Honours Project Report

AfriMeet: Online Meetings in Africa

Zafika Manzi
zmanzi@cs.uct.ac.za

Supervised by: Dr Hussein Suleman
hussein@cs.uct.ac.za

	Category	Min	Max	Chosen
1	Software Engineering/System Analysis	0	20	6
2	Theoretical Analysis	0	25	0
3	Experimental Design and Execution	0	20	16
4	System Development and Implementation	0	15	13
5	Results, findings and Conclusion	10	20	16
6	Aim Formulation and Background Work	10	15	9
7	Quality of Report Writing and Presentation	10		10
8	Adherence to Project Proposal and Quality of Deliverables	10		10
9	Overall General Project Evaluation	0	10	0
Total marks		**80**		**80**

Department of Computer Science
University of Cape Town

2011

Abstract

Online meeting is a form of meeting in which clients collaborate remotely via the Internet in real-time. However, hosting online meetings with remote clients is often a nightmare when using South African Internet connections, notorious for unstable and low bandwidth. Commercial and Open source tools have been developed to host online meetings based on fast, stable connections with large amounts of bandwidth but these tools often fail due to the fact that they do not deal with low bandwidth environments.

The aim of this project was to design and implement a bandwidth-aware application tool to host online meetings where multiple clients can share audio, video, presentations, desktop as well as send instant messages to one another in a low bandwidth and unstable Internet connections. Compressing audio and video, images , presentation slides and pre-fetching those slides can help improve the client experience as well as allow efficient downloading of slides, sharing desktop, sharing audio and video in a low bandwidth meeting environment causing no delays, no freezing of slides, no loss of slide content, no loss of text messages and also a good sound quality.

This part of the project only covers two features of the tool, the presentation application which is responsible for pre-loading of presentation slides and the chat application which is responsible for sending text messages between clients. In order to make these two applications to function in a low bandwidth environment, what is being proposed in these paper as a solution to solving the bandwidth problem is to compress PDF files that are being uploaded to JPEG image files and then store these image files on a central server where multiple clients can then retrieve these image files, thus reducing latency as well as bandwidth. All messages for the chat application will be sent to the central server and the server will automatically broadcast these messages as soon as it receives the messages to the clients who are logged on the same conference room. This will allow for the messages sent not to be lost and cause no delay in sending or receiving. Thus the time and space it takes for each packet to be sent or received will be reduced.

From the above solution as a result of the final outcome after designing and implementing the two applications, chat and presentation it was found that the chat application uses 1.9bps of bandwidth on average, and the presentation application uses 108148.6 bps of bandwidth on average. For pre-loading presentation slides, the bandwidth required is 73238.05bps on average.

Keywords

Bandwidth, Retrieving, Uploading, Broadcasting, Web conferencing tools, PDF compression

Acknowledgements

First and foremost, I would like to give thanks to the mighty God, to whom, without his grace upon me this work would not have been possible. I would also like to thank my supervisor, Hussein Suleman for his commitment and efforts as well as providing invaluable direction to this work. None of this would have been possible without him. Thank you for being such an inspiration and being patient with me. To my family whose support has been unbounded over the couple of year's, thank you for always being there to support me. Working in the Department of Computer Science was inspiring, many thanks to the department for the long and ongoing support.

I would not have made it this far was it not for the unfading support I received from my family and friends in the past few years. To single out, I would like to thank my father and mother whose support is always there whenever I need it. My friends, Shepherd and Sicelo, continually give me good advice whenever I need it and kept me sane in all those long nights I spent during the writing of this project. My boyfriend Khetho has been an inspiration to me; he always looks upon me and always wants to see me achieve my best every day. Singling out each and every person who has made my life change for the better would require acres of space but any lack of direct mention should in no way diminish my profound gratitude to everyone who has helped shape my life in a positive way.

Contents

List of Figures

List of Tables

List of Acronyms

PDF	Portable Document Format
JPEG	Joint Photographic Experts Group
GUI	Graphical User Interface
bps	bytes per second
kps	kilobytes per second
QoS	Quality of Service
GNU	General Public License
MDGs	Millennium Development Goals
ICT4D	Information Communication Technologies for Development
MPEG	Motion Photographic Experts Group
MJEG	Motion Joint Photographic Experts Group
IDE	Integrated Development Environment
J2SE	Java 2 Standard Edition
PNG	Portable Network Graphics
HDTV	High Definition Television
DPCM	Differential Pulse-Code Modulation
DVMR	Distributed Virtual Meeting Room
TCP	Transmission Control Protocol

Chapter 1

Introduction

The use of online meetings is increasing each and every day and has grown to have a positive effect in our daily lives. This is due to the fact that it helps reduce travelling expenses and that it is accessable to everyone even those who are far away. Online meetings occur on the Internet in real time streaming. The Internet is a group of millions of computers connected by networks (Wikipedia, 27 October 2011). These connections within the Internet can be large or small depending upon the cabling and equipment that is used at a particular Internet location. Bandwidth is defined to be the rate at which data is transferred from the website over the Internet and is measured in bps (Prasad et al, 2003). The size of each network connection determines how much bandwidth is available. Internet is usual slow (Lai and Baker, 1999). Some of this slowness is due to properties of the end points, like slow servers, but some is due to properties of the network, like propagation delay and limited bandwidth. Since online meetings occur over the Internet if bandwidth is low it results in the performance being low and causing some interruptions leading to tasks taken by clients over the Internet being a complete failure.

Open source tools such as Adobe Connect and Zoho Show and many more do exist to support online meetings (Wikipedia, 11 October 2011). But when using these tools in South Africa they seem to fail since South Africa's bandwidth is very low. They were built with the assumption of fast, usable and stable connections in a high bandwidth environment. In this document the aim it to mainly build a different application which has more or less similar features but the only difference is that it is designed for a low bandwidth environment.

The QoS relies on the hardware environment such as the computer, webcam and microphone. Computers require electricity to work and thus the Internet is vulnerable to power outages and results in connections of clients and information being lost. In this document another aim is to solve the issue by the fact that every piece of information gets saved on the server and clients can always retrieve this information.

Only the design and implementation and evaluation of chat and presentation applications will be discussed in this document. Audio and video application has been implemented by Tresor and a client list and desktop sharing application has been implemented by Flora. The applications are built so that later they can be easily combined to form a complete online meeting tool.

1.1 The AfriMeet Project

Due to the fact that currently developed open source tools fail in a low bandwidth environment. The AfriMeet project is about designing and implementing an online meeting tool that works in a low bandwidth environment. These tools have features such as audio, video, presentation slides, chat, desktop sharing, hand-raising and a participant list. The AfriMeet project is about building these features in such a way that they work effectively and efficiently in a low bandwidth environment.

1.1.1 Motivation

The use of conferencing technology continues to grow as accessibility increases and costs decline. Open source and commercial tools often fail in a low bandwidth environment and they exists a great need to build a tool that works efficiently in such conditions. The aim of building a bandwidth-aware application tool is to help improve the user experience and also help in development projects such as ICT4D. ICT4D projects try to build applications that can impact developing countries and also tries to help achieve broader development goals, such as the MDGs. Web conferencing can sometimes be affected from a client perspective (Baecker et al, 2006) since client experience can degrade rapidly as the number of conference clients rises, since it becomes harder to support video and audio from all clients.

1.1.2 Research Question

Looking at the features of an online meeting, research was conducted to determine how changing the internal architecture can positively enhance the usability, performance, responsiveness and user experience. The research was mainly focused on investigating how to reprioritize features of an online meeting in order to get the best tradeoff between quality and usability with constrained Internet connections and low bandwidth environment.

The research questions that the AfriMeet project investigates are:

- Is it possible to build an effective text chat tool that can work with minimal bandwidth?
- Is the pre-loading of static data feasible with low bandwidth?
- Is it possible to build an effective audio-conferencing tool that works with low bandwidth conditions?
- Is it possible to build an effective video-conferencing tool that works with low bandwidth conditions?
- Is it possible to construct a system that manages meeting procedures (presence, hand-raising, etc.) efficiently despite varying Internet conditions?
- Is it possible to build an effective desktop sharing application tool that works with low bandwidth conditions?

In order to test the above research questions, a number of components needed to be developed. In answering these research questions, to optimize the bandwidth usage, each client will either receive or send a stream. All the static content presentation slides will be compressed and preloaded at all client posts. This approach will avoid real-time streaming of the content and help preserve the bandwidth. Even though some online tools to host online meetings already exist, the project assumed that the application will run in a low or high bandwidth environment with fast or slow Internet connections. Further details on

the implementation and design on how the system can work reliably will be discussed later in the document.

1.1.3 System overview

The implementation and design of the system is separated into three parts, Audio and video; Chat and presentation; and Client list and desktop sharing. Figure 1.1 shows the overall framework of the system, what applications features are built and who is responsible for building that application. The three parts are all equal in terms of length and difficulty. These parts are all based on one notion and that is to be able to function in a low bandwidth environment. Each of these parts will later be merged to form the overall system. This part of the report focuses on building the Chat and presentation applications. Tresor report focuses on building the Audio and video applications and Flora report focuses on building the Client list and desktop sharing applications.

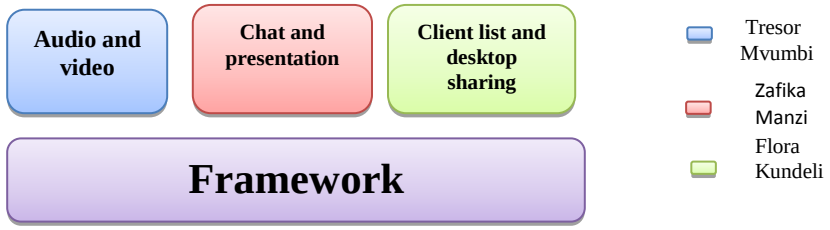


Figure 4.1: AfriMeet system overview

Audio-video

This part is responsible for maintaining audio and video communication. Audio is responsible for recording and producing sound during the meeting and video is responsible for capturing still images in motion to allow participants to be able to see participants in the meeting. The aim is also to provide an acceptable audio and video quality. Reprioritizing audio stream over video and applying key framing and compression techniques to provide a usable low bandwidth video stream.

Chat and presentation

This part is mainly allowing text messages to be sent between participants when audio and video channel fail to work in a low bandwidth environment. This is because a chat facility is considered not to use up much bandwidth (Scholl et al, 2006). The presentation is responsible for viewing slides being presented and allowing only the ID of the current page to be sent across the network. The idea is to compress these slides and upload them to the server so that they can later be retrieved by other participants.

Client list and desktop sharing

The client list is a list of all logged on clients in the meeting. Some features that are implemented in the list are hand-raising and floor control which allows the meeting to proceed in a proper manner and also identifies who is currently handling the floor. The desktop sharing allows for clients to remotely share their computer's desktops.

1.2 Chat and Presentation

1.2.1 Problem Statement

Chat application

When bandwidth is very low, text messages is sometimes lost, as there is unreliable transmission of packets. Important things to consider when building the chat application are that:

- Clients must be able to send and receive text messages. No messages should be lost.
- The client must be able to send a private text message to another client reliably without being lost or sent to an incorrect client.

The usability, performance, and responsiveness of the system will then be evaluated over low bandwidth environments using multiple users.

Presentation application

When bandwidth is very low, uploading of slides is normally unsuccessful; slides freeze, not all static information is downloaded and static content sometimes is lost. Important things to consider when building the presentation application are that:

- Every client must have access to the presentation slides and be able to upload and retrieve all available presentation slides in a proper manner.
- A client must be able to see which slide is currently being viewed.
- A presenter must be able to control the presentation slides.

The usability, performance, and responsiveness of the system will then be evaluated over low bandwidth environments using multiple users.

1.2.2 Motivation

Chat application

When bandwidth is low and a client cannot use audio and video applications, the sound quality is not good and if this is the case the client has the ability to use a chat application where the communication is through instant messaging. Chat application will allow for clients to still communicate and the meeting to continue even if the bandwidth is low as it is known not to use a lot of bandwidth.

Presentation application

An application such as presentation slides can be directly impacted in a low bandwidth environment-an example will be if the presentation slides freezes, and if the content is lost, causing the user experience to be unpleasant. If such problems are not dealt with it can lead to the Internet meeting tools being useless.

1.2.3 Research Question

The research questions for the chat and presentation application are:

- Is it possible to build an effective text chat tool that can work with minimal bandwidth?
- Is the pre-loading of static data feasible with low bandwidth?

In answering the above questions some modifications of the applications were done and then the applications were evaluated based on those modifications.

Chat application

All text messages will be sent to the server and reliably broadcast to all clients in the same meeting room. This will prevent messages being lost as the server will automatically broadcast to all clients as soon as the server receives those messages. The bandwidth usage by each client will be calculated as the total amount of text messages that the clients send at a specific time interval. A graph is then plotted of amount of text messages over time for each client to calculate the bandwidth that each client uses and the total bandwidth required by the overall application, this is fully being discussed in Chapter 4.

Presentation application

For presentations, PDF files will be compressed to image files and then uploaded to the server where each client can then retrieve the slides. This will help prevent downloading at the same time, reducing the bandwidth usage as time to download will be less. The bandwidth that the clients used will then be given by the total amount of byte data that the clients send in a specific time interval. A graph is then plotted of amount of data over time that the presenter uploads; this gives the bandwidth usage for the presenter. A graph of amount of data over time for each client for downloading the presentation slides, gives the bandwidth that each client's uses. The bandwidth required by the overall application is then given by the total bandwidth usage of all clients. More details about this are fully being discussed in Chapter 4.

1.2.4 Chat and presentation application overview

The two applications-chat and presentation-are built separately from each other but they both have a similar structure, consisting of a client and server application. The client application represents the GUI. Clients request services of the server independently but use the same interface. The server passively awaits for requests from the clients and then act on them. The server is responsible for maintaining communication between clients, responsible for sending information from one client to other clients who are waiting to receive. The communication between both the client and server is established over the TCP using HttpPost for the chat application and HttpURLConnection for the presentation application.

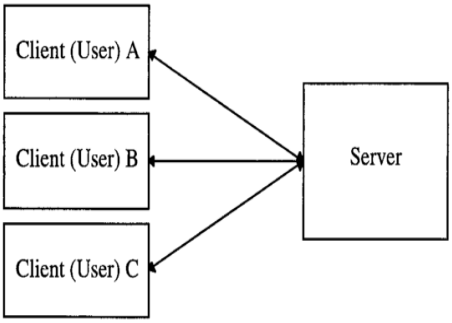


Figure 1.2: Client-Server system architecture (Scott, 1998)

1.2.5 Ethical, Professional and Legal Issues

User testing was conducted, in order to assess the research questions and evaluate the system design, implementation and evaluation phases. Ethics Clearance for all planned user experiments was obtained from the Science Faculty Ethics Community. Users were asked to take part in the experiment and they were provided with enough details on the project, why it is being conducted, test procedures and the confidential nature of collected data. Users were told to feel free and ask questions if something was not clear and appropriate answer was given. Questionnaires were used as source of evaluation of the system to measure user satisfaction.

With regards to legal and copyright issues surrounding use of libraries and software, a number of factors must were considered. The libraries and software used were licensed under permissive free software.

1.3 Outline

This document outlines the design, implementation, and evaluation of the chat and presentation application in a low bandwidth environment. Chapter 2 provides the background information of tools built so far and a comparative analysis of them; previous work done on audio and video, chat and presentation, client list and desktop sharing are also discussed. The design, implementation, evaluation of the chat and presentation application based on an iterative approach is discussed in Chapter 3.In Chapter 4 the chat and presentation applications are evaluated and also contains the overall findings. Chapter 5 discusses some future work and finally, Chapter 6 provides a conclusion discussion of the overall project.

Chapter 2

Background

2.1 Introduction

Much work has been done in the field of building Web conferencing tools which use fast, stable connections in a high bandwidth environment. Section 2.2 covers the work that has been done so far on audio and video, client list and desktop sharing in section 2.4 and the work on chat and presentation is provided in section 2.5. Finally in section 2.6 a summary of the chapter is provided.

2.2 Audio and Video

Audio conferencing and multipoint video conferencing are typical examples of methods that are part of Web conferencing (Baecker, 2003), which are mostly used for real-time communication, collaboration, and knowledge sharing over the Internet. Audio conferencing allows the real-time multipoint transmission of voice. Yet it lacks the media richness, sense of presence, and ability to engage clients that is afforded by video and other dynamic media. Internet desktop video conferencing supports real-time multipoint audio and video, chat communications as well as shared workspaces. Yet it still does not provide reliable Internet video performance, and is not scalable to large numbers of clients.

In packet-switched networks, audio transmissions are typically subjected to several latency components, for example sampling, pre-processing silence-suppression and compression, network transmission, and network propagation delay being typically the least-predictable and most dominant component for audio transmission over the Internet. As the bandwidth available on networks and the speed of computers increases, real-time transmission of video between general purpose workstations becomes a more realistic application. However, even with a high speed network, video has to be compressed before transmission (Turletti and Huitema, 1996). Video conferences usually occur in environments with a static background. Thus, there is little local motion per frame. Also, multiple videos may originate from a conference location. These factors can be taken into consideration in developing a compression scheme meant for conferencing (Kamath, 2005).

A study conducted by (Scholl et al, 2006) were chat as compared with video, it was found that when using chat for informal communication along with video, chat uses less bandwidth. This is because the technical quality of the video conference depends critically on bandwidth availability (Baecker et al, 2006). There is a little knowledge about the effects of narrow-bandwidth digital videoconference systems

on group decision quality and thus it is not correct to assume that low-quality communication yields high quality group decision (Baecker et al, 2006), because if the system is "poor" enough to filter out not only "noise" but necessary information for the task, the decision quality may be lower than with face-face (Takao, 1999).

One way in which video can be reprioritized in order to make it work in a low bandwidth environment is to compress the video streams. Different algorithms exist for video compression and the standardized ones that exist are mainly JPEG for still images or MPEG and H.261 for moving images. MPEG- 1 coding is suited for high definition video storage and retrieval. MPEG-2 extends MPEG-1 to HDTV coding applications. The H.261 standard describes a complex video compression algorithm that allows achieving a very high compression rate (Turletti and Huitema, 1996).

In 2005, Kamath (2005) proposed a compression scheme for video conferencing called MJPEG-DPCM with segmentation. Most compression schemes compress one video at a time but in the case where more than one video stream originates from a location in the situation of video conferencing the proposed method was implemented to exploit redundancies in portions of the video without motion and static background between consecutive video frames. The system allowed client interaction at both the encoder and decoder in order to allow selection of video streams. The system was based on modified motion JPEG in order to achieve low complexity. Compression ratios of the order of five times that of motion JPEG were obtained. MPEG was compared with the proposed method and it was found that there were improvements from MJPEG to the proposed method with little degradation in the image quality. But further processing will be carried out to reduce the degradation due to scaling and possible saturation during reconstruction.

In a low bandwidth environment audio is presented ahead of video in some video conferencing systems since audio requires less time to process (Chen, 2003) thus reducing latency as well as bandwidth ; what the client says is not what is seen on the video. The conventional approach to synchronizing audio and video is to delay the audio so that the audio and video latencies are matched, however the time required to process video can exceed the maximum perceived audio latency that is acceptable in a conversation. Video conferencing systems may not synchronize the audio with the video (Chen, 2003) since supporting perceptually instantaneous audio is more important than maintaining lip synchronization. In 2003, Chen (2003) produced a video conferencing system to achieve lip synchronization with minimal perceived audio latency. The system maintained a balance between minimizing audio latency and supporting lip synchronization. Audio latency was minimized by stretching the time at the beginning of each utterance so that the audio and video latencies are matched.

2.3 Client list and desktop sharing

Client list provides a list of all logged on users in a meeting. A client list is implemented so it will allow for floor control. Floor control allows users of networked multimedia applications to utilize and share resources (Hans and Garcia, 1997) such as remote devices, distributed data sets, telepointers, or continuous media such as video and audio without access conflicts. A lot of features cans be implemented

in floor control such as hand-raising, allowing presenter to control the meeting. A client list allows chat communication between clients to be easy and efficient as the client list shows which client is on line.

Some real-time remote desktop sharing software has recently been getting popular. Desktop sharing allows clients to exchange desktop screen images between remote computers through the Internet. Ichimura and Matsushita (2005) designed and implemented a desktop sharing tool called QuickBoard that allows clients to deliver computer screen images of any application to more than 100 Web browsers in real-time. Not only did they implement QuickBoard but they also designed and developed ICMHP, a method integrating interframe compression mechanism with Web-server-side technology where the aim was to reduce network traffic on a Web-based real-time presentation system. The first prototype that they implemented was QuickBoard capture which is illustrated in figure 2.6. The generated image file was automatically transmitted to the Web server immediately through Windows network file sharing, and placed onto the Web directory which is open to the public.

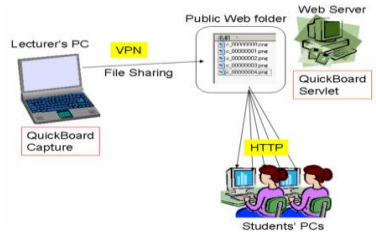


Figure 2.1 QuickBoard capture interface (Ichimura and Matsushita, 2005)

From this the network traffic increases in proportion to the size of the image file when sending a large image file through a Web system. It was discovered that the more efficiently an image file is compressed, the less the network bandwidth needed for transmitting it. PNG is known to have an excellent compression ratio despite the fact that it is a lossless compression format. From this experiment PNG is discovered to be better than JPEG. This was because the compression ratio of PNG is better than that with JPEG when a snapshot image of computer screen is compressed under normal conditions. Hence that's why the PNG file format was used in the QuickBoard system.

2.5 Chat and Presentation

A study in 2006, (Scholl et al, 2006) was conducted to compare chat and audio usage within multimedia conferencing systems using two case studies of informal group communication in a naturalistic setting. The first case study is of workplace clients participating in a "virtual shared office", and the second case study focused on an educational setting where students are provided with tutoring via a multimedia conferencing tool instead of having an instructor physically available. The results from these studies showed that in a media-rich environment supporting both audio and chat alongside a video channel more clients preferred chat to audio and found chat to be more useful than audio for both private and public communication. It was found that chat enables asynchronous communication, can lower the cost of interrupting others, and can make it easier to communicate in a second language.

Yang (2001) proposed a data retrieving engine and came up with an algorithm for determining the proper objects to be retrieved as well as the pre-fetch time of the object under client actions. The proposed data retrieving engine adopts the just-in-time policy to efficiently make use of the data buffers and network bandwidth. The policy requires the data retrieving engine to finish retrieving the object right before the object's playback time to provide smooth progress of the presentation. The objects that should be retrieved depended on the client action since different client actions result in different playback patterns and different playback time of objects. The pre-fetch time for an object depends on both the object's playback time under the client action and the network condition. The server estimated the bandwidth for the object to the data-retrieving engine by some bandwidth measuring mechanism. The data retrieving engine also estimated the time for the request packet arrival at the data server. The total time to retrieve an object was given by the summation of the delay of the request packet and the transmission time of the object from the server to the client. The duration of play action, fast forward, fast backward, pause/restart, were calculated by the data retrieving engine algorithm. The problem with the data-retrieving engine is that it had to recompute the pre-fetch time of objects each time a new user action is made-this was very inconsistent however the reasoning behind this problem are explained by the fact that the computation time of the algorithm is linear, and it can therefore be finished within a short time.

2.6 Summary

This chapter gives an overview of the literature that covers the different software tools that exist and also some aspects features of Web conferencing. The main aspects covered include audio, video, client list, desktop sharing, chat and presentation. From doing the literature it is found that a lot of work has been done on video and audio compression. This is because video and audio consumes a lot of bandwidth. There is little work done on chat and client list and this is because they are assumed that they do not use up a lot of bandwidth. The notable outcomes of this chapter were that a chat application is indeed reasonable to have since it does not consume a lot of bandwidth and therefore can be used when audio and video cannot work in a low bandwidth. It is also noticed that after compressing an object or stream the quality of the object results to be poor.

Chapter 3

Design and Implementation

3.1 Introduction

In order to answer the research questions a chat and a presentation application was built. This chapter gives an overview of how the research was conducted. The methodology used to design, implement and evaluate the prototypes developed is discussed. The chat and presentation applications are designed separately from each other. This is because the aim was to evaluate each application alone. The design process was based on an iterative approach which is shown in Figure 3.1. Three iterations were used-each iteration comprising of a design, implementation and evaluation stage of the applications. The first iteration is proof of concept and was done using sockets, the second iteration is a preliminary implementation and is done using servlets and HttpClient and the third iteration represents the complete system. The reason why the iterative approach was used was to build an effective and productive system based on interested user testing.

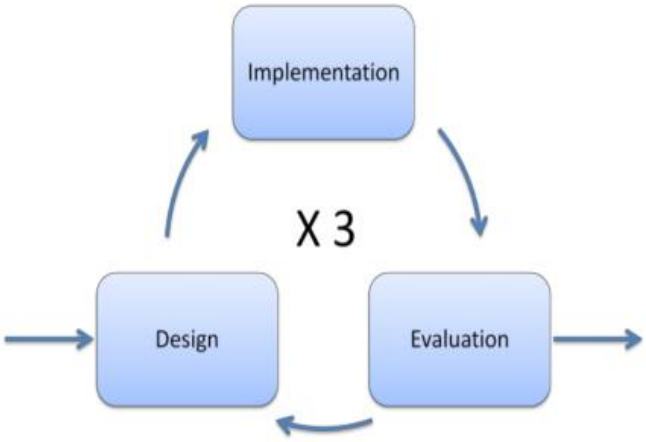


Figure 3.1: Iterative design process

The goal of this project is to design and implement chat and presentation applications for a Web conferencing tool. The design of the applications needs to be usable and easily understood by the clients. The interface ought to be simple enough so that users can interact with the system without requiring excessive training or manuals.

First, in section 3.2 the system requirements are that are needed in order to build the applications are discussed. The client-server application is discussed in section 3.3. Following that, in section 3.4 the iterative design process for the chat application is discussed. The presentation application is discussed in section 3.5. When one is designing an application some design challenges occur and these challenges are

discussed in section 3.6. Section 3.7 provides some strengths and weaknesses of both the applications. Finally, this chapter ends by providing a brief summary of the overall chapter in section 3.8.

3.2 System Requirements

The software required were mainly, Apache commons libraries, the Ghostscript software for PDF compression to JPEG image files, Windows or Linux operating system, and a Java development platform. The applications were implemented using J2SE and Netbeans IDE.

Why Java development platform?

A Java platform was used because Java provides swing classes which are the only classes to use to provide the best graphical user interface on the client side and also because Java is platform independent and supports distributed computing. Since the applications being built are remote applications via the Web and all the meetings may have different clients, the client-server architecture is the best approach (Lu et al, 2010) to use since a server will maintain all the conference meetings and save all the required information for it to be available to other clients.

3.3 Client-Server Application

The client application represents the GUI which will enable the client to interact with the system more easily. The HttpClient network protocol was used. This was because HttpClient functions as a request-response protocol and the fact the application were not supposed to run on the Web browser, because the system that was developed would require some low level interface with the hardware that is hardly realizable via a Web browser for instance for the presentation application an external tool needs to be called to convert PDF to JPEG image files. The server application runs on a single machine hosting the Web site and handles all the requests made by the client, stores the content and processes the request and also perform tasks associated with these request and then send back a response to the client. Figure 3.2 shows a request-response protocol. The client sends a request to the server and the server acts on that request and sends back a response related to the request. The server which was used was Apache tomcat.

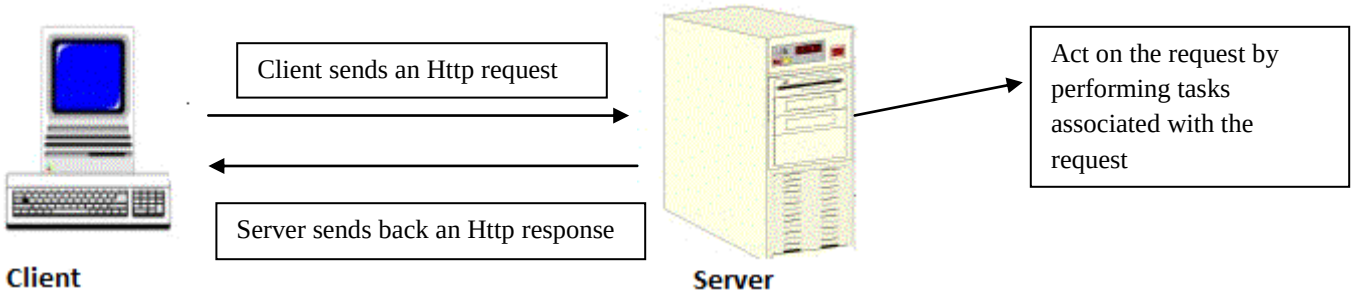


Figure 5.2: Http Request- Http Response Protocol

3.4 Chat Application

3.4.1 First Design Iteration

Design

The first iteration was proof of concept to show that one can send and receive both public and private messages. A lot of chat applications have been designed and, looking at these chat application such as Facebook, Skype, Gtalk chat and more, most of them have an interface area to write the messages and an interface area for displaying the messages. In terms of how the message is sent they normally have a send button or use the keyboard enter key. The AfriMeet chat is also designed in a similar way. The design of the chat application interface is made up of:

Features

- **Writing interface**: Interface area for writing both public and private messages.
- **Display interface**: Interface area for displaying both public and private messages.
- **Send button**: Allows for messages to be sent.
- **Back button**: Allows for clients to go back to the logging window which is shown in Figure 3.4.
- **Exit button**: Allows for clients to exit the room.

The features that were designed are shown in Figure 3.3.

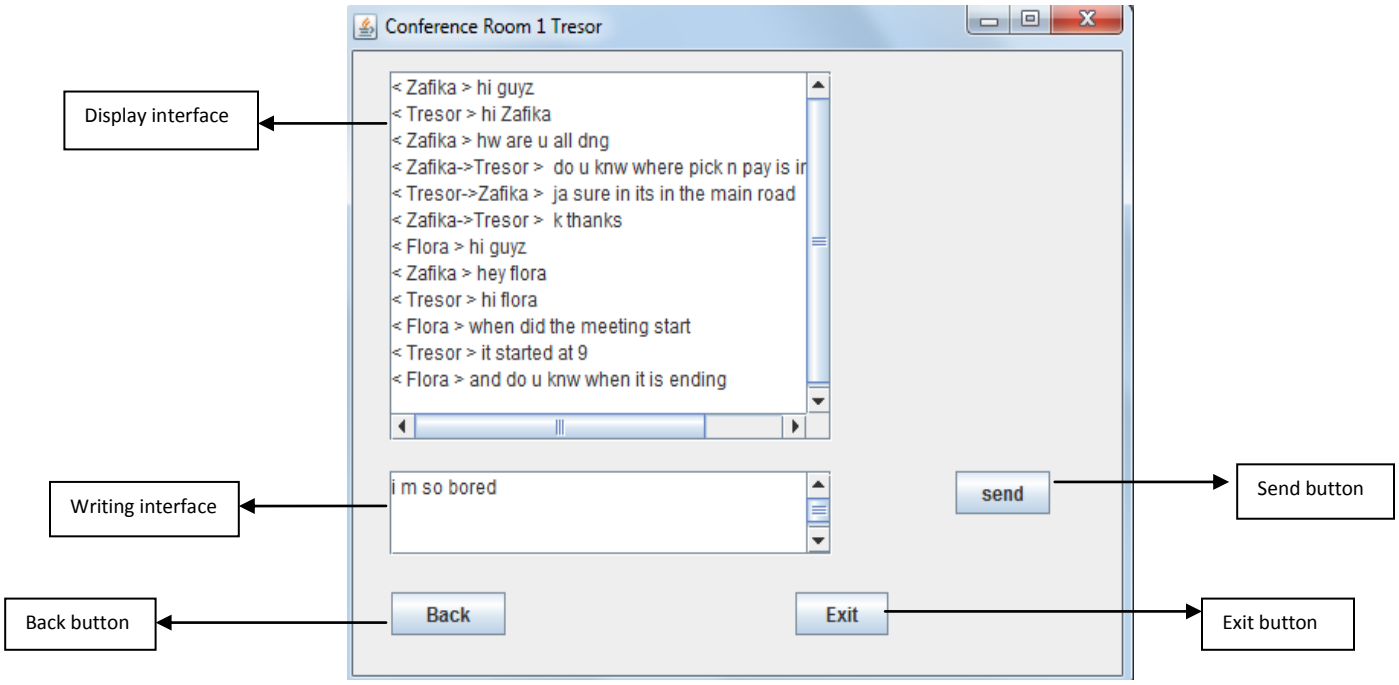


Figure 3.3 AfriMeet chat application interface with functionality

Functionality

On testing, all the functionality was working, the client was able to write and send the message using the send button and the clients were able to receive the messages and the messages also got displayed on the display interface. This is illustrated in Figure 3.3 above.

Implementation

The initial design of the chat application interface and functionality was implemented using socket programming instead of HttpConnection and Java Swing. The aim was to test if the chat application worked on a normal Java application. It was then going to be easy to port the application to a Web application. No libraries were needed to implement this application. First the connection between the clients had to be successful and then the clients had to log on to the application-the interface for logging on is shown in Figure 3.4.

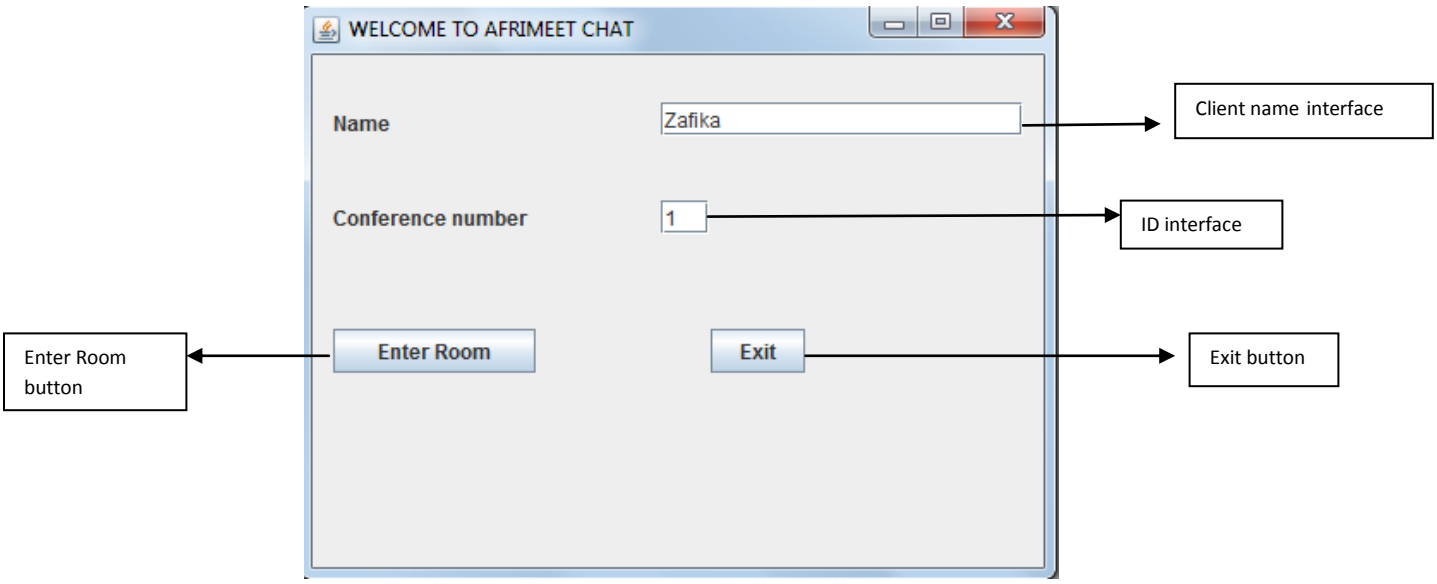


Figure 3.4: Logging in interface

Enter Room button: Allows clients to enter the meeting room and chat and then introduced them to the chat interface, which is shown in Figure 3.3.

Client name interface: Allows client to enter their name.

ID interface: Allows clients to enter the conference room number to which they wish to be in.

Exit button: Same function as describe above in feature section.

Public message:

The client writes the message in the writing interface and then presses the send button. The message is then sent to the server were the server which broadcasts the message to all other clients including the client who sent it in the same meeting room-this is illustrated in Figure 3.5.TCP is used to send and receive messages. TCP is the dominant transport protocol for networking. TCP is used as a transmission protocol because it is considered to provide a reliable end-to-end delivery of data between applications (Aweya, 2003).

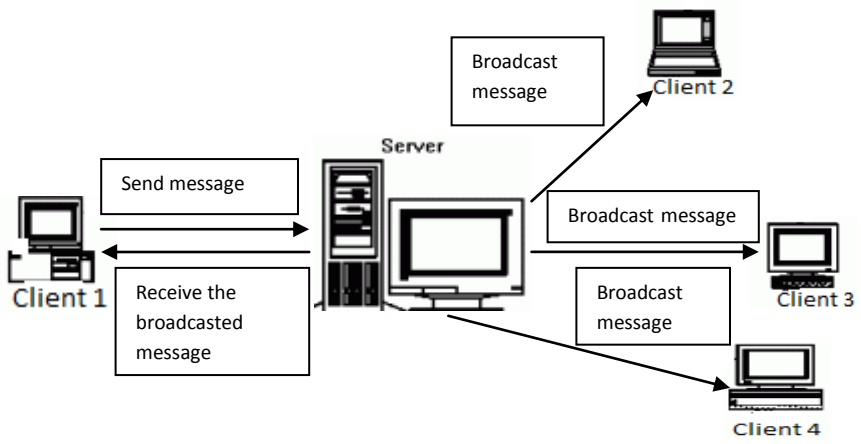


Figure 3.5: Public message implementation

Private message

To send a private message the clients enters @ as the first character in the writing interface followed by the client name to which the message is being addressed and then space and then the message, for example, @zafika “hi”. Once the send button is clicked the message as well as the client name is sent to the server and the server only sends the message to that client and also to the client who sent it but not to the other clients. Figure 3.6 shows this.

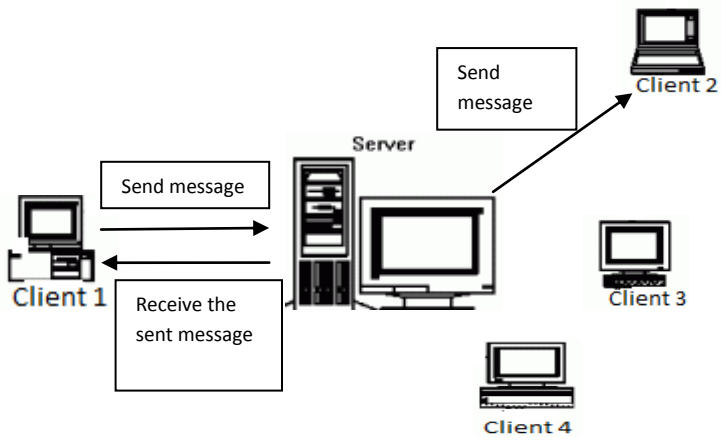


Figure 3.6: Private message implementation

Evaluation

The application was then evaluated by Hussein Suleman and Antoine Bagula. The feedback that was received was to use a Web Application instead of a Java application. They were no comments on the design. However the application did fail to send the private message and this was because when one sent the private message (the @ followed by the name of the person to which the message is being sent to) it turns out that the name entered was wrong. This is due to the fact that the design does not have a list showing who is in the meeting. However the list cannot be imported and this is because the participant list

was being done by another member. So it did not make sense to include it in the chat design. However when doing the second iteration evaluation the users were warned about this.

3.4.2 Second Design Iteration

Design

After user evaluation and testing from the first design, the application design did not change. The same features and functionalities remained. Only the implementation had to change.

Implementation

The application had to be implemented as a Web application using servlets and HttpClient. The transmission of messages was still done using TCP. The sending and receiving of public and private messages were implemented the same way from the first iteration.

Evaluation

The experiment

The second design iteration and implementation was evaluated using 6 users. The aim of the evaluation was to determine the usability, responsiveness and presence of the application. Three computers were set up and users had to use the application by performing some tasks such as sending public and private messages. The users were given 5 minutes to send messages to one another and were then asked to answer some questions about the application. These questions are available in the appendix B1.

User feedback

Responsiveness

Users were happy about the response of the system, they were able to send and receive both public and private messages and they were happy about the performance to send and receive messages.

Usability

4 users had some difficulties in sending private messages. They did not know how to send a private message. They requested that there should be some form of document that explains how to send a private message and they also thought that having a list of all logged on clients will be more convenient so that they can see who is in the meeting.

Presence

All of the users felt that they were in a meeting.

Extra Features

- **List**: Users suggested that a list showing all clients in the meeting should be added.

- **Help documentation**: A help documentation that specifies how to send a private message.
- **Text size:** A feature to change the size of the text messages in their display interface since some people cannot see small words.
- **Text colour:** A feature to change their message colour.
- **Text font:** A feature to change their message font.
- **Emotions and smileys:** Have beautiful icons where they can express their feelings.

3.4.3 Third Design Iteration

Design

After doing user evaluation and testing with 6 users from the second design, the application design will not change as users were happy with the interface. Everything was clear about the design and they could easily see where they can write and how they can send. This was because all the 6 users had used a chat application before. The design had to incooperate new features but the core functionalities remained the same. Figure 3.7 shows the final design including some features that the users suggested.

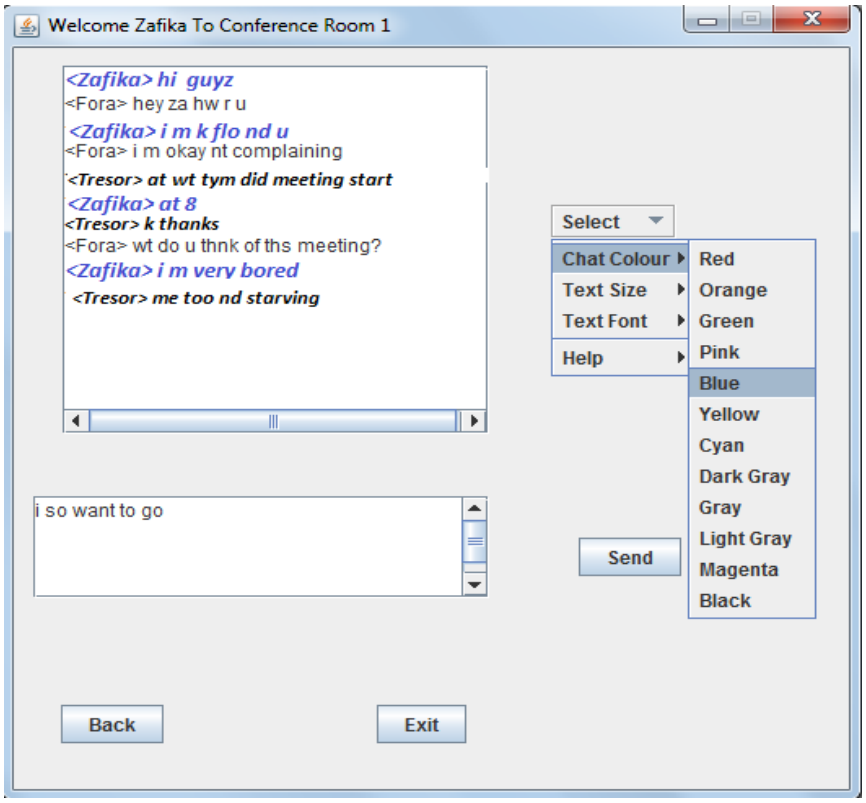


Figure 3.7: Chat Application Final design

Implementation

The features that were added were implemented by sending the chat colour, font along with the message, and when the server broadcasts the message it broadcast the message with the chat colour or font. This was done by parsing. The colour and font were parsed along with the message. The emoticons however could not be implemented.

Evaluation

The experiment

The same approach to experimentation was used as the one in the second design iteration, but now using 12 users instead of 6. Different users were used. The main aim was to also determine if the application can handle more than one meeting. It was not easy to find a user who has never used a chat application before-all of the users had some experience.

User feedback

Responsiveness

Users were again happy about the response of the system, they were able to successful send and receive both public and private messages and use the features such as chat colour, text size and font.

Usability

None of the users had difficulties in sending private messages. Although some users said that sometimes the private messages could not be sent and said that this was because they will sometimes forget the format or enter a wrong name and thought that a list will be wise to have. However they were happy about the application.

Presence

All of the users felt that they were in a meeting and they enjoyed using the application.

Extra Features

Only one feature that they thought should be added was the emoticons and smileys. The implementation of emoticons and smileys is not that easy and this is because it is not easy to add an icon to a text area and the other thing is one has to know the position for were to exactly put it. However most of the work that has been done on the emoticons and smileys are html based and there is little work done on Java Swing. The users also thought it would be great to have one display interface for the public message and display interfaces for each user for the private messages because it is confusing when all the messages are displayed in one interface.

3.5 Presentation Application Design and Implementation

3.5.1 First Design Iteration

Design

The first design iteration is a proof of concept design in terms of the possibility to compress PDF files to JPEG files. The design is very simple and has these features and functionalities and is shown in Figure 3.8:

Features

- **Display interface:** This interface displays the first slide
- **Upload button:** This button is responsible for uploading a PDF
- **Back button**: Allows for clients to go back to the logging window which is shown in Figure 3.4.
- **Exit button**: Allows for clients to exit the room.

Functionality

When the client presses the upload button the clients selects a PDF file, which then gets compressed and converted to JPEG files and the GUI only displays the first image file in the Display interface-this is shown in Figure 3.8.

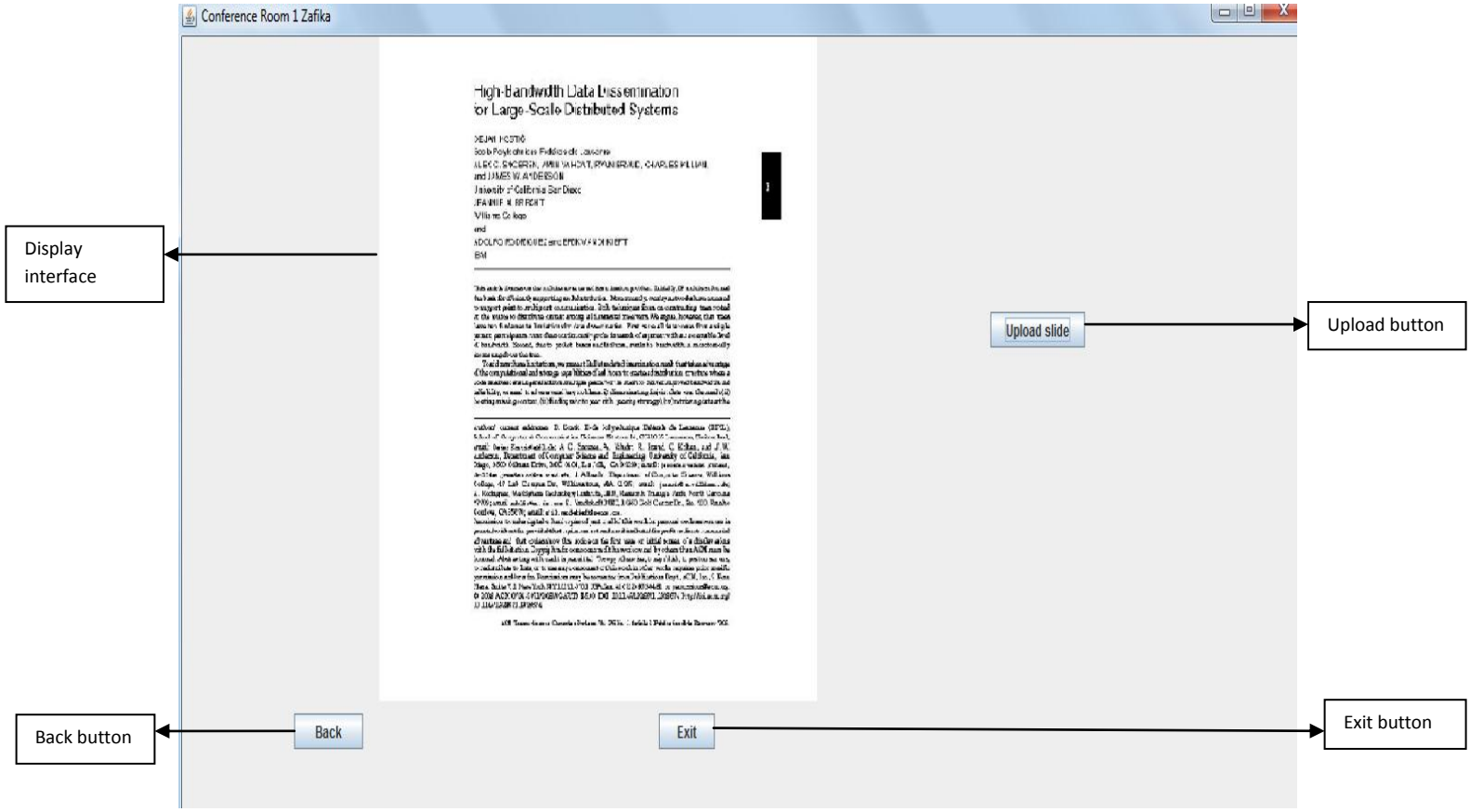


Figure 3.8: AfriMeet presentation application initial design

Implementation

The client uploads a file and then the file gets compressed and converted to JPEG files by the Ghostscript software-this is shown in Figure 3.9. The command for this was inserted in the code and this is the output. The file upload implementation was done using sockets.

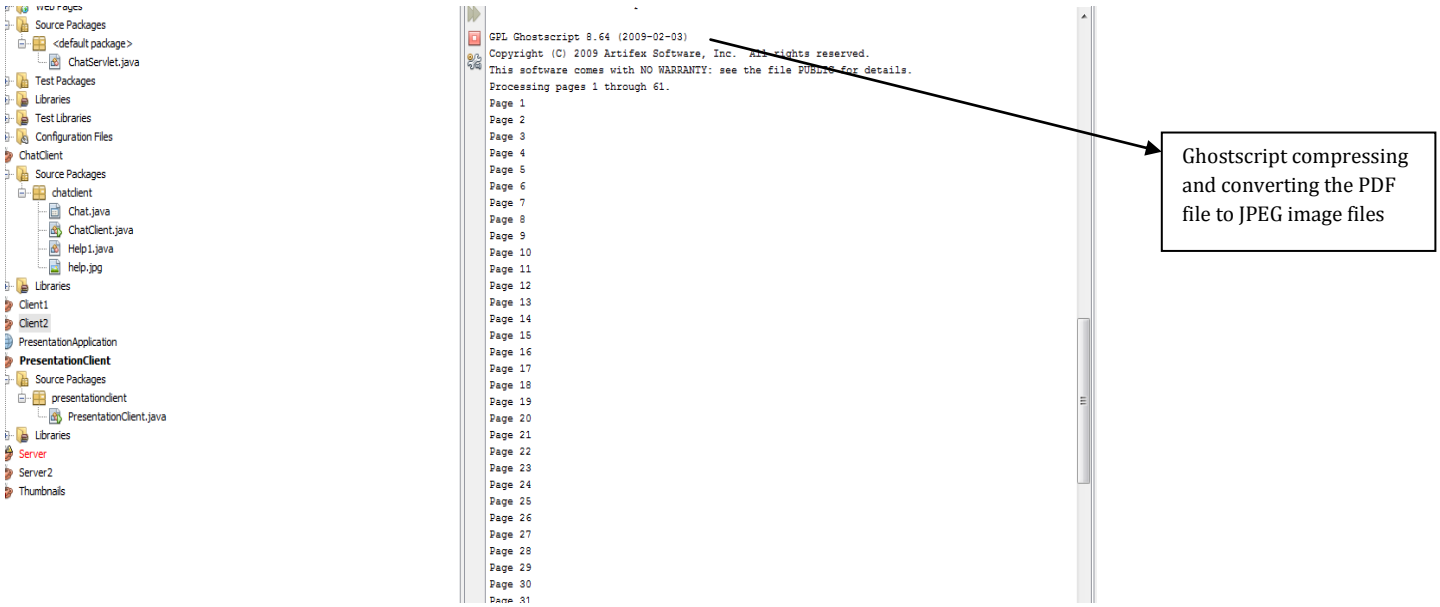


Figure 3.9: PDF files compressed and converted to JPEG image files

Evaluation

The evaluation of the first design was to see if the PDF file does get converted. The application was evaluated by Hussein Suleman and Antoine Bagula. The feedback that was received was to use a Web Application instead of a Java application. The other feedback was to also consider other document file such as PowerPoint, Open office and Microsoft word and this was because not every client has PDF and a client will not always upload a PDF. The other feedback was to have thumbnails, which will show the slides being uploaded.

3.5.2 Second Design Iteration

Design

The design of the application had to in co-operate new features and functionalities, which were suggested from the first design iteration. The new features include:

Features

- **Thumbnails interface:** Interface responsible for displaying all retrieved presentation slides.
- **Retrieve button:** This button is responsible for retrieving all the presentations slide, which were stored in server memory.

Functionality

When the upload button is clicked the PDF files are converted to JPEG image files and these image files are the presentation slides, which are then stored in server memory and are retrieved when the retrieve button is clicked. Once the presentation slides are retrieved from the server they are then displayed as thumbnails in the thumbnail interface and the first presentation slide appears in the display interface.

Figure 3.10 illustrates the features and functionalities of the second design iteration.

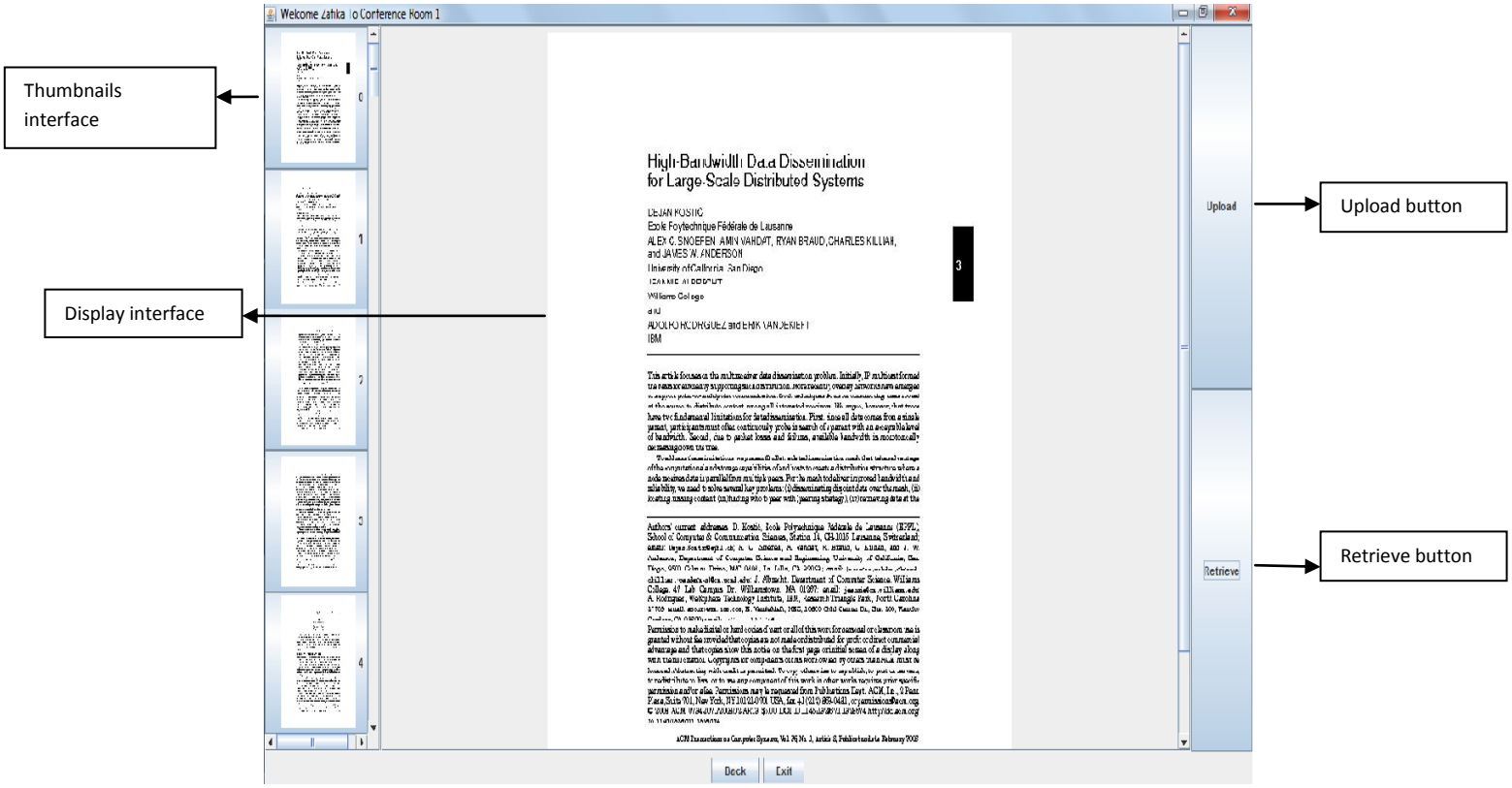


Figure 3.10: Second iterative design

Implementation

The presenter uploads a PDF file, which gets compressed and converted to JPEG image files and then this file is stored in the server memory-this is illustrated in Figure 3.11. The client can then retrieve these image files, which get displayed on the thumbnails interface. Figure 3.12 shows how the files are retrieved and Figure 3.10 shows the presentation slides being displayed in the thumbnail interface after retrieving. When the files get compressed they are saved in a directory folder of the meeting room. If another client in the same meeting room uploads a file, the old files get overridden by the new ones.

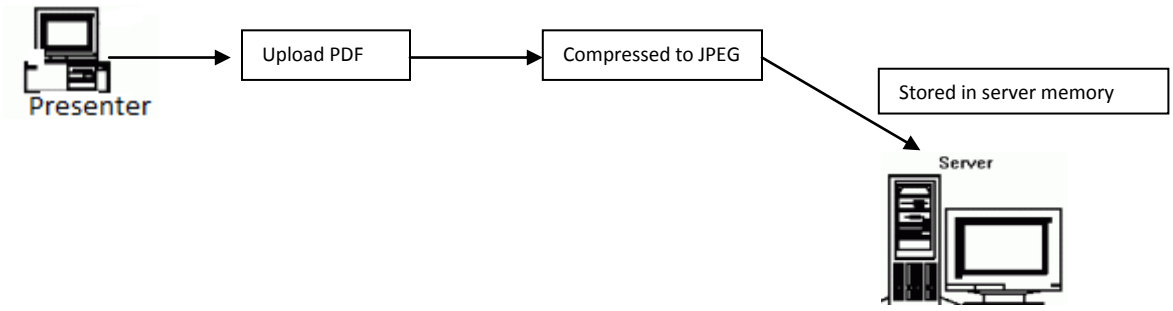


Figure 3.11: Uploading and saving file in server memory

After the files have been saved in memory, any client in the same meeting room can then retrieve the presentation slides, which are JPEG image files, after, before or during the meeting. The clients request for downloading the presentation slides to the server. The serve then goes to memory and checks if there is a file and if there is no file, it sends an error message. However if there is file it sends the presentation slides as JPEG image files to client, were they are then displayed on the thumbnail interface. The reason why uploaded files are being saved to server memory is to allow for clients to download the slides at any time and to prevent the slides from being lost. In the case of a low bandwidth environment if the slides were not uploaded and the client logs out unexpectedly they can easily log in and retrieve the slides again.

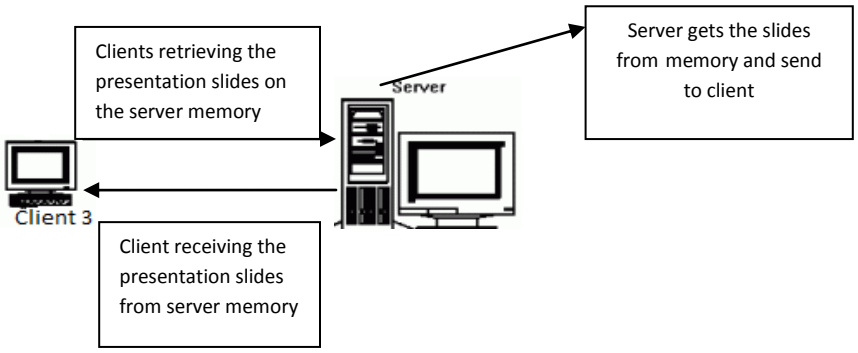


Figure 3.12: Retrieving the presentation slides

Evaluation

The Experiment

The second design iteration and implementation was evaluated using 6 users. The main aim of the evaluation is to test whether users can upload and retrieve the presentation slides and also view the presentation slides in the thumbnails interface. The usability, responsiveness and presence of the application were also evaluated. Three computers were set up and users had to use the application by performing some tasks such as uploading a PDF and retrieving the PDF. Before the experiment started one user was asked to be a presenter and upload while the other users would retrieve the presentation

slides. The experiment took 5 minutes and then the users had to answer some questions. These questions are available in the appendix B2.

User feedback

Responsiveness

Users were happy about the response of the system. Most users were able to upload and retrieve the slides and they could see all the presentation slides uploaded in the thumbnail interface. Users also thought that they should be told that the uploading was successful and that the file is saved in memory.

Usability

The users were happy about the usability of the application-they were able to understand the application and also what was needed of them.

Presence

The users did not feel like they were in a meeting this was because they were only uploading and retrieving the presentations slides-they were no presentation going on. So the suggestions were to be able to send across the presentation slide that is currently being presented.

Extra features

The users thought that it would be a great idea to see the presentation slide being presented. They also thought that it would be great to have some buttons which will allow them to go to the next, previous, first and last slide. They also thought that having a full screen view of the presentation slide will also be great.

3.5.3 Third Design Iteration

Design

After evaluating the design of the second iteration with 6 users, the design had to include some extra features and functions. The design had to implement one function, which was to send the current slide being presented. The features that the application had to in co-operate were: going to the first, previous, back and last slide from current slide. A full screen view of the presentation slide was also designed, which did not have the thumbnail interface, upload button and retrieve button. Figure 3.13 illustrates the final design.

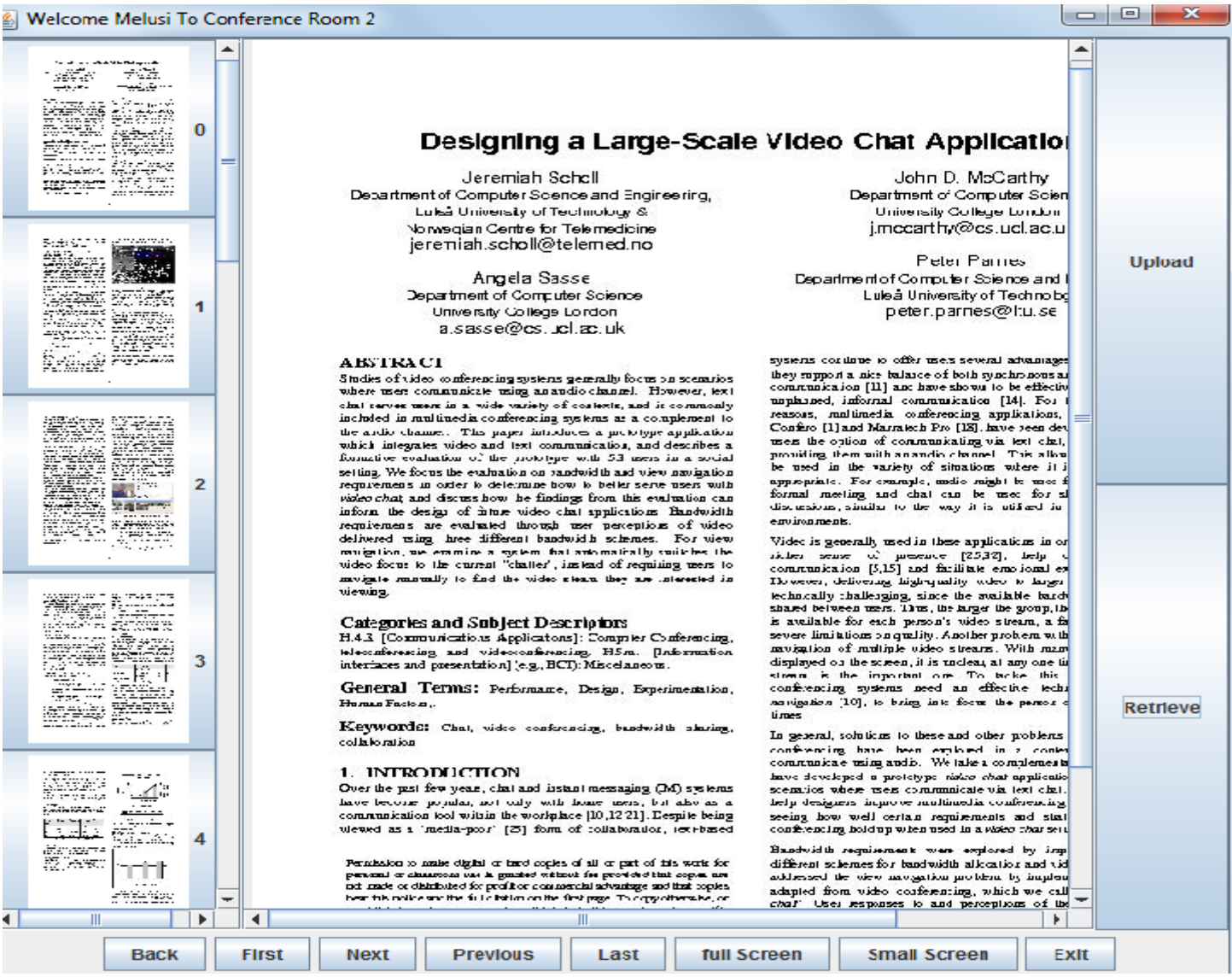


Figure 3.13: Presentation Application Final Design

Features

- **First button:** Allows presenter to go to the first slide of the presentation slides
- **Last button:** Allows presenter to go to the last slide of the presentation slides
- **Next button:** Allows presenter to go to the next slide of the presentation slides
- **Previous button:** Allows presenter to go to the previous slide of the presentation slides

- **Full Screen button:** Allows client to see the full view of the presentation slide without the thumbnail interface, upload and retrieve button.
- **Small Screen button:** Allows client to see the small view of the presentation slide with the thumbnail interface, upload and retrieve button. A small screen view is the one shown in Figure 3.13.

Functionality

The thumbnails and also the next, last, previous and first buttons can all be clicked and when clicked the slide is displayed in the display interface. The next, last, previous and first buttons are usable in both full and small screen view while the thumbnails are only usable in a small screen view.

Implementation

The uploading and retrieving implementation is implemented in the same way as the one in section 3.4.2. The id to be sent across to server is obtained by doing the following:

- By clicking on the thumbnails button, it returns an index and that index is the id.
- For the first button the id is the first slide presentation from the thumbnails.
- For the last button the id is the length of the thumbnails minus one.
- For the next button the id is the id of the presentation slide in the display interface plus one.
- For the previous button the id is the id of the presentation slide in the display interface minus one.

The presenter sends the id across to server and the server broadcasts the id as soon as it receives it automatically to other clients. Figure 3.14 shows this. The presentation slide then gets displayed in the display interface for all clients including the presenter-this is shown in Figure 3.15.

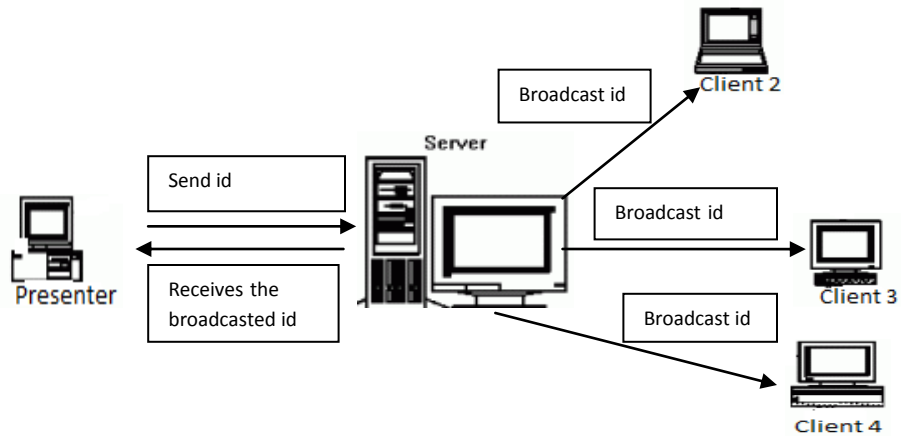


Figure 3.14: Sending the id implementation

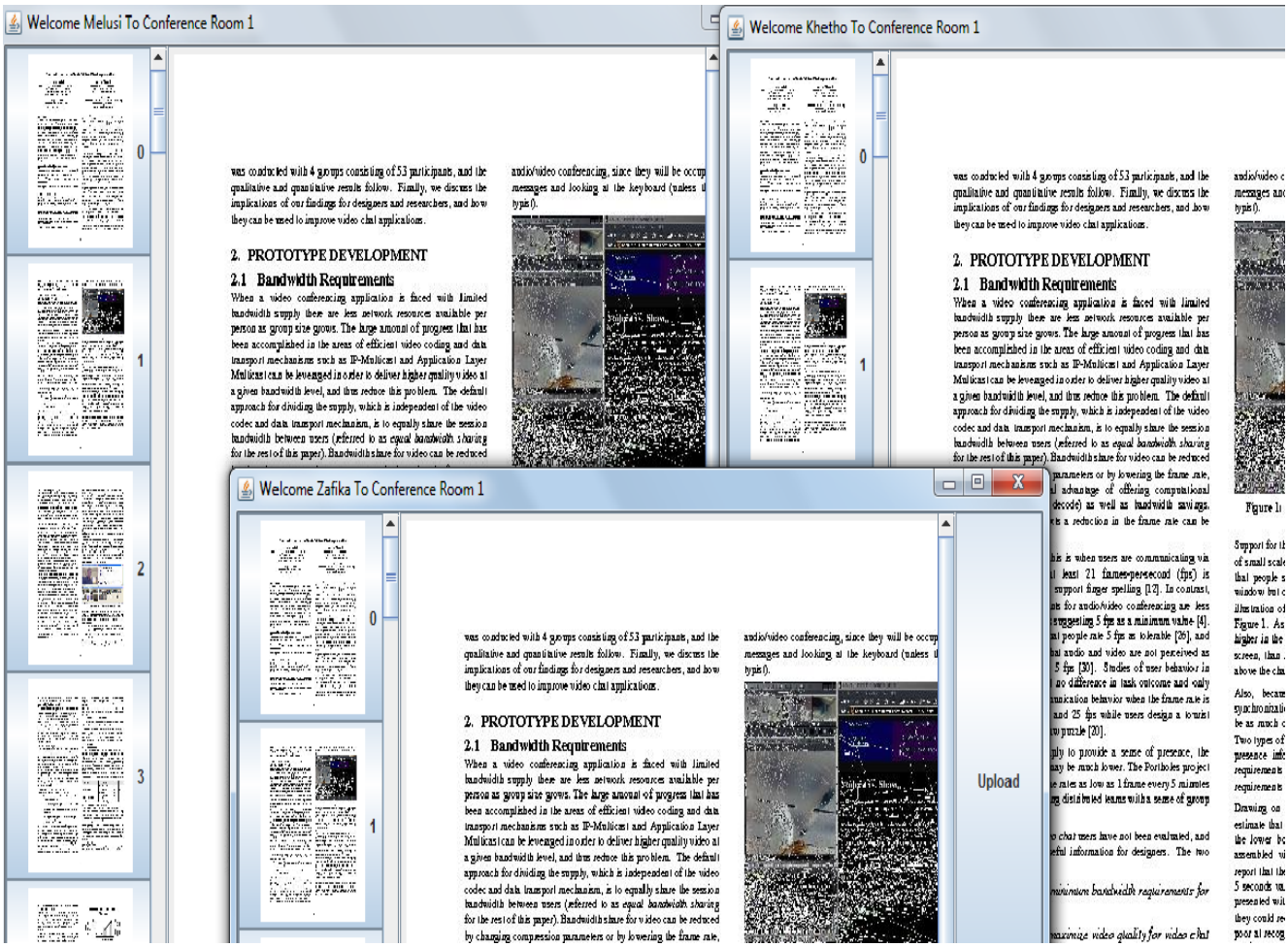


Figure 3.15: Overview of the id being received

Evaluation

The experiment

The same approach to experimentation was used as the one in the second design iteration but now using 12 users instead of 6. Different users were used. The main aim was to see if the users can see the presentation slide being presented and also determine if the application can handle more than one meeting.

User feedback

Responsiveness

Most users were happy about the response of the application; they were able to successfully upload and retrieve the presentation slides and also view them in the thumbnails interface. They could also view the slide that was currently being presented.

Usability

Users found the application to be very easy and usable.

Presence

All of the users now felt that they were in a meeting and they enjoyed using the application. They found the application interesting.

Extra Features:

They were happy with the features but some thought that it would be good to allow them to create slides on the application and then send those slides to the server. They also thought having an interface to write notes will also be good to have in the system. This was however not implemented and could be future work.

3.6 Design and Implementation Challenges

When designing and implementing the applications a number of challenges were faced and are listed below:

- Making the application to be a Web application was challenging as it was new to me.
- For the chat application it was an issue to put the menu in a combobox. The menu did not want to appear on the panel of the combobox. In debugging this issue a new frame had to be created and the combobox did appear on that frame but it appeared at the left hand corner and did not want to appear at the center. After asking for some help it ended up working and the problem was that the layout of panel where the combobox menu was going to be added was not set.
- Adding an image icon to a text area was an issue and leading to the emoticons and smileys features not being implemented.
- Getting a Ghostscript version for Windows was a mission. One almost thought that they did not exist one.
- For the presentation application the implementation of sending and receiving images was hard to accomplish. One had to add a file upload and image io library.
- There were problems in deploying the Web application in Linux-this was because the Tomcat in Linux runs separately on its own unlike in Windows were the Tomcat is embedded in Netbeans. In Windows one could see the log files on Netbeans; in Linux the log files were not shown and the server had to run on the terminal and then access the log files on the terminal.

3.7 Strengths and Weaknesses of the applications

Strengths

Chat application

- Accepts an unlimited number of clients and meetings at the same time.
- Allows the client to send public and private messages.
- Allow participants to change their message style such as colour, font and size.
- The applications works on Windows and Linux.
- Works in a low bandwidth environment this is discussed in Chapter 4. Allows client with a low bandwidth to use the application.

Presentation Application

- Accepts an unlimited number of clients and meetings at the same time.
- Allows client to upload and retrieve presentation slides. Pre-fetching of static data such as slides.
- Provides a view for the presentation slides as thumbnails.
- Allows presenter to select a slide and other clients can see the slide currently being presented.
- Provides a full screen and small screen view.
- Allows client to go to next, previous, last and first slides.
- The applications works on Windows and Linux.
- Works in a low bandwidth environment this is discussed in Chapter 4. Allows client with a low bandwidth to use the application.

Weaknesses

Chat application

- Does not have a list to see all clients in the meeting and this is because the client list is being implemented by Flora. Since there is no client list users find it hard to send private messages because the message fails to send if the receiver name is incorrect.
- Does not allow for clients to add emoticons and smileys.

- There is no encryption of messages and any client can log on to a meeting even one without permission-no access control.

Presentation Application

- Does not create a presenter-every client is assumed to be a presenter. There is no control of the presentation slides, however this can simply be integrated since Flora is implementing this.
- There is no encryption of static data and any client can log on to a meeting, even one without permission-no access control.
- The quality of slides is not good and this is because they have been compressed. The aim is to be able to upload and retrieve slides i.e. pre-load and pre-fetch static data. This problem factor is common was addressed in the background chapter.
- This application only considers uploading PDF file documents.

3.8 Summary

The applications were designed by looking at other tools. The aim was not to compare and make a better design but to look at the features that they have and try to implement those features in such a way that they are usable in a low bandwidth environment. The design and implementation restructured the features by:

Chat application

All messages sent were sent to the server, which broadcasts those messages as soon as it receives them to other clients preventing loss of data when the clients logs out unexpectedly or if the connection is lost.

Presentation application

The uploaded PDF file was compressed to JPEG image files and stored in server memory allowing for any client to then pre-fetch the presentations slides at any time, before or during the meeting. This was to avoid all clients trying to download at the same time which was going to cause delays and cause interruptions in downloading. This was also to help reduce the time it takes to receive the data and thus providing for less bandwidth to be used.

Evaluation

The aim of this evaluation was to build an effective chat and presentation applications. The evaluation was done by user testing and it was mostly based on the responsiveness, usability and presence. In the next chapter the bandwidth that is required by the applications is calculated, this evaluation is to try and see if the applications being built in this chapter do work in a low bandwidth environment.

Chapter 4

Evaluation

4.1 Introduction

In doing the design, implementation and evaluation of the chat and presentation application in an iterative manner, the evaluation in chapter 3 was more on the usability, responsiveness and presence of the applications. In doing some user testing it is found that the applications built are effective in terms of performance, responsiveness and usability. When the user testing was done the bandwidth was not controlled; in other words the testing occurred in a high bandwidth environment. The research questions have not been answered yet and in this chapter we try to perform some analysis and then finally come up with a conclusion. Going back to the research questions discussed in chapter 1:

- Is it possible to build an effective text chat tool that can work with minimal bandwidth?
- Is the pre-loading of static data feasible with low bandwidth?

The research questions have not been answered yet even though it was possible to build an effective text chat tool and pre-load static data. The question still remains if these applications can work in a low bandwidth environment. Simulating a low bandwidth environment requires additional research and implementation and is not what we are aiming for in this document. Using software that simulates bandwidth is sometimes not accurate.

There are basically two approaches in evaluating the applications:

1. Define what we mean by a low bandwidth environment in terms of how many bytes per second we consider as being a low bandwidth environment and called that value Z. Then calculate the required bandwidth for each application-chat and presentation-and then call this value X. The conclusion can be drawn based on the fact that if X is less than Z then we have shown that it is possible to build a text chat application, which works in a low bandwidth environment and pre-loading of static data is feasible with low bandwidth. However if X is greater than Z than we have failed to build a text chat application that works in a low bandwidth environment and pre-loading of static data was not feasible with low bandwidth.

2. Another approach is to perform some analysis based on the data; by data we mean the size of each packet sent and the time it took to send that packet. The analysis is performed by Calculating the bandwidth that each client is consuming and then the total bandwidth required by the chat and presentation applications will be the sum of all the client's bandwidth usage.

From the above two approaches the first one cannot always be accurate to use as the value of Z is not the same for some bandwidth environments, however from some literature (Chen, 2002) and (ElluminateLive, 2010) describes the value Z as being100kps as their low bandwidth environment, and evaluate their system based on this value as their low bandwidth environment. The evaluation of the applications will also be based on this value. The second approach on the other hand is more convenient and accurate to use, as all the required information can be found.

In summary the approach to be taken in analyzing the chat and presentation applications is to calculate the required bandwidth that the applications are using and then conclude that if the client has this bandwidth then he/she can use the applications. For each application some calculations are done that provide some analysis on the bandwidth required. In this document, only sending data to the server is considered (pre-loading) and not receiving the data (pre-fetching). This is because receiving of data is part of latency, which is the time it takes to receive a packet and that is not what we are interested in.

The required bandwidth for chat application is discussed in section 4.2 and for presentation application in section 4.3. The results and findings are discussed in section 4.4.Finally this chapter end by providing a brief summary of the overall chapter in section 4.5.

4.2 Chat Application Evaluation

Recall that bandwidth is the amount of data being transferred over a specified time period and is given by bytes per second.

$$\text{Bandwidth} = \frac{\text{Amount data in bytes}}{\text{Time taken to send in seconds}} \quad (4.1)$$

Firstly proper conversions must be done. The text messages are strings and a string is made of characters. One character equals to one byte so if the length of the string is 8 then it means there are 8 characters and therefore 8 bytes. The time it takes to send a packet is given in milliseconds and therefore the time should be divided by 1000 to give us seconds. The number of bytes that are sent as well as the time it takes to send a text messages are stored in a log file with extension ".csv" and separated by a comma. This is because we want the file to be opened by Microsoft Excel. We then plot a graph of amount of data sent in bytes versus time it takes in seconds for each client and obtain the following graphs:

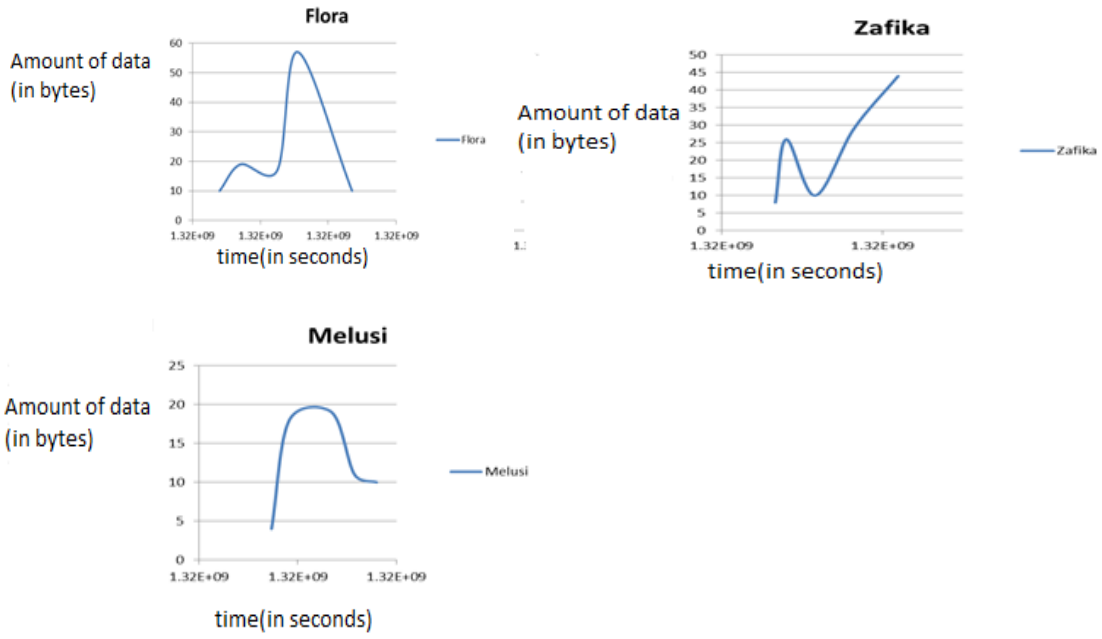


Figure 6.1: Bandwidth used by each client to send both public and private messages

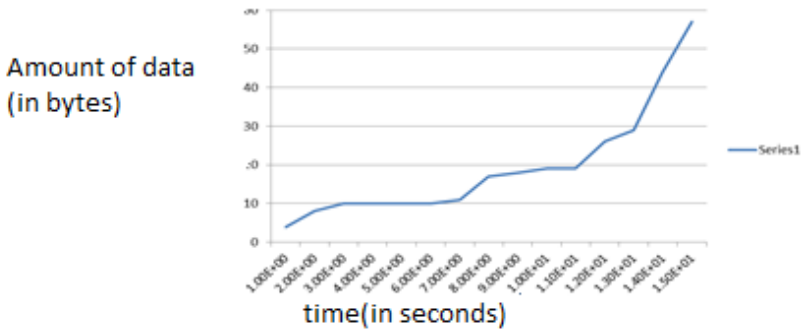


Figure 4.2: The total bandwidth used by all clients in sending public and private messages.

Discussion

Figure 4.1 shows the graph of each client's bandwidth usage and from the graphs we can see that the graph is not linear but parabolic. What we see from the graphs make sense as it shows that the client used the bandwidth to a point where it is minimum or maximum. Figure 4.2 shows the overall bandwidth usage for all clients and from the graph we see that the amount of bandwidth used for all clients fluctuates over time based on amount of data sent. All the graphs are not linear, however the minimum, maximum and average bandwidth used can be calculated.

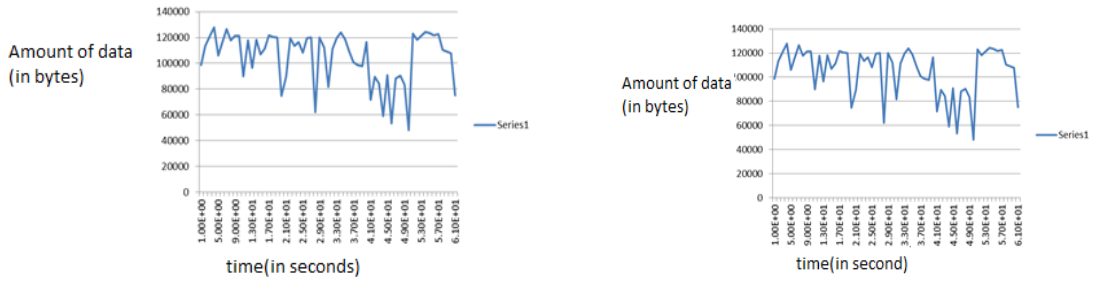
We now then calculate the bandwidth required by the application. Begin by setting a specific time period and then calculate the bandwidth used in that time period. We set the time period to be in minutes and then we allow for clients to chat within that minute and the total amount of data sent is given by the total text messages that the clients send. The total bandwidth used by the system is then calculated using the equation in section 4.2.

Table 4.1 The bandwidth used by the chat application

Total Amount of data sent(in bytes)	Time period(in minutes)	Bandwidth(bps)
116	1	1.9
216	2	1.8
626	5	2.1
1142	10	1.9
1617	15	1.8

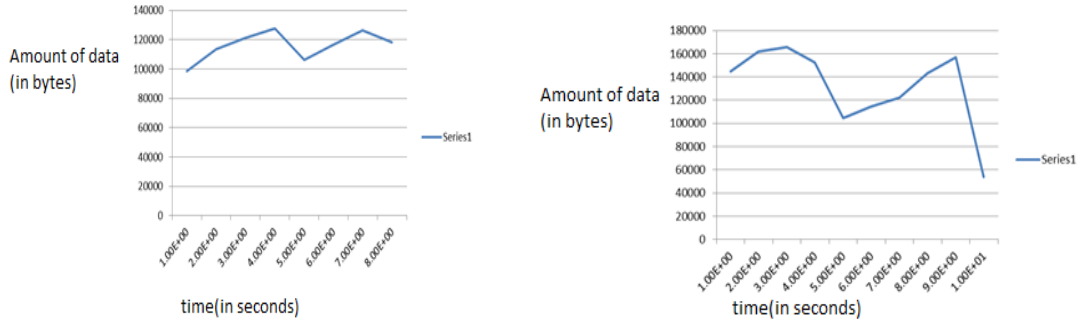
4.3 Presentation Application Evaluation

Using the same phenomena and formula to calculate bandwidth as the chat application, we get the bytes for each image file, the time it takes to send each image file and also the name of the client who is sending the file and store all this information in a log file with extension ".csv" and then determine the bandwidth that each client has used. The following graphs are obtained.



(a)Pre-loading a big file (b) Pre-fetching a big file

Figure 4.3: Pre-loading and Pre-fetching a big file



(a)Pre-loading a small file (b) Pre-fetching a small file

Figure 4.4: Pre-loading and Pre-fetching a small file

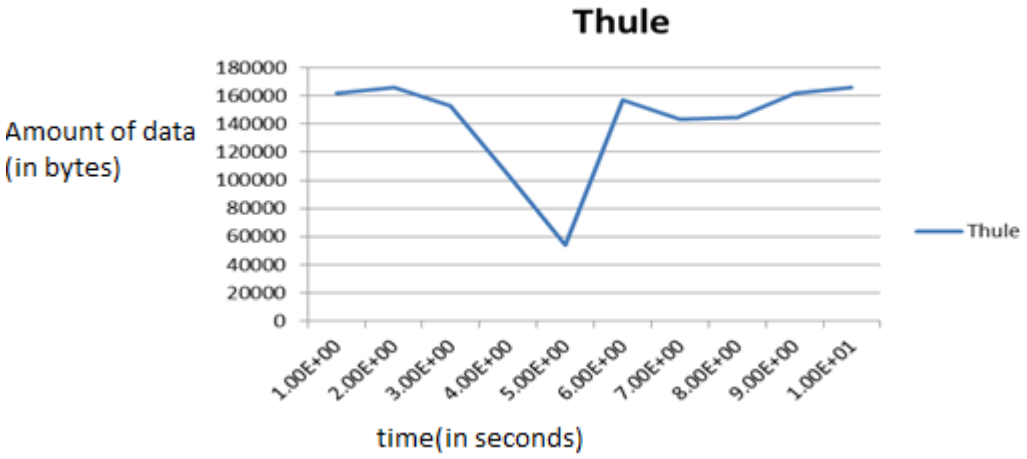


Figure 4.5: Sending across the presentation slide being currently presented

Discussion

For the above graphs only two presenters uploaded the file to server. One presenter uploaded a big file, which the size of 6426614 bytes and the other presenter uploaded a small file which the size of 928300 bytes. The above graphs show the amount of data of image files sent versus the time it took to pre-load and pre-fetch PDF file. We notice that the graph fluctuates depending on the amount of data sent. If the data sent is small the time it takes to send it is also small. Assuming that it took 1 minute to pre-load and pre-fetch big and small presentation slides, and 1 second to send across the id of current slide, the

calculations for the bandwidth usage of the presentation application in uploading and downloading the presentation slides is provided in the Table 4.2 send the files.

Table 4.2 The bandwidth used by the presentation application

Total Amount of data (in bytes)	Time period(in seconds)	Bandwidth(bps)
6426614 (Pre-loading)	10	642661.4
928300(Pre-loading)	3	309433.3
6426614(Pre-fetching)	6	1071102.3
928300(Pre-fetching)	2	464150
61pages(Sending across 1 id at a time)	1	61
10pages(Sending across 1 id at a time)	1	10

4.4 Results and Findings

On average the chat application uses a bandwidth of 1.9 bps, this was found by adding all the total bandwidth from the Table 4.1 and dividing by 5, since 5 time period intervals were used i.e. 1, 2, 5, 10, and 15. The presentation application uses a total bandwidth of 108148.6 bps. This was found by adding all the total bandwidth from the Table 4.2. However pre-loading of presentation slides uses on average 73238.05, this was found by averaging the bandwidth for preloading only.

The bandwidth for chat application is very small and this proves the point from the background work that a chat application does not use a lot of bandwidth. For both applications it was noticed that if the amount of data sent increases the time increases, leading to the amount of bandwidth used to also increase-this is described by the points on the graphs as they fluctuate.

For the presentation application, if a presenter sends a big file the amount of bandwidth used is very high. This makes sense as the image files carry a lot of bytes unlike strings characters.

4.5 Summary

In this chapter we calculated bandwidth used by each client and also by the application as a whole based on the definition of bandwidth. Two steps were taken one was to plot the graph of amount of data versus time and the other step was to calculate the amount of data sent over a specified time period. A table was built, which provided the total amount of bandwidth used by the applications over a specified time period. The calculations and the evaluations provided enough information to conclude on the research questions that's it is possible to build a text chat tool that can work in a low bandwidth environment and pre-loading of static data such as presentation slides is feasible in a low bandwidth environment.

Chapter 5

Future Work

There are several ways in which the work done in this project can be extended. This involves additional features of the chat and presentation interface design and functionalities.

5.1 Extra features

The emoticons and smileys could not be added in the chat application, however not implementing this features was a good idea because one have to consider how to implement them such a way that they do not use a lot of bandwidth. Different interfaces for private messages for each client and one interface for displaying public messages should also be designed and implemented instead of having just one interface displaying public and private messages.

For the presentation application an interface which allows for clients to write notes and be able to save them can be implemented.

5.2 Implementation

The application only considered uploading a PDF file and in future maybe Microsoft PowerPoint, OpenOffice and Microsoft Office document files can be considered as well. The easiest and most efficient way is to convert theses document files to PDF. Some clients may not have a PDF plugin and they might need to install it. So what is more convenient is to simply have software that will do the conversions of these document files to PDF. The other thing is that the applications can be made to work on other operating systems since in this document we only considered Windows and Linux operating systems.

5.3 Evaluation testing

In order to get to more accurate results on the bandwidth required by the applications more calculations on the bandwidth usage at a specific time period needs to be done as well as more graphs for chat application. For the presentation application more files should be for pre-loaded and pre-fetched, more graphs needs to also be drawn and also the bandwidth usage that each file consumes in uploading and downloading.

Chapter 6

Conclusion

This project was aimed at building an effective text chat application that can work in a low bandwidth environment and also allow for the pre-loading and pre-fetching of presentation slides in a low bandwidth environment to be feasible. The usability, performance and responsiveness of the system were tested using users, concluding from the results obtained that the chat and presentation applications were effective and usable since the design of the applications were based on an iterative approach and each method and feature was evaluated and improved on.

In answering the research questions, it was found that it was possible to build an effective text chat application that could work in a low bandwidth environment since on average the chat application uses 1.9bps(0.009kps) of bandwidth which is less than the Z value of ElluminateLive, (2010) which was 100kps.

For the presentation application the bandwidth used relies heavily on the size of data sent. However the presentation application uses 108148.6 bps of bandwidth. For pre-loading presentation slides, the amount of bandwidth used by the presentation application is 73238.05 bps which is 73kps. This is a very low bandwidth compared to ElluminateLive, (2010) bandwidth of 100kps; therefore it was feasible to pre-load presentation slides in a low bandwidth environment.

Appendices

Appendix A: Consent Form

Consent form

University of Cape Town

Department of Computer Science

Please read the following before signing:

1. I agree to participate in this experiment at my own free will.

2. I agree for the responses that I provide to be used for research purposes.

3. I am fully aware that my personal details and responses that I provided will only be visible to me and the researcher.

4. I have been given full information about the experiment and told to ask questions.

5. I understand that I have the right to withdraw from this experiment at any stage.

6. I have read this consent form and the information it contains.

Name of Participant:
Signature of Participant:
Date:

Appendix B: Evaluation Questionnaires

B1: Chat Application Questionnaire

Thank you for taking part in this experiment. Please take note that the answers being provided will only be visible to me only. And once done the information provided will be kept safe where no one except me have access to it. If you have any problems please call me and ask. The aim of this experiment is to see if you can use the application.

You are required to:

Step 1: Enter your name and conference number 1 and then press the enter button.
You are now in a chat window.
Step 2: You can now chat to other participate. Please note that to send a private message you have to use @name of person you want to send to followed by space and then the message

After 5 minutes you can then stop chatting and answer the questions below:

1. Have u ever used a chat application before? If yes please provide which application(s) it was e.g. Facebook, Gtalk, etc.

2. Did you have any difficulties in using the application? If yes please specify

3. Can you send and receive both public and private message?

4. Did you have issues using the private message? If yes please specify

5. Are u happy about the system response? Are you happy the way the error messages are handled?

6. Did it feel like you were in a meeting? Please explain

7. Is there anything that you think should be added to the system?

8. Is there anything that you think should be removed?

9. What do you think about the design?

10. If you were to build an application of this nature, how would you do it differently?

Thank you again for taking part in this experiment. Have a great day

B2: Presentation Application Questionnaire

Thank you for taking part in this experiment. Please take note that the answers being provided will only be visible to me only. And once done the information provided will be kept safe where no one except me have access to it. If you have any problems please call me and ask. The aim of this experiment is to see if the presenter can upload a PDF and if participants can retrieve the PDF.

You are required to:

Before we start who would like to be a presenter? Answer yes or no to the following

Are you a presenter?

Are you a participant?

Enter your name and conference number 1 and then press the enter button. You are now in the presentation window.

Presenter:

Step 1: Upload a PDF file from the desktop.
Step 2: Retrieve that PDF.
Step 3: Start presenting. Always double click to change slide

Participants:

Step 1: Retrieve the PDF that was uploaded by the presenter.
Step 2: Look at the slides that are being presented.

After 5 minutes the presenter will stop presenting and then you can answer the questions below:

1. Have u ever used any Web conferencing tools such as Adobe Connect, Zoho, etc. before? If yes please provide which tool(s) it was.

2. Did you have any difficulties in using the application? If yes please specify

3. **Presenter**: were you able to upload and retrieve the PDF?

Participant: were you able to retrieve the PDF uploaded by presenter?

4. Are u happy about the system response? Are you happy the way the error messages are handled?

5. Did it feel like you were in a meeting? Please explain

6. Is there anything that you think should be added to the system?

7. Is there anything that you think should be removed?

8. What do you think about the design?

9. If you were to build an application of this nature, how would you do it differently?

Thank you again for taking part in this experiment. Have a great day

References

Aweya, J. (2003). Transmission Control Protocol.[Online].Available: http://onlinelibrary.wiley.com.ezproxy.uct.ac.za/doi/10.1002/0471219282.eot202/full. [21 October 2011]

Baecker, R. (2003). A principled design for scalable internet visual communications with rich media, interactivity, and structured archives. CASCON '03 Proceedings of the 2003 conference of the Centre for Advanced Studies on Collaborative research , (pp. 16 – 29): IBM Press 2003.

Baecker, R., Baran, M., Birnholtz, J., Chan, C., Laszlo, J., Rankin, K., et al. (2006). Enhancing interactivity in webcasts with VoIP. Proceeding CHI EA '06 CHI '06 extended abstracts on Human factors in computing systems, (pp. 235-238) Canada: ACM New York, NY, USA 2006.

Barakonyi, I., Fahmy, T., & Schmalstieg, D. (2004). Remote collaboration using Augmented Reality Videoconferencing. GI '04 Proceedings of Graphics Interface 2004, (pp. 89-96). Canada: Canadian Human-Computer Communications Society School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada 2004.

Becker, K. (2007). Using Elluminate in CS. Journal of Computing Sciences in Colleges , 23 (2): (pp. 73-75), December 2007.

Bekkering, E., & Shim, J. (2006). Trust in videoconferencing. Communications of the ACM - Services science , 49 (7): (pp. 103 – 107), July 2006.

Chen, M. (2002). Achieving effective floor control with a low-bandwidth gesture-sensitive videoconferencing system. Proceedings of the tenth ACM international conference on Multimedia - MULTIMEDIA '02, (pp. 476 - 483). ACM New York, NY, USA 2002

Chen, M. (2003). A low-latency lip-synchronized videoconferencing system. CHI '03 Proceedings of the SIGCHI conference on Human factors in computing system, (pp. 465 - 471). ACM New York, NY, USA 2003

Ciocco, M. D., Toporski, N., & Dorris, M. (2005). Developing a synchronous web seminar application for online learning. SIGUCCS '05 Proceedings of the 33rd annual ACM SIGUCCS fall conference (pp. 36 - 39). ACM New York, NY, USA 2005.

ElluminateLive. Best Practices for Users on Low-bandwidth and Dial-up Connections. http://cairnssde.eq.edu.au/docs2010/elluminate/lowbandwidth.pdf, 2010. Accessed: 28 October 2011

Egido, C. (1988). Video conferencing as a technology to support group work: a review of its failures. CSCW '88 Proceedings of the 1988 ACM conference on Computer-supported cooperative work (pp. 13 - 24). ACM New York, NY, USA 1988.

Gong, F. (1994). Multipoint audio and video control for packet-based multimedia conferencing. MULTIMEDIA '94 Proceedings of the second ACM international conference on Multimedia (pp. 425-432). ACM New York, NY, USA 1994.

Hans, P., & Garcia. L. (1997). Floor control for multimedia conferencing and collaboration . Mutimedia Systems, 5 (1): (pp. 23-38).1997.

Hargreaves, D. M., & McCown, B. R. (2008). Low-cost, low-bandwidth online meetings between farmers and scientists. OZCHI '08 Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat (pp. 271-274). Sydney: ACM New York, NY, USA 2008.

Heeler, P., & Hardy, C. (2005). A preliminary report on the use of video technology in online course. Journal of Computing Sciences in Colleges , 20 (4): (pp. 127-133), April 2005.

Ichimura, S., & Matsushita, Y. (2005), Lightweight Desktop-Sharing System for Web Browsers. Third International Conference on Information Technology and Applications (ICITA'05), Vol. 2: (pp.136-141), 2005

Kamath, S. (2005). A compression scheme for video conferencing. ACM-SE 43 Proceedings of the 43rd annual Southeast regional conference, Vol.2: (pp. 345 - 346). ACM New York, NY, USA 2005.

Koh, E. (2010). Conferencing room for telepresence with remote clients. GROUP '10 Proceedings of the 16th ACM international conference on Supporting group work (pp. 309-310). USA: ACM New York, NY, USA 2010.

Lai, K., & Baker, M. (1999). Measuring bandwidth. INFOCOM '99 Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies Proceedings IEEE, Vol.1: (pp.235-245), March 1999

Lu, Y., Zhao, Y., Kuipers, F., & Van Mieghem, P. (2010). Measurement study of multi-party video conferencing. NETWORKING 2010 Lecture Notes in Computer Science , 6091(2010): (pp. 96–108), April 2010

Nijholt, A., Rienks, R., Zwiers, J., and Reidsma, D. Online and off-line visualization of meeting information and meeting support. In Proceedings of The Visual Computer. 2006, (pp. 965-976).

Olsen, L. (2006). Being there: a "teach them to fish..." approach to training and support using WebEx™, videoconferencing, and the telephone. SIGUCCS '06 Proceedings of the 34th annual ACM SIGUCCS fall conference (pp. 295 - 300). ACM New York, NY, USA 2006.

Sabin, M., & Higgs, B. (2007). Teaching and learning in live online classrooms. SIGITE '07 Proceedings of the 8th ACM SIGITE conference on Information technology education (pp. 41-48). ACM New York, NY, USA 2007.

Scholl, J., McCarthy, J., & Harr, R. (2006). A comparison of chat and audio in media rich environments. CSCW '06 Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work (pp. 323 - 332). ACM New York, NY, USA 2006.

Scholl, J., Parnes, P., McCarthy, J. D., & Sasse, A. (2005). Designing a large-scale video chat application. MULTIMEDIA '05 Proceedings of the 13th annual ACM international conference on Multimedia (pp. 71 - 80). ACM New York, NY, USA 2005.

Scott, M. (1998). Frameworks for component-based client/server computing. ACM Computing Surveys (CSUR) Surveys Homepage table of contents archive , 30 (1): (pp. 3-27).ACM New York ,NY,USA 1998.

Seeling, P. (2010). Web conferencing traffic- an analysis using Dimdim as example . International Journal of Computer Networks & Communications (IJCNC), 2 (6), November 2010.

Sun, J., & Regenbrecht, H. (2007). Implementing three-party desktop videoconferencing. OZCHI '07 Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining Client Interfaces (pp. 95 - 102). New Zealand: ACM New York, NY, USA 2007.

Prasad, R., Dovrolis, C., Murray, M., & Claffy, K. (2003). Bandwidth estimation: metrics, measurement techniques, and tools. Network, IEEE, 17 (6): (pp. 27-35).November-December 2003.

Takao, S. (1999). The effects of narrow-band width multipoint videoconferencing on group decision making and turn distribution. WACC '99 Proceedings of the international joint conference on Work activities coordination and collaboration (pp. 109 - 116). Japan : ACM New York, NY, USA 1999.

Turletti, T., & Huitema, C. (1996). Videoconferencing on the Internet. IEEE/ACM Transactions on Networking (TON) , 4 (3): (pp. 340 – 351), June 1996.

Yang, C. (2001). Client-interaction supported data-retrieving engine for distributed multimedia presentations, IEEE International Conference on 2001, Vol.10: (pp. 3244 - 3250), 2001