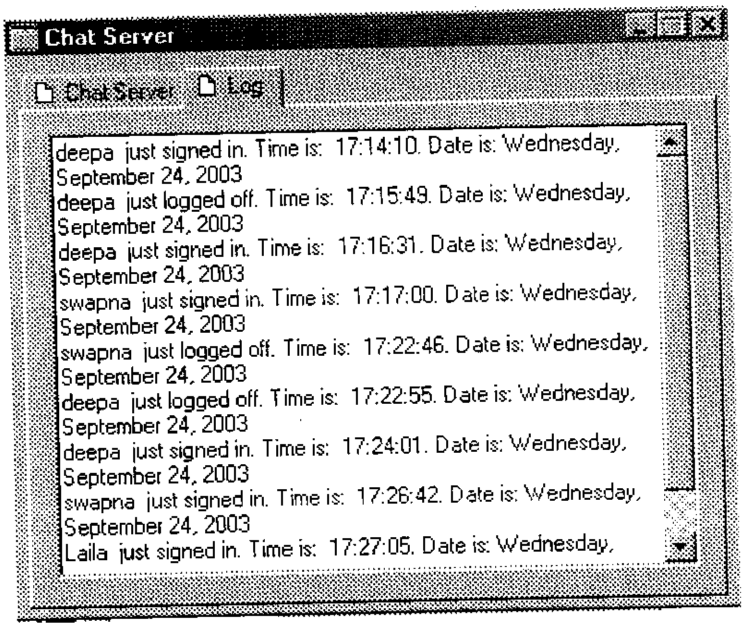
Choose a question: What is your last name?

Answer: m

Group chat session

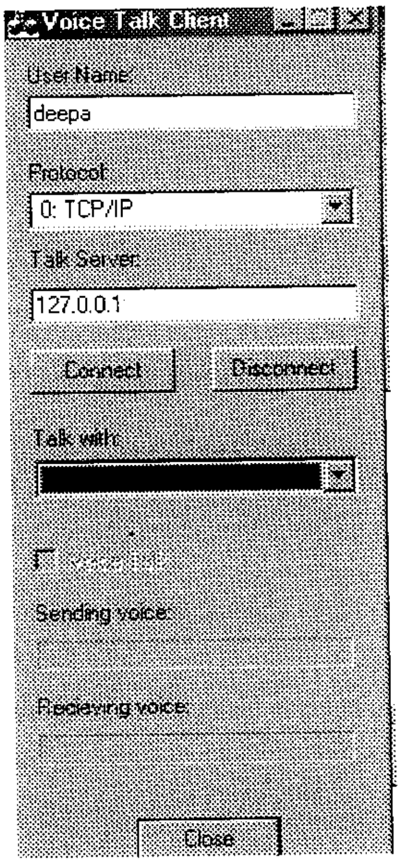


Server: Connected. Please Sign in now ...
Server: Laila just signed in.
Laila: hai how are u



```
deepa  just signed in. Time is:  17:14:10. Date is: Wednesday,
September 24, 2003
deepa  just logged off. Time is:  17:15:49. Date is: Wednesday,
September 24, 2003
deepa  just signed in. Time is:  17:16:31. Date is: Wednesday,
September 24, 2003
swapna  just signed in. Time is:  17:17:00. Date is: Wednesday,
September 24, 2003
swapna  just logged off. Time is:  17:22:46. Date is: Wednesday,
September 24, 2003
deepa  just logged off. Time is:  17:22:55. Date is: Wednesday,
September 24, 2003
deepa  just signed in. Time is:  17:24:01. Date is: Wednesday,
September 24, 2003
swapna  just signed in. Time is:  17:26:42. Date is: Wednesday,
September 24, 2003
Laila  just signed in. Time is:  17:27:05. Date is: Wednesday,
```

Voice chat session


Voice Talk Client

User Name: deepa

Protocol: 0: TCP/IP

Talk Server: 127.0.0.1

SOURCE CODE

```
void CChatRoomDlg::OnSignIn()
{
CSignInDlg dlg;
if(!bConnected) return;
if(bSignIn) return;
if(dlg.DoModal() == IDOK){
m_strScreenName = dlg.GetName();
if(m_strScreenName.IsEmpty()) return;
SendPkg(NEW);
bSignIn = TRUE;
SetDefaultCharFormat4Input();
SetWindowText(m_strScreenName);
PopulateFriends();
}




void CChatRoomDlg::SendPkg(int request)
{
if(!bConnected || !bSignIn){
m_wndInput.SetWindowText("");
return;
}

CPkg pkg;

AssemblePkg(pkg);

// send a quest to the server
pkg.request = request;

try{
pkg.Serialize(*m_pArchiveOut);
m_pArchiveOut->Flush();
m_wndInput.SetWindowText("");
}
catch(CFileException* e){
TCHAR lpszError[255];
e->GetErrorMessage(lpszError,255);
AfxMessageBox(lpszError);
exit(0);
}
}
```

```
void CChatRoomDlg::SendPkg(int request, CString Name, CString ToWhom)
{
if(!bConnected || !bSignIn){
m_wndInput.SetWindowText("");
return;
}

CPkg pkg;

AssemblePersonal(pkg, Name, ToWhom);

// send a quest to the server
pkg.request = PERSONAL;

try{
pkg.Serialize(*m_pArchiveOut);
m_pArchiveOut->Flush();
m_wndInput.SetWindowText("");
}
catch(CFileException* e){
TCHAR lpszError[255];
e->GetErrorMessage(lpszError,255);
AfxMessageBox(lpszError);
exit(0);
}
}

void CChatRoomDlg::OnUpdateConnect(CCmdUI* pCmdUI)
{

pCmdUI->SetCheck(bConnected);
}



void CChatRoomDlg::OnUpdateSignIn(CCmdUI* pCmdUI)
{

pCmdUI->Enable(bConnected);
pCmdUI->SetCheck(bSignIn);
}

void CChatRoomDlg::OnAppAbout()
{
```

```
dlgAbout.DoModal();

}

void CChatRoomDlg::OnSignOut()
{
if(bConnected && bSignIn){
bConnected = FALSE;
bSignIn = FALSE;
CString tmp;
tmp.Format("%s just logged off.",m_strScreenName);
m_wndShow.ShowMessage("Server",tmp);
SendPkg(OFF);
m_nameString.RemoveAll();
m_wndContactList.ResetContent();
return;
}
}

void CChatRoomDlg::LoadIniFile()
{
CFile iniFile;

if(iniFile.Open("Config.dat",CFile::modeNoTruncate|CFile::modeCreate|CF
ile::modeRead)){

if(iniFile.GetLength() == 0) return;

CArchive* ar = new CArchive(&iniFile,CArchive::load);

*ar >> m_strServerIP;
*ar >> m_strScreenName;

*ar >> bCharBold ;
*ar >> bCharItalic ;
*ar >> bCharUnderline ;
*ar >> bCharStrikeOut ;
*ar >> clrChar ;
*ar >> m_fontName ;
*ar >> m_fontSize ;

//for combo
int nIndex;
*ar >> nIndex;
m_wndRichTextBar.m_cmbFontSize.SetCurSel(nIndex);
*ar >> nIndex;
m_wndRichTextBar.m_cmbFontName.SetCurSel(nIndex);

delete ar;

iniFile.Close();
}


}

void CChatRoomDlg::SaveIniFile()
{
CFile iniFile;
if(iniFile.Open("Config.dat",CFile::modeNoTruncate|CFile::modeCreate|CF
ile::modeReadWrite)){

if(iniFile.GetLength() == 0) return;

CArchive* ar = new CArchive(&iniFile,CArchive::store);

*ar << m_strServerIP;
*ar << m_strScreenName;

*ar << bCharBold ;
*ar << bCharItalic ;
*ar << bCharUnderline ;
*ar << bCharStrikeOut ;
*ar << clrChar ;
*ar << m_fontName ;
*ar << m_fontSize ;

//for combo
int nIndex = m_wndRichTextBar.m_cmbFontSize.GetCurSel();
*ar << nIndex;
nIndex = m_wndRichTextBar.m_cmbFontName.GetCurSel();
*ar << nIndex;

ar->Flush();

delete ar;
iniFile.Close();
}
}
```

```
void CChatRoomDlg::OnOptionsTransferfile()
{
// TODO: Add your command handler code here
CFileDialog cmnDialog(true);
int a = cmnDialog.DoModal();
if(a>0)
{
SendFile(cmnDialog.GetPathName(), cmnDialog.GetFileName());
}

}

void CChatRoomDlg::SendFile(CString FileName, CString Name)
{
//int send = m_pClientSocket->Send("s", 1);
CFile myFile;
char buf[2024];
CString buff;
myFile.Open(FileName, CFile::modeRead | CFile::typeBinary);

int myFileLength = myFile.GetLength(); // Going to send the correct File Size

myFile.Read(buf, myFileLength);
buf[myFileLength]='\0';
buff = buf;
/*Name = Name + "*";
buff.Insert(0,Name);
MessageBox(buff);*/
buff = Name + "*" + buff;
SendPkg(MYFILE, buff);

}

void CChatRoomDlg::OnBtnSpeak()
{
UpdateData(true);
//CPkg pkg;
//      CString m_strRcvd = pkg.clrText;
ISpVoice * pVoice = NULL;
if (FAILED(CoInitialize(NULL)))
{
AfxMessageBox("Error to intiliaze COM");
return;
}
```

```
HRESULT hr = CoCreateInstance(CLSID_SpVoice, NULL, CLSCTX_ALL,
                    IID_ISpVoice, (void **)&pVoice);
                         if( SUCCEEDED( hr ) )
                                 {
          hr = pVoice->Speak(Msg.AllocSysString(), 0, NULL);
                              pVoice->Release();
                              pVoice = NULL;
                                 }

                         CoUninitialize();
                                 }

                void CChatRoomDlg::OnDblclkList()
                                 {
                                   CString str;
                                 CPkg pkg;


                 Name = m_wndContactList.GetData();

                            CString From;
                        GetWindowText(From);

                 SendPkg(PERSONAL, Name, From);

          //        index = m_ctrlClients.GetCurSel();
              //        str = m_ctrlClients.GetText(
                                 }

void CChatRoomDlg::AssemblePersonal(CPkg &pkg, CString Name, CString
                                 From)
                                 {
                                  pkg.Init();
                         //the next 5 are for client side
                                CString msg;
                    m_wndInput.GetWindowText(msg);
          CString text = /*ToWhom + ": " +*/ From + ": " +msg;
            //m_wndInput.GetWindowText(pkg.strText);      //1
                             pkg.strText = text;
                pkg.strName = Name;                        //2
        pkg.bAway = bAway;                                            //3
        pkg.iMyIcon = m_iMyIcon;                                      //4
           pkg.bBold = bCharBold;                                  //5
              pkg.bItalic  = bCharItalic;                       //6
              pkg.bUnderLine  = bCharUnderline;                 //7
              pkg.bStrikeOut = bCharStrikeOut;                  //8
        pkg.clrText  = clrChar;                              //9
       pkg.fontName  = m_fontName;                           //10
          pkg.fontSize  = m_fontSize;                      //11

//      pkg.ipAddress                                            //12
                            reserved for Server
//      pkg.request = UNDEFINED;                        //13    reserved
//      pkg.pSocket                                              //14
                            reserved for Server
//      pkg.port                                                 //15
                            reserved for Server

                                  }


                void CChatRoomDlg::PopulateFriends()
                                  {

                            CRSClients rsClient;
                              CDatabase db;
                          CString Query, FromDb;
                            CString Friend;
                           CString ThisUser;
                             CString User;
                              CString Me;
                              int flg = 0;

                        GetWindowText(ThisUser);

    int open = db.OpenEx("DSN=ChatDSN", CDatabase::useCursorLib);
                                if(open)
                                  {
                           CRSClients UserRs(&db);
        Query.Format("Select * from clients where UserName = '%s'",
                           ThisUser);
                               int len1, len2;
       if(UserRs.Open(CRecordset::snapshot, Query, CRecordset::none))
                                     {
                        CString temp = UserRs.m_Friend;
                                   do{
                              len1 = temp.GetLength();
                          Friend = temp.SpanExcluding("*");
                              len2 = Friend.GetLength();
                                  if(Friend!="")
                         m_wndContactList.AddFriends(Friend);
```

```
                    temp = temp.Right(len1 - len2 - 1);
                        }while(Friend!="");
                            }
                        }
        else MessageBox("Cannot open the database, Sorry!");

                        }

void CChatRoomDlg::AssembleFilePkg(CPkg &pkg, CString File)
                        {

                        pkg.Init();
                //the next 5 are for client side
                    CString text = File;
        //m_wndInput.GetWindowText(pkg.strText);      //1
                    pkg.strText = text;
            pkg.strName = Name;                       //2
    pkg.bAway = bAway;                                      //3
    pkg.iMyIcon = m_iMyIcon;                                //4
      pkg.bBold = bCharBold;                              //5
        pkg.bItalic  = bCharItalic;                     //6
        pkg.bUnderLine  = bCharUnderline;               //7
        pkg.bStrikeOut = bCharStrikeOut;                //8
      pkg.clrText  = clrChar;                             //9
      pkg.fontName  = m_fontName;                         //10
        pkg.fontSize  = m_fontSize;                     //11
                        }

  void CChatRoomDlg::SendPkg(int request, CString File)
                        {

                        CPkg pkg;

                AssembleFilePkg(pkg, File);

                // send a quest to the server
                    pkg.request = request;

                    .       try{
                pkg.Serialize(*m_pArchiveOut);
                    m_pArchiveOut->Flush();
                m_wndInput.SetWindowText("");
                            }
                catch(CFileException* e){
                    TCHAR lpszError[255];
                e->GetErrorMessage(lpszError,255);
```

```
AfxMessageBox(lpszError);
        exit(0);
        }
    }

void CChatRoomDlg::VoiceChat()
    {
STARTUPINFO       siStartupInfo;
PROCESS_INFORMATION piProcessInfo;

memset(&siStartupInfo, 0, sizeof(siStartupInfo));
memset(&piProcessInfo, 0, sizeof(piProcessInfo));

siStartupInfo.cb = sizeof(siStartupInfo);
if(CreateProcess("talkclient.exe", // Application name
    0,          // Application arguments
        0,
        0,
        FALSE,
    CREATE_DEFAULT_ERROR_MODE,
        0,
    0,                  // Working directory
        &siStartupInfo,
    &piProcessInfo) == FALSE)
        {
    MessageBox("Error");
        }
        }
```