

TABLE OF CONTENT

TABLE OF CONTENT	1
Acknowledgement	3
Abstract	4
Chapter 1: Introduction	5
1.1 Problem Statement	7
1.2 System and Scope	8
1.3 Scope	9
1.4 Limitation	10
1.5 Planning	11
Chapter 2: Literature Review	12
2.1 Internet Overview	12
2.2 Networking Overview	13
2.3 Digital Telephony	16
2.4 Voice Compression	18
2.5 Security Issues	22
2.6 Authentication	24
2.6.1 Speech Coding	24
2.6.2 Bandwidth	29
2.6.3 Encryption	33
2.6.4 Firewall	33
2.6.5 Summary	35

TABLE OF CONTENT

Summary	38
Chapter 3: System And Requirements Analysis	39
Introduction	39
TABLE OF CONTENT	1
Acknowledgement.....	3
Abstract	4
Chapter 1: Introduction.....	5
1.1 Problem Definition	7
1.2 System and Module Objectives.....	8
1.3 Scope	9
1.4 Limitation	10
1.5 Planning.....	11
Chapter 2 : Literature Review.....	12
2.1 Internet Overview.....	12
2.2 Networking Overview.....	13
2.3 Digital Transmission.....	15
2.4 Voice Digitization	18
2.5 Security Issue	22
2.6 Application Study Case.....	24
2.6.1 Speak Freely Application	24
2.6.2 NetMeeting	29
2.7 Software Development Tools.....	33
2.7.1 Visual Basic	33
2.7.2 Java	35

Summary 38

Chapter 3: System And Requirements Analysis 39

 Introduction 39

 3.1 Prototyping Development Method 39

 3.2 System Requirements..... 41

 3.2.1 Functional Requirements 41

 3.2.2 Non-Functional Requirements 42

 3.2.3 Hardware and Software Requirements 43

Chapter 4: System Design..... 44

 Introduction 44

 4.1 System Architecture..... 44

 4.2 System Functionality Design..... 47

 4.3 User Interface Design 51

Project Discussion 57

Conclusion 58

Project Discussion 2 [WXES 3182] 59

 Actual System Implementation 59

 Active-X Control [NetMeet] Implementation 72

Conclusion 2 [WXES 3182]..... 77

Appendix..... 78

References 89

Acknowledgement

Computer technology is a very dynamic and changing field. There are always new technologies being developed and adopted on all fronts, which makes it an exciting field. Here, I would like to thank everyone who had helped me in completing this project. In particular, I would like to express my sincere gratitude to my supervisor, Encik Noorzaily for his greatly beneficial guidance, support comments throughout the planning and development of this project. Besides, I would like to express my appreciation and thanks to my entire team members Lee Chee Cheng, Chong Bing Shuang in helping me to finish the report. Also, others who has lent me their hand on helping me in this research. Finally, I would also like to thank my family for giving undivided support to me throughout the period in succeeding the project.

Chapter 1: Introduction **Abstract**

Computer technology is a very dynamic and changing field. There are always new technologies being developed and adopted on all fronts, which makes it an exciting field to be in. So, in order to catch up with the technology, I choose myself to investigate some of the telecommunication technology specifically voice networking on the computer.

Through this project, I hope I can learn everything regarding to voice networking and thus can build a complete voice communication package for the whiteboard system project as an enhancement to the system. In order to accomplish this project, me together with two of my friends Chong Bing Shuang and Lee Chee Ching have joined the whiteboard system project. We have decided to separate the project into 3 parts on hopes of getting more details for the research, which are the whiteboard module, voice communication module and the server module. Chong Bing Shuang will be in charge of the whiteboard module while Lee Chee Ching will be involved in the server module. I myself will be doing the voice communication module.

My voice communication module will involved me in on doing research on some history of the voice networking, current data technology or technique used for voice networking, idea of enhancement and so on. After the research, I will be building or enhancing a voice communication module for the whiteboard system as a case study for my research.

Chapter 1: Introduction

It was as early as 1854 when Philip Reise was able to send sound over wires. Many consider him to be the inventor of the telephone. However, it wasn't until 1876 that Alexander Bell obtained the patent for it. This was similar to when the inventors of the first digital computer, John Atanasoff and Clifford Berry approached IBM and were told that IBM would never be interested in electronic computers. These two technologies have now become the most successful creation ever been in history. Many others sector especially the business sector depend the most on it to gain profit from the market. What is more interesting are the start of the networking era and the growing of Internet had makes these two technologies undivided.

So, what is network? A network can simply be defined as the interconnection of two or more independent computers or switches. One reason to network is that the sharing of resources can be done easily. With networks, it can also allow us to be mobile log into a network system to perform certain tasks. But it cannot be done without the facility provided by the communication technology. Besides all the new communication information data such as video, data and image, voice still play a major role as a type of information data over the new network communication system. But a lot had change since then, the analog data such as voice have now been digitized for reason of cost, data integrity, capacity utilization, security and privacy and integration.

Due to my interest in the voice communication through network technology and the fascination from both of the network and computer technology, I have decided to get myself involved in building a voice communication package for a whiteboard system in my project. This voice communication application involved utilizing some of current technology used to build the voice communication system over the networked computer. The whole whiteboard system involved me and another two of my colleagues and it has been divided into 3 parts after some discussion. Lee Chee Cheng, will be in charge of building the server module for the whiteboard system while Chong Bing Shuang will do the whiteboard module for the system. I will be in charge of doing voice communication module. The last integration module will involved three of us to integrate all of the module into a perfect whiteboard system. The voice communication package I am going to build will tend to improve the performance, feasibility and usability to the user.

1.1 Problem Definition

Currently when talk about whiteboard system; people seldom get the voice communication module built into it. This can be due by several reasons such as cost, efficiency, usability and the overhead in maintaining of resources available. Voice data involved using a lot network resources and face a lot of delay over the network. In order to get the best quality of service, we need a better computer and a better network solution, which bring us back to cost factor and become a limitation.

Secondly, the efficiency of a voice communication can bring down of the implementation. Many application involved voice communication failed to achieve 100% quality because of network or the computer. This limitation causes the voice communication system purposely be left out to save resources and focus on extending other areas. Resource maintaining is also a major concern of a good voice communication implementation. This can help maximize throughput with a minimize usage of precious resources such as the network resources and the processing power. Another turndown for the voice communication involves the usability and feasibility of the voice communication. A clear example is people prefer using phone to computer as a communication medium because of the lack in quality of service, security and feasibility in communication application. A successful voice communication need not be sophisticated, as this will only bring problems to the user in order to familiar with it. Extra features are important but too much with brings negative effects to the application as well.

After clarify all problems found in the voice communication definition,

As a conclusion, all problems above can be classified as hardware problem and software problem. All these problems need to be solved in order to convince people on using the voice communication over computer. Below is a problem definition is being list out:

- Delay in the voice module
- Cost of building the a full features voice communication of voice modules not worth
- Efficiency of the voice module
- Resources problem
- Feasibility and usability
- Sophistication of the module itself.

1.2 System and Module Objectives

The Whiteboard system that we are intended to build consists objectives as follow:

- Support major drawing functions
- Support major collaboration functions
- Support real-time audio conferencing and major functions of it
- System Management add-in (Client-Server Architecture)
- Easy to maintained and used

After clarify all problems found in the voice communication definition,

System Integration Section

1. Identify the impact of the voice technology to the IT sector:

Sub-objectives:

- I. Learn about the fundamental of voice technology.
- II. Identify the drawbacks of implementing voice communication.
- III. Observe the functions, attributes and implementation of voice application available in the market.

2. To build a voice communication application:

Sub-objectives:

- I. To build a user-friendly, easy to implement, secure and feasible voice communication package.
- II. Implement the voice communication package into the whiteboard system.
- III. To further enhance the quality of service of the voice communication package.

1.3 Scope

Voice Communication Module Section

This voice communication module consists of the exact implementation of the voice communication package. It involves building the voice communication package that fit criteria such as reusable, extendable, secure, easy to implement and have better quality of service. Basically, it will consist of receiving and transmitting or broadcasting voice over a network. Then the focus will go to the quality of voice over network. I will try to do some investigation and improvement to the module. After that, some extra features for the easy implementation will be considered. Reusability and extendable of the module will be the concern in the last part.

System Integration Section

This integration section consists of the actual implementation of the voice communication module into the whiteboard system. It can be consider as an actual testing environment of the voice communication module. In this module, integration with others such as the Server Module and the Whiteboard Module will be done to further extend total Quality of Service of the whiteboard system. Some advance feature may be added for the good of the total quality of service. Beside of this, another consideration involved in this module is resource sharing among us.

1.4 Limitation

Due to the imperfect network situation, the whole system will be limited to simple LAN to avoid extreme traffic consideration in actual implementation. Platform implementation chosen will be Windows. This is because that the time limitation problem has prevented us on putting effort into study other platform to perfectly build the system. A solution to this will be using Java and this will also be given due consideration. Programming language is also limited Visual Basic to due to the time constraint and platform compatibility. Besides that, Visual Basic is also a very good RAD tool.

1.5 Planning

Project Schedule

ID	Task Name	2001							
		May	June	July	Aug	Sept	Oct	Nov	Dec
1	Problem Definition	■							
2	Literature Review	■	■						
3	System Analysis & System Design		■	■	■	■			
4	System Implementation			■	■	■	■		
5	System testing				■	■	■	■	
6	Documentation	■	■	■	■	■	■	■	■

Figure 1.5.1 Project Schedule

Chapter 2 : Literature Review

Literature reviews is a very important step for information gathering before going to others phase such as the system design & analysis, system implementation. It served as an overview of the project and broadens the thought of the implementers before going for actual implementation.

Throughout this phase, the implementers will study background of the project, its relationship to others, comparison of other systems, mark down details of 'going to be' system and analysis of the enhancement probability. Most of the times, it will require the implementers to go separately and search from other people research, electronic resources, journal, book and personal observation. Below is the outcome after the hard work.

2.1 Internet Overview

What is Internet? There is debate over the precise definition for it. Some say it is only the high-speed backbone of the network, some say it is the backbone plus the major arteries, some say it is everything down to the personal computers in homes and businesses, and some say it is a global computer network that includes the users and data.

But what is interesting enough is the growth of the Internet. Recent studies show that there are approximately 33,000 new users using the Internet everyday. About 177 countries linked to the Internet since March 1997. In August 1996, there were 72,600,000 users using the Internet and this figure growth to 90,000,000 in June 1998.

Due to the tremendous growth rate of Internet and its extensive and diverse usage, institutions are reorganizing communications and workflow to take advantage of the coverage and return on investment features of the Internet. This change of processes necessitates the conversion or transformation of conventional media to a digital format that can be used in a particular computer environment and transferred efficiently over a network. This media adaptation thus involves the use of audio.

Conclusion

This topic was chosen as our first investigation as this is the major factor of the growth of multimedia over the network. Although there is not much useful information obtain to the project, it is certainly served as a good brief opening to the project.

2.2 Networking Overview

The communication functionality required for distributed application is quite complex. This functionality is generally implemented as a structured set of modules. The modules are arranged in a vertical, layered fashion, with each layer providing a

particular portion of the needed functionality and relying on the next-lower layer for more primitive functions. Such a structure is referred to as a protocol architecture. One motivation for the use of this type of structure is that it eases the task of design and implementation. A clearer example of this would be the TCP/IP.

So what is TCP/IP? TCP/IP is a result of protocol research and development conducted on the experimental packet-switched network, ARPANET, funded by the Defense Advanced Research Projects Agency (DARPA), and is generally referred to as the TCP/IP protocol suite. It is a 5 layers protocol suite for communication and there are physical layer, network access layer, internet layer, transport layer and application layer. The physical layer covers the physical interface between a data transmission device and a transmission medium. It involved how to nature of the signals and the data rate. The next layer is network access layer and it is concerned with the exchange of data between an end system and the network to which it is attached. Flow control and error control of the frame will also be implemented here. Internet layer involves in exchange of the data for internetwork and thus provides the routing function. The famous internet protocol (IP) is implemented in this layer. As for the Transport layer, it provides the reliable and transparent of data exchange between 2 end point in the internetwork environment, data flow control and data error recovery control. The famous Transmission Control Protocol (TCP) protocol is the widely used protocol for this layer. The last layer would be Application layer and this layer contains the logic needed to support the various user applications.

Talking further about Transport layer, there are two commonly datagram implemented in the layer, there are the Transmission Control Protocol datagram and the User Datagram Protocol datagram. As mentioned above Transmission Control Protocol provides reliable end-to-end connection, it will always follow up messages sent from an end point to another. Voice data is produced in real time as this thus give a problem as the TCP will tend to recover if the some of data were lost and this will force a delay on the real time architecture of the voice communication application. A solution of this is to implement the UDP datagram instead of TCP. UDP is a connectionless based protocol and its lack in data lost recovery ability will fit this job perfectly.

Conclusion

Networking is certainly a very broad field. We start with this investigation just to broaden our view on how the network communication functions. Though this investigation we know why UDP is more suitable compare to TCP for voice communication. And this investigation we had also expose to us choosing network solution for implementation. To limit the scope of investigation, network technology will be limited to LAN and thus no investigation had been done to the network technology to assist the project.

2.3 Digital Transmission

Most of us would ask why go for digital signal if we already can transmit using the analog form. There are plenty of reasons for using digital signal for transmission.

First, digital transmission produces fewer errors than the analog signal. This happens because the analog signals are far more susceptible to noise and signal impairment compare to digital signals. With the use of digital repeater rather than analog amplifier, the effects of noise and other signal impairment are not cumulative. Thus it is possible to transmit data longer distances and over lesser quality lines by digital means while maintaining the integrity of the data.

Secondly, digital transmission is easier to secure than analog transmission. This is because digital data is easier to encrypt compare analog data. Furthermore, there are various of encryption techniques can readily be applied to digital data and to analog data that have been digitized. For instance the DES (Data Encryption Standard), PGP key exchange, Blowfish encryption, IDEA (International Data Encryption Algorithm) and Key file are some of the encryption methods use in to encrypt digital data.

The third reason of for using digital signal for transmission would be the capacity utilization. Digital transmission permits higher maximum transmission rates. It has become economical to build transmission links of very high bandwidth, including satellite channel and fiber optic. Thus a high degree of multiplexing is needed to utilize such capacity effectively, and this is more easily and cheaply achieved with digital (time-division) rather than analog (frequency-division) techniques.

Another reason for using digital signal for transmission is the integration. Integrating voice, video and data on the same circuit is far simpler with digital transmission. Thus economies of scale and convenience can be achieved by integration voice, video, image, and digital data. For instance, the ability to send speech over a digital circuit has many attractions; in particular it provides the ability to use the same ISDN network that is used for data transmission. In fact once a speech signal is in a digital form, the network is unaware of whether the traffic that it is carrying is data, graphics or speech.

One last reason for using digital signal for transmission is the implementation cost. The growth in semiconductor especially VLSI sector has caused a continuing drop in the digital circuitry but analog equipment has not shown a similar drop. Furthermore, the maintenance costs for digital circuitry are a fraction of those for analog circuitry.

Conclusion

After the investigation we can surely know why digital transmission are the more welcome when come to transmitting voice data. Though, analog transmission is still widely use as most of the transmission media transmit data on electromagnetic wave which also in the analog basis. But factors had shown above have certainly open our view on choosing the digital voice data over the analog voice data for the transmission. So our next investigation will cover on digitization of analog data

2.4 Voice Digitization

Voice digitization involve about techniques used for converting analog voice signal into a digital form of voice signal and vice versa. These processes are normally called as the encoding and decoding of analog data.

What is encoding and decoding scheme involved? The encoding is actually ways involved in converting an analog signal into digital form before being transmitted over the network. As for decoding, it is the reverse of the encoding scheme.

PCM (Pulse Code Modulation)

Pulse Code Modulation is a digital scheme for transmitting analogue data. The signals in PCM are binary; that is, there are only two possible states, represented by logic 1 (high) and logic 0 (low). This is true no matter how complex the analogue waveform happens to be. Using PCM, it is possible to digitize all forms of analogue data, including full motion video, voices, music, telemetry, and virtual reality.

To obtain PCM from an analogue waveform at the source (transmitter end) of a communications circuit, the analogue signal amplitude is sampled at regular time intervals. The sampling rate, or number of samples per second, is several times the maximum frequency of the analogue waveform in cycles per second or hertz. The instantaneous amplitude of the analogue signal at each sampling is rounded off to the nearest of several specific, pre-determined levels. This process is called quantization. The number of levels is always a power of 2 -- for example, 8, 16, 32,

or 64. These numbers can be represented by three, four, five, or six binary digits (bits) respectively. The output of a pulse code modulator is thus a series of binary numbers, each represented by some power of two bits.

At the destination (receiver end) of the communications circuit, a pulse code demodulator converts the binary numbers back into pulses having the same quantum levels as those in the modulator. These pulses are further processed to restore the original analogue waveform.

DPCM (Differential Pulse Code Modulation)

Differential pulse code modulation (DPCM) is procedure of converting analog to digital signal in which analog signal is sampled and then difference between actual sample value and its predicted value (predicted value is based on previous sample or samples) quantized and then encoded forming digital value. DPCM code words represent differences between samples unlike PCM where code words represented a sample value.

Basic concept of DPCM - coding a difference, is based on the fact that most source signals shows significant correlation between successive samples so encoding uses redundancy in sample values which implies lower bit rate. Realization of basic concept (described above) is based on technique in which we have to predict current we have to encode the sample value based upon previous samples (or sample) and difference between actual value of sample and predicted value (difference between

samples can be interpreted as prediction error). Because its necessary to predict sample value DPCM is form of predictive coding. DPCM compression depends on prediction technique, well-conducted prediction techniques leads to good compression rates, in other cases DPCM could mean expansion comparing to regular PCM encoding.

ADPCM (Adaptive Differential Pulse Code Modulation)

A major limitation of the DPCM system considers so far is that the predictor and the quantizer are both fixed. DPCM schemes can be made adaptive in terms of the predictor or the quantizer or both. Adaptive prediction usually reduce the prediction error prior to quantization, and thus, for the same bit rate, the reduced dynamic range of the quantizer input signal results in less quantization error and better reconstructed image quanlity. On the other hand, adaptive quantization aims at reducing the quantization error directly by varying the decision and reconstruction levels according to the local image statistics.

DM (Delta Modulation)

With very high over sampling rates, the changes between sample periods are made very small, thus the quantizer can be reduced to low-bit. A 1-bit DPCM coder is known as a delta modulator (DM) thus its process is known as Delta Modulation. In Delta Modulation (DM) each sample is stored as a single bit which represents either an increase of one step or a decrease of one step. In other words each sample can only be one bit higher or lower than the previous (and never the same).

2.5 Security Issue

ADM (Adaptive Delta Modulation)

The performance of DM can be improved by varying the step size.

For example, a succession of bits moving in the same direction means the output has to rise or fall at its maximum value.

This rise or fall can be encoded in less space if it is assumed that :

- A single bit rise indicates a small step size.
- A two bits rise indicates a larger step size.
- A three bits rise indicates a larger step size still and so on.

During recording the encoder looks at the rate of rise or fall over a period and then uses a certain number of bits to encode that rise or fall.

On playback the process is reversed. When the decoder sees several rises or falls in succession it sets the steepness of the slope accordingly.

Conclusion

Through this investigation, we gain knowledge of voice data communication and the methods being used to send a digitized voice data over a network. These methods are PCM, DPCM, ADPCM, DM and ADM with each has its good characteristic over the other likes compression and precision. This helps a lot on choosing the best voice digitization method to ease encryption factor discussed later, which usually extend the length of the original voice data.

2.5 Security Issue

When talking about security issue in voice communication the main concern would be privacy. Since voice data has been digitized before transmission, encryption can be done on it before transmission. Due to the consideration of voice communication application involve in RT (Real Time), the time factor play the most important role in choosing a good encryption algorithm.

In cryptography view, encryption mechanisms can be divided into 2 groups, which is the secret-key cryptography and public-key cryptography. Secret-key cryptography is also referred as symmetric cryptography because the key for encryption and decryption of a message is actually the same key. As for Public-key cryptography, also known as asymmetric cryptography uses different set of key for encryption and decryption. These keys are the private key and the public key. From the point of speed, symmetric cryptography is much faster compare to asymmetric cryptography that is about 10 to 100 times.

Each of these groups can also be sub-divided into another 2 groups based on their encryption algorithm characteristics and they are the block cipher and the stream cipher. Block cipher tends to separate the plaintext into the blocks of fixed length, ex. 64 bits and each of it is encrypted independently and it is ready for transmission as soon as the encryption process finish. To further enhance its security, the block cipher also employs the CBC (Cipher Block Chaining). These CBC served to prevent of repeated patterns over the same encrypted blocks. The easiest sample to this

would be implementing the XOR operation. Though to prevent eavesdropper might get some useful information when transmission to two destinations for a same message situation happens, additional texts served as an initialization vector will be added to the front of each message before the encryption. CBC required a reliable connection and the decryption would fail if any blocks of ciphertext were lost hence it is not suitable for real time application.

Another type of the encryption algorithm is the stream cipher. For some application such as the voice communication encryption in blocks is inappropriate because of data are produced in real time in small chunks. So Stream ciphers are encryption algorithms that can perform encryption incrementally, converting plaintext to cipher text one bit at a time. The trick is to construct a keystream generator. A keystream is an arbitrary-length sequence of bits that can be used to obscure the contents of a data stream by XOR-ing the keystream with the data stream. If the keystream is secure, then so is the resulting encrypted data stream.

Below is some current use of encryption algorithms:

I. DES (Data Encryption Standard)

- Designed at IBM around 1970.
- DES designed to be implemented only in hardware, and is therefore extremely slow in software.
- DES accepts a 64-bit key, the key setup routines effectively discard 8 bits, giving DES a 56-bit effective key length.

II. Blowfish encryption

- Design by Bruce Schneier.
- Combines a Fiestal network, key-dependent S-Boxes, and a non-invertible $f()$ function.

III. IDEA (International Data Encryption Algorithm)

- Developed in Zurich, Switzerland by Xuejia Lai and James Massey.
- Utilizes a 128-bit key and is designed to be resistant to differential cryptanalysis.

Conclusion

Through this investigation we now have a clearly view on choosing better encryption algorithm for the voice communication. The stream cipher is good for real time application and the secret-key cryptography is normally used because of the encryption time are smaller compare the public-key cryptography. On choosing the actual algorithms both these factor will be given due consideration.

2.6 Application Study Case

2.6.1 Speak Freely Application

Speak Freely is an Internet telephone for Microsoft Windows that allows you, with appropriate hardware and software, to send and receive audio, in real time, over a computer network. If you're connected to the Internet by a sufficiently high-speed link, you can converse with anybody else similarly connected anywhere on Earth without paying long-distance phone charges. Users can find one another, even if

they have dial-up connections to the Internet, by publishing and searching directory entries on a Look Who's Listening server. You can designate a bitmap file to be sent to users who connect so they can see whom they're talking to.

Speak Freely was written by John Walker and Brian C. Wiles. John Walker is the founder of AutoDesk Inc (ACAD-NASDAQ), one of the largest personal computer software companies, has become a leader in the computer aided design industry; its first product, AutoCAD, is the de facto worldwide standard for computer aided design and drafting. While Brian C. Wiles is the Chief Technical Officer of Planetlink, Inc. of California, a corporation that owns an Internet Service Provider and a software development company in California. He took over the Speak Freely project in April of 1998, two years after John Walker's last release. He released his first beta version of 7.0 on October 23, 1998, and has since been actively enhancing the product into what it is today. Below are some of the features provide by the Speak Freely.

Send sound file wav or au

Send sound file over the network besides real time voice communication.

Auto detection half / full duplex connection

Autodetect soundcard capability for half or full duplex support connection to ease user's configuration setting.

Voice Activation as a smart management of full duplex connection

Network capacity (bandwidth) is finite and precious; it's important not to waste it. Even though users with full-duplex audio hardware could, in theory, transmit continuously, this would be irresponsible as it would double the load on the network with no real benefit. This feature gives a smart management of the connection.

Ring remote user

To establish a connection with a remote user who's running Speak Freely, but who's accidentally turned the volume down to zero by setting the output volume of the receiving machine to the maximum value upon receiving to get the user's attention.

Multicasting / Broadcasting

Multicasting is implemented, allowing those whose networks support the facility to create multi-party discussion groups to which users can subscribe and drop at will. For those without access to Multicasting, a rudimentary Broadcast capability allows transmission of an audio feed to multiple hosts on a fast local network.

Speak Freely's Broadcast facility provides an alternative which requires no special network configuration. Without the benefit of Multicasting is it forced, however, to send duplicate packets to each recipient, which usually works only on fast local networks. Since many educational institutions and enterprises have such networks, broadcasting can be an effective way to transmit classes, seminars, and meetings to multiple destinations within the organisation.

View extended status

Some of the extended status such as the number of connection, message queue size, audio sample size and rate for input and output channel, packet sent, packet received, samples per packet, packet size and packet lost can be view through a extended status view.

Monitor tools for audio level and spectrum

Speak Freely's Audio Monitor panel displays audio input and output levels in real time, allowing user to observe what Speak Freely is receiving from your microphone and sending to your speakers. In conjunction with your sound card's gain setting utility, this should help you set the levels so things work acceptably.

Compression enable

For asynchronous serial communication, the data rate in bytes per second is about one tenth the speed in bits per second so it's clear that even a 64 Kb line can't transmit uncompressed sound at 8000 bytes per second. Speak Freely provides three forms of compression which can be selected independently or in combination to reduce the data rate.

Encryption enable.

Speak Freely provides three different kinds of encryption, including the same highly-secure IDEA algorithm PGP uses to encrypt message bodies. By using PGP to

automatically exchange session keys, you can Speak Freely to total strangers, over public networks, with greater security than most readily available telephone scramblers provide.

Text chat enable

As a back-up form of communication which will work on all but the most unreliable Internet connections.

Answering machine enable

A built-in answering machine use to save everything into a .sfm format file for later replay.

Platform and Communication Compatibility

Speak Freely for Windows is 100% compatible Speak Freely for Unix, currently available for a variety of Unix workstations. Windows users can converse, over the Internet, with users of those Unix machines. In addition, Speak Freely supports the Internet Real-Time Protocol (RTP) and the original protocol used by the Lawrence Berkeley Laboratory's Visual Audio Tool (VAT); by selecting the correct protocol, you can communicate with any other network voice program, which conforms to one of these standards.

2.6.2 NetMeeting

NetMeeting helps small and large organizations take full advantage of the global reach of the Internet or corporate intranet for real-time communications and collaboration. Connecting to other NetMeeting users is made easy with the Microsoft Internet Locator Server (ILS), enabling participants to call each other from a dynamic directory within NetMeeting or from a Web page. The following are the features that can be found in NetMeeting

Internet phone/H.323 standards-based audio support.

Real-time, point-to-point audio conferencing over the Internet or corporate intranet enables you to make voice calls to associates and organizations around the world. NetMeeting audio conferencing offers many features, including half-duplex and full-duplex audio support for real-time conversations, automatic microphone sensitivity level setting to ensure that meeting participants hear each other clearly, and microphone muting, which lets users control the audio signal sent during a call. This audio conferencing supports network TCP/IP connections.

Support for the H.323 protocol enables interoperability between NetMeeting and other H.323-compatible audio clients. The H.323 protocol supports the ITU G.711 and G.723 audio standards and Internet Engineering Task Force (IETF) RTP and RTCP specifications for controlling audio flow to improve voice quality. On MMX-enabled computers, NetMeeting uses the MMX-enabled audio codecs to improve

performance for audio compression and decompression algorithms. This will result in lower CPU use and improved audio quality during a call.

Participants can share applications, exchange information through a shared

H.323 standards-based video conferencing

With NetMeeting, a user can send and receive real-time visual images with another conference participant using any video for Windows-compatible equipment. They can share ideas and information face-to-face, and use the camera to instantly view items, such as hardware or devices, that the user chooses to display in front of the lens. Combined with the audio and data capabilities of NetMeeting, a user can both see and hear the other conference participant, as well as share information and applications. This H.323 standard-based video technology is also compliant with the H.261 and H.263 video codecs.

Participants can share Windows-based applications transparently without any special knowledge of the application

Intelligent Audio/Video Stream Control

NetMeeting features intelligent control of the audio and video stream, which automatically balances the load for network bandwidth, CPU use, and memory use.

This intelligent stream control ensures that audio, video, and data are prioritized properly, so that NetMeeting maintains high-quality audio while transmitting and

receiving data and video during a call. Through system policy features, IS

organizations can configure the stream control services to limit the bandwidth used for audio and video on a per-client basis during a meeting. For more information

about limiting bandwidth usage, see Chapter 10, "Network Bandwidth Considerations."

Multipoint data conferencing

Two or more users can communicate and collaborate as a group in real time.

Participants can share applications, exchange information through a shared clipboard, transfer files, collaborate on a shared whiteboard, and use a text-based chat feature. Also, support for the T.120 data conferencing standard enables interoperability with other T.120-based products and services

Application sharing

A user can share a program running on one computer with other participants in the conference. Participants can review the same data or information, and see the actions as the person sharing the application works on the program (for example, editing content or scrolling through information.) Participants can share Windows-based applications transparently without any special knowledge of the application capabilities.

The person sharing the application can choose to collaborate with other conference participants, and they can take turns editing or controlling the application. Only the person sharing the program needs to have the given application installed on their computer.

Shared clipboard

The shared clipboard enables a user to exchange its contents with other participants in a conference using familiar cut, copy, and paste operations. For example, a

participant can copy information from a local document and paste the contents into a shared application as part of a group collaboration.

2.7.1 Visual Basic

File transfer

With the file transfer capability, a user can send a file in the background to one or all of the conference participants. When one user drags a file into the main window, the file is automatically sent to each person in the conference, who can then accept or decline receipt. This file transfer capability is fully compliant with the T.127 standard.

Whiteboard

Multiple users can simultaneously collaborate using the whiteboard to review, create, and update graphic information. The whiteboard is object-oriented (versus pixel-oriented), enabling participants to manipulate the contents by clicking and dragging with the mouse. In addition, they can use a remote pointer or highlighting tool to point out specific contents or sections of shared pages.

Chat

A user can type text messages to share common ideas or topics with other conference participants, or record meeting notes and action items as part of a collaborative process. Also, participants in a conference can use chat to communicate in the absence of audio support. A new "whisper" feature lets a user have a separate, private conversation with another person during a group chat session.

2.7 Software Development Tools

2.7.1 Visual Basic

Visual Basic is very powerful application programming language founded by the Microsoft used to write Windows-based computer programs. With Visual Basic, programmers could for the first time implement Windows applications in an intuitive, graphical environment by dragging controls onto a form. By enabling both professional and casual programmers to maximize their productivity, Visual Basic ushered in a renaissance of Windows-based application development. The Visual Basic software development is very broad and most of this research will focus on network software development.

Winsock

Winsock is an open network API standard. It was first designed to create a standard programming interface for TCP/IP on all versions of Microsoft Windows including Windows 2000, Windows NT, Windows 95/98, Windows CE and Windows 3.x. The major benefits of Winsock to date have come in 3 areas. First, by providing an open API standard rather than a closed proprietary API, it has helped foster the success of TCP/IP on Microsoft Windows operating systems. Second, application developers have been able to easily port applications from BSD Sockets source code to run on all Windows platforms. Third, it has made it much easier for end users and IT managers to find a wide selection of the applications to choose from that can run without modification "out of the box".

WinSock is far from the only network API available to programmers on Microsoft Windows. There are any number of APIs available that provide access to high-level application protocols--like HTTP, FTP, SMTP or SSL--and many object-oriented languages have class libraries specifically designed for this purpose (e.g. Java 2's "net" core class library, MFC's CSocket/CAsyncSocket classes, and many Active-X libraries for Visual Basic). A common question is, "why use WinSock when there are other APIs that can simplify development?" There are two primary answers to this question: control and performance.

WinSock is as close to the TCP/IP (Internet Protocol Suite) as you can get, without implementing them yourself. As a result, your application minimizes the layers of other software it needs to access to gain access to the Internet transport and control protocols and services. This is important to maximize performance. Having first-hand access to error values and incoming and outgoing data adds to the complexity of development, but it also increases the control your application has. This increased control means you are not dependent on third party software if you encounter bugs and provides a means to give more meaningful user feedback as your network application executes, both of which mean fewer customer support calls.

There are a number of other considerations as well. Java is a relative newcomer to provide cross-platform portability. Berkeley Software Distribution (BSD) Sockets has been the de facto Internet API since 1986. Although WinSock is not 100% compatible with BSD Sockets, it supports a major subset of the APIs. As a result,

there is a large body of working source code to draw from and a large population of network application developers that are already familiar with the concepts and mechanics.

2.7.2 Java

Java is programming language founded by the Sun Microsystems. It robust and versatile, enable user to:

- Write software on one platform and run it on another
- Create programs to run within a web browser
- Develop server-side applications for online forum, stores, polls, processing HTML forums, and more.
- Write application for cell phones, two-way pagers, and others consumer devices.

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Java bytecodes —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java bytecode instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed.

Java Media Framework API

The Java Media Framework API (JMF) is an application programming interface for incorporating audio, video and other time-based media into Java applications and applets. This API was developed by Sun Microsystems, Inc., Silicon Graphics Inc., and Intel Corporation. The 1.0 API focuses on the playback of streaming and stored media, i.e. supports the synchronization, control, processing and presentation of such media. JMF is an optional package which extends the multimedia capabilities on the Java™ platform.

The JMF is a set of three new APIs being co-defined by the JMF Working Group members -- Sun, Silicon Graphics, and Intel. These APIs eventually will include Java Media Player, Capture, and Conferencing. The first to be delivered, the Player API provides a framework for implementors to build media players and provide them in a standard way on all Java platforms. The JMF specification is flexible enough to allow developers to extend players by adding their own nodes (such as images filters, audio reverb effects, and so on) or to use the standard players without making any additions.

Before the JMF Player API, multimedia playback support in Java was extremely limited. Programmers had to generate their own GUI controls. (JMF returns a standard set of controls in the form of a ControlPanel and other Control objects.) The supported media types in the core Java API were limited (Sun's muLaw format for sound, and no media option for video), so developers were forced to implement their own players without any underlying framework to assist them.

With the JMF Player API, however, Java programmers can implement support for almost any audio or video format by building upon an established media playback framework. In addition, standard implementations provide built-in support for common Web formats such as muLaw, Apple AIFF, and Microsoft PC WAV for audio, as well as Apple QuickTime video, Microsoft AVI video, and Motion Picture Expert Group's MPEG-1 and MPEG-2 for video. MIDI currently is supported in the Silicon Graphics IRIX implementation and is slated for support in Intel's Windows implementation. If you want to use one of these standard Web-based formats, you are now able to easily integrate multimedia playback into applets and applications alike with only a few lines of code.

JMF allows player implementors to use native methods as need be underneath the covers for greater speed. This lets the implementors optimize performance on each platform. At the same time, the common Java Media Player API ensures that applets and standalone applications will run on any Java platform.

Conclusion

WinSock is the choice among network application engineers for a number of reasons. It provides the ultimate in binary portability among Microsoft Windows platforms. It provides source compatibility with many legacy Internet applications, and a familiar interface for many network programmers. It can maximize performance and control, and the enhanced APIs in WinSock version 2 support the newest in protocol services in a simplified fashion (such as Quality of Service and IP Multicast).

The JMF Player API is both powerful and simple to use -- evidence of the elegance of object-oriented programming and Java. The Player API provides the core functionality that Java needs to be a powerful multimedia processing and display platform.

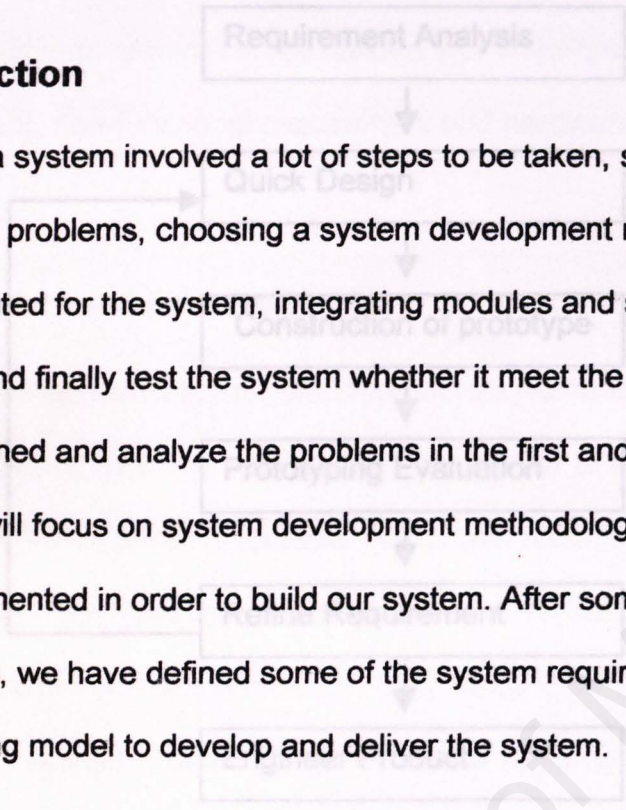
Everyone, from applet programmers with personal Web sites to player implementors at software companies and Web designers at content providers, will benefit in their newfound ability to deliver audio and video functionality in their programs. Future enhancements promise to add Capture and Conferencing tools, bringing Java in line with (or possibly ahead of) other languages in its multimedia processing and presentation capabilities.

Summary

The purpose of this research intended to further the knowledge and information to help the project development. This knowledge has covered the Internet and networking overview, advantage of digital transmission, voice digitization, security issue, case study of voice applications and programming languages study. All this information has help me in the sense of knowing the steps need to be taken towards building a voice communication package.

Chapter 3: System And Requirements Analysis

Introduction



Building a system involved a lot of steps to be taken, such as defining problems, analyzing problems, choosing a system development methodology to be implemented for the system, integrating modules and sub modules into complete system and finally test the system whether it meet the expected outcome or not. We have defined and analyze the problems in the first and second chapter, and this chapter will focus on system development methodology and requirement aspect to be implemented in order to build our system. After some discussion among ourselves, we have defined some of the system requirement and agreed to choose prototyping model to develop and deliver the system.

3.1 Prototyping Development Method

Prototyping is an approach where the system is being implemented as a prototype model and will constantly be revise, reanalyze and redesign for time-to-time in the system development life cycle. Through this development life cycle, the prototype model will eventually evolve into a complete system.

Figure 3.1 shows the basic flow among phases in a prototyping development. It consists of six phases, which is the requirement analysis, quick design construction of the prototype, prototyping evaluation, refine requirement, engineer product. The iteration of phases will start from quick design to refine requirement until a proper system has been developed.

3.2 System Requirements

The system requirements are divided into three groups, which is the functional requirement, non-functional requirement and hardware and software requirement.

3.2.1 Functional Requirements

Functional requirements are fundamental requirements of a system. It consist requirements of what the system must do to interact with its environment and fulfilling the system objectives. Functional requirement for this project is sub

divided into three modules: Whiteboard Module, Server Module and Voice Communication module. The Voice Communication module will be discussed here while other modules such as the Whiteboard module and Server module can be found in Chong Bing Suang and Lee Chee Cheng's reports.

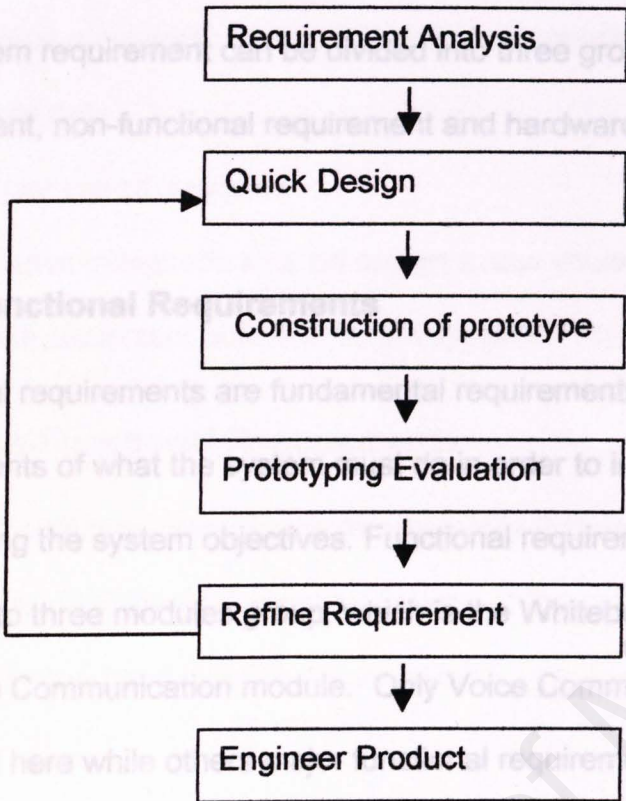


Figure 3.1: Prototype Model

There are several reasons on choosing the prototyping development method as the methodology for this project they are:

- Fasten the time for development and delivery of a system
- Reduce time cost on further investigation
- Involving user into the development process to enable the designer and user collaboration
- Manageable module can be defined easy (Key feature based first)
- VB as a RAD tools is very suitable for this type of development

3.2 System Requirements

The system requirement can be divided into three groups, which is the functional requirement, non-functional requirement and hardware and software requirement.

3.2.1 Functional Requirements

Functional requirements are fundamental requirements of a system. It consist requirements of what the system must do in order to interact with its environment and fulfilling the system objectives. Functional requirement for this project is sub divided into three modules group, which is the Whiteboard Module, Server Module and Voice Communication module. Only Voice Communication module will be discussed here while others major functional requirements such as the Whiteboard module and Server module can be found in Chong Bing Suang and Lee Chee Cheng's reports.

The functional requirements of Voice Communication module are

- Capable of sending and receiving real-time audio data.
Outgoing audio will be translated into audio data before sending across a network while an incoming audio data will be translated into sound and send to the speaker.
- Encryption and decryption of an audio data
To protect data confidentiality, encryption is available to handle all outgoing audio data and decryption to all incoming data.
- Compression and decompression of an audio data

- To save network resources and time consumed on sending an audio data across a network. Compression and decompression is available to handle all audio data.
- Object based designed
- To save designer's time on design a new voice communication object and make implement able into others system.

3.2.2 Non-Functional Requirements

Non-Functional requirements are requirements that are affecting the overall system performance, usability and security. Several of these non-functional requirements for this project have been listed down as below:

- Attractive / Interactive Interface

System must be in a user-friendly environment and can be used by anyone without prior knowledge.

- Efficiency

System must achieve efficient collaborative environment. Smooth processes of user-to-user collaboration must be achieved.

- Customizable

System enable novice user to customize their own layout to suit their needs

- Reliable

System able to performed all intended operations systematically and accurately. Fault-tolerance, fault-avoidance and fault detection must also be provided to increase the system reliability.

- **Response Time**

System able to performed all intended operations in a limited time constraint without any delay. Prompt can be made to user if an action takes a very long time length.

- **Adaptive Environment**

The system must be able to be changed as user requirements change

3.2.3 Hardware and Software Requirements

The hardware and software requirements of this project are as follows:

	Development Environment	Runtime Environment
Hardware Requirement	<ul style="list-style-type: none"> • Pentium II 500 Mhz PCs • 128 MB RAM • Standard input and output devices • A SVGA Graphic Adapter • Sound card 	<ul style="list-style-type: none"> • Pentium II and above. • 64MB RAM and above. • Standard input and output devices. • A SVGA Graphic Adapter • Sound card • Network setup
Software Requirement	<ul style="list-style-type: none"> • Visual Basic 6.0 IDE • Ms Office 2000 and Html Help Workshop for documentation • Windows 98se/2000 platform • Macromedia Flash 5.0 • Adobe Photoshop 6.0 	<ul style="list-style-type: none"> • Windows 98/98se/2000 • Ms Office 2000

Chapter 4: System Design

Introduction

System Design is a phase where the system is being design based the system and requirements analysis done on previous chapter. It consists of system architecture design, system functionality design and the user interface design. Though, this design will only cover mainly basic features of the system and the database design and others advance features are purposely being left out in this prototype to simplify the prototype implementation.

4.1 System Architecture

Large system normally divided into modules that provide some related set of services. The initial design process of identifying these sub-systems and establish a framework is called architectural design. Sub system normally defined as a system that do not depend on services provided by other sub systems but it still can actually interact to each other through a proper interfaces defined by designer. As for modules, it is a system component that provides one or more services to other modules. Besides that, it also makes use of to other modules services.

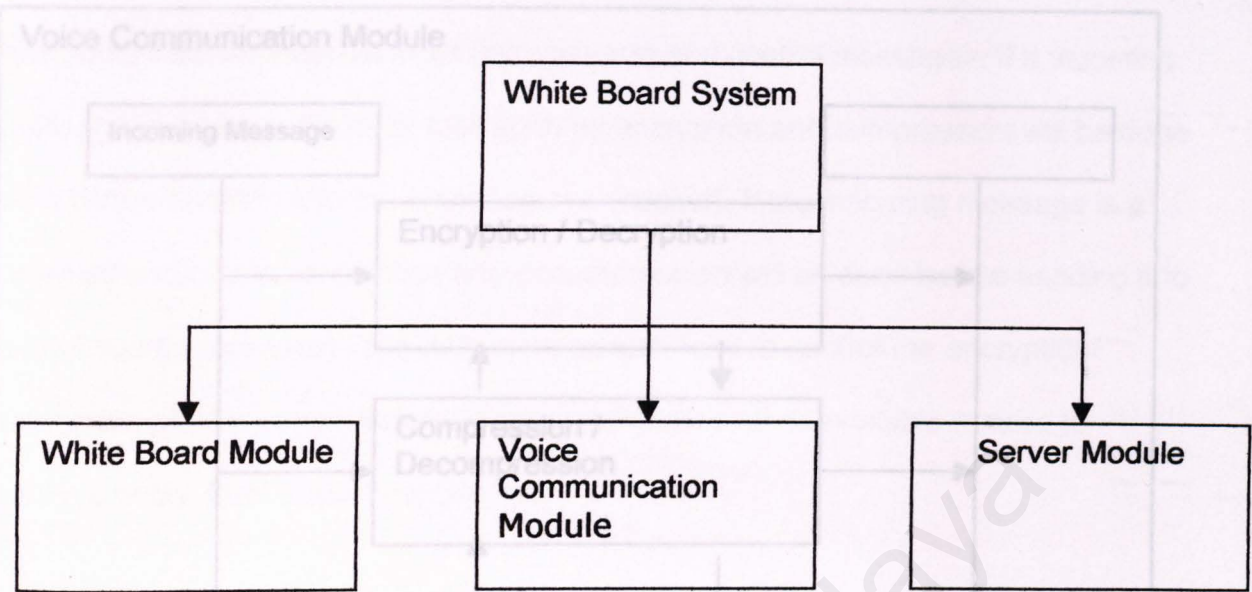


Figure 4.1.1 System Architecture of Whiteboard

As show in Figure 4.1.1, Whiteboard system can be decomposed into 3 modules, which are the Whiteboard module, Voice Communication module and the Server Module. Each of these 3 modules is dependent on each other to act as a complete system. Only Voice Communication Module will be discussed here, please refer to Chong Bing Suang and Lee Chee Ching's report for other modules descriptions.

Figure 4.1.2 Voice Communication Architecture

The Voice Communication Module serves as a module to perform real audio talks to the user. An incoming message for the system will activate the voice communication module to perform certain tasks on it and then pass back as

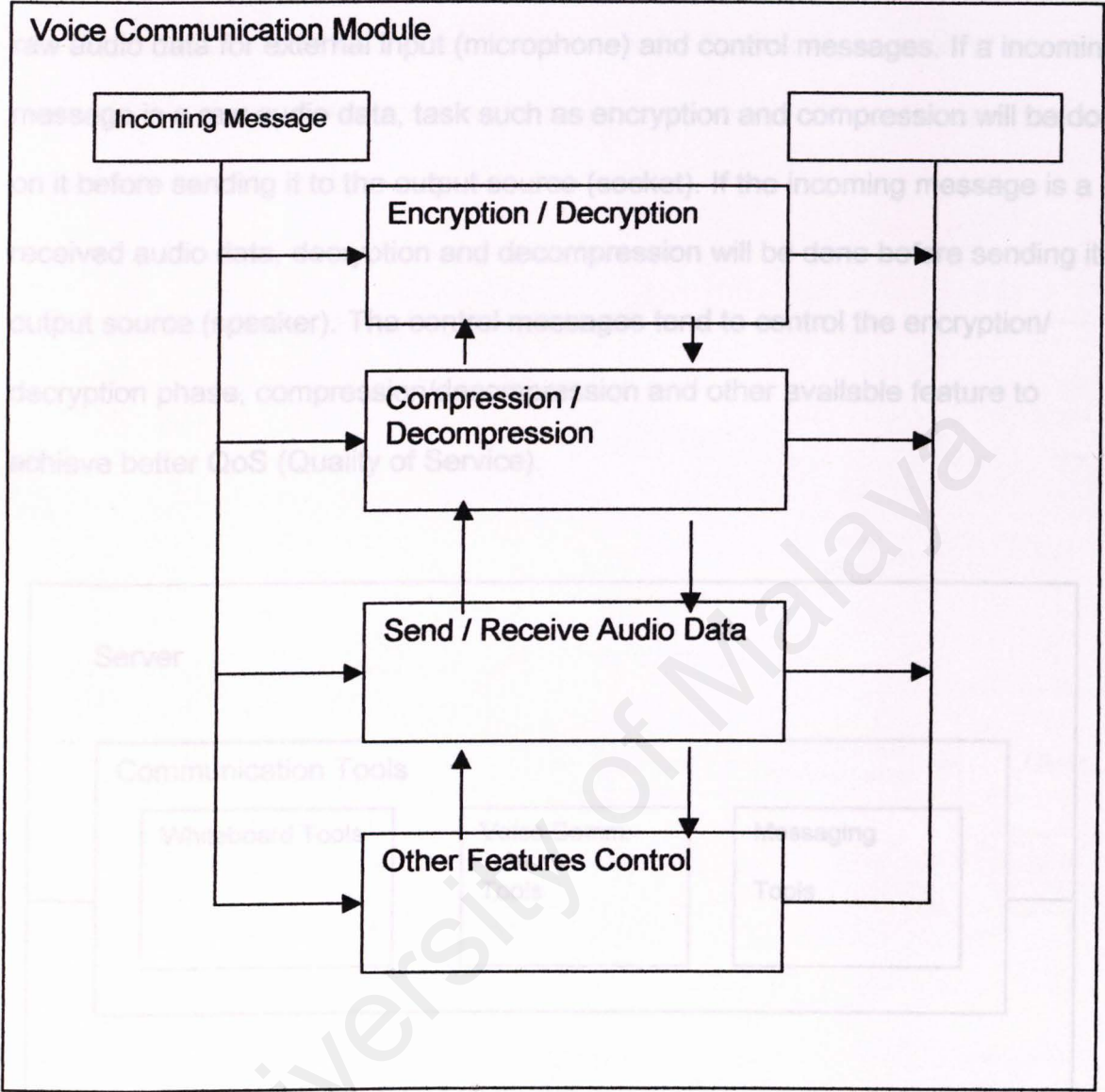


Figure 4.1.2 Voice Communication Architecture

The Voice Communication Module serves as a module to perform real audio functions to the user. An incoming message for the system will activate the voice communication module to perform certain tasks on it and then pass back as

outgoing message to output. Incoming message examples are received audio data, raw audio data for external input (microphone) and control messages. If a incoming message is a raw audio data, task such as encryption and compression will be done on it before sending it to the output source (socket). If the incoming message is a received audio data, decryption and decompression will be done before sending it to output source (speaker). The control messages tend to control the encryption/ decryption phase, compression/decompression and other available feature to achieve better QoS (Quality of Service).

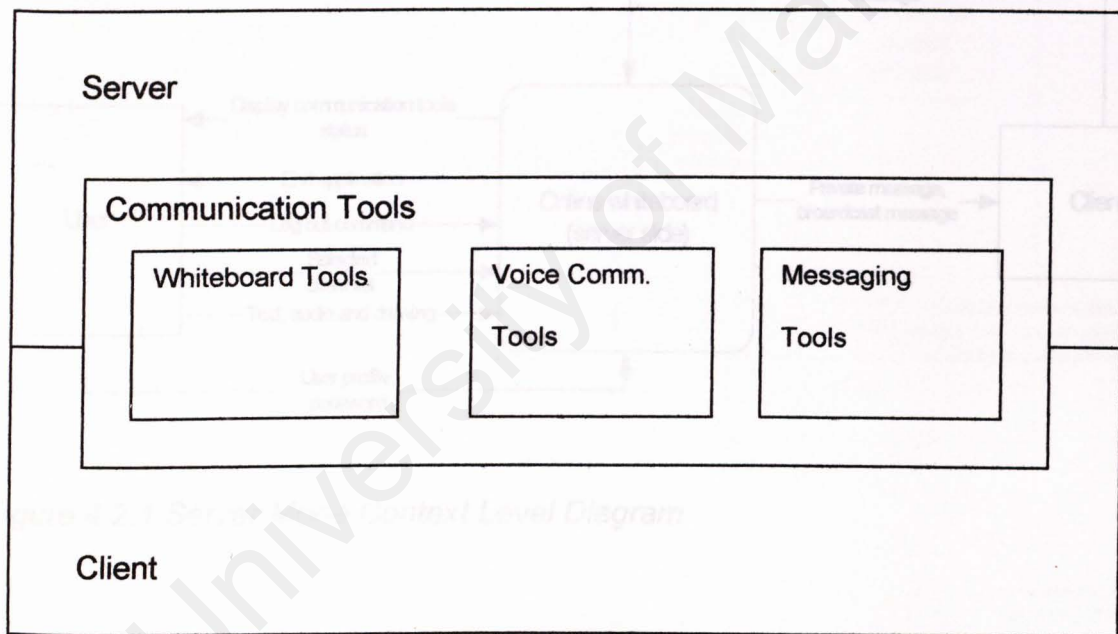


Figure 4.1.2 Communication Tools Layout.

Due to this system is a distributed system, so the client and server architecture is being considered in designing the system. Figure 4.1.2 shows the view of the Whiteboard System's communication tools layout based on client server architecture

design. All of the module are combined into a system as a communication tools and interact with each other as a client or server mode. Further explanation of the data flows of the communication tools in client and server mode will be discussed later in system functionality design.

4.2 System Functionality Design

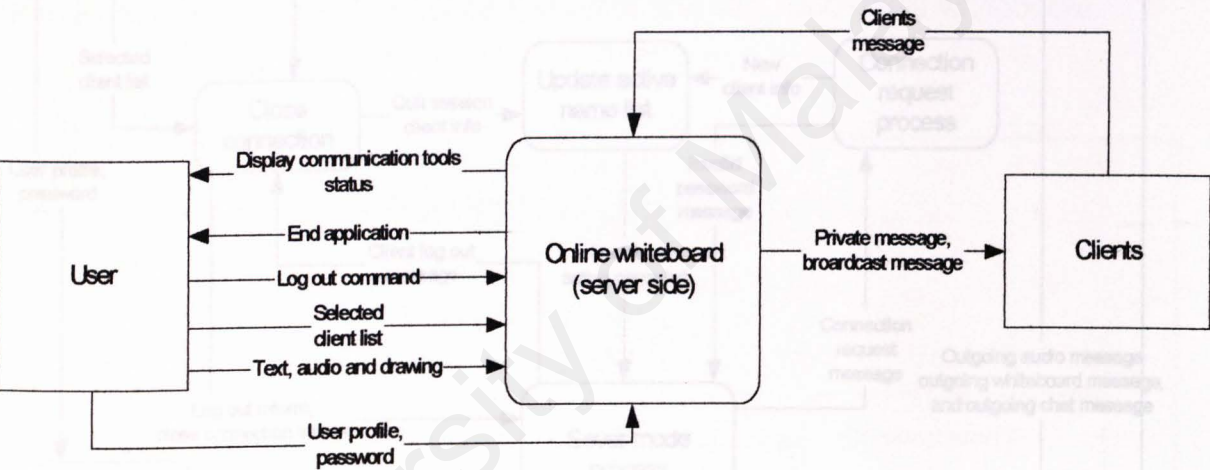


Figure 4.2.1 Server Mode Context Level Diagram

Figure 4.2.1 shows a server mode context level DFD of the Whiteboard System. To login to a whiteboard server mode, the user has to provide a user profile (username) and a group password to the system. After logging into the server mode there are actually three types of input able to be made by a user now on. There are the log out command, selected client list and data (text, audio or drawing). On receiving the

client message the server could either response to it using private or broadcast form to handle the sending.

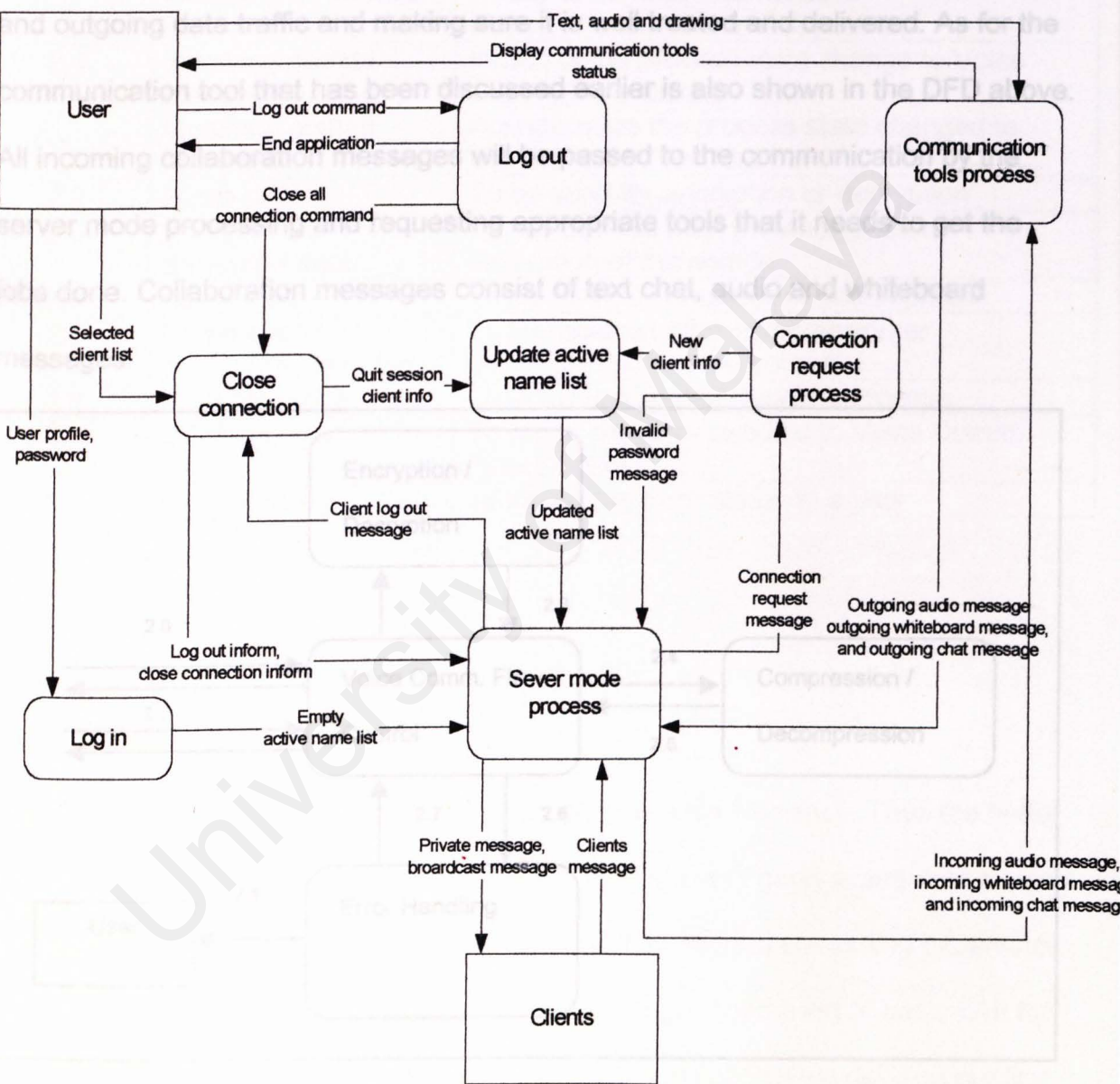


Figure 4.2.2 Server Mode Level 0 DFD

Figure 4.2.2 above shows a more detail of the context level diagram. It consists of more detail server mode data flows and the processes involved in handling the data.

Some of these processes are logging in, logging out, connection request, connection closed, update active name list, server mode processing and communication tools request. The core of the entire diagram is the server mode processing and the communication tools request. The server mode processing in control all incoming and outgoing data traffic and making sure it is well treated and delivered. As for the communication tool that has been discussed earlier is also shown in the DFD above. All incoming collaboration messages will be passed to the communication by the server mode processing and requesting appropriate tools that it needs to get the jobs done. Collaboration messages consist of text chat, audio and whiteboard messages.

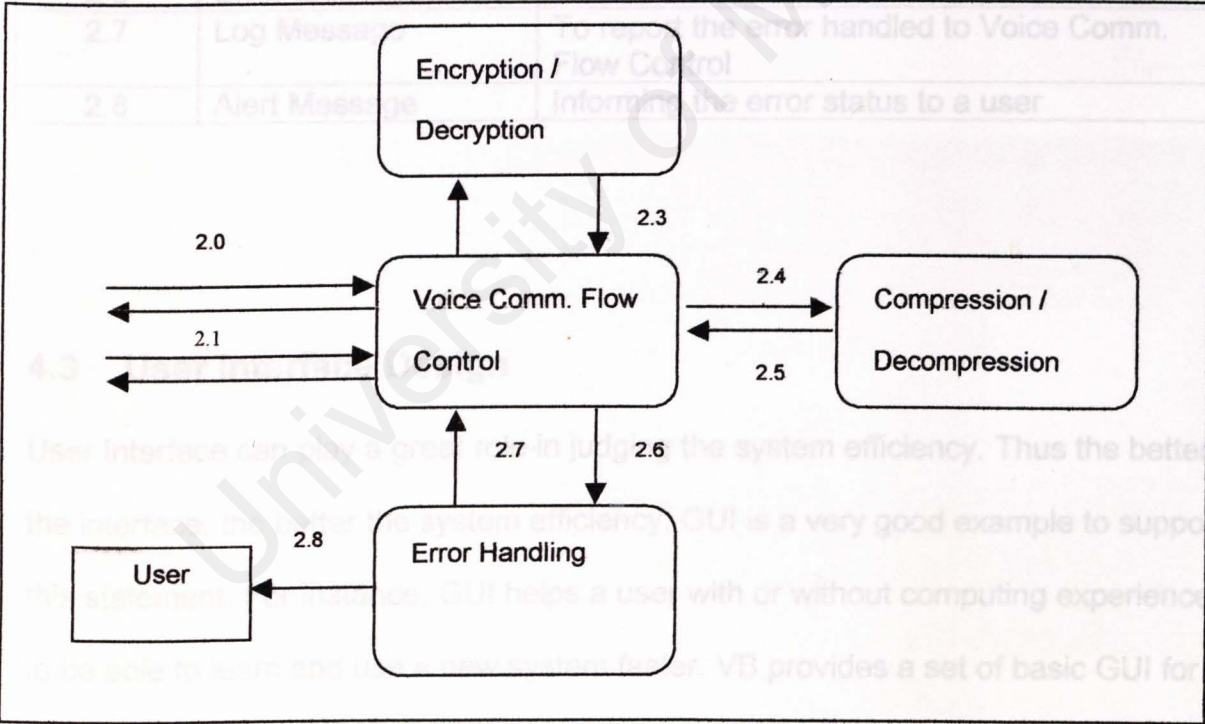


Figure 4.2.3 shows the communication tools level 2 DFD and below is a description chart on all the data flows in the voice communication tools level 2 DFD.

Reference	Type of Data	Description of Usage
2.0	Incoming raw audio Incoming audio data Outgoing audio data Outgoing raw audio	Going through encrypt and compress process Going through decrypt or decompress process Ready to send across network Ready to send to output (Speaker)
2.1	Incoming control message Outgoing control message	Inform of the process state change to Voice Comm. Flow Control Acknowledge the process state changed to Voice Comm. Flow Control
2.2	Audio data	To be send for encryption or decryption
2.3	Encrypted or decrypted data	To be received after the encryption or decryption of the source
2.4	Audio data	To be send to compress or decompress
2.5	Compressed or decompressed data	To be received after compression or decompression of the source
2.6	Error Message	To activate the error handling process
2.7	Log Message	To report the error handled to Voice Comm. Flow Control
2.8	Alert Message	Informing the error status to a user

4.3 User Interface Design

User Interface can play a great role in judging the system efficiency. Thus the better the interface, the better the system efficiency. GUI is a very good example to support this statement. For instance, GUI helps a user with or without computing experience to be able to learn and use a new system faster. VB provides a set of basic GUI for developer to develop their system and though these set of GUI we develop our first prototype model Interface.

Our first prototype model consists of two forms, which is the login form, and the main form. Figure 4.4.1 shows the login form of our system. A user will be prompted to insert a user name and a group password. Then the user will have to choose either as either server mode to start the application.

A screenshot of a 'Login' dialog box. It has a title bar with 'Login' and a close button. Inside, there are two text input fields: 'User Name:' and 'Password:'. Below these are two radio buttons: 'Server Mode' and 'Client Mode'. At the bottom are two buttons: 'Login' and 'Cancel'.

Figure 4.4.1 Login form

The second form of our prototype model consists of five tabs, which are the Session, Paint, Audio, Chat and Preference.

A screenshot of the 'WhiteBoard System V 1.0 - Not connected' window. It has a title bar with the text and a close button. Below the title bar are five tabs: 'Session', 'Chat', 'Paint', 'Audio', and 'Preference'. The 'Session' tab is selected. Inside the 'Session' tab, there are three text input fields: 'Server Name:' (containing 'myServer'), 'Server IP address:' (containing '192.168.0.1'), and 'Server Port:' (containing '1234'). To the right of these fields are two buttons: 'Stop Service' and 'Start Service'. Below these fields are two status indicators: 'Local status:' with a dropdown menu showing 'Available', and 'Remote status:' showing 'Not connected'. On the right side of the window, there are two list boxes: 'Active List' (containing '[Not Connected]') and 'Contact List' (empty). Between these two list boxes is an 'Add To List' button.

Figure 4.4.2 Session Layout

other through text. All received message will be put on the Received messages box

Figure 4.4.2 show the first tab of the main form, which is the Session tab and it consists of the all-basic connection setup and status figures. If it is in a server mode, after the user press the start button, it will start listening for connection establishment. While in client mode, the start button as to connect to the dedicated server. Server name or the server IP address must be supply by the client in order to establish connection. The password enter previously in the login form will be used to determined if it is allowed to enter the group or not.

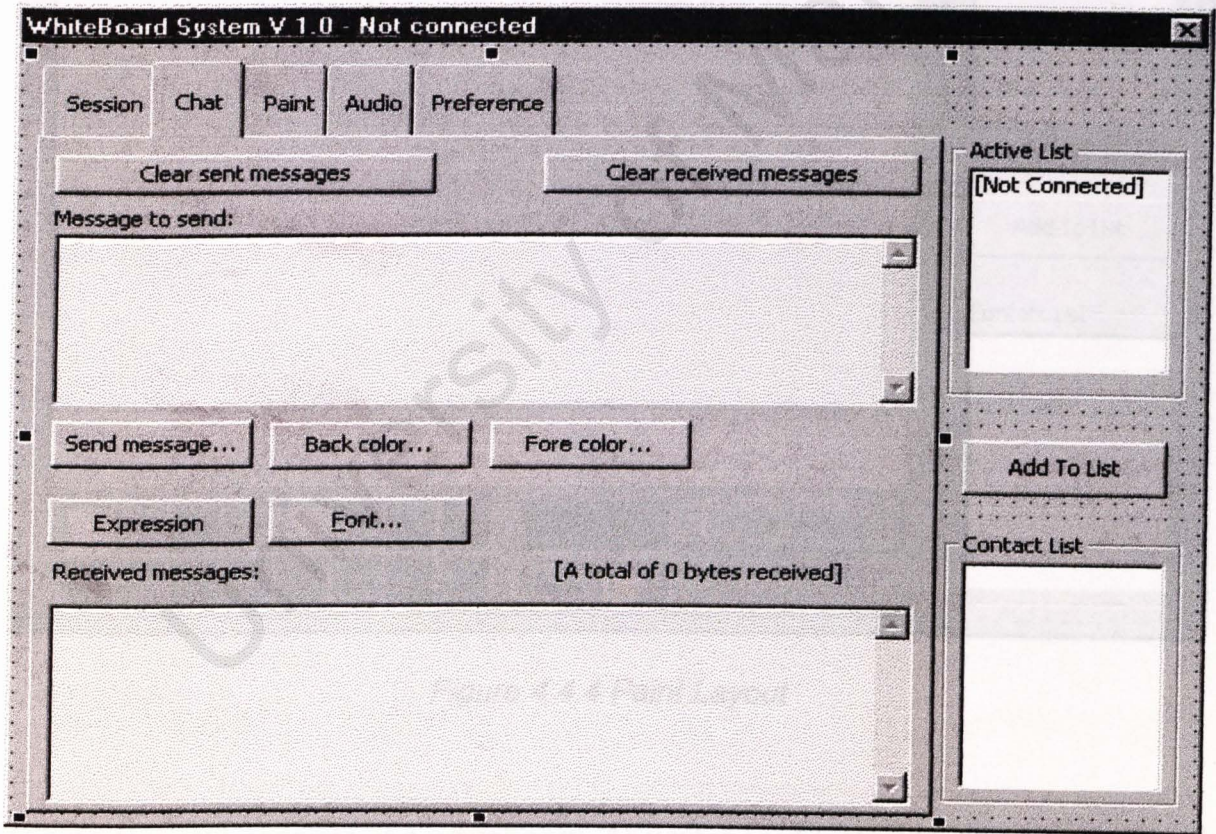


Figure 4.4.3 Chat Layout

Figure 4.4.3 show the second tab of the main form, which is the Chat tab. This is layout consist most of the messaging services that enable user to interact to each

other through text. All received message will be put on the *Received messages* box and outgoing message will be put on *Message to send* box. Basic text messaging capability such as show expression, change text font, change text color are also available in this prototype.

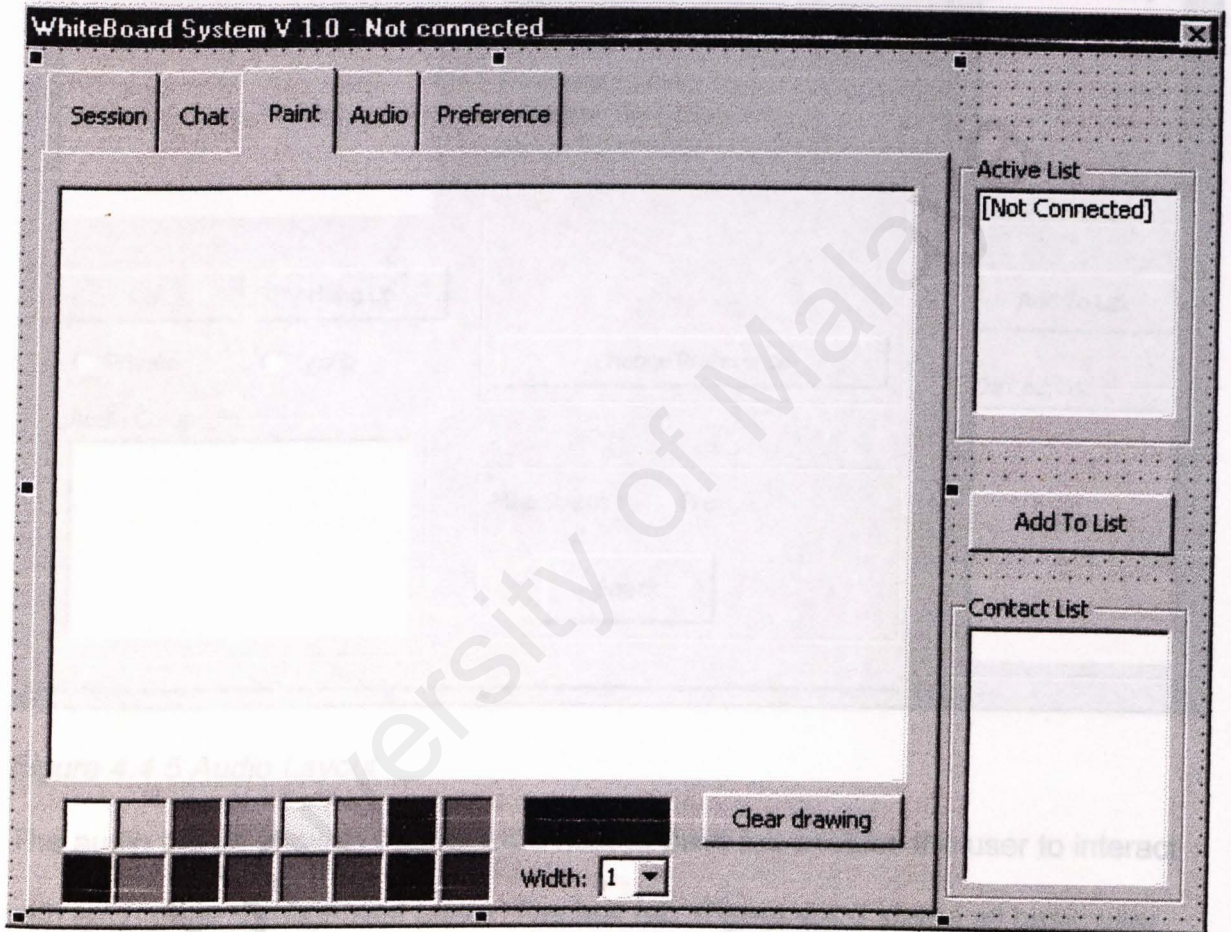


Figure 4.4.4 Paint Layout

The paint layout consists of a whiteboard. All whiteboard features will be constantly add in to this layout. For the time being, a whiteboard user is only allow to change the color and size of a pen.

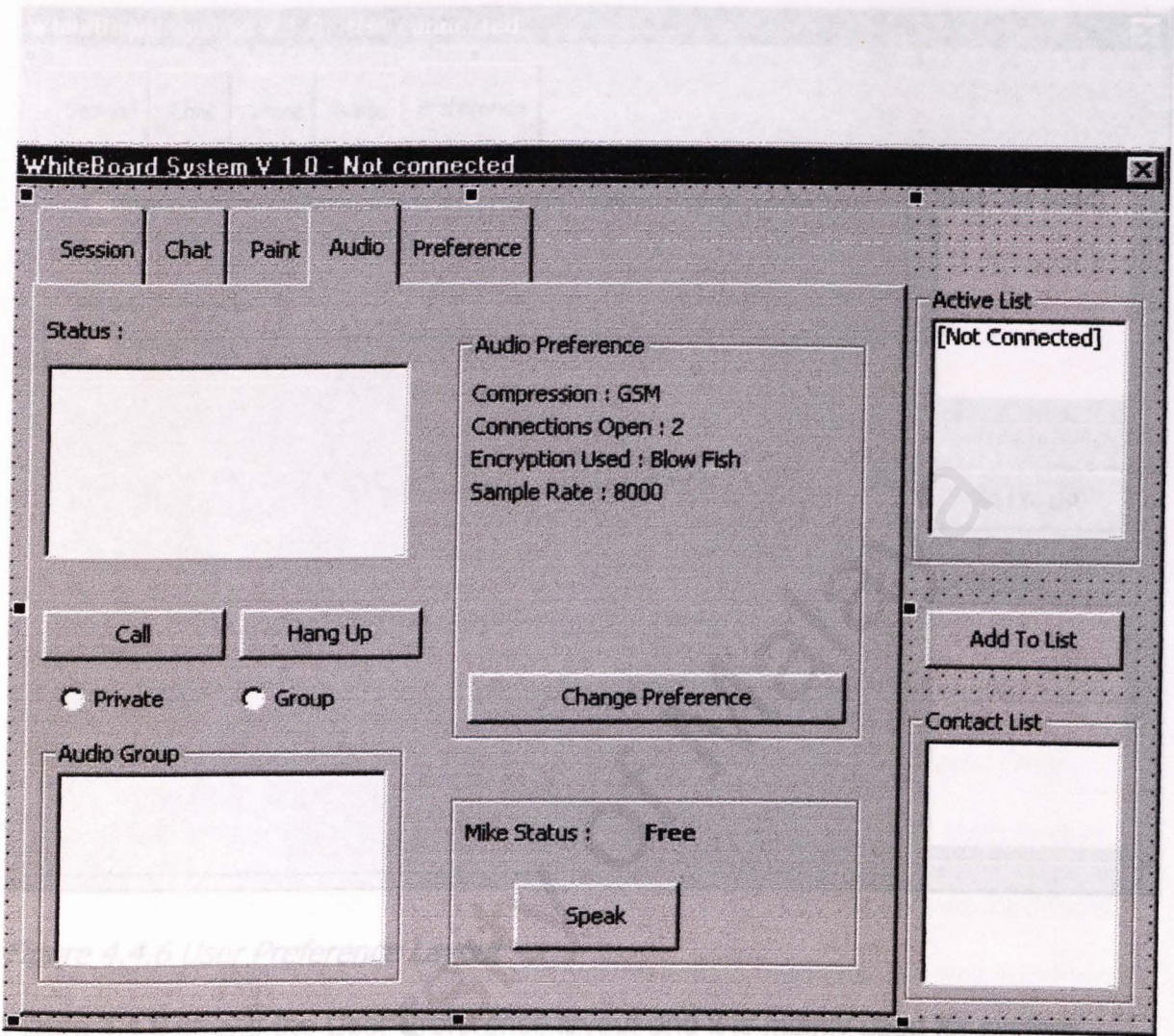


Figure 4.4.5 Audio Layout

The audio tab as show in Figure 4.4.5 next to paint tab enables the user to interact with each other through real audio. The last tab, Figure 4.4.6 stated all basic user preference.

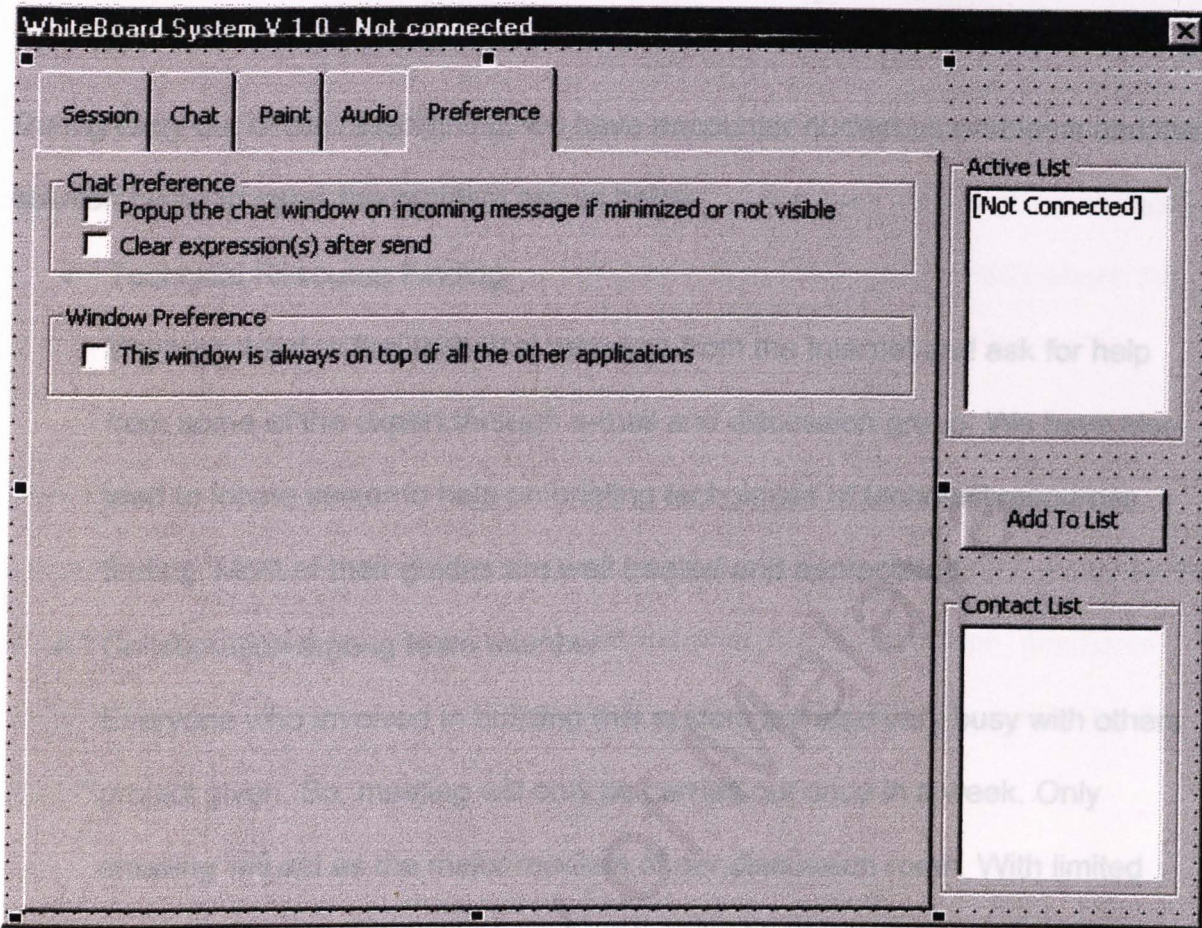


Figure 4.4.6 User Preference Layout

Figure 4.4.6 shows the User Preference layout and it is mainly used for system preference setup and control.

Project Discussion

During carry out of the researching, we have encounter numerous problems and the solution taken to solve the problem are as below:

- Technical Resource Finding

We have tried to find technical resource from the Internet and ask for help from some of the expert through e-mail and discussion group. We have also tried to locate senior to help on briefing techniques of technical resources finding. Most of their guides are well treated and appreciated.

- Collaboration among team member

Everyone who involved in building this system are also very busy with others project given. So, meeting will only be carried out once in a week. Only emailing will act as the major medium of our discussion room. With limited discussion time, we still manage to solve most of the problem we had faced like task distribution, schedule, module tolerate, resource tolerate and many others.

- Integration problem.

Since we may not be building the system together in a same environment, the system may not be exactly what we expected to be. Further more each of us has his own idea of viewing the system abstractly will also make the system even more complex. A solution to this is to build simple prototype to help us view the system objectively and test on the module tolerance ratio. With prototyping, we can first build a sample consists of major system functions and slowly increasing functionality and enhance it to achieve our goals

Conclusion

This research is done in order to help us to adapt to the R & D process and hence capable of develop and deliver any system in the future when we are given chance to. Besides that, it also help us a lot in the next half thesis (WXET 3182) where the actual implementation is done.

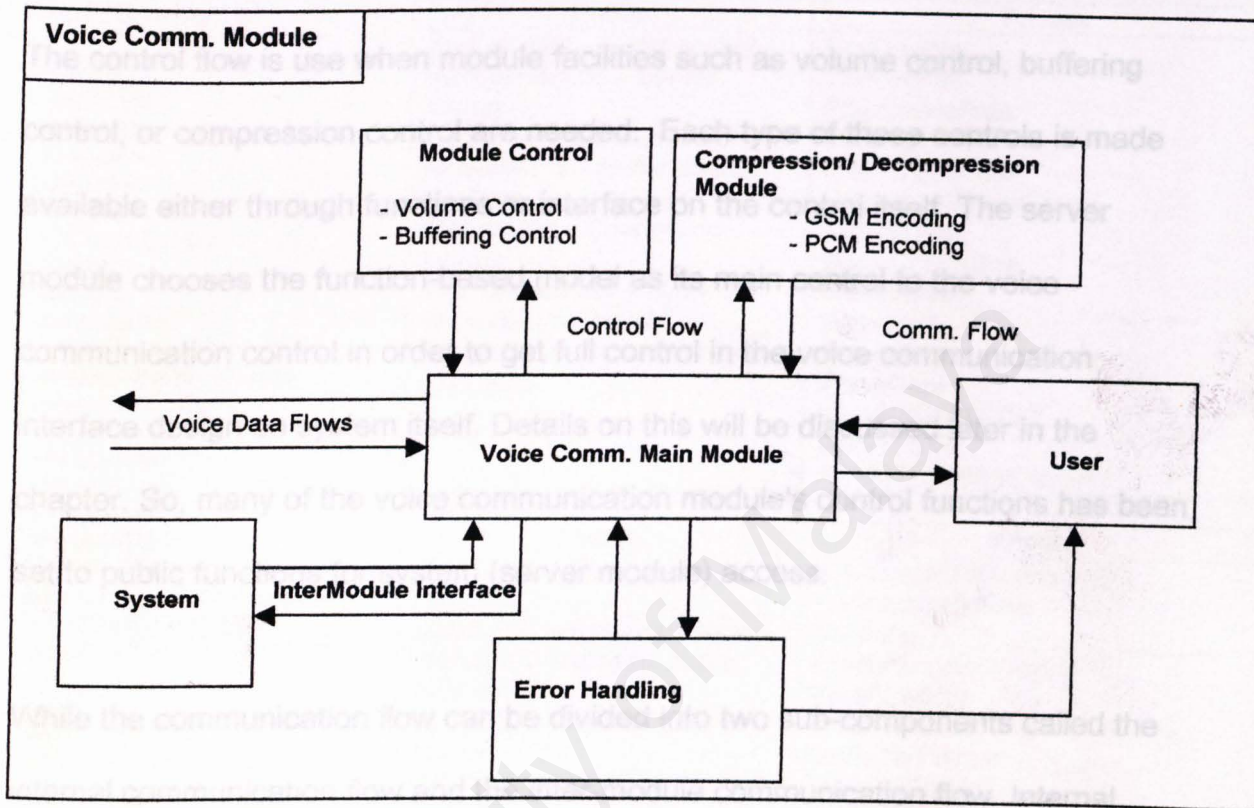
After the research, I now have a clearer idea on developing a system and at the same time gain a lot of knowledge in voice communication over network. Analysis on the development tools, requirements specification, architectural design, functional design, database design, user interface design will be very helpful in continuing the project.

Most of time we have spent in this project is in resources finding. It took me a lot of time, as the voice communication over network does not really have much technical resource that able to guide me through this project. Though this does not stop me to continue with this project. Some guidance from our senior does help me a lot in succeeding this research.

Lastly, I do hope that this project will progress smoothly and reach our entire expected outcome.

Project Discussion 2 [WXES 3182]

Actual System Implementation



PD 1.0 Finalize System Diagram

The diagram above shows the actual voice communication module implemented into the online-whiteboard system. From the diagram, we can clearly see that this module consists 4 types of data flow, which are the voice data flow, control flow, internal communication flow and inter-module communication flow. Voice data flow mainly used for capturing incoming voice data and releasing outgoing voice data in the network. Two sockets that are the *sckConnect* (Client Mode) and *sckConnection* (Server Mode) have been made available to the server module for handling such data flow and connection. When it is connected and a recording or

receiving event is signaled the recording processes and playing processes would be run independently to the system.

The control flow is use when module facilities such as volume control, buffering control, or compression control are needed. Each type of these controls is made available either through functions or interface on the control itself. The server module chooses the function-based model as its main control to the voice communication control in order to get full control in the voice communication interface design on system itself. Details on this will be discussed later in the chapter. So, many of the voice communication module's control functions has been set to public functions for system (server module) access.

While the communication flow can be divided into two sub-components called the internal communication flow and the inter-module communication flow. Internal communication flow acts as the data flow between the actual user and the module through the module interface it self (Ex. User dragging the volume-sliding bar implemented by the module and not the system itself). Volume control and error handling are among the data found the most in this type of flow. External or the inter-module communication flow is use when its facility needed external support from the system in order to run accordingly. Data such as the buffering control, control locking and connection control are among the functions who uses this type of flow (send to other connected clients to inform of stage changed).

Integration Module		
Global	MUT_CTRL; VOL_CTRL	
Enum : MUT_CTRL	MICROPHONE, MMICROPHONE, SPEAKER, MSPEAKER	
Enum : VOL_CTRL	MICROPHONE, SPEAKER	
Class : UserControl API used : Kernel32.dll - GlobalAlloc - GlobalFree - GlobalLock - GlobalUnlock - RtlMoveMemory - Sleep Winmm.dll - waveInOpen - waveInPrepareHeader - waveInUnprepareHeader - waveInAddBuffer - waveInStart - waveInStop - waveInReset - waveInClose - waveOutOpen - waveOutPrepareHeader - waveOutUnprepareHeader - waveOutWrite - waveOutClose - waveOutReset - waveOutPause - waveOutRestart - mixerGetNumDevs - mixerGetDevCaps - mixerOpen - mixerClose - mixerMessage - mixerGetLineInfo - mixerGetID - mixerGetLineControls - mixerGetControlDetails - mixerSetControlDetails Msacm32.dll - acmStreamOpen - acmStreamClose - acmStreamPrepareHeader - acmStreamUnprepareHeader - acmStreamConvert - acmStreamReset - acmStreamSize User32.dll SetWindowLong CallWindowProc	Properties :	
	Public	Private
	SckConnect – Client socket SckConnection – Server socket MbServer – Desired Mode MbBuffering – Buffer Enable Boolean MiNumCnn – Connection Loaded MlByteReceived – Bytes of voice data received MlByteSent - Bytes of voice data sent	CurPlayPos CurRecPos PlayDeviceFree Playing PlayWaveBuffer RecordingDeviceFree TIMESLICE WaveChunkSize WaveCodec Wave WaveArray WaveData WAVEFORMATX WAVEHDR ACMSTREAMHEADER
	Methods :	
	Public	Private
	CleafStuff – Clear state (Reset) MuteSet – Set Mute to device MuteGet – Get Mute status from device VolumeSet – Set Volume to device VolumeGet – Get Volume from device SetTalkCaption – Set Talk Button Caption SetTalkEnabled – Set Talk Button Enable	AddStreamToQueue AudioErrorHandler Cplaying CrecordingAndPlaying FreeACMHdr FreeWaveHdr IncBufferPointer InitACMCodec InitACMCodec InitWaveFormat InitWaveHdr LoadPlayBuffer PlayWave PlayWaveB RecordWave RecordWaveB SaveStreamBuffer RemoveStreamFromQueue SendSound SendSoundToAll SendToAll StreamInQueue WaitForACMCallBack WaitForCallBack
	Event:	
	Public	Private
	MyEvent- For user to implement code	

PD 1.1 Simplified Class Diagram

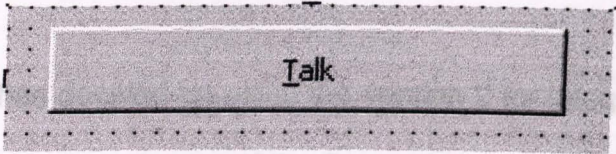
Interface

In order to ease the integration processes, information hiding or the Object Oriented model has been used to accomplish the job. Figure PD 1.1 clearly shows the object model view of my Voice Communication module. All the functions, properties and control in my module are combined into a single embedded object namely the Active-X control that are ready to used in the online-whiteboard system.

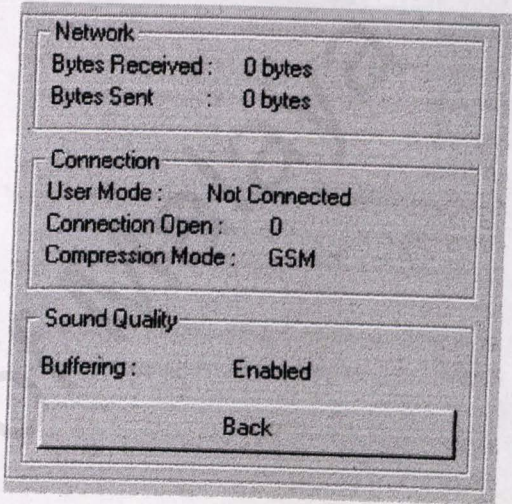
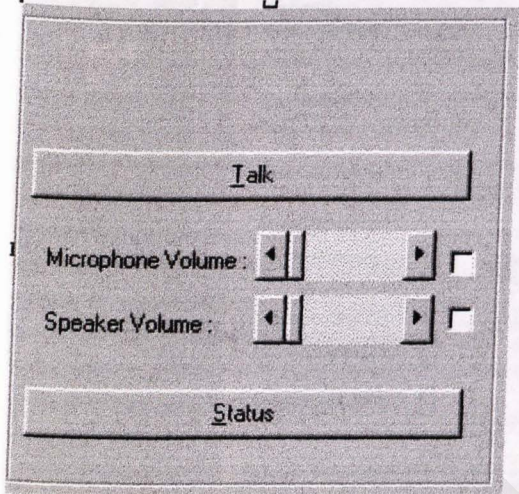
Active-X control implemented served as the embedded object in the system and most of my codes had been implemented into the control and thus eases the server module when integrating the system. Hiding the codes from the server module will certainly help in integrating as different people involved in developing different module and thus showing all the codes in a system can be a mess as debugging and error tracing is harder to be done. Meaningless functions to the system are hide and good interface is being developed for system (server module) access so that easier data management and control can be achieved.

From the diagram, I have also included a set of lower window API functions uses by the module in order to provide the module facilities. Most of the API functions are interact by the system (server module) indirectly through my interface (functionally). This can ease the system from understanding the whole unwanted and meaningless processes to the system. To be more clearly, some of the methods and properties are set public while some others are not depending on the system needs.

Interface as the developer favor and thus more powerful because developer involved in manipulate the control function in its integration.



Active X Control Version 2

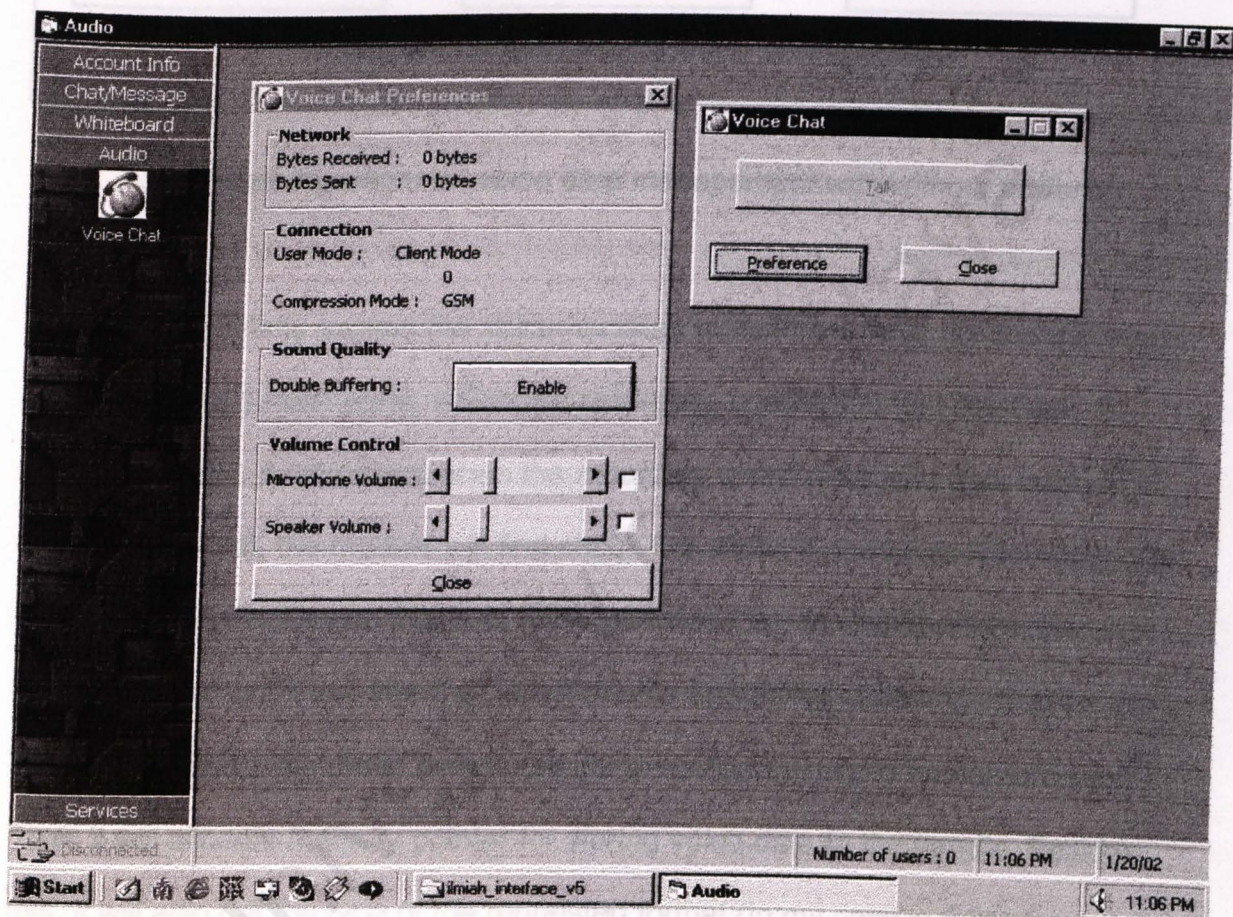


Active X Control Version 1

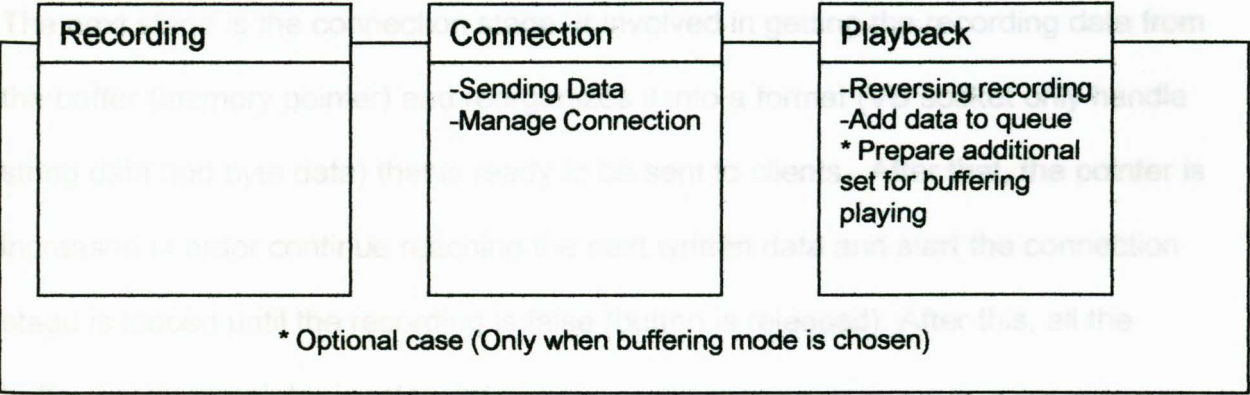
As shown above there are two version of the Active-X control being developed for the voice communication module. The mainly difference of the these two control is that version 1 based on the interface control which means most of the available status and control are already made available to the developer through the control itself and are not functionally achievable. While in version 2, all of the controls are functionally achievable to the developer. The advantage of version 1 over version 2 is that the developer can handle all of the integration only in a few lines of code because everything is already done for him. While the advantage on version 2 control over version 1 control is that it allows developer to redesign the interface of

the control as the developer favor and thus more powerful because developer involved in manipulate the control function in its integration.

The server module has decided on using the version 2 for integration into the system and the figure below is the outcome of the figure show.



Algorithm



Simplified Algorithm on voice data management with/without buffering

To be able to briefly explain the voice-handling algorithm, I have divided the whole processes into 3 main parts, which are recording, connection and playback. In recording stage, it happened when the recording event is flagged (talk buttons is clicked and hold), the module will first check the status it need (not Recording, not Playing, RecordingDevice free and PlayingDevice free) in order perform the recording function, if it is free then it will open a handle for capturing voice for the microphone and a handle to acm(audio compression manager) for compression. Then it will allocate memory pointer for storing voice data to the handle. Besides this, it will also get the type of the requested format (In my case is PCM format) for the handle in order to prepare the recording device that supports such format. PCM is chosen because it worldwide recognized and support by mostly all multimedia devices. Then the recording is issued yang data captured is passed to the acm handle for compression (converting to GSM format) and it is written into the buffer through the memory pointer.

Technical Documentation

The next stage is the connection stage. It involved in getting the recording data from the buffer (memory pointer) and reorganizes it into a format (VB socket only handle string data and byte data) that is ready to be sent to clients. After that, the pointer is increased in order continue reaching the next written data and start the connection stage is looped until the recording is false (button is released). After this, all the buffer, memory pointer is released.

Meanwhile on the client side that recently received the voice data, reversing process of the connection stage is done. Then the data in PCM format in put into the stream queue before it is free to play (in a way similar to recording process of handling).

Buffering is an additional feature added into the voice module that makes the voice received become smoother and thus enhanced the sound quality. But the trade off is that it will consume more memory and processing power and delaying the voice received. Without buffering the sound quality will decrease but the delay in receiving voice is solved. As conclusion to the buffering issue, one can considered of using buffering when its network condition is good and the processing power and memory are more that enough to handle the buffering. If the network condition is bad, buffering disable is the choice for better quality of service.

Technical Documentation

Through building this system, I personally feel that using Active-X in this project is a very wise step taken. This is because it is more or less implements some of the Object Oriented features and thus good for both side (the server module and voice module) in the whole integration process (interfacing) as I have discussed earlier in the chapter. Besides this, the Active-X also allowed its user to develop his or her library choosing either Active-X dll or Active-X control. Active-X dll is a fully Object Oriented concept where all the classes developed is put in the library and it is fully focus on function-based. While Active-X control take advantage on the basic control and thus focus more on control-based. If the user prefers less coding than ActiveX control is the choice and if the user prefers powerful in controlling than ActiveX dll is the choice use for development. One of the major disadvantages of the VB ActiveX control is the error management. The error management in this type of control is not tight enough and some time it do have unsolvable flaw in it that made in imperfect.

Since WXES 3181 last finding, not much resources on wave format have been found useful in VB. This had cause the development in the module can only be done in either in PCM or GSM format. Less documentation had also brought failure in developing others (fail on test using) than this two format. Most resources regarding to the wave format definition are written in C++ language format and differences in the wave format interface between C++ and VB make it even harder to trace the error occurred.

Besides this, most of the time had been spent on discovering the windows API function needed to directly interact with the multimedia device. Two of windows API were found to be very useful and powerful for developing this type of project is the wsacm32.dll and the winmm.dll. Winmm.dll is the library that allowed its user to directly interact with the device to perform most of the multimedia function while wsacm32.dll is useful in convert wave data from one format to another. Most of the technical resource in the MSDN library on these topics are also limited thus means more time is spent to do testing and researching on the use of these libraries.

The third issue that I am going to discuss here is the VB winsock control. VB winsock control is a very developer friendly tool to use for developing network application. But as the result of developer friendly (can even handle string data type), it has also become a trade off to its usability in control to the network application being developed. Furthermore the winsock control based on the event handling still have some unidentified flaws yet not being discovered in it and that will cause the affect quality of application developed using this control. After finished developing the module, I have also found out that this winsock control is not very suitable tool to user for developing this kind of project because all the data sent and received are time critical. A better solution to this matter is to use lower API function to send or retrieve data through network and more likely to increase the speed of the module thus take more time to investigate and test. Since the server module and the whiteboard module uses this control in its development, I choose follow in order to standardize the system.

Due to the time constraint, encryption and decryption of the data is replaced by

Another issue to be discussed here is the connection used to develop this module.

This module uses only one connection for handling the sending and receiving of voice data. This is because the system architecture that we had agree on WXES 3181 is a client/server architecture and thus rely on server to perform most of the data passing. If it were to be implemented more than one connection per client to handle the voice data, server load will be increased and inefficient. Regarding on the simplex connection with event triggered, I choose this in my implementation because I believe that it can further cut down the use of resources of my module. The processing, memory plus network resources are only activated when the event is triggered and so most of the resources are still free to achieve by other application and module. For duplex connection, another connection (as thread) is need to be implemented in the module and run synchronously and continuously. This means a certain amount of resources is already allocated to the module even the module function is not activated (needed). For example, 2 threads are constantly using the memory, processor and network resources and thus slowing down other module in the system. Besides this, event triggered can also help to avoid annoying sound echo due too near communication over 2 pc. To implement duplex using VB, thread handling need to be done. But thread in VB is not easy to use even to the VB expert and not encouraged because any mishap will crash the whole application. So, in consider to this matter, duplex connection is not implemented in this module.

Due to the time constraint, encryption and decryption of the data is replaced by implementation of compression and decompression that I think is even more meaningful to the module itself. Comparing encryption/decryption and compression/decompression, the latter not only consuming processing power but it does cut down the network resources for communication over network. Furthermore, doing encryption/ decryption in VB is inefficient because its structure is not design to handle fast computation compare to Visual C++ or java.

Another problem I faced is the polymorphism in VB. Object Oriented model allows the method to be implemented by the user to the object in Object Oriented programming language such as C++ and java. But in VB, I have not done much research regarding to this matter due to time constraint factor. I need this because in my module I needed the user to written a function that needed to been run together with my control when an event is activated. Another theoretical way to solve this problem is to pass a function pointer to my control so that it can located the function and perform together in the event handler when the control event is activated. Luckily, by using the public access event feature the problem is managed to solved.

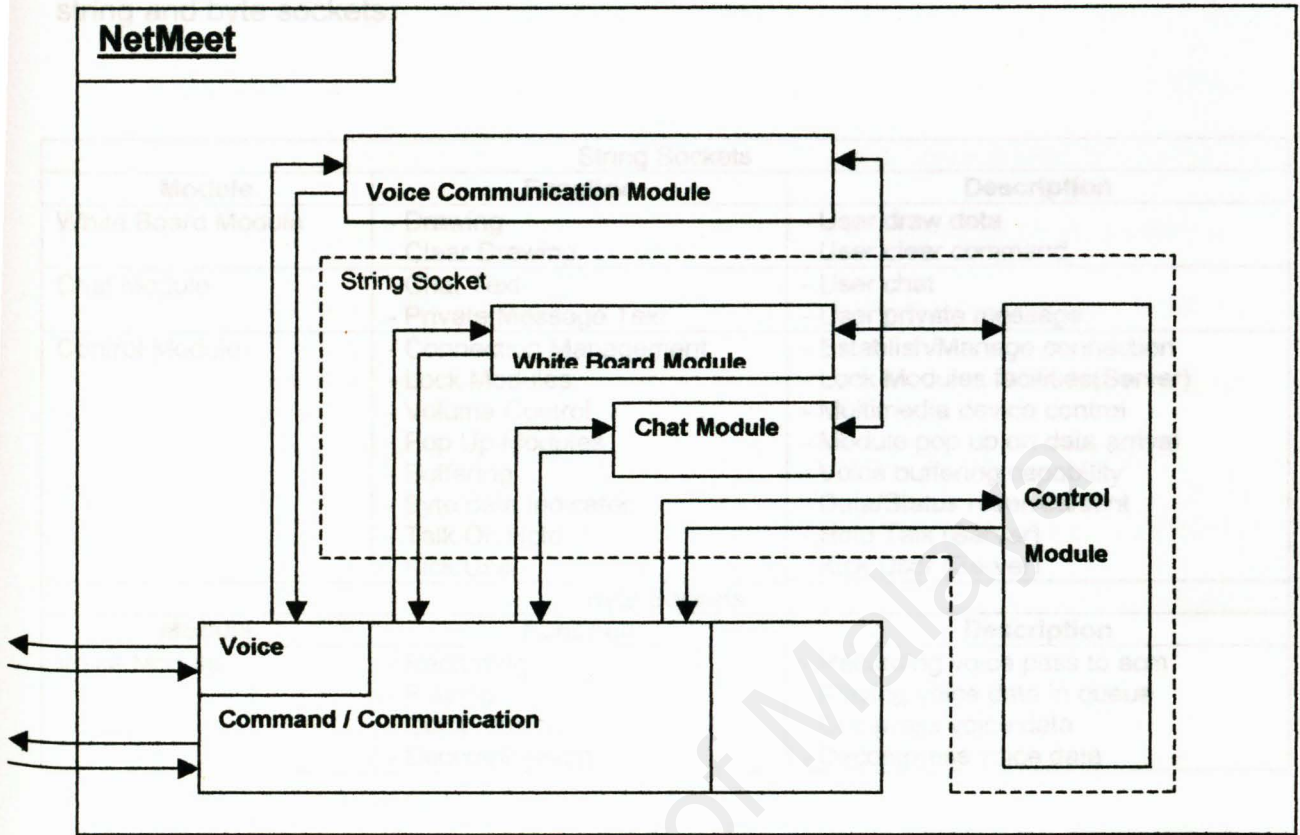
Active-X Control [NetMeet] implementation

NetMeet

To sum up the technical documentation I have enclosed a comparison table for future developers who consider development in this type of project according programming language preference.

		VB	Java	Visual C++
System	App. Speed	Fast	Slow	Fastest
	Development Speed	Fastest	Fast	Slow
	Readability	Easy	Intermediate	Hard
	Socket	Less. Control based winsock. Data type - String - Byte	Good. Object Based Data type -File Stream -Object Stream -Data Stream	Good. Object Based
	Error handling	Less. Flaws encountered	Good.	Good.
	Powerful	Less. Control Based	Yes. Object Based	Yes. Object Based
Voice Module	Module speed	OK. Utilizing API	OK. JMF package is provided	Best. Build from scratch
	Development speed	Fast	Fastest	Slow
	Reusability	Best. ActiveX is introduced. Either control based or Object based	Good. Object Based	Good. Object Based

Active-X Control [NetMeet] Implementation



Active-X Control Diagram

Diagram above shows the full Active-X Control (NetMeet) built with some of the whiteboard and chat features similar to NetMeeting control. This Active-X is made in full control based and that that means it is ready to be used without any configuration just as the net meeting control. In order to accomplish this control, 2 sets of sockets (sckConnect and sckConnection) are used. The control module, chat module and the whiteboard module, uses the first set of the sockets(String sockets) to handle data type in string while the voice module uses the second set of the sockets (Byte sockets) to handle data in byte. Below is a table that listed down some

of the functions that NetMeet capable to perform and it is separate according to string and byte sockets.

String Sockets		
Module	Function	Description
White Board Module	<ul style="list-style-type: none"> - Drawing - Clear Drawing 	<ul style="list-style-type: none"> - User draw data - User clear command
Chat Module	<ul style="list-style-type: none"> - Chat Text - Private Message Text 	<ul style="list-style-type: none"> - User chat - User private message
Control Module	<ul style="list-style-type: none"> - Connection Management - Lock Modules - Volume Control - Pop Up Modules - Buffering - Byte data Indicator - Talk On Hold - Kick User 	<ul style="list-style-type: none"> - Establish/Manage connection - Lock Modules facilities(Server) - Multimedia device control - Module pop up on data arrival - Voice buffering capability - Data/Status received/sent - Hold Talk (Server) - Kick User (Server)
Byte Sockets		
Module	Function	Description
Voice Module	<ul style="list-style-type: none"> - Recording - Playing - Compression - Decompression 	<ul style="list-style-type: none"> - Recording voice pass to acm - Playing voice data in queue - Compress voice data - Decompress voice data

Algorithm

In here, I will briefly discuss on the connection establishment, management and the interaction and communication between 2 sets of socket used in the control. This control mainly utilize some of the server module's idea written by Lee Chee Cheng on establishment where the host's string socket and byte socket is set to listening for connection. The user determines the string socket port while the control will in charge in picking a free random port for byte socket to perform listening. User that wish to establish connection will only need to connect to the host's string socket and on connection accepted, the host will reply with a byte port number to the initiate

client for further connecting the byte socket. The diagram below will help in clarified the whole process of a full connection establishment.

On connection management process, every new client than succeeded in connecting to the host will prompt the host to send a notification to other online clients. Then a new list of all connected user is send to the entire connected client by the host. If any of the clients is disconnected, the host again will send a notification to other online clients and follow a new updated online user list to the entire connected clients. When the host is disconnected, clients that detect the disconnection will also flushing all its state and back to the disconnected state again.

Regarding to the whiteboard module in NeetMeet, idea is taken from the online-whiteboard system whiteboard module written by Chong Bing Suang. The algorithm is revised and consulted by Chong Bing Suang before it is really implemented into the NeetMeet. While the control module and chat module is taken from Lee Chee Cheng's Server module in order to accomplish NetMeet. Then I do integrate all the modules this time, as I am the one that has the knowledge on this NetMeet ActiveX control.

Technical Documentation

The ActiveX control that I have developed for the application is a fully control based ActiveX control. This means all of the functions embedded can only be invoked through the ActiveX control interface. This is very similar to the netmeeting control designed by Microsoft. My control namely the "Netmeet" is also similar to the netmeeting control in way that it has the chat and whiteboard features included in it besides the voice communication package. Also as pointed out earlier my control uses 2 sets of TCP sockets per client to enable the communication package to be able function according. While the amount of sockets netmeeting use is far greater than the netmeet control.

Compared to Netmeeting, my control provides more server administration control to the developer. Function such as locking whiteboard and locking chat room are among the administration function that can be performed by the server. Besides this, Netmeeting architecture is peer-to-peer (directory based) while my NeetMeet is implemented in client / server architecture (details in earlier technical documentation).

The reason that I develop this ActiveX control is to make it similar to NeetMeeting control where user do not have to code (user friendly) to use the control yet can even extend and enhance or redesign the control as prefer to suit their need when

provided the source code. Its flexibility also allowed it user to even convert the control to fully function-based library to extend the capability of the NetMeet.

One of the major defect in this control is that is that the error handling process. VB is powerful but flaw do happen in the ActiveX control and such defect make the developer try avoiding in using it to their implementation. My way to counter this problem is to redesign the ActiveX control in Visual C++. Visual C++ as pointed out earlier in this chapter is a very powerful language. Its capability in the error and socket handling is undeniable compare to VB. Since it is fully Object Oriented Model, extendibility and reusability is even greater than VB. But using Visual C++ causes more time spent on researching and developing because it is not an easy language to use even for an expert in programming. It involved a lot of time to research and test on every sub modules developed.

So my forecast on enhancing this control will be on redesign a library (dll) in Visual C++ in order to increasing its capability before implementing it as ActiveX control to counter flaws in VB.

Conclusion 2 [WXES 3182]

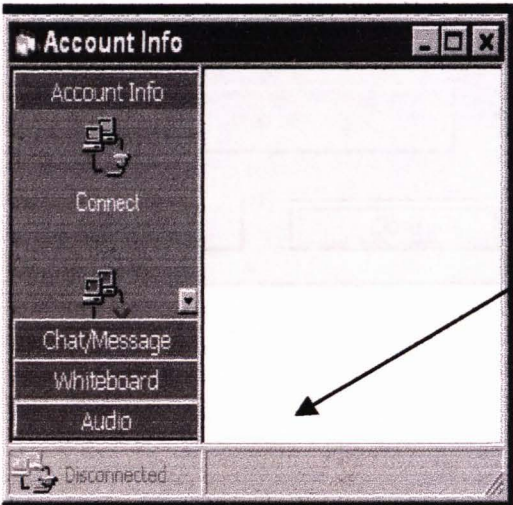
This project had really opened up our view in discovering the R & D process and environment in software engineering. After this project, I had gained a lot of knowledge not only in voice communication over network but also in the ActiveX field. My hope is that those who use my project as reference will also benefit from this.

Since most of my time had been spent on R & D on the developing tools, this has also help on sharpen and familiarizing me to the all programming language specification and power to determine what type of programming language that is suitable for a project. This experience certainly helps me a lot in future development of a project in consideration to time, speed and application performance that is preferable to client.

Lastly, I do hope that this project will benefit future developer and implementer that want to implement and use this project as their template or reference.

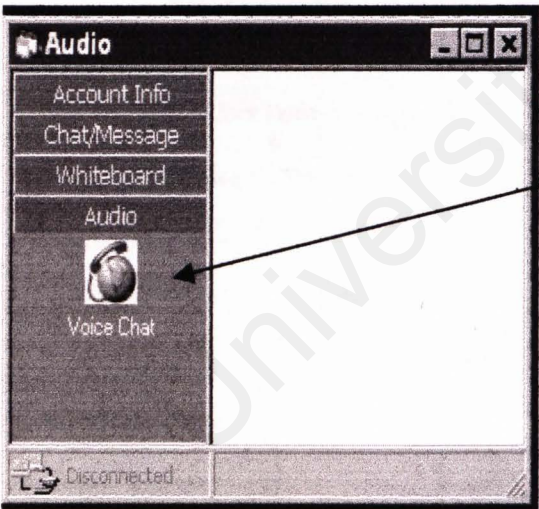
Appendix

ONLINE WHITEBOARD VOICE MODULE USER MANUAL

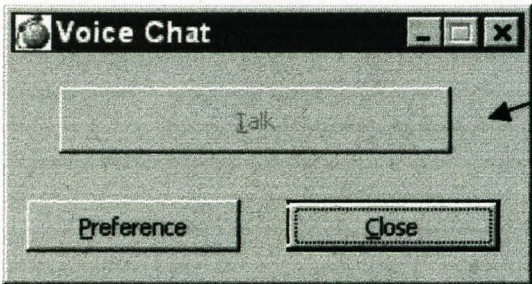


In order to facilitate the voice chat application, user must first in connected stage. (Please refer to Lee Chee Cheng Server Module on the regarding connection startup).

Assume that you are in connected stage to start using the voice chat, please select the audio panel to reach for the voice chat tool



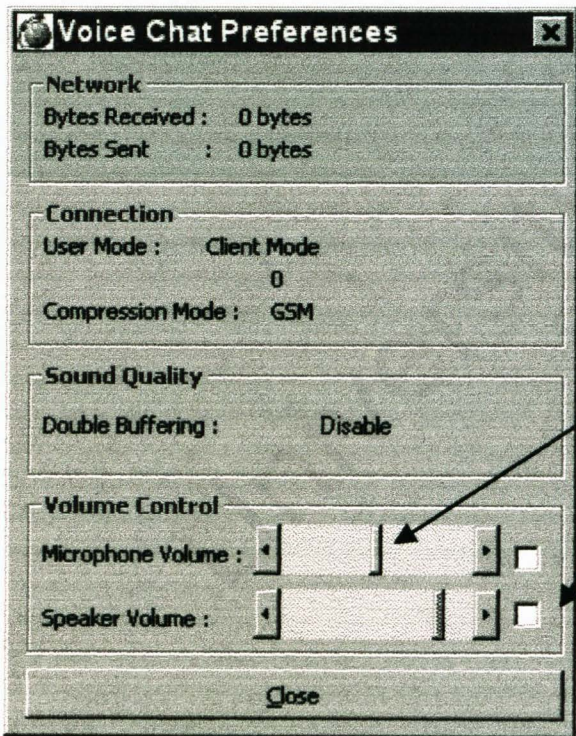
Click on the Voice Chat to start using the application. After clicking on the voice chat icon, the voice chat main form will appear in the program.



Talk Button : The talk button is only available to the user in order for a user to click and start send voice over a connected network.

Preference Button: The preference button is available for user to check the status on the voice chat application and changing the setting of the voice chat application

Close Button: This is mainly use to close the voice chat form. (Info: closing the form will never stop the received sound process, in order do so, please using the mute check box provided in the preference form

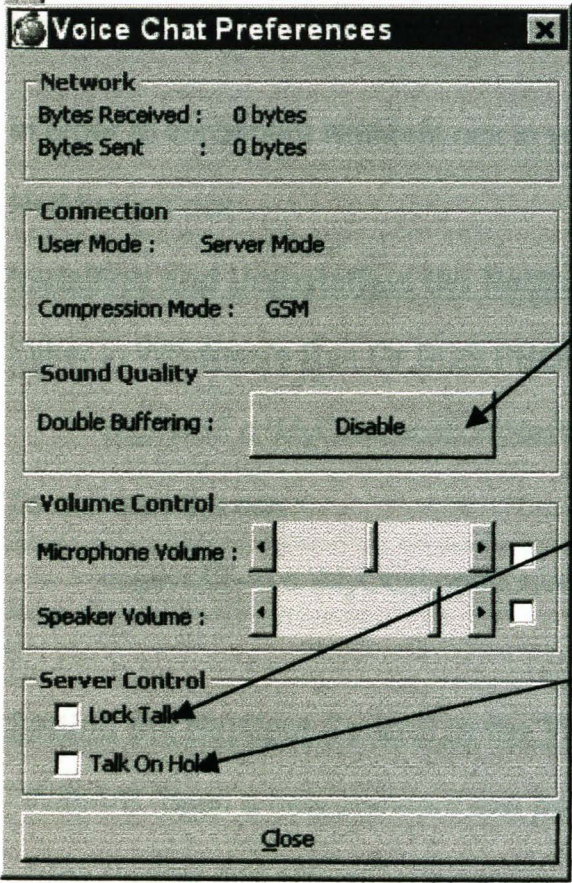


Voice Chat Preferences: Client Mode

* Client is only provided with the capability of controlling the multimedia devices such as the Microphone and Speaker

The slider is for changing the volume of the specific device.

The checkbox beside is for muting and un-muting the specific device.



Voice Chat Preferences: Server Mode

* Extra Control feature in Server Mode is provided.

Sound Quality

Double buffering: A button is given for the host in order to disable or enable the double buffering feature in Voice Chat Application.

Server Control

Lock Talk : This checkbox is available to the server in order for the hosting side to lock the talk button of a clients.

Talk On Hold: This is a feature for server in order the hosting to do more than 2 specific thing at one time. Click the first time to start activating the voice chat then click again to stop the voice chat.

NetMeet [Developer Guide]

In order to implement NetMeet.ocx in your application, one must first install the NetMeeting.ocx

Installing and Uninstalling the NetMeet.ocx

Either in Windows's Run Or DOS PROMPT, type

```
regsvr32 c:\MyOcxLocation\NetMeet.ocx
```

To uninstall the ocx

```
regsvr32 -u c:\MyOcxLocation\NetMeet.ocx
```

Getting the NetMeet Control in the VB Project Toolbox

- With VB, open a new standard exe project.
- Click Project and select Components
- Search for NetMeet from the list then check on it and click Apply
- The NetMeet Control is now is reachable in the tool box.

* A Sample of a project is included in the CD [Folder Simple Application]

NetMeet [Developer Guide] Internet Deploy

- A set of pre-package NetMeet Control ready for Internet deploy is also included in the file. (* In order to repackage the file use tools include in the Visual studio Deployment & Package Wizard)

- To be able to run the NetMeet over the Internet, for the package file into the web folder(Inetpub\wwwroot).

* All a set 3 files is included in the CD [Folder InternetDeploy]

- Netmeet.htm, Netmeet.cab, folder support

CAUTION: Internet Deploy is not recommended under Internet Security Issue and not fully tested (Not in project scope). In order for client to download and use the control, setting the Internet security to lowest is a must. Besides this, the control is only tested on IE Browser.

NetMeet [User Guide]

1. Choosing to be a host or a client
- 2 . If you want to host click host button from the main form else click on the connect button. Please make sure the connecting port is same as the host port before connection start.
3. Checked status window to make sure of your current status is connected before starting
4. Below is a brief guide to the buttons available in voice chat:

Server Mode

Client Mode

Host :
Click this to create a host

Connect :
Click this to connect to a Host

WhiteBoard:
Click this to go to WhiteBoard main

Chat :
Click this to go to Chat Main

Preferences :
Click this to go to preferences

Talk :
Click and hold to start voice chat

Form1

User Name : Server

IP Address : neo1

Connection Port : 112

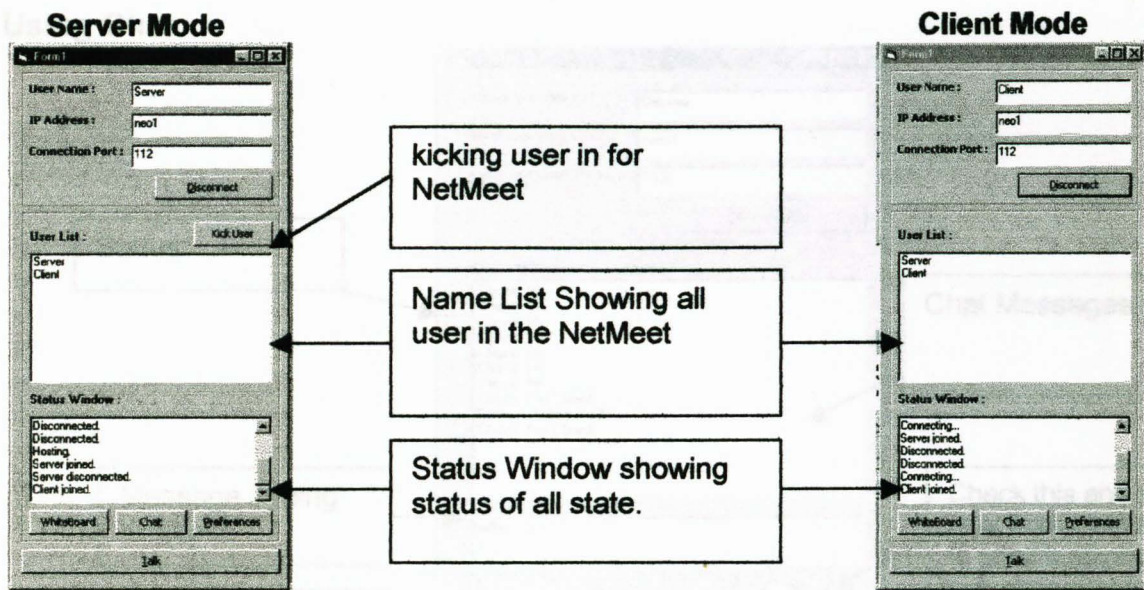
Host Connect

User List :

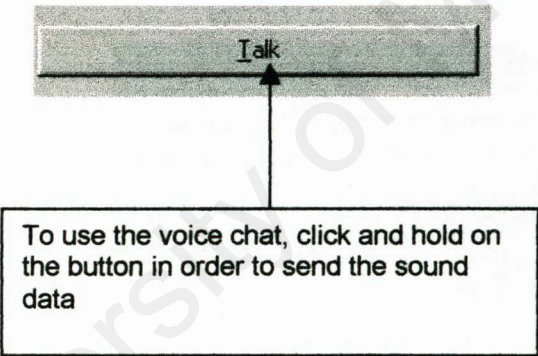
Status Window :

WhiteBoard Chat Preferences

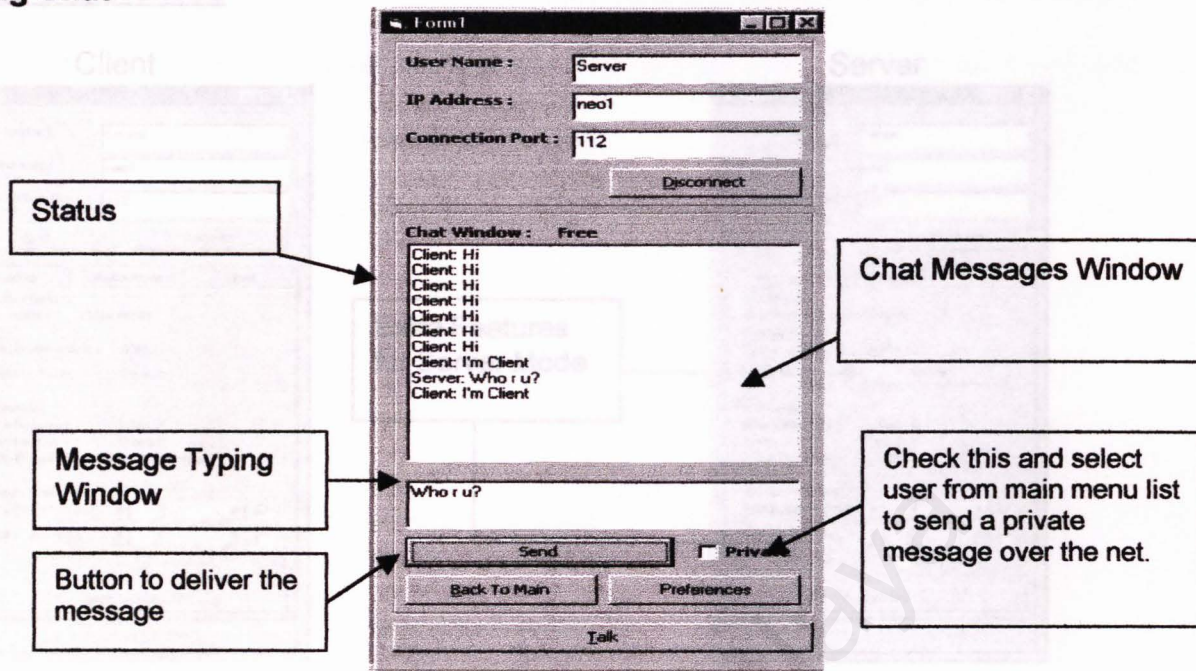
Talk



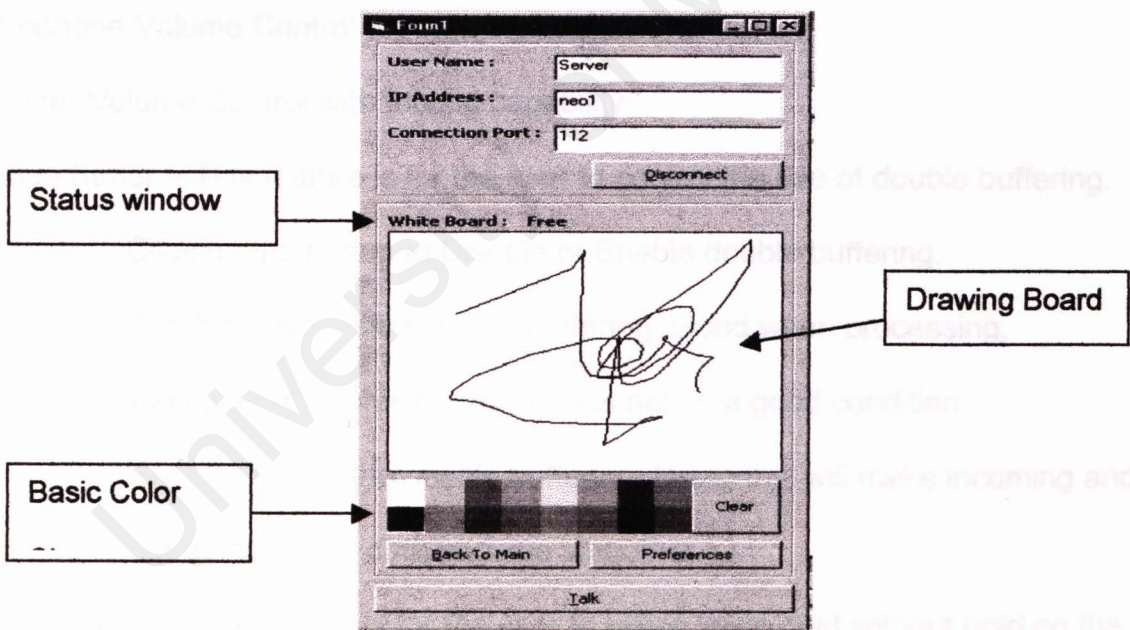
Using Voice Chat



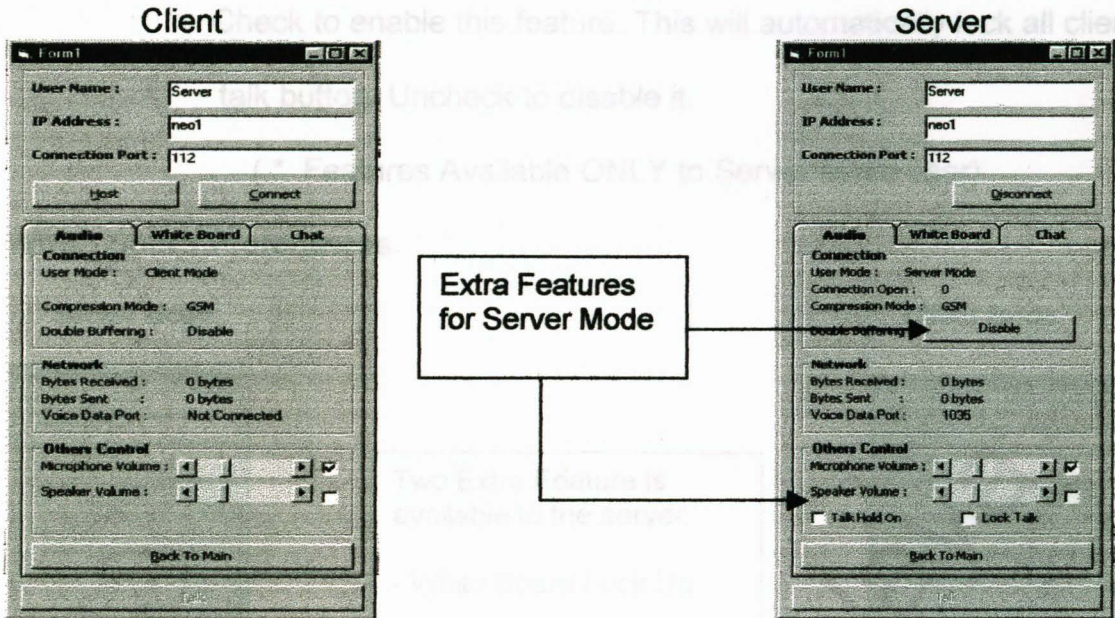
Using Chat



Using White Board



Audio Preferences



Others Control

- Microphone Volume Control with muting capability
- Speaker Volume Control with muting capability
- * Double Buffer – This feature is for the user to control the use of double buffering.

Click on the button to Disable or Enable double buffering.

Disable : Not using double buffering. Good when processing, memory and network resource is not in a good condition

Enable : Using the double buffering. Using this will make incoming and out going voice data become smoother.

- * Talk On Hold – This feature is for the user to utilize voice chat without hold on the mouse click on talk button.

Check to enable this feature. The click the first time to start talking and the second time to stop the talking. Uncheck to disable it.

* Lock Talk – This feature is for the user to lock the talk button of client side users.

Check to enable this feature. This will automatically lock all client side talk button. Uncheck to disable it.

(* Features Available ONLY to Server Mode user)

White Board Preferences

Server Mode

Two Extra Feature is available to the server.

- White Board Lock Up
- Presentation Mode

Client Mode

Inside here user can define that own color for the background and pen color.

Color Setting

- Fore Color - Changing the color of the drawing pen on White Board
- Background – Changing the color of the White Board background

White Board

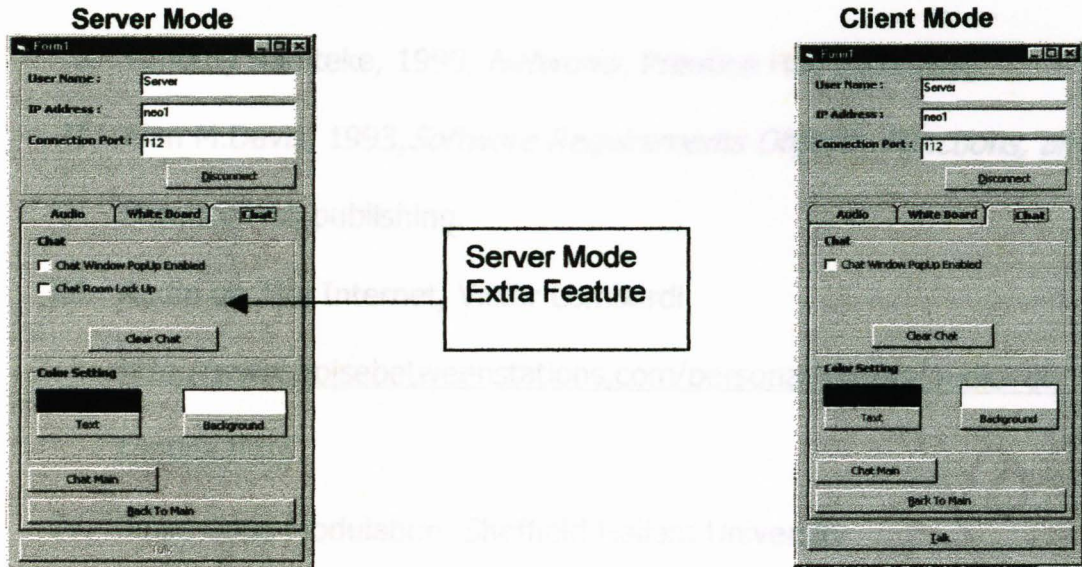
- WhiteBoardPopUpEnabled – This is for the user automatic to be notified by changes happened in White Board.

* Lock White Board - This is to enable the server to lock or unlock the white board.

* Presentation – This feature is to enable client to verify actual presenter such as the happened

(* Features available ONLY to Server Mode User)

Chat Preferences



Chat

- Chat Window PopUp Enabled -
- Clear Chat – Clear all the message in the chat window
- * Chat Room Lock Up – Disable the client's chat window capability

Color Setting

- Text - Setting the color of the text
- Background – Setting the color of the chat window background

(* Feature Available ONLY to server)

References

- I. Timothy Ramteke, 1999, *Networks*, Prentice Hall publishing
- II. Alan M.Davis, 1993, *Software Requirements Objects, Functions, and States*, Prentice Hall publishing.
- III. Audio on The Internet, Victor Lombardi
http://www.noisebetweenstations.com/personal/essays/audio_on_the_internet/index.html
- IV. Pulse Code Modulation, Sheffield Hallam University
<http://www.shu.ac.uk/schools/eng/teaching/rgh/infoeng/pcm.html>
- V. DES, <http://raphael.math.uic.edu/~jeremy/crypt/contrib/stj.html>
- VI. IDEA and others, <http://www.geocities.com/ResearchTriangle/4806/fyp.html>
- VII. PCM Source,
<http://www.cs.tcd.ie/courses/baict/bac/jf/projects/99/pcm/LINKS.html>
- VIII. DPCM Source, http://fly.cc.fer.hr/~eddie/Sem_DPCM.htm
- IX. ADPCM Source,
<http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/dpcm/dp3.htm>
- X. Emerging voice over Instant Messaging,
<http://www.networkmagazine.com/article/NMG20010416S0005>
- XI. Voice Digitization, http://student.monterey.edu/Students_S-Z/scottcharmaignel/world/1CLSWEBP.1.html
- XII. Voice Digitization,
<http://ocean.ucc.ie/02/jtk1/cs2050/assignment2/encoding.html>