HONOURS PROJECT REPORT

AfriMeet: An Internet meeting tool designed for low bandwidth and unstable network conditions

Tresor Mvumbi

Supervised by

Hussein Suleman

	Category	Min	Max	Chosen
1	Requirement Analysis and Design	0	20	5
2	Theoretical Analysis	0	25	0
3	Experiment Design and Execution	0	20	15
4	System Development and Implementation	0	15	15
5	Results, Findings and Conclusion	10	20	15
6	Aim Formulation and Background Work	10	15	10
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

A Web meeting software tool is an Internet-based tool offering a virtual environment for remote meeting and collaborative work among geographically dispersed participants. Online meetings can save time and expenses that an ordinary face-to-face meeting would require. However, carrying real-time communication within the Internet packet-switched network is a challenging task. This fact is particularly true for South Africa and many other developing countries, which often rely on low-bandwidth and unstable connections.

The aim of this project is to develop a Web conferencing solution that could reliably host meetings with constraining Internet conditions typical of Africa. Approaches used to achieve this goal include: reprioritization of multimedia streams, half duplex communication mode, image differentiation, and audio-video compression.

The experimental prototype developed delivers a clear audio stream (radio quality) at 28 kbps (2.5 KB/s). The sound quality resulted is perceived as either good or excellent by 84% of users who evaluated the system. The overall sense of presence and feeling of involvement in an actual meeting is ranked at least “good” by 91% of testers who assessed audio-video communication. Performance experiments show that a server with 512 kbps (64 KB/s) total bandwidth can handle an audio meeting with 18 simultaneous participants; or 11 participants when a low frame rate (0.2 FPS) video stream is used as more.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF GRAPHS

CHAPTER 1 – INTRODUCTION

Internet technologies have developed quickly during the last decades; they are currently mature enough to support low-cost real-time communication services. These developments have literally changed the way people meet and collaborate to make decisions (Suduc et al. 2009). New collaborative and conferencing environments based on the Internet are now common tools for many people and organizations around the world.

A Web meeting system is an Internet-based tool offering a virtual environment for remote meeting and collaborative work among geographically dispersed participants. Remote conferencing can avoid travel expenses and time required for face-to-face meetings (Gurhan et al. n.d.). This explains the worldwide spread of Web-based conferencing tools.

A Web meeting environment offers tools and methods to support remote meetings. Some of the common features include audio and video communication, slide show presentation sharing, a shared white board for annotation, screen sharing and text chatting (Jain et al. 2003). Figure 1.1 illustrates a possible Graphical User Interface (GUI) of a Web meeting system.

Figure 1.1: Example of graphical user interface of a Web meeting system

1.1 Problem statement

The Internet provides a public packet-switched network with a relatively high probability of loss and random delay in packet delivery (Foo et al. 1999). These transmission problems directly affect any service relying on the Internet for communication. Another important factor affecting Internet services is the amount of bandwidth available. The bandwidth is a measurement of data quantity that a link can transmit per unit of time. This factor is particularly important for South Africa and many other developing countries where the bandwidth is relatively low.

Despite the growth in use of ICTs on a global scale, Internet access is still limited in most African countries (Ajuwon & Rhine 2008). In Africa, around 50 million Internet users share the estimated total bandwidth of 40 Gbps (Gray & Minges 2008). This represents an average of only 0, 84 Kbps per person!

As a service relying on the Internet, a Web meeting tool is directly impacted by underlying networking problems. Web meeting tools offer several features that are differently affected:

- Audio conferencing: unpleasant or even unintelligible sound playback;
- Video conferencing: blocking and jerky video playback;
- Text chat: messages received with high delay; and
- Screen and presentation sharing: very poor image quality, not readable.

These problems can seriously degrade the communication quality, making an Internet conferencing solution practically useless. That is why most current commercial and open source Internet meeting solutions cannot reliably be used in the African context (Edigo 1988).

1.2 Research questions

The features or services offered by Internet meeting tools have different needs in terms of real-time and bandwidth usage. Consequently, they are not all affected the same way by networking problems. Since the primary objective of a Web meeting tool is to support human communication, special emphasis need to be put on user experience and satisfaction. One research objective in this project is to study how the user experience can be positively enhanced despite networking problems. Investigations are focused on how to reprioritize features and offer the best tradeoff between quality and usability. The main research questions are:

- Is it possible to construct an effective audio-video conferencing tool that can cope with low bandwidth and unstable networking conditions?
- Is pre-loading of all static documents feasible when the bandwidth is low?
- Is it possible to build a system that can manage meeting procedures efficiently despite constraining Internet conditions?

1.3 System overview

In order to respond to the research questions, an experimental prototype was developed. It was first assumed that the underlying Internet connection is low bandwidth and unstable. This assumption motivated core system design choices, which focused on delivering an acceptable and satisfactory user experience. The resulting system has 3 modules, developed by 3 group members. Figure 1.2 illustrates the global system overview.

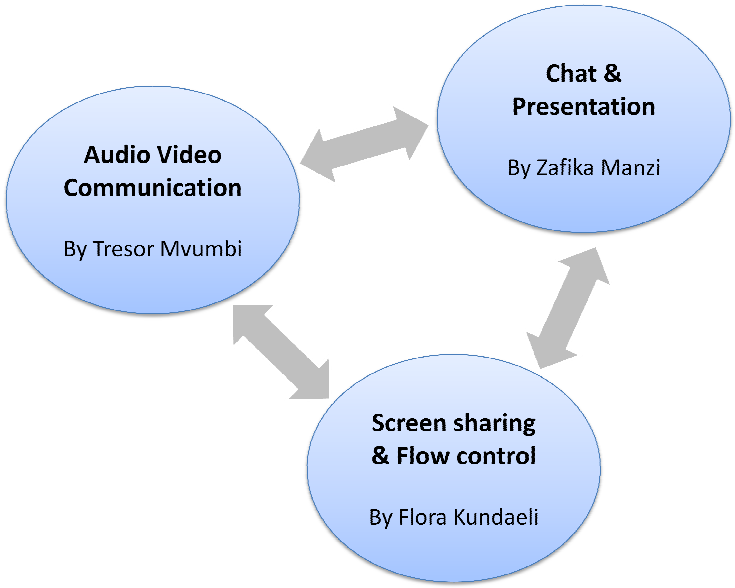


Figure 1.2: System overview

The audio module aims to provide a clear sound stream conveying an intelligible speech when using the smallest bandwidth possible. The video module complements audio and enhances the sense of presence by allowing participants to see each others during the meeting. When the bandwidth becomes insufficient, the audio stream is prioritized.

1.4 Report outline

This report is divided into 4 major chapters:

The background chapter reviews the current state of the art in audio-video meetings over the Internet. Existing tools are presented and compared; followings, is a discussion on networking problems and their effects on the quality of user experience. The rest of the chapter is an in-depth review of research and work done on how to enhance the quality of experience of audio-video Internet-based tools.

The design chapter presents the detailed and high level system architecture. It describes approaches used for audio and video recording, compression and streaming. The design of a congestion control system is presented at the end.

The implementation chapter addresses technical details regarding how the system was developed. Technologies and programming libraries used are presented, along with the implementations of algorithms. Outcomes from the 3 development iterations are presented at the end of this chapter.

The evaluation chapter presents the approach used to test the system and assess the quality of user experience. The first part tests system performance and bandwidth requirements. The second part presents results of system evaluation by users.

CHAPTER 2 – BACKGROUND

Web meeting components rely on the Internet environment for communication. However, carrying real-time communication within the Internet packet-switched network is challenging. This task is even more difficult when using a low bit-rate or low-bandwidth Internet connection (Foo et al. 1999).

Real-time audio and video communication is the most affected features as a result various networking problems (Gurhan et al. n.d.). Varying and long delays combined with packet loss in the underlying network results in unintelligible audio and jerky video playback. These problems badly degrade the user experience, preventing Web meeting tools from being widely accepted (Foo et al. 1999).

In this chapter, some of most successful current Web meeting tools are presented and compared. Following is a discussion on the effects of poor network quality of service on real time audio and video communication. This chapter mainly reviews research aiming to enhance the user experience for real-time audio and video communication operating under constraining Internet conditions. The last paragraph is a comparison, critique and discussion of the reviewed literature.

2.1 Current Web-meeting tools

The evolution of Internet technologies made real-time and low-cost communication a reality. More than ever, real-time multimedia communications are used daily across the world, supporting all kind of needs. To support this growing communication mode, lots of software packages have been developed and deployed. Table 2.1 lists and compares five of the currently most successful Web conferencing tools (Anon 2011).

Table 2.1: Top 5 most successful current Web meeting systems

Product	Description
1) Citrix GoToMeeting 4.8	User-friendly graphical user interface with all the key features. Good for Windows and Mac
2) Adobe Connect 8	Can run on most of current operating systems. Offer intuitive GUI with a large range of features.
3) BeamYourScreen 4.0	Online solution with a user-friendly interface. Feature all important functionalities. Can run on Windows, Mac and Linux
4) Cisco WebEx Meeting Center 8.5	Offer several features, but complex solution. Can run on most of current operating systems.
5) RHUB Go MeetNow 4.3	Good solution with broad range of features, can run on Windows and Mac

2.2 Effects of networking problems on the quality of user experience

The following networking problems can directly affect the quality of stream delivered (Watson & M. A. Sasse 1997) :

- varying delays or jerks in packet delivery;
- high latency;

- high rate of packet loss;
- low bit-rate transmission channel;
- packet duplication; and
- unordered packet delivery.

The above problems can have one or more of the following effects on quality of audio and video (Gurhan et al. n.d.):

- Frequent interruptions and unintelligible audio stream;
- Jerky video play-out; and
- Audio playback not synchronized to video stream.

The quality of audio and video stream can be objectively measured and expressed using factors like packet delay, loss rate, frame rate or bit-rate. But the quality as perceived by users is more complex to predict using objective measurements.

The notion of quality for a user is very subjective and closely related to the task undertaken. Figure 2.1 lists some common factors affecting user opinion on the quality of audio and video streams (Watson & M. A. Sasse 1997).

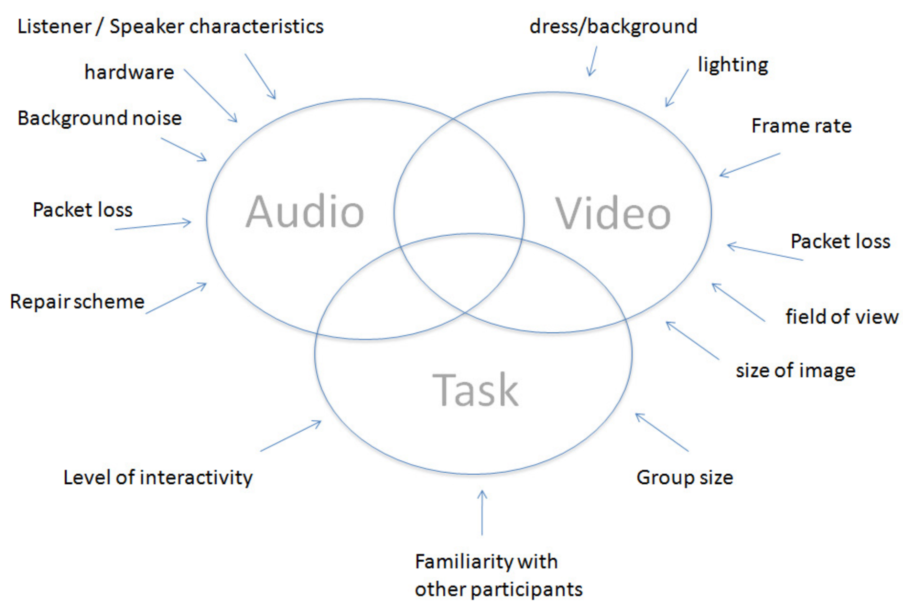


Figure 2.1: Factors affecting the quality as perceived by the end-user (Watson & M. A. Sasse 1997)

Some of the above factors have greater impact than others. For audio streams for example, the rate of packet loss is the most influential factor, with a direct influence over speech intelligence and the level of listening effort. For video, the frame rate is the most important factor that can affect the video quality as perceived by the user. Frame rate aside, other influencing factors include image size, lighting and the degree of synchronization with the audio.

Table 2.2 summarizes how the network quality of service (QoS) degradation can affect the user experience (Scholl et al. 2005)(Watson & M. A. Sasse 1996) (Hargreaves & McCown 2008).

Table 2.2: Effects of QoS degradation on the user experience

QoS degradation	**Consequence on user experience**
Long and random delays and packet loss	Directly affect fluency of multimedia stream, leading to poor user concentration
Frequent interruptions	Badly alter the sense of “presence”, can be very irritating for the user
Low bandwidth	Delivery of poor quality content which cuts off the user enjoyment and immersive experience

Over the Internet, it is impossible to avoid all networking problems and ensure a perfect quality of service. This fact motivated several research progress on how to enhance the user experience when using an imperfect underlying network. The next paragraph discusses the current state of the art in this challenging research area.

2.3 Enhancing the quality of experience for low bit-rate real-time audio and video conferencing

Several research avenues have been explored to enhance the user experience and perceived quality of audio and video communications. These works can be summarized as the following three main domains:

Transmission rate adaptation: research in this area studies how to efficiently use the available bandwidth to broadcast a multimedia stream to several recipients.

Error tolerance: the aim here is to develop correction schemes to make transmission errors less noticeable by the user.

Compression: a good compression scheme reduces the network load while keeping reasonable content quality. Findings in this area help to reduce the effect of low bandwidth on the user experience.

2.3.1 Transmission rate adaptation

Research in this domain aims to optimally adapt the multimedia stream rate to user bandwidth capacity in order to provide the best QoS possible (McCanne et al. 1996). The main challenge is to accommodate heterogeneous environments where several users have different bandwidths (Gill et al. 2008).

2.3.1.1 Source based rate adaptation

This approach is the simplest solution to the problem. A uniform representation of the signal is sent to all interested receivers using IP multicast (Deering & Cheriton 1990). So the sender or source broadcasts at a fixed rate without regard to changing network conditions. The source based rate adaptation is very simple to implement and supposes that receivers have almost the same bandwidth, which does not change a lot over time. In realistic conditions though, the bandwidth is not stable and receivers have different amounts of bandwidth. So, the current method performs poorly in heterogeneous environments since low capacity regions of the network suffer congestion while high capacity regions are underutilized (McCanne et al. 1996).

2.3.1.2 Receiver based rate adaptation

The Internet's heterogeneity makes multi-point communication design a difficult problem. Receiver based rate adaptation struggles to solve this problem. The objective is to broadcast a live signal from any particular sender to an arbitrarily large set of receivers along paths with potentially high variability in bandwidth (McCanne et al. 1996). To achieve this, the transmission rate is adjusted to match the available capacity in the networks.

The general principle is to broadcast several flows and each receiver, depending on its actual available bandwidth, will subscribe to one or more streams. Hence, it is up to the receiver to adapt the flow rate in this approach (Amir et al. 1997).

There are several implementations for the receiver based rate adaptation, which can be grouped into two main models: simulcast and multilayer.

2.3.1.2.1 The simulcast model

In this model, the sender transmits multiple copies of the same signal simultaneously at different rates (resulting in different qualities) (Gill et al. 2008). Depending on its available bandwidth, the receiver subscribes to only one flow, which is the optimal quality for the actual capacity. Should the network condition change, the receiver can adapt by subscribing to another "flow channel" to avoid either congestion or underutilization of the bandwidth. This approach requires a good bandwidth at the sender side to supply parallel streams.

2.3.1.2.2 The multilayer model

The multilayer model encodes the stream into a number of layers that can be incrementally combined to provide progressive refinement (McCanne et al. 1996).

The receivers can then connect to more than one layer at the same time. Each upper layer to which the user connects provides further content refinement. Figure 2.2 illustrates a situation where three receivers with different amounts of bandwidth (R1: 512 Kbs/s, R2: 256 Kbs/s and R3: 128 Kbs/s) connect to a sender S with 512 Kbs/s bandwidth.

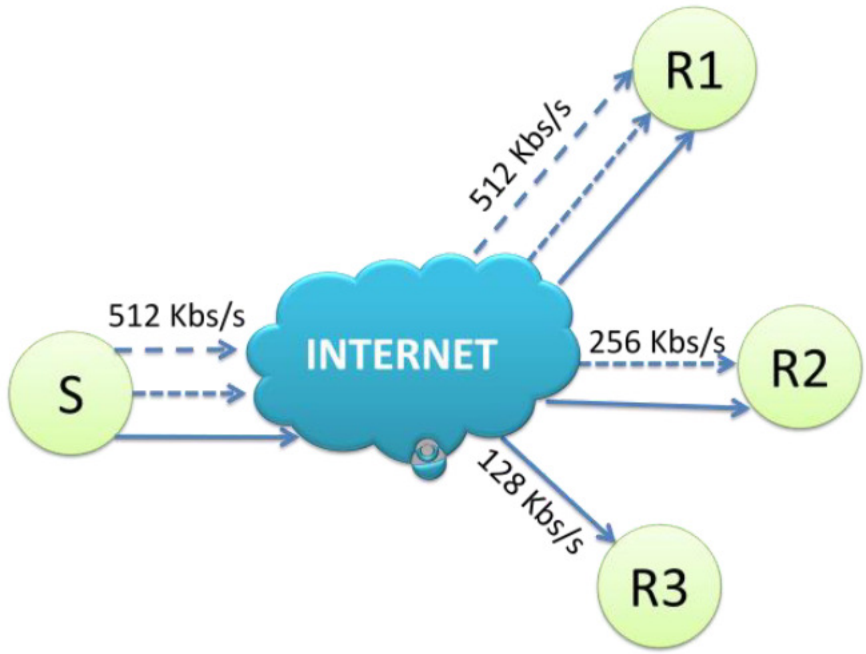


Figure 2.2: Rate adaptation based on multilayer model

In the implementation of this model, the receiver connects initially to the first or base layer. If reception is good (no congestion, small error rate, etc.), it adds the second upper layer, which adds a certain refinement to the stream and the reception quality is analyzed again. This process is repeated to eventually reach the best tradeoff between bandwidth and quality (McCanne et al. 1996). In contrast, the bandwidth decreases while receiving, the user can simply successively drop the upper layers to adapt the stream to the actual capacity.

Another important advantage of this model is the ability to associate a priority with each layer. The basic layer can get the highest priority and upper layers lower priorities. So when the bandwidth decreases, the routers can drop the upper layer packets that carry refinement information first. This technique leads to a graceful degradation of the stream over low and unstable bandwidth, substantially improving the quality of the user experience.

2.3.2 Error tolerance

The Internet often suffers from packet loss and random delay (jitter), degrading the user's perceived quality of multimedia stream (audio and video) (Foo et al. 1999). The following sections discuss some approaches used to enhance the quality of user experience under usual Internet transmission problems.

2.3.2.1 Stream buffering

In this approach, transmitted frames are buffered in memory by the receiver, allowing each frame to be played out with a constant latency, this achieving a steadier stream (Guan-Ming 2005). The added latency can badly affect interactive communication. So the buffer size should be chosen to provide the best perceptual quality, taking into account the tradeoff between decreased jitter and increased latency.

2.3.2.2 *Error tolerance and correction*

This approach privileges real time traffic by minimizing the delay as much as possible. Some error correction methods insert redundant information into the packet for later recovery. But this approach augments the bandwidth usage. A second approach is to retransmit erroneous packets. Under high error rate conditions, this approach worsens the jitter problem and generates network overhead. A third approach is to generate small packets, so when an error or loss occurs, the packet is merely ignored (replacement with a silent packet for audio stream, for example). Under low error rates, the small ignored packets will not be noticed by the user. But above a certain error threshold (depending on the nature of the stream), this approach is almost useless (Claypool & Tanner 1999).

2.3.3 Compression

In online meeting context, the video stream consumes more bandwidth then audio. The compression aims then to reduce the bandwidth needed while keeping an acceptable quality for the user experience.

2.3.3.1 *The Discrete Cosine Transform (DCT)*

The Discrete Cosine Transform algorithm is used in most videoconferencing systems. A modern DCT compressor requires roughly 100 Kbps for a 320*240*15 fps video of a person's upper body (M. Chen 2002). Below this rate, the video image motion may appear jerky (emergence of blocking artifacts).

To cope with very low bandwidth, two alternative approaches have been developed. In the first approach, only the outlines of images are encoded. Experimentation has shown that people can actually recognize the identity and facial expression of a person by the outlines of facial features (Xia et al. 2011). An implementation of this idea can deliver usable video at less than 10 Kbps. In the second approach, some key facial features are encoded in order to animate a 3D model of the person's head.

2.3.3.2 *Frame rate minimization*

The bandwidth requirement can be lowered by decreasing video frame rate (which is the number of images displayed per second). Experiments have demonstrated that a frame rate as low as one update every five minutes is enough to provide environmental awareness. But studies on user behavior suggest that 5 frames per second (FPS) is the acceptable lower bound for a direct human to human interaction (Ou et al. 2008).

An enhancement of this approach is to use a dynamic frame rate (Xin & Lin 2005). The idea is to detect user gestures and increase the frame rate only when movements are spotted. Otherwise keep a very low frame rate.

2.4 Comparison and discussion of the reviewed literature

Current research results (in multiple domains) can contribute to improve the quality of user experience when using poor Internet connections for Electronic Meeting Systems. This section compares and discusses some of the main findings and their possible application to the Web meeting context.

Research on rate adaptation aims to provide the best streaming broadcast quality adapted to the available bandwidth. The source-based approach uses the available bandwidth poorly. On the other hand, the multilayer model approach offers the best rate adaptation since it allows the definition of different QoS on layers (leading to graceful signal degradation).

Error correction techniques based on retransmission and redundancy are simply not applicable in Web meeting contexts, as they increase the network overhead. The buffering approach is a better solution.

Since meeting exchanges are not as interactive as a phone call, for example, the small delay introduced by the buffering will almost not be perceived by other participants, as long as the delivered stream is smooth.

There is good progress on video compression too, but the bandwidth required (100 Kbps) to maintain acceptable quality can still be too much (for African contexts especially). It is better to combine a compression scheme with frame rate minimization to deliver a usable video stream in very low bandwidth conditions.

CHAPTER 3 – SYSTEM DESIGN AND IMPLEMENTATION

3.1 Introduction

The main research aim of this project is to study the feasibility of a usable audio-video conferencing system with very constraining Internet conditions typical to Africa and the developing world. This chapter discusses design choices and their corresponding implementation to meet the above objective. The final experimental prototype should deliver:

- an audio stream conveying perceptible speech using the smallest bandwidth possible; and
- a video stream that provides a good sense of presence and improves the user experience.

3.2 General architecture

The system is designed using a Client-Server architecture. This model involves a server component providing services to one or many clients. The clients initiate the communication by requesting server services. For the audio-video communication sub-system, each component plays the following role:

- Clients: record stream (audio or/and video), compress stream, send packets to the server, receive packets from the server, decompress and play back the streams; and
- Server: receives and buffers packets coming from a specific client; and broadcasts packets to the rest of clients.

During a meeting, only one participant is allowed to talk at a time while the others are listening. So the communication with the server, as presented in Figure 3.1 is half-duplex. This approach improves bandwidth usage by a factor of two.

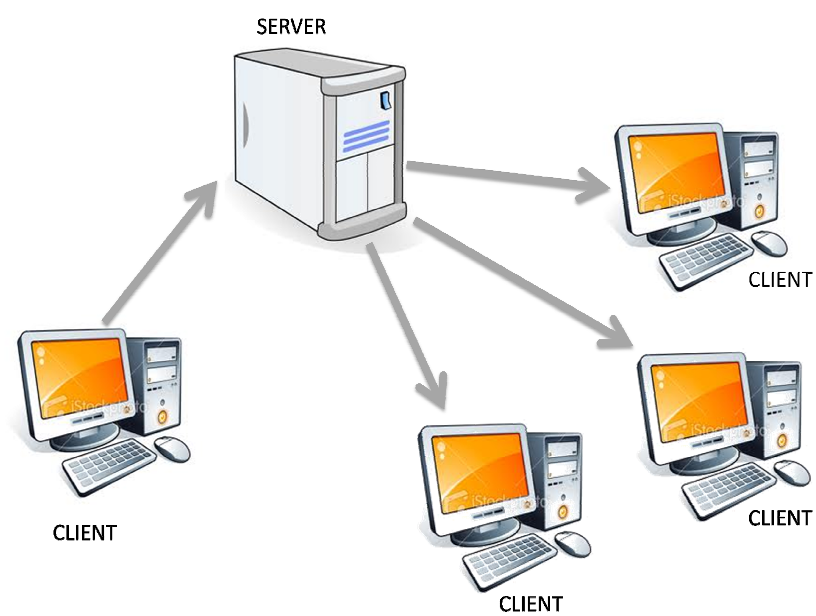


Figure 3.1: Half-Duplex client-server communication

3.2.1 Technology choices

This project is implemented with Java Technologies. The term "Java" refers to a collection of products and specification defined by Sun Microsystems, a branch of Oracle Corporation. These products include the Java programming language, developing application software and enterprise server applications (Anon 2011b).

The choice of Java Technologies over its major competitor (that is, Microsoft .Net) is motivated by the ability to develop a solution that can run on virtually any other platform. Sun made Java free and open source under the General Public License (GPL) since October 2006 (Anon 2011b). Java is a proven and mature technology that has good support from a very large developer community.

3.2.2 Client-Server implementation

The end user uses the client to interact with the Web meeting system. The scope of this project is limited to PCs; the possibility to expand the system to mobile devices is discussed on the future work chapter.

The end user interface can be based on either a Web page interface or desktop Windows-based graphical interfaces. Web pages are very simple to deploy: once installed on the Web server they can easily be accessed by any Web browser (which are preinstalled on virtually all modern operating systems). This is partly possible because Web pages can rely on largely accepted Web standards; like HTML, CSS, JavaScript, etc. For this project, the client should be able to access hardware resources including the webcam and microphones. The current Web standards do not allow native access to these resources, making it compulsory to use third party non-standard plugins. The literature survey and first prototype revealed that the current third party technologies do not give access to low level interfaces needed for the project implementation. These facts motivated the choice of a desktop-based Graphical User Interface (GUI) instead of Web interfaces. The GUI is developed using Java Swing, that is the standard GUI and widget library for Java.

All the communication with the server is built on top of TCP/IP. The approach in this project is to privilege quality over the real-time aspect. It is assumed that during a meeting, hearing a clear and smooth sound with a certain delay is better than getting an unintelligible real-time audio stream. This assumption motivates the choice of the Transport Control Protocol (TCP) that provides reliable and ordered delivery of data packets.

At a higher level, all communications happens on top of HTTP protocol. This choice facilitates the system to work behind a firewall or proxy without any network reconfiguration. The server runs a Web server service to handle and respond to all client requests. Processing at the server side is implemented with Java servlet.

A servlet is a compiled Java class, managed by a container and running at the server side. The container manages all the interactions between the servlet and Web clients, using a request-response model based on the Hypertext Transfer Protocol (HTTP)(Davidson & Coward 1999). The servlet can receive requests and data from the client and generate dynamic responses. As illustrated in Figure 3.2, a servlet for this project receives continuous real time multimedia stream from clients and forwards the stream toward destinations.

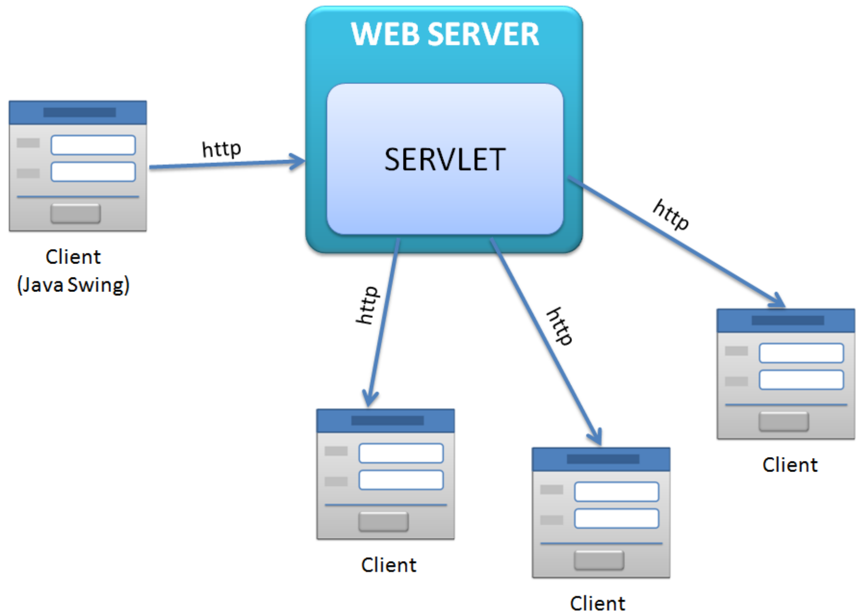


Figure 3.2: Global system architecture: communication between Client and Servlet via HTTP

3.3 Design and implementation of the audio sub-system

The objective is to deliver an audio stream clear enough conveying an intelligible speech, when using the smallest bandwidth possible. The audio communication will go through six main steps: recording, compression, transmission, decompression, buffering and playback. The next paragraphs discuss the design of these steps.

3.3.1 Sound recording and formatting

The sound is a continuous mechanical wave that travels through a solid, liquid or gas medium. A sound wave is actually differences of pressure travelling through a medium. Therefore, one way to detect the sound is by measuring the air pressure at a specific location (Marshal 2001). Figure 3.3 illustrates a sound wave with some of its main characteristics.

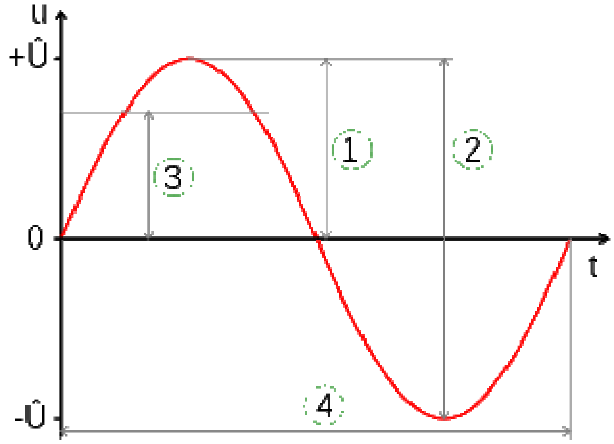


Figure 3.3: Sound wave characteristics: 1 – Peak amplitude, 2 – Peak-to-peak amplitude, 3 – Root Mean Square (RMS) amplitude, 4 – Wave period (Anon 2011b)

A sound can be recorded using a microphone, which transforms the physical waveforms in the air into an electrical signal. For this signal to be processed by and transmitted between computers, it needs to be digitized by converting it into a stream of numbers (Marshal 2001). This process is accomplished in 2 steps:

- Sampling: consists of dividing time axis (horizontal axis) into a number of discrete blocks called samples; and
- Quantization: consists of dividing the vertical axis that represents the signal strength into several discrete levels. For example, an 8 bit quantization will produce 256 different levels. The sound amplitude at each sample will be associated with the nearest discrete value.

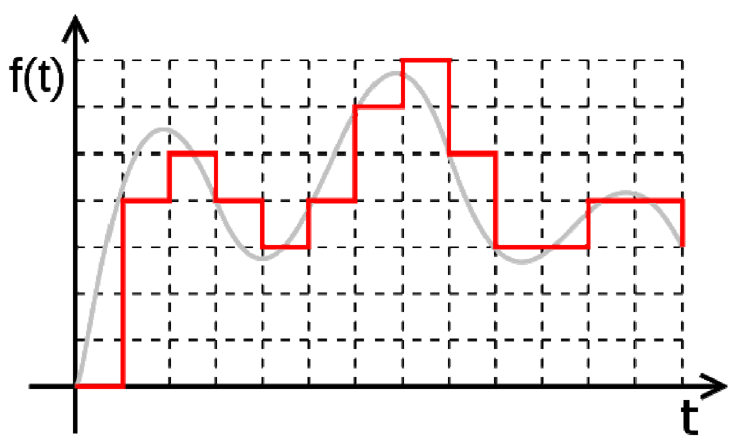


Figure 3.4: Sound digitization

The above figure illustrates an analog sound wave (in gray) and its corresponding digital signal (in red). A digital signal is then made up of a stream of numbers, with each value representing the discrete amplitude recorded for each sample. This stream can be recorded into a file, transmitted via a network or played back through speakers. The number of samples and quantization of each sample directly determine how close a numerical signal is to its analog equivalent.

In this project, the digitization of voice will use the following parameters to provide clear sound capture:

- Sampling rate: 8000 samples per second
- Sample size: 8 bits = 256 levels
- Number of sound channels: 1

3.3.2 Audio compression and streaming

The sound bit rate is the amount of data per unit of time needed to store or transmit a digitized sound stream. It can be calculated using the following formula:

Bit rate (in bits per second) = Number of channels X Sampling rate X Sample size

Based on the parameters selected for audio recording, the Bit rate will be:

Bit rate = 1 X 8000 X 8 = 64 000 bits/sec = 64 Kbps

This means a bandwidth of at least 64 Kbps will be needed for transmission of the uncompressed audio stream; which is too much for a low bandwidth context. The solution to this problem is stream compression, which reduces the quantity of data needed. Different algorithms and formats have been proposed for audio compression, including: A-Law, M-Law, MP3 and Groupe Speciale Mobile (GSM) (Foo et al. 1999). During prototyping, these formats were tested:

- A-Law and M-Law required recording the sound at 16 Khz. The uncompressed sound stream needs 128 kbps. After compression, the stream is delivered at 64 kbps which is not enough for the context of limited bandwidth.
- MP3 offers different level of compression. But with 32 kbps, the sound quality degrades substantially and the speech is hardly intelligible.
- GSM offered a constant compressed stream of 13 kbps; but as for MP3, the speech is hardly intelligible.

Therefore, for this project, every sound packet is compressed using the ZIP format. This format can compress without data loss. The resulting sound is then clear, using 44% of the uncompressed stream size.

Figure 3.5 summarizes the main steps from audio recording to streaming with selected parameters for the project.

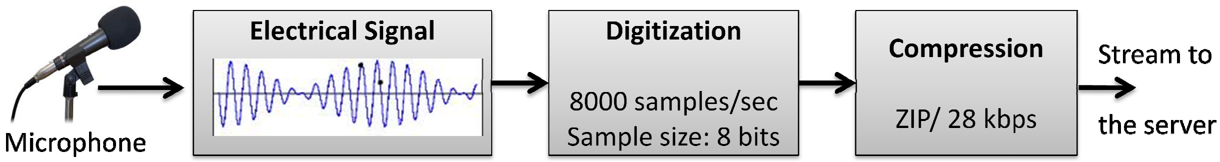


Figure 3.5: Sound recording, compression and streaming

Sound recording is done with Java Sound, the standard Java API for manipulation of input and output audio streams. It offers high level interfaces for sound capture, playback and some special effects.

3.3.3 Stream reception, decompression, buffering and playback

The audio packets received at the destination are decompressed and buffered into a queue collection, which is an implementation of a first-in first-out stack. The queue size is dynamic and grows or decreases when needed. Audio playback is performed using Java Sound. The packets from the queue form a stream used as input for the sound API. Figure 3.6 illustrates the whole process, from reception to playback.

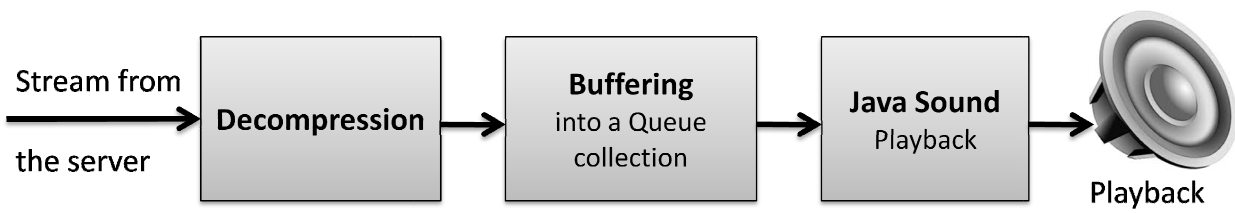


Figure 3.6: Sound reception and playback

Since communication in a meeting is not as interactive as a telephone call, the size of the buffer can be increased to avoid the delay coming from the IP Internetwork and end systems (Anon 2006). This approach privileges fluent play back over the real time aspect.

3.4 Design and implementation of the video sub-system

3.4.1 Video recording

The video is recorded using a webcam or any digital video recorder (any camera connected to the computer, for example). The webcam provides a raw and uncompressed live video stream, which will be used as a data source for the compression module.

Two API are used to record the video:

- Java Media Framework (JMF): is the standard Java library developed by Sun that offer audio, video and other time-based media capability to Java application. The JMF API features media capture, play, stream and transcode. Unfortunately, this project does not receive enough support from Sun; the last update was done in 1999. This drawback results in important limitations. The recent Linux webcam drivers are not supported, for example; forcing the use of a different library for Linux projects.

- Video 4 Linux 4 Java (V4L4J): is a library developed by Google that provides simple access to video capture devices using the interface V4L (Video for Linux). V4L4J features stream recording, formatting and streaming; but it can work only on the Linux platform.

Images recorded using either JMF or V4L4J go through the differentiation process before being compressed with JPEG. The rate at which images are recorded is dynamic: either automatically evaluated or set by the user. Figure 3.7 summarizes the video recording process.

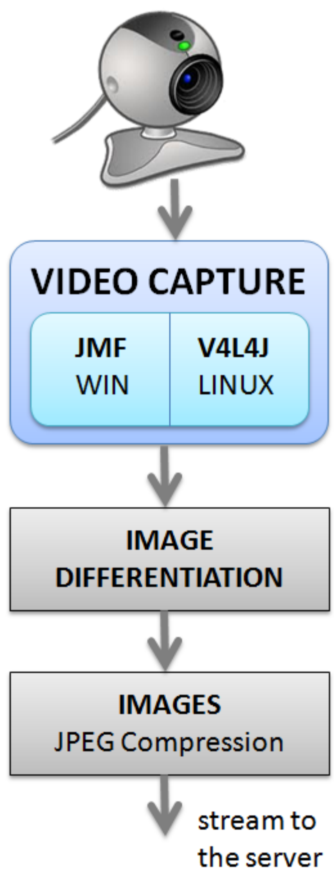


Figure 3.7: Video recording, compression and streaming

3.4.2 Video compression

The raw video stream from the webcam is not compressed and natively has a high bit rate (around 18 Mbps for a 320X240 video at 15 frames per sec). Such a bit rate can obviously not be afforded in the video conference context.

Several video compressions algorithms had been proposed and are currently used. H263 is among the most used in video conferencing, especially for low bandwidth. A test of H263 with a 320X240 video at the lowest frame rate supported (5 FPS) gives a video stream with around 64 kbps.

For this project, to have full control over the compression and stream process and achieve a greater compression, each image is recorded individually. This technique allows delivering of even very low frame rates (far below 1 frame per sec); this could help to convey at least a certain sense of presence when the bandwidth available is very limited. Each individual image then goes through the differentiation process described in the next paragraph, before being compressed in the JPEG format.

3.4.2.1 Image differentiation algorithm

In a meeting context, images coming from the camera are very similar to one another. The speaker often stands still and does not move a lot during the meeting. This fact can be exploited to achieve better compression. Instead of transmitting a different image each time; it is better to calculate, compress and transmit only the differences between two consecutive images. The destination reconstitutes the original image by aggregating differences. This process eventually degrades the image quality. To avoid complete degradation, a key frame is sent after a certain number of iterations, as illustrated in Figure 3.8.

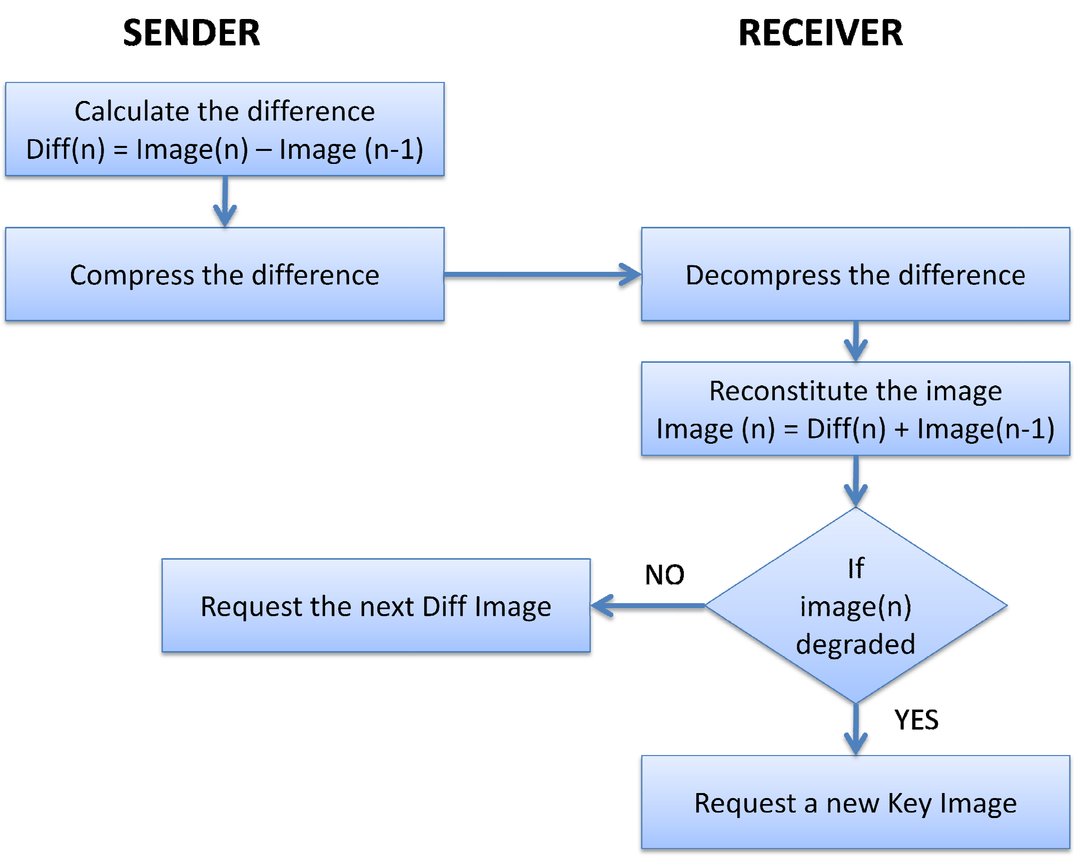


Figure 3.8: Image differentiation algorithm

A picture in the computer memory is modeled by a two dimensional matrix of samples or pixels. Each pixel can be a single value or a vector modeling the color and brightness. For a colored picture, each pixel can be modeled using a vector of 3 values representing 3 primary colors: Red, Green and Blue.

Each color component is encoded with 8 bits with values ranging from 0 to 255. Figure 3.9 represents the pixel model, using 24 bits in memory.

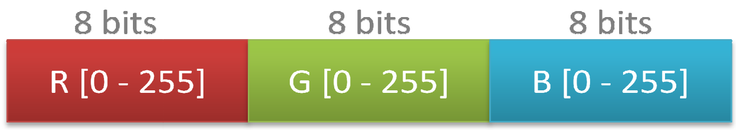


Figure 3.9: Pixel representation in memory

Calculating the difference between 2 images amounts to evaluate the difference between their corresponding matrices. The difference between two matrices A and B is evaluated by the below formula:

$$\boxed{\text{Diff AB } (i , j) = \text{Mat A } (i , j) - \text{Mat B } (i , j)}$$

Where (i,j) represent the matrix element at the i_{th} line and j_{th} column.

3.4.2.1.1 Out of range color value problem

When calculating the difference, the result ranges from -255 to 255 while the valid color value ranges are from 0 to 255. To avoid the invalid color range (-255 to -1), the difference is calculated using the following formula:

$$\boxed{\text{Diff} = 127 - (\text{Color A} - \text{Color B}) / 2}$$

With the above formula, the resulting difference ranges from X to Y, where:

$$X = 127 - (-255) / 2 = 127 + 127{,}5 = 254{,}5$$

$$Y = 127 - (255) / 2 = 127 - 127{,}5 = -0{,}5$$

To avoid the decimals, X and Y are rounded giving a final range of 0 to 255. That is a valid color range.

Note: when dividing the difference between color A and B by 2, there is small loss in precision. But the loss is almost not noticeable to a human eye.

3.4.2.1.2 Image reconstitution

The difference is calculated by:

$$\boxed{\text{Diff} = 127 - (\text{Col}_n - \text{Col}_{n-1}) / 2}$$

Col_n represents the color from the current image. This value exists only on the client capturing the image.

Col_{n-1} is the color from the preceding image. This value exists on both the sender and receiver sides.

Only the value of Diff will be sent across the network; Col_n can be evaluated knowing Col_{n-1} and Diff, calculated as follows:

$$\text{Diff} = 127 - (\mathbf{Col_n} - Col_{n-1}) / 2$$

$$127 - (\mathbf{Col_n} - Col_{n-1}) / 2 = \text{Diff}$$

$$- (\mathbf{Col_n} - Col_{n-1}) / 2 = \text{Diff} - 127$$

$$(\mathbf{Col_n} - Col_{n-1}) / 2 = 127 - \text{Diff}$$

$$\mathbf{Col_n} - Col_{n-1} = 2\,(127 - \text{Diff})$$

$$\mathbf{Col_n} = Col_{n-1} + 2\,(127 - \text{Diff})$$

3.4.3 Video reception and playback

Images coming from the server can either be key-frames or intermediate frames resulting from image differentiation. After decompression, the playback of key frames is straightforward as they are directly displayed on the screen. Intermediate images require the calculation of the original image, using the formula in section 3.4.2.1. Figure 3.10 illustrates the video reception and playback process.

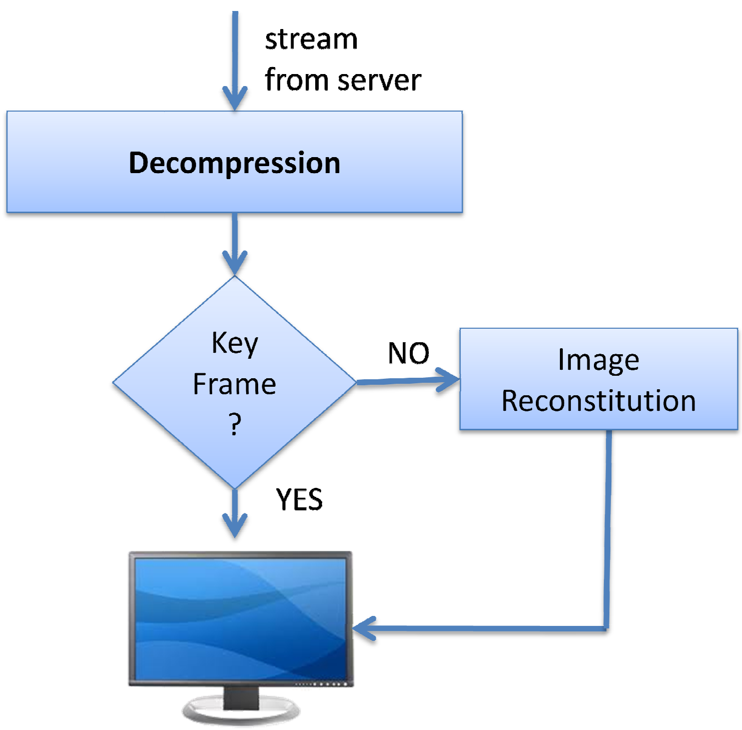


Figure 3.10: Video reception and playback

3.5 Implementation of the server application

The main function of the server is to receive packets from each client and buffer them in memory before sending to the rest of clients. All these features are implemented within a servlet, a Java class compiled and running at the server side.

Each request from a client triggers the creation of a servlet thread responsible for generating a response. In consequence, the servlet thread responsible of collecting packets from a client is different from the one forwarding data to destination. In order for different servlet threads to communicate, a common and shared memory is needed. Java Servlet specification offers two Interfaces: setAttrute(String key, Object value) and getAttribute(String key), allowing writing and reading of Java Objects into a shared memory.

The server handles communications from several users participating in different audio-video meetings. To manage everything without confusion, the shared memory is fragmented as described in Figure 3.11.

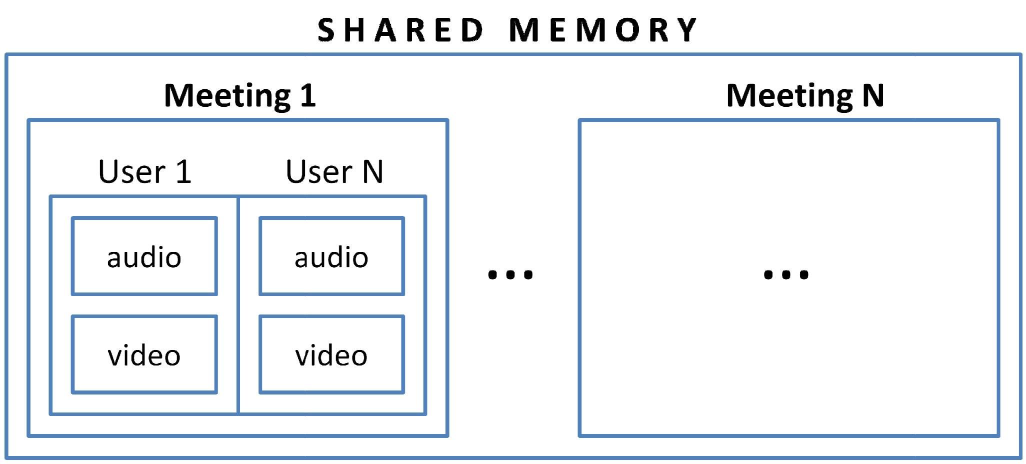


Figure 3.11: Shared memory fragmentation

This hierarchy is implemented using a composite key value for the interface "setAttribute". The key is defined as this:

Key = meeting + user + media type

3.5.1 Communication with the server

All the communication with the server is built on top of HTTP. The server runs a servlet responsible for receiving and responding to client requests. The Java API URLConnection is used to connect and communicate with the server. This library can initiate new connections and requests to the server, send data and collect responses.

URLConnection can also take advantage of long-lived connections and persistent connections, two features introduced from HTTP version 1. Long-lived connections allow opening and keeping a connection alive for a long time. Persistent connection allows sending several requests without initiating the entire TCP connection process each time, which is very resource intensive.

3.5.1.1 Send data to the server

Both audio and video streams are stored in memory buffers in the form of byte arrays. Audio data is sent to the server at the constant pace of one packet per second; but the rate for video streaming is dynamic, depending on the current frame rate.

The byte arrays are sent to the server as binary data using the HTTP POST method. Since the server handles different packets from several users and meetings, each packet is sent with 3 parameters: the user name, the meeting id and type of stream (either audio or video). The transmission of these parameters is based on the HTTP GET method. This consists of appending the parameter names and values to the URL.

3.5.1.2 Receive data from the server

URLConnection offers an interface for reading the response from the server. The response body is natively accessed via a binary Java Stream. To facilitate high level communications, the binary stream is enclosed into a Java object stream in order to transmit and receive Java objects. Both audio and video packets are decompressed before playback.

3.6 Bandwidth control and mitigation techniques

Many networking problems are susceptible to perturbation in packet delivery. These problems may include, for example: connection interruptions, delays and drastic decrease of the available bandwidth. Detecting packet delivery problems and evaluating their level can help to trigger appropriate counter-actions and adapt streams to the actual network conditions.

3.6.1 Design of bandwidth control system and congestion detection

Evaluating the actual bandwidth available between two nodes on the Internet is a complex task (Brakmo & O'Malley 1994). The bandwidth is often evaluated by uploading or/and downloading a file with a specific size and determining the time it takes. But in the context of low bandwidth, this approach is inefficient as it consume the limited network resource. In addition, the bandwidth may vary a lot, and then repeated tests based on data exchange will generate an overhead.

The bandwidth and congestion control proposed for this project is based on monitoring sound data packet delivery. The sound stream is delivered at a constant pace of one packet per second. The receiver records the time when each packet is received. In the ideal network, the time span between two deliveries should constantly be one second. When this time increases far above the second, it means that there is a problem preventing good packet delivery; network congestion can then be suspected.

There is no perfect network and even within a Local Area Network (LAN), a small variation in delay is normal. To make the congestion detection system more realistic, the last ten time values are recorded and the average delay is evaluated as shown by the following formula.

$$Delay = \frac{1}{10}\sum_{i=1}^{10}(Time_i - Time_{i-1})$$

3.6.2 Congestion avoidance techniques

In the perfect setting, the delay should be one second. But as stated, some variations may be normal. Table 3.1 defines 3 levels of congestion associated with the corresponding action in the system.

Table 3.1: Levels of congestion and corresponding action in system

Level	Threshold	Action
LEVEL 1: No congestion	Delay < 2 seconds	Nothing
LEVEL 2: Small congestion	2 sec < Delay < 5 sec	Decrease the video frame rate by a factor of 5
LEVEL 3: High congestion	Delay ≥ 5 Sec	Shut down the video stream

3.7 Development cycles

The project development went through 3 iterations of design, implementation and evaluation.

3.7.1 First implementation iteration

The aim of this first iteration was to assess the project feasibility; by testing possible underlying technologies. The implementation result in the first prototype featured: audio and video recording, stream transmission based on Java sockets and playback. The screenshot in Figure 3.12 shows the first prototype running; at left is the live video and right is the video playback after streaming.

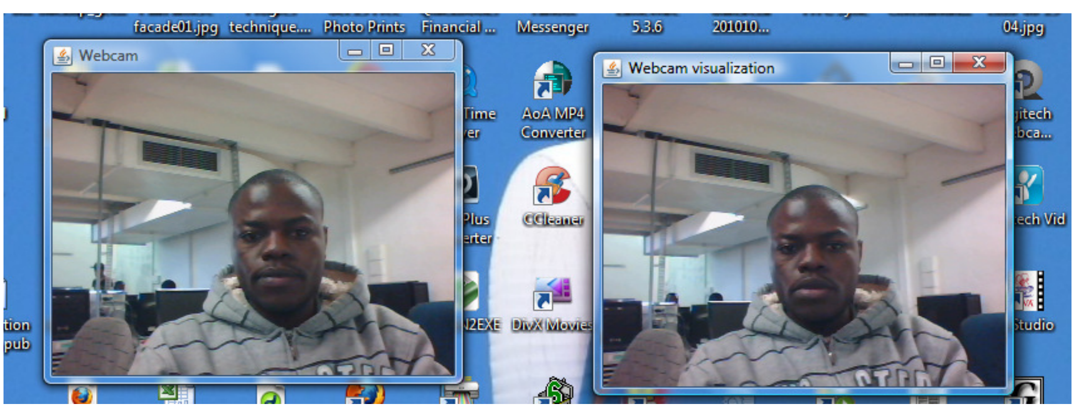


Figure 3.12: Screenshot of the first prototype

The conclusion from the first prototype was that the needed underlying technology is available and works, so the project is feasible.

3.7.2 Second implementation iteration

The second iteration is an audio-video conferencing system featuring:

- Audio/video communication;
- Audio buffering, with possibility to pause the live stream for later playback with delay;
- User and meeting management;
- Bandwidth control and mitigation action (on Figure 3.13, the green circle on right windows means there is no congestion detected, otherwise it would have been red);
- Possibility to shut down audio, video or both at the sender and receiver side;
- Possibility to set the video frame rate the sender and receiver side;
- A setting panel;

The screen shot on Figure 3.13 illustrates the second prototype running.

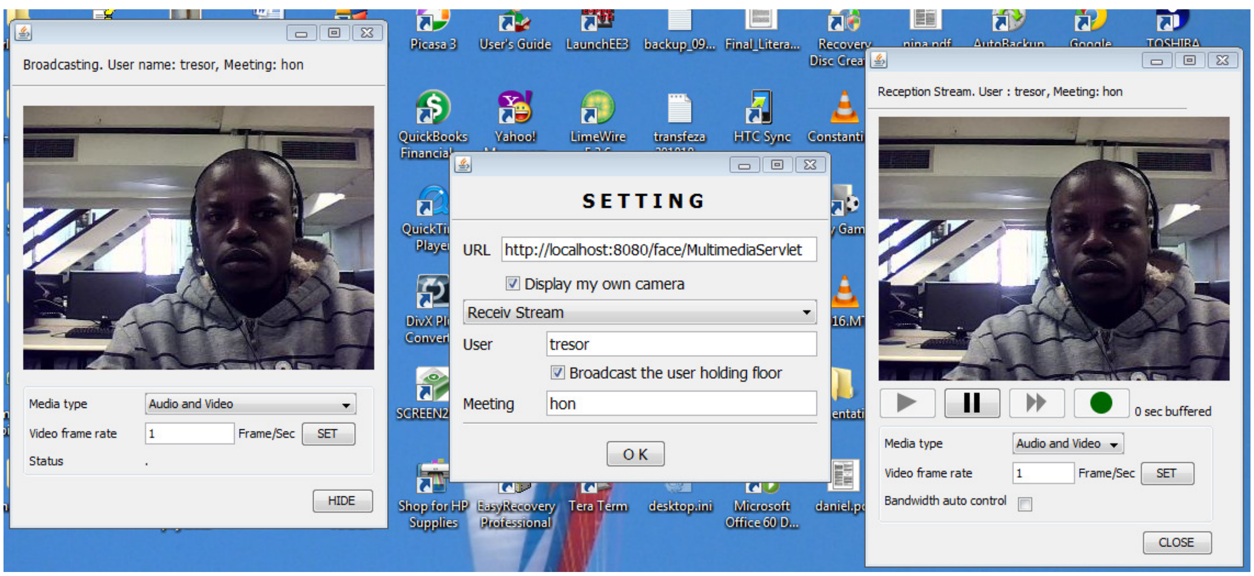


Figure 3.13: Screenshot of the second prototype

Testing and user evaluation of the second prototype lead to the following remarks and insights that were implemented in the final prototype:

- Trigger feedback action (confirmation alert box, for example) to notify the user when a configuration is changed;
- The congestion control should be activated by default;
- Compress the audio (64 Kbps is still too much for a low bandwidth context);
- The receiver cannot display more than 1 frame per second, optimize video play back; and,
- The quality of images played back is too high-increase image compression;

3.7.3 Last implementation iteration

The last prototype is an optimization of the second one, will doing all its features plus enhancements resulting from feedback during the second evaluation. The last prototype is presented in detail in the next section.

3.8 Implementation of the final prototype

The final prototype is a module providing audio and video communication among participants attending a virtual meeting. This module can be combined to work together with other components: screen sharing, presentation, text chat, floor control and participant list.

3.8.1 Settings panel

When launched, the prototype presents the settings screen as illustrated in Figure 3.14.

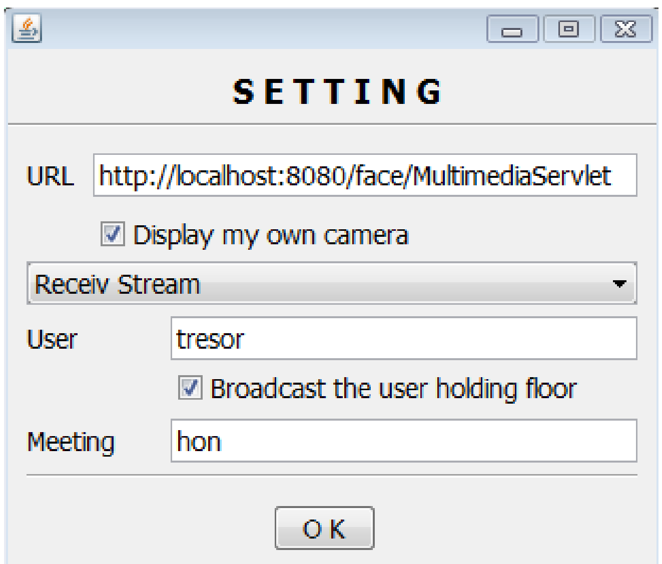


Figure 3.14: Screenshot of the setting panel

The settings screen allows the user to log in and join a meeting. The following parameters can be set:

- URL: is the address of the server side application, responsible for handling requests and responding.
- The check box “Display my own camera” allows the user to either display his own live camera or not.
- The list box contains two values: “Broadcast Stream” and “Receive Stream”
    - Broadcast stream: allows the user to start his camera and microphone, and stream them when he holds the floor.
    - Receive stream: allows the user to receive audio or/and video stream from either a specific user or the current user holding the floor (when the option “Broadcast the user holding the floor is checked”),

- The field User records the user name.
- The field Meeting records the meeting name.

After filling the settings form, the user clicks on the “OK” button, to either open a new window for streaming the camera and microphone or receive the stream from other participants.

3.8.2 Send stream to the server

Figure 3.15 shows the user interface controlling audio and video streaming with the server.

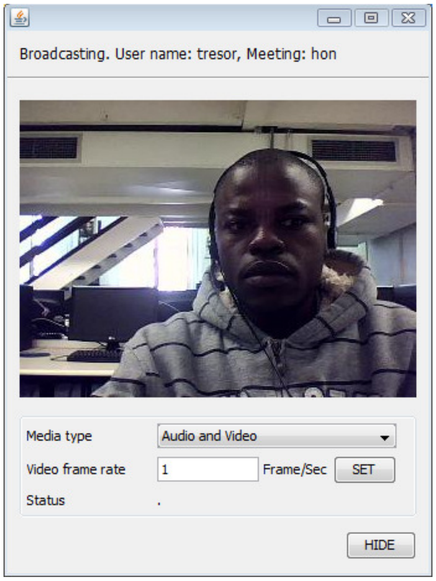


Figure 3.15: Screenshot of streaming control panel

When this form is opened, the application automatically detects the default microphone installed on the system. Should a headset is plugged, its microphone will be used for sound recording. The application also detects all video devices installed on the system. Should more than one video recording device be available; the user is presented with a list of all installed video devices to select one.

When both the microphone and video device are initialized and the user hold the floor, the system can start to stream audio or/and video to the server. The options available on the bottom of the interface allow users to:

- Select which media stream to send to the server; it can be audio and video, audio only, video only or even nothing;
- Se the video frame rate; and
- Hide the streaming control panel; it can be displayed back using the setting panel.

The user can send both audio and video stream to more than one server simultaneously. He can also participate in more than one meeting.

3.8.3 Receive the stream from the server

Figure 3.16 illustrates the interfaces responsible for receiving and managing multimedia streams from the server.

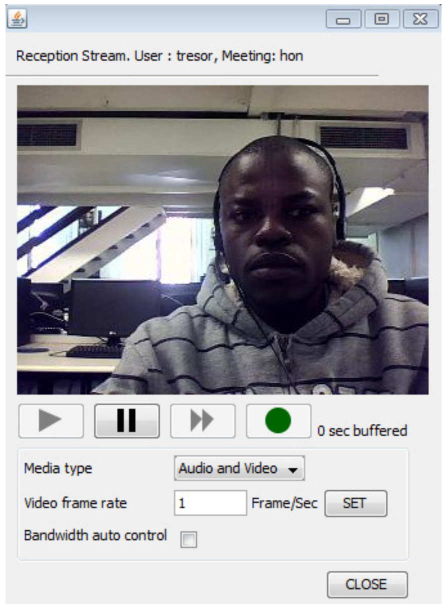


Figure 3.16: Screenshot: video stream reception and playback

As stated above, the user can choose to receive streams from either a specific user or the current user holding the floor. If the last option is selected, the reception module will switch the user automatically when the floor his handed over.

When a camera is available at the remote participant side, the images are displayed on the screen. Otherwise, a still picture representing the participant is displayed. The parameters on the bottom of the interface allow the user to:

- Control audio flow and playback:
    - Pause button: stops the playback and record the coming audio stream in a buffer;
    - Play button: restart the playback, with a delay corresponding to the precedent pause time. Playing from a buffer offers a smooth playback when the connection is unstable, by sacrificing the real time aspect; and
    - Fast forward: empty the buffer and catch up the live audio stream.
- Select the media type to receive and play. Can be both audio and video, audio only, video only or nothing.
- Set the video frame rate at destination. This option allows the user to manually adapt the video stream to the available bandwidth.
- Activate or deactivate congestion control: when congestion control is activated, the system takes automatically actions to reduce bandwidth usage when congestion is detected.

The settings panel allows displaying of more than one participant camera simultaneously.

3.9 Design summary

The system is based on client-server model. The client can either record and send audio-video stream or receive and play it back. The server acts as a central hub: it receives multimedia streams from one client and forwards to the rest of participants.

The sound is recorded at frequency of 8000 samples per second, with a sample size of 8 bits. The resulting audio bit rate is 64 kbps, which is compressed with the ZIP format into a 28 kbps audio stream. For video, each image is captured individually. Then the difference between two consecutive images is calculated, compressed with JPEG and sent across the network. A key frame is sent after a certain number of iterations, to cope with image degradation resulting from accumulation of differences.

Congestion detection is based on the average reception time of audio packets. When the average time between two audio packet receptions goes beyond 2 seconds, congestion is suspected and actions are taken to reduce bandwidth usage.

CHAPTER 4 – EVALUATION

4.1 Introduction

The research question for this sub-project is: “Is it possible to build an effective audio-video conferencing tool that works with low bandwidth conditions?”. To address this question, a prototype aiming to provide acceptable audio and video communication with low bandwidth was developed and evaluated. The first section presents a set of performance tests and results. The second section presents system requirement in terms of bandwidth usage. And the last section focuses on user evaluation.

4.2 Bandwidth usage evaluation and performance test

The objective of this test is to measure what are the actual bandwidth usages under diverse settings. The results will help to infer system requirement in term of bandwidth usage. To record data, any packet reception and transmission between clients and the server is logged into a text file. Figure 4.1 shows a log file sample.

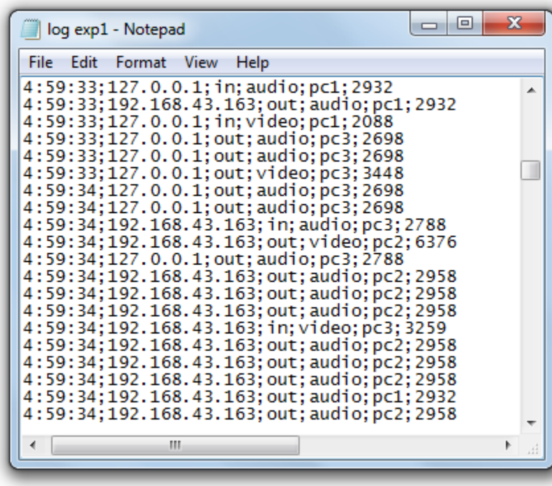
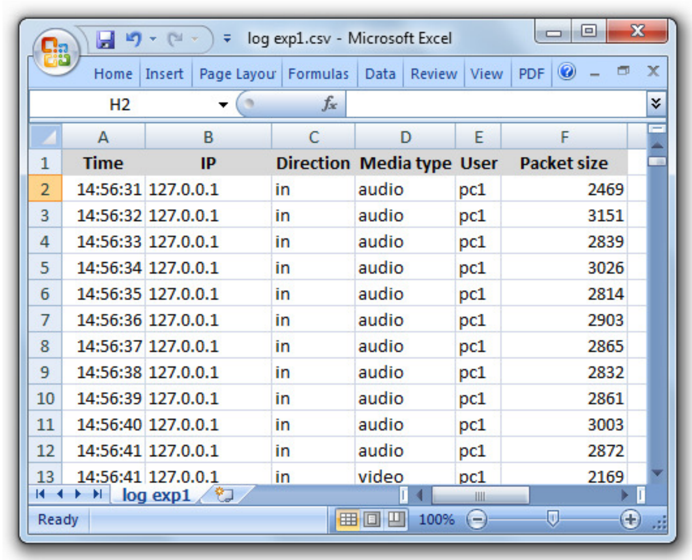


Figure 4.1: Log file structure

The log file is recorded from the server. It is a text file, with “.csv” extension. A “.csv” file can be easily opened and converted into Ms Excel format for further analysis. When a packet is received or sent by the server, a new line is recorded in the log file. The line records several information field separated by semicolons. The fields recorded are respectively: time, IP address of remote user, packet direction (in for packet received and out for packet sent), type of media (audio or video), user name and size of packet (in byte).

After each experiment, the generated log file is converted into an Excel file: a header is introduced and the semicolon helps to separate fields in different columns. Figure 4.2 shows a sample of the Excel file generated from a log file.



	Time	IP	Direction	Media type	User	Packet size
1	Time	IP	Direction	Media type	User	Packet size
2	14:56:31	127.0.0.1	in	audio	pc1	2469
3	14:56:32	127.0.0.1	in	audio	pc1	3151
4	14:56:33	127.0.0.1	in	audio	pc1	2839
5	14:56:34	127.0.0.1	in	audio	pc1	3026
6	14:56:35	127.0.0.1	in	audio	pc1	2814
7	14:56:36	127.0.0.1	in	audio	pc1	2903
8	14:56:37	127.0.0.1	in	audio	pc1	2865
9	14:56:38	127.0.0.1	in	audio	pc1	2832
10	14:56:39	127.0.0.1	in	audio	pc1	2861
11	14:56:40	127.0.0.1	in	audio	pc1	3003
12	14:56:41	127.0.0.1	in	audio	pc1	2872
13	14:56:41	127.0.0.1	in	video	pc1	2169

Figure 4.2: Excel data file generated from the log file

Microsoft Excel is used to perform calculation, analyses and generate graphs presented in the following sections. Unless stated otherwise, all bandwidth values are in kilobit per second (kbps).

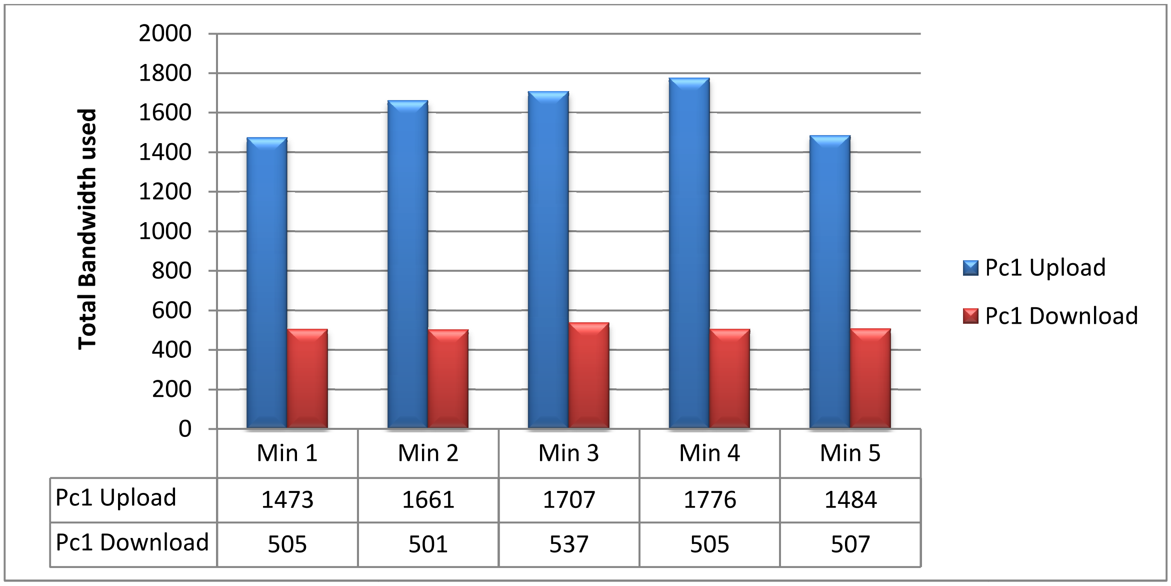
4.2.1 Audio sub-system bandwidth usage

4.2.1.1 Experiment 1: Audio meeting with 2 participants using full duplex mode

In full duplex mode, both participants simultaneously transmit and receive streams from each others. For experiment 1, 5 minutes of audio meeting activity was recorded. Table 4.1 and Graph 4.1 give the bandwidth used by the first client and server per minute. The download bandwidth at pc2 corresponds to the upload of pc1, and the upload on pc2 is equal to download of pc1. In consequence, the total bandwidth used by pc1 and pc2 are equal. Since the server is receiving and forwarding the stream from both clients, its bandwidth usage is twice the bandwidth of each client.

Table 4.1: Bandwidth used per minute (in kilo bit) for an audio meeting with 2 participants using full duplex communication mode

	Pc1 Upload	Pc1 Download	Total Pc1	Server
Min 1	1473	505	1978	3955
Min 2	1661	501	2163	4325
Min 3	1707	537	2244	4489
Min 4	1776	505	2281	4562
Min 5	1484	507	1991	3982



	Min 1	Min 2	Min 3	Min 4	Min 5
Pc1 Upload	1473	1661	1707	1776	1484
Pc1 Download	505	501	537	505	507

Graph 4.1: Client bandwidth usage per minute (in kilo bit) during an audio meeting with 2 participants communicating in full duplex mode

During the meeting, the user on pc2 was mainly listening and silent most of the time. The audio compression approach used achieves better compression in the case of silence, so the download bandwidth on pc1 (used to receive packets from pc2) is significantly lower. Graph 4.1 summarizes maximum, minimum and average bandwidth needed by the client and server during experiment 1.

Table 4.2: Maximum, minimum and average bandwidth usage (in kbps) during an audio meeting with 2 participants using full duplex mode

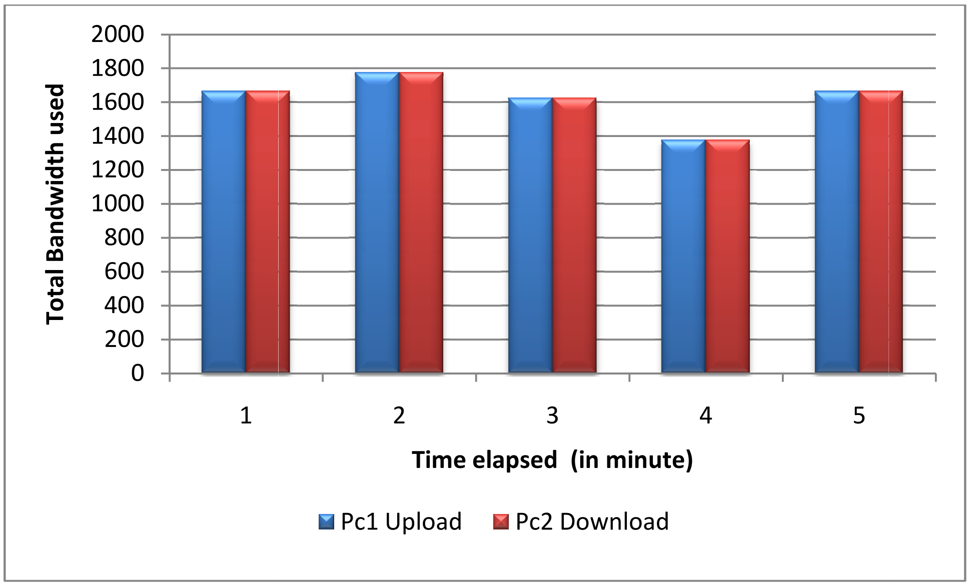
	Client	Server
Maximum	38	76
Minimum	33	66
Average	36	71

4.2.1.2 Experiment 2: Audio meeting with 2 participants using half duplex mode

In half duplex mode, each participant can only either transmit or receive streams. For experiment 2, 5 minutes of audio meeting activity is recorded. Table 4.3 and Graph 4.2 give the total amount of bandwidth used by each participant and server per minute.

Table 4.3: Bandwidth usage per minute for each user and the server (in kilo bits) during an audio meeting in half duplex between 2 participants

	Pc1 Upload	Pc2 Download	Server
Min 1	1663	1663	3327
Min 2	1776	1776	3552
Min 3	1625	1625	3249
Min 4	1376	1376	2753
Min 5	1664	1664	3327

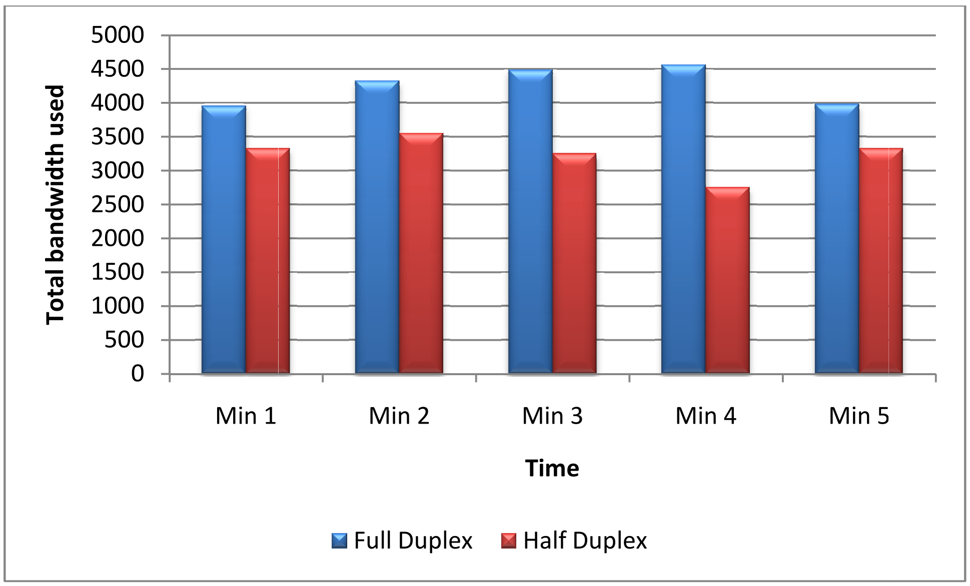


Graph 4.2: Bandwidth usage per minute (in kilo bit) during an audio meeting in half duplex mode between 2 participants

In half duplex mode, the user can either be sending or receiving the audio stream. The amount of data sent by Pc1 is exactly what is received by Pc2. This fact explains why the total bandwidth used by both clients is the same. The server requires exactly double the client bandwidth, to receive and forward the same packet size. Table 4.4 compares bandwidth usage between full and half duplex transmission mode.

Table 4.4: Server bandwidth usage: comparison between full and half duplex bandwidth used during an audio meeting with 2 participants

	Full Duplex	Half Duplex
Min 1	3955	3327
Min 2	4325	3552
Min 3	4489	3249
Min 4	4562	2753
Min 5	3982	3327
Total	**21313**	**16209**

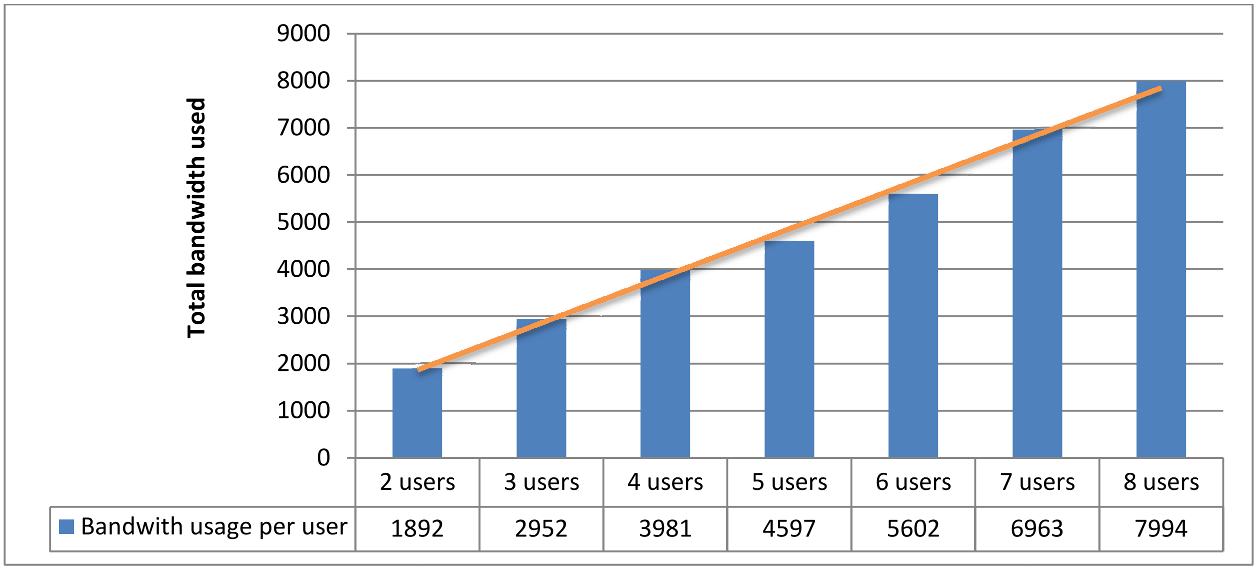


Graph 4.3: Server bandwidth usage: comparison between half and full duplex mode for an audio meeting with 2 participants

Even if the server in half duplex mode uses half of the channels compared to full duplex, the bandwidth required by half duplex is not half of full duplex. This observation is explained by the fact that, even in full duplex, both participants do not speak at the same time. Audio compression for the participant not talking is very efficient. Silent sound packets use around 8 kbps whereas normal speech needs 28 kbps. Finally, the total bandwidth used in half duplex during 5 min was 16,2 Mbits, representing a gain of around 24% compared to full duplex (21,3 Mbits).

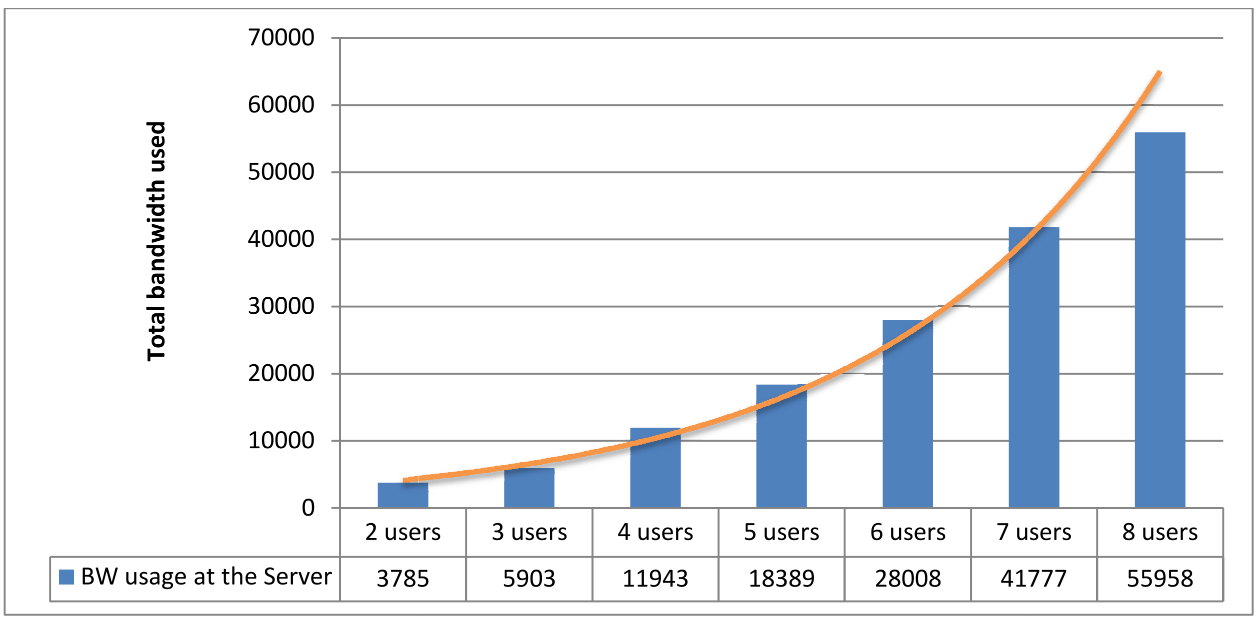
4.2.1.3 Experiment 3: Meeting with 8 participants using full duplex mode

In experiment 3, a meeting is started with 2 participants. After every minute, a new participant joined the meeting. The whole experience took 7 minutes; at the end, 8 participants were attending the audio meeting using full duplex mode. Graph 4.4 shows the average total amount of bandwidth used by each participant per minute.



Graph 4.4: Average total bandwidth used per user per minute (in kilo bits) during a meeting where a new participant joined in full duplex mode

Every minute, as the number of users increases, each participant needs to receive more data. The server needs to receive and forward streams from almost all the clients. Graph 4.5 gives the total amount of bandwidth needed by the server every minute as a new participant is joining the meeting.

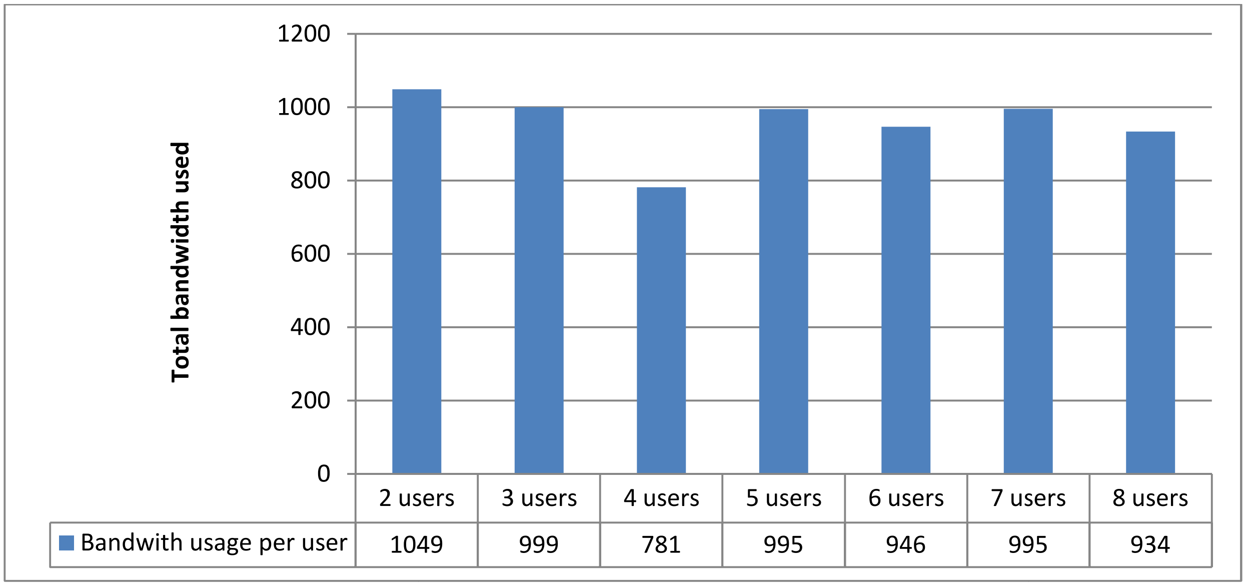


Graph 4.5: Total bandwidth used on the server per minute (kilo bits) in a meeting where a new participant joined every minute in full duplex

As illustrated on Graph 4.5, the bandwidth usage on the server is exponentially increasing for each new user joining the meeting.

4.2.1.4 Experiment 4: Meeting with 8 participants using half duplex mode

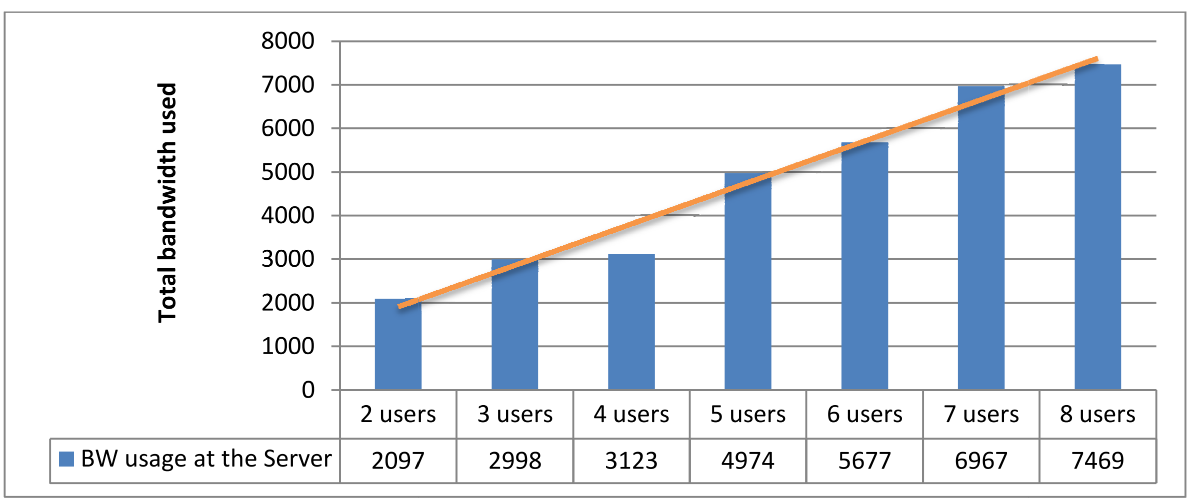
This experiment is exactly the same with the one in 4.2.1.3, except the fact that here half duplex mode is used. Graph 4.6 shows the average total amount of bandwidth used by each participant per minute.



Graph 4.6: Average total bandwidth used per user per minute (in kilo bits) for a meeting where a new participant joined every minute in half duplex mode

Each user can only either be sending or receiving an audio stream. Consequently, a new participant joining the meeting does not affect the bandwidth requirement for the rest of the users. Each participant asks for only one audio stream for either sending or receiving.

The server receives the audio stream from only one user, and forwards the stream to the rest of participants. Graph 4.7shows bandwidth usage at the server side.



Graph 4.7: Total bandwidth used on the server per minute (in kilo bits) during an audio meeting where a new participant joined every minute in half duplex mode

The total bandwidth usage for the server can then be approximated with the following formula:

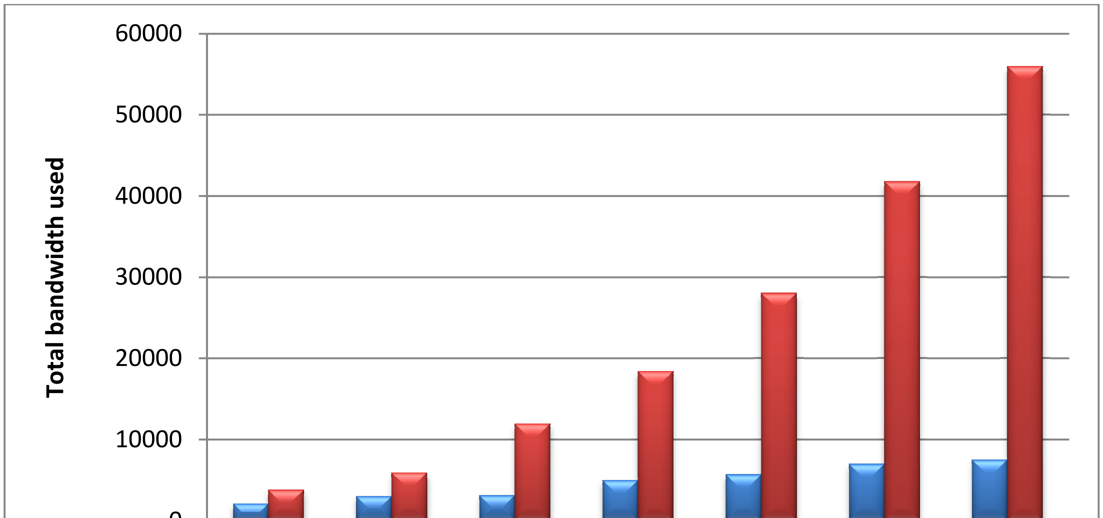
Bandwidth for reception = audio bit rate

Bandwidth for sending = audio bit rate X (number of participants - 1)

Total server bandwidth = reception + sending

Total server bandwidth = audio bit rate x number of participants

Graph 4.8 compares total server bandwidth usage per minute between half and full duplex mode.



	2 users	3 users	4 users	5 users	6 users	7 users	8 users
half duplex	2097	2998	3123	4974	5677	6967	7469
Full duplex	3785	5903	11943	18389	28008	41777	55958

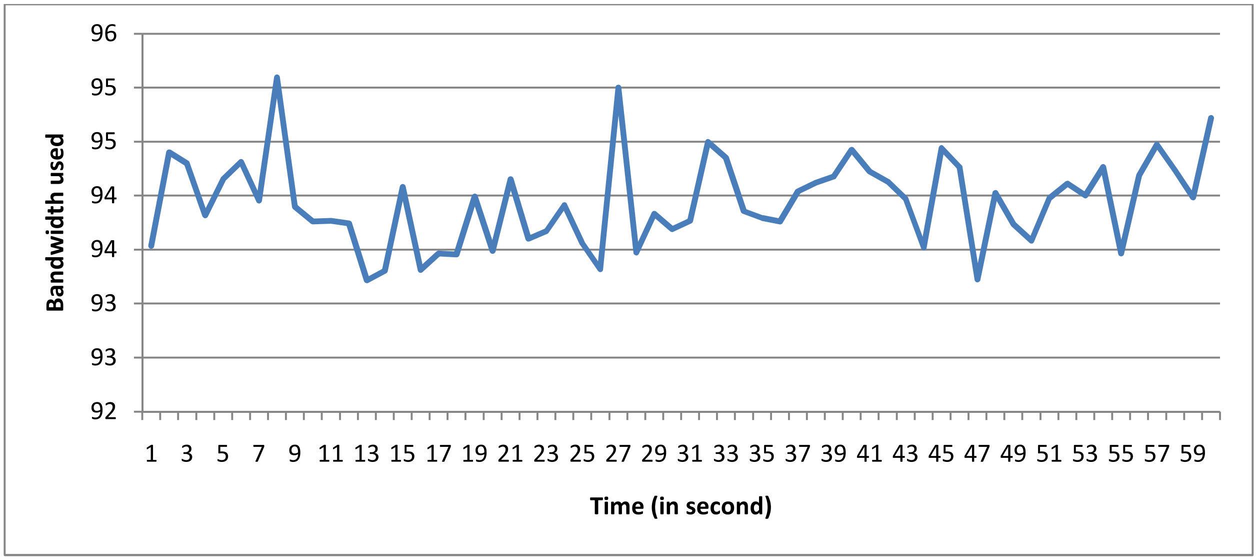
Graph 4.8: Comparison of server bandwidth (in kilo bit) usage between half and full duplex mode for an audio meeting where a new participant joined every minute

The result on Graph 4.8 motivates the choice to use half duplex mode, as it allows using the bandwidth efficiently when the number of participants increases. The choice is also motivated by the fact that a participant focuses mostly on the presenter during the meeting.

4.2.2 Video sub-system bandwidth usage

4.2.2.1 Experiment 5: Video conference with 2 participants without image differentiation

In experiment 5, two participants are having one minute of video conferencing. The video is sent and received at 1 frame per second (1 FPS). Graph 4.9 gives the bandwidth usage at the client side.



Graph 4.9: Client bandwidth (in kbps) usage during a video conference at 1 FPS

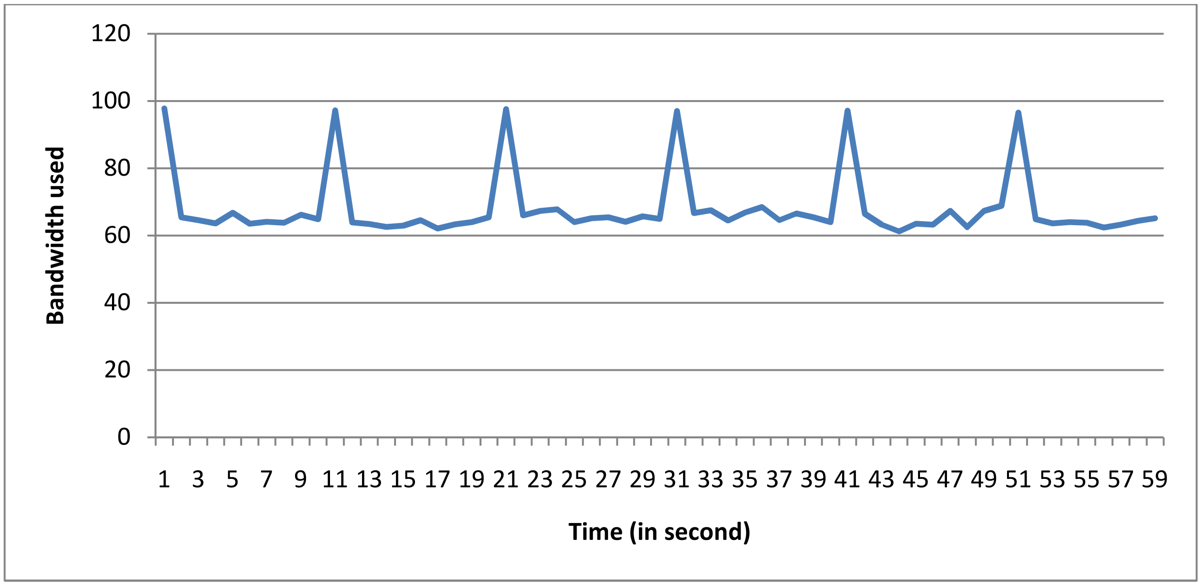
Each point of the curve in Graph 4.9 represents the size of images (in kilo bits), compressed using JPEG. Table 4.5 below gives the maximum, minimum and average bandwidth required in kbps.

Table 4.5: Maximum, minimum and average client bandwidth (in kbps) usage for video stream at 1FPS

Maximum	95
Minimum	93
Average	**94**

4.2.2.2 Experiment 6: Video conference with 2 participants using image differentiation

In experiment 6, two participants are having a video conference with image differentiation activated. The video is sent and received at 1 FPS. Graph 4.10 indicates the bandwidth usage for the video stream.



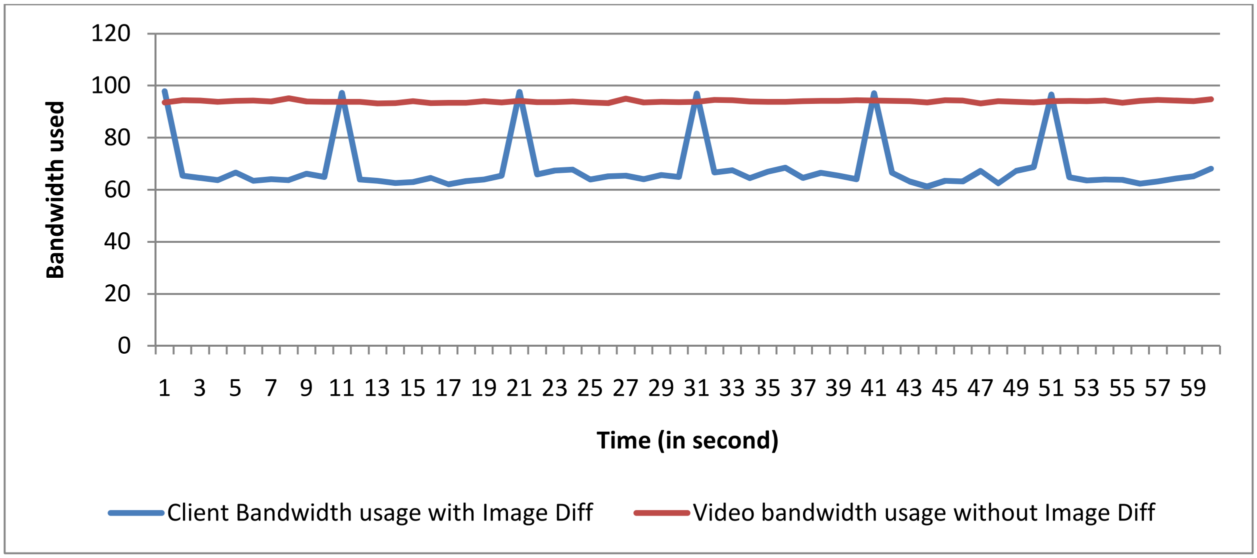
Graph 4.10: Client bandwidth (in kbps) usage during a video conference at 1 FPS with image differentiation

The image differentiation algorithm sends one key frame (the peaks in Graph 4.10) followed by 9 middle frames (differences between 2 pictures) that receive a better JPEG compression. Table 4.6 below gives the maximum, minimum and average bandwidth required in kbps.

Table 4.6: Maximum, minimum and average video bandwidth (in kbps) usage with image differentiation activated

Maximum	98
Minimum	61
Average	**68**

Graph 4.11 compares bandwidth usage between the video without image differentiation and the one with image differentiation; both are at 1FPS.



Graph 4.11: Client bandwidth (in kbps) usage: comparison between video stream with image differentiation and without image differentiation

The network activity represented in Graph 4.11 is summarized in Table 4.7.

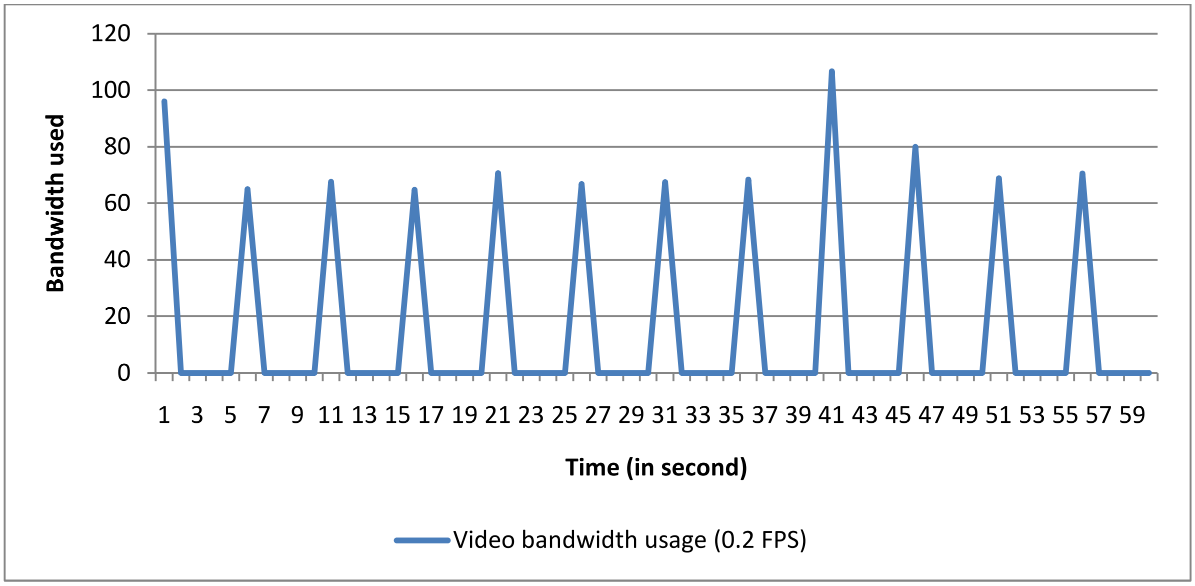
Table 4.7: Maximum, minimum and average bandwidth (in kbps) usage : comparison between video with image differentiation and without image differentiation

	Image diff	No image diff
Maximum	98	95
Minimum	61	93
Average	**68**	**94**

Image differentiation offers around a 28% reduction of bandwidth requirement on a video recorded, at 1 FPS. Now the gain is even better because images are very close to each other allowing a better compression of differences.

4.2.2.3 Experiment 7: Video conference with 2 participants at lower frame rate (0.2 FPS)

In experiment 7, two participants are having a video conference with 0,2 FPS; that is sending an image every 5 seconds. Graph 4.12 and Table 4.8 give the bandwidth usage for this case.



Graph 4.12: Client bandwidth (in kbps) usage for a video stream at 0.2 FPS

The two highest peaks at 0 and 41 represents key frame and the rest of peaks are middle frames. At 0.2 FPS, an image is sent every 5 second. In consequence, during the 4 seconds separating peaks the bandwidth usage is null.

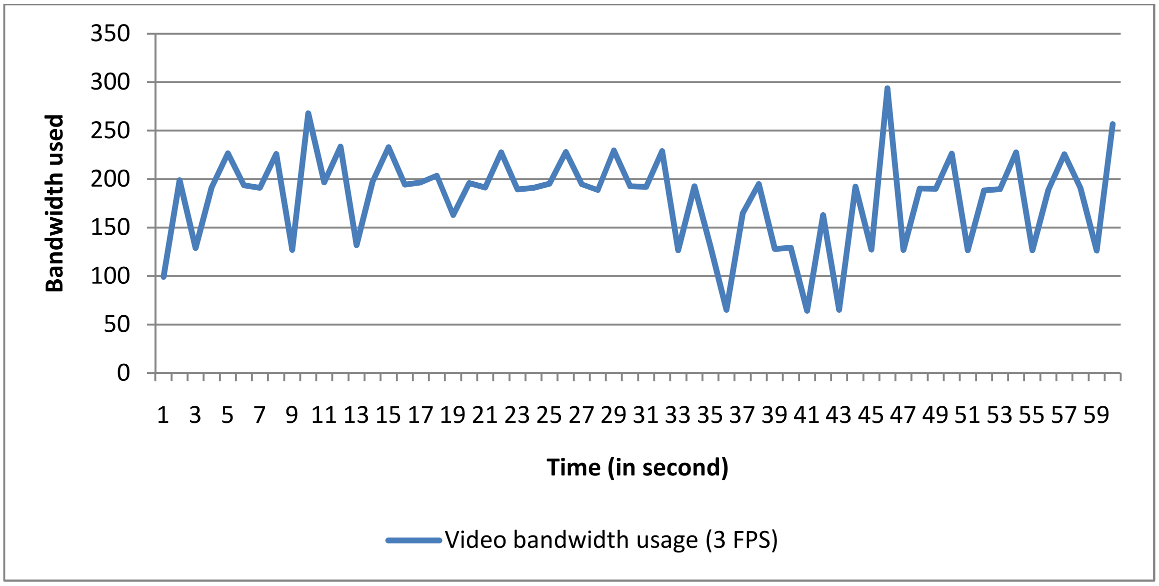
Table 4.8: Maximum, minimum and average client bandwidth (in kbps) usage of video stream at 0.2 FPS

Maximum	107
Minimum	0
Average	**15**

The resulting video stream uses only 15 kbps, which is smaller than the average audio stream (28 kbps). Such stream can help to convey a certain sense of presence for meetings where the available bandwidth is very limited.

4.2.2.4 Experiment 8: Video conference with 2 participants at 3 FPS

In experiment 8, one minute of video conference between 2 participants is monitored. Graph 4.13 and Table 4.9 summarize such bandwidth usage during the meeting.



Graph 4.13: Client bandwidth (in kbps) usage of video stream at 3 FPS

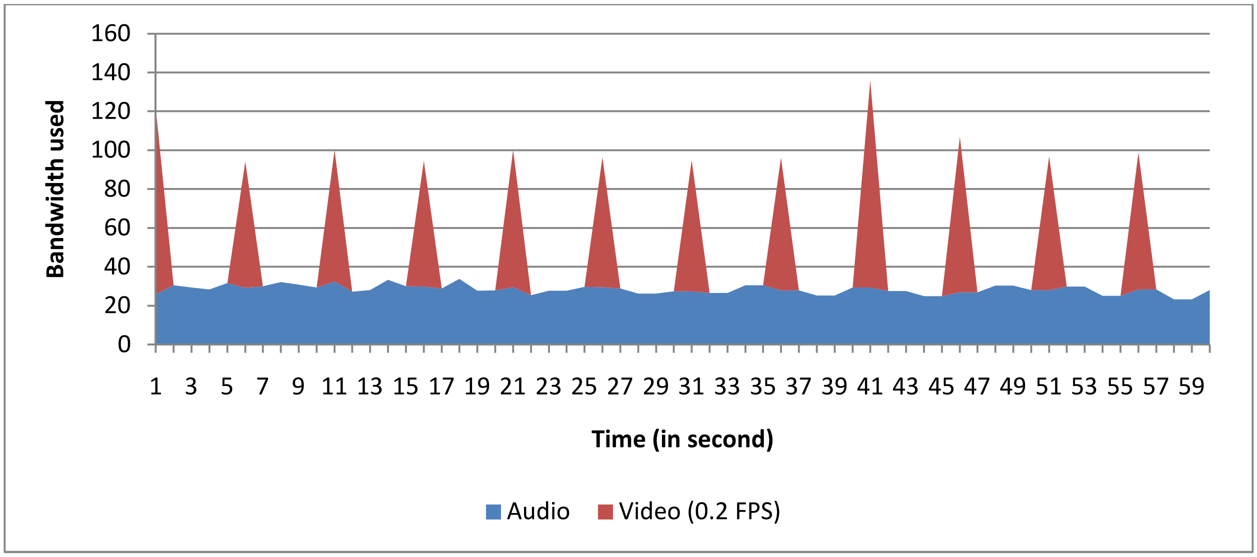
Table 4.9: Maximum, minimum and average client bandwidth (in kbps) usage for a video stream at 3 FPS

Maximum	294
Minimum	64
Average	**181**

4.2.3 Audio-video bandwidth usage

4.2.3.1 Experiment 9: audio-video conferencing with 2 participants with low frame rate (1 FPS)

In experiment 9, two users participate in an audio-video conference for one minute. The video is streamed at 0.2 FPS. Graph 4.14 and Table 4.10 summarize bandwidth usage of both audio and video stream as still monitored.



Graph 4.14: Client bandwidth (in kbps) usage audio and a 0.2 FPS video stream

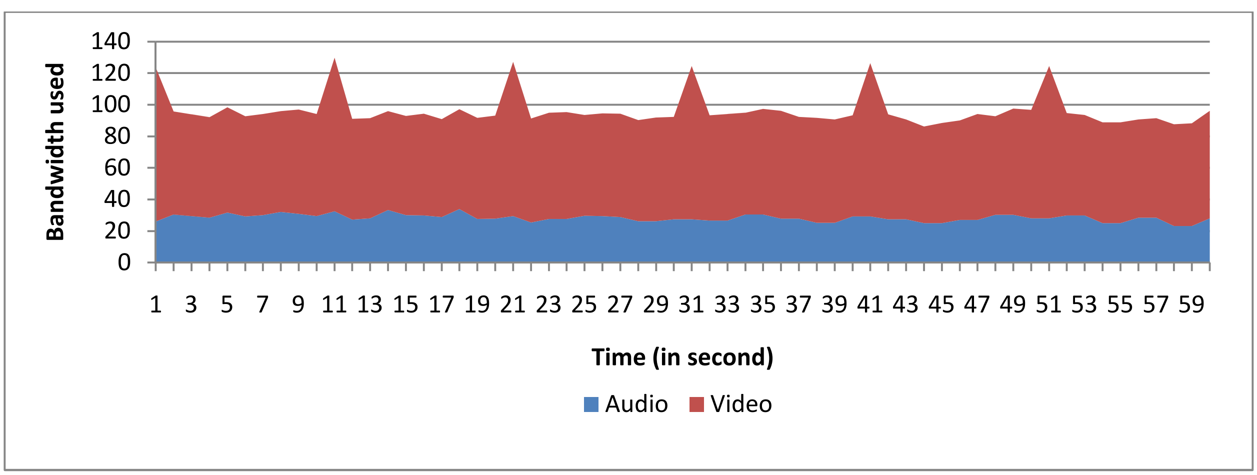
Table 4.10: Maximum, minimum and average client bandwidth (in kbps) usage for audio and 0.2 FPS video stream

Maximum	136
Minimum	23
Average	**43**

In this setting, the video stream uses a relatively few bit rate that represents 35% of the total bandwidth usage; and almost the half of audio stream bit rate.

4.2.3.2 Experiment 10: audio-video conferencing with 2 participants at 1 FPS

In experiment 10, two users participate in an audio-video conference for one minute. The video is streamed at 1 FPS. Graph 4.15 and Table 4.11 summarize bandwidth usage of both audio and video stream monitored.



Graph 4.15: Client bandwidth (in kbps) usage for audio and 1 FPS video stream

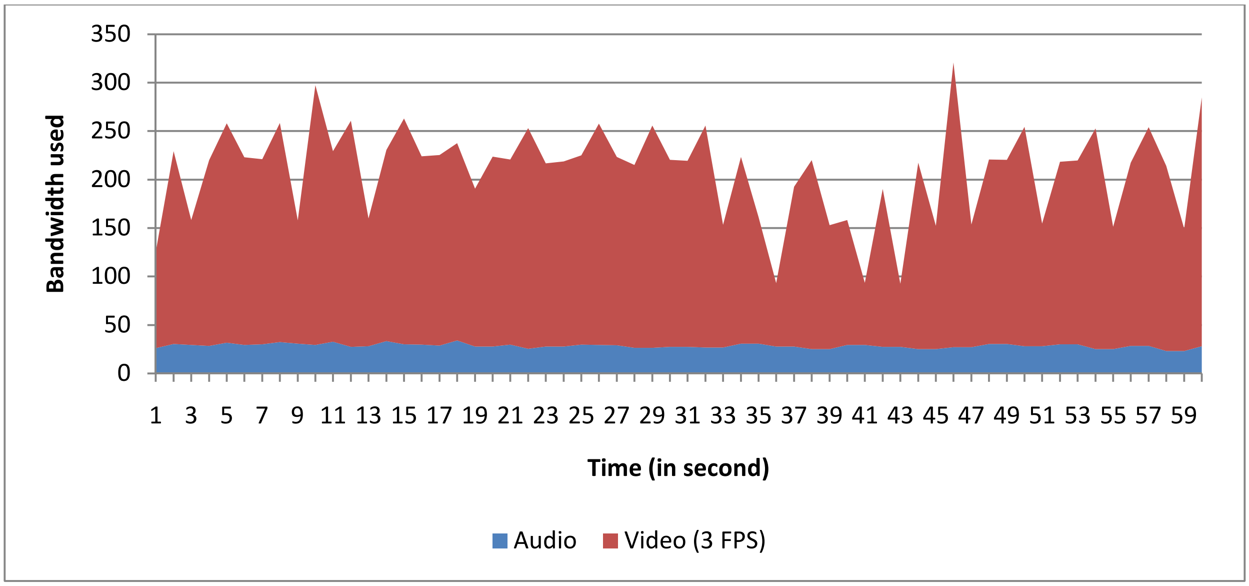
Table 4.11: Maximum, minimum and average client bandwidth (in kbps) usage for both audio and 1 FPS video stream

Maximum	130
Minimum	86
Average	**96**

At 1 FPS, the video playback is 5 times more fluent then 0.2 FPS and can fairly well convey body movements. At this frame rate, the video stream represents 71% of the total bandwidth and is more than twice the audio usage.

4.2.3.3 Experiment 11: audio-video conferencing with 2 participants at 3 FPS

In experiment 11, two users participate in an audio-video conference for one minute. The video is streamed at 3 FPS. Graph 4.16 and Table 4.12 summarize bandwidth usage of both audio and video stream monitored.



Graph 4.16: Client bandwidth (in kbps) usage audio and 3 FPS video stream

Table 4.12: Maximum, minimum and average client bandwidth (in kbps) usage for both audio and 3 FPS video stream

Maximum	321
Minimum	93
Average	**209**

The 3FPS video stream represents 87% of the total bandwidth usage.

4.3 System bandwidth requirement for audio and video conferencing

From preceding experiments, the system bandwidth requirement is approximated and summarized in Table 4.13.

Table 4.13: Client and server bandwidth requirement

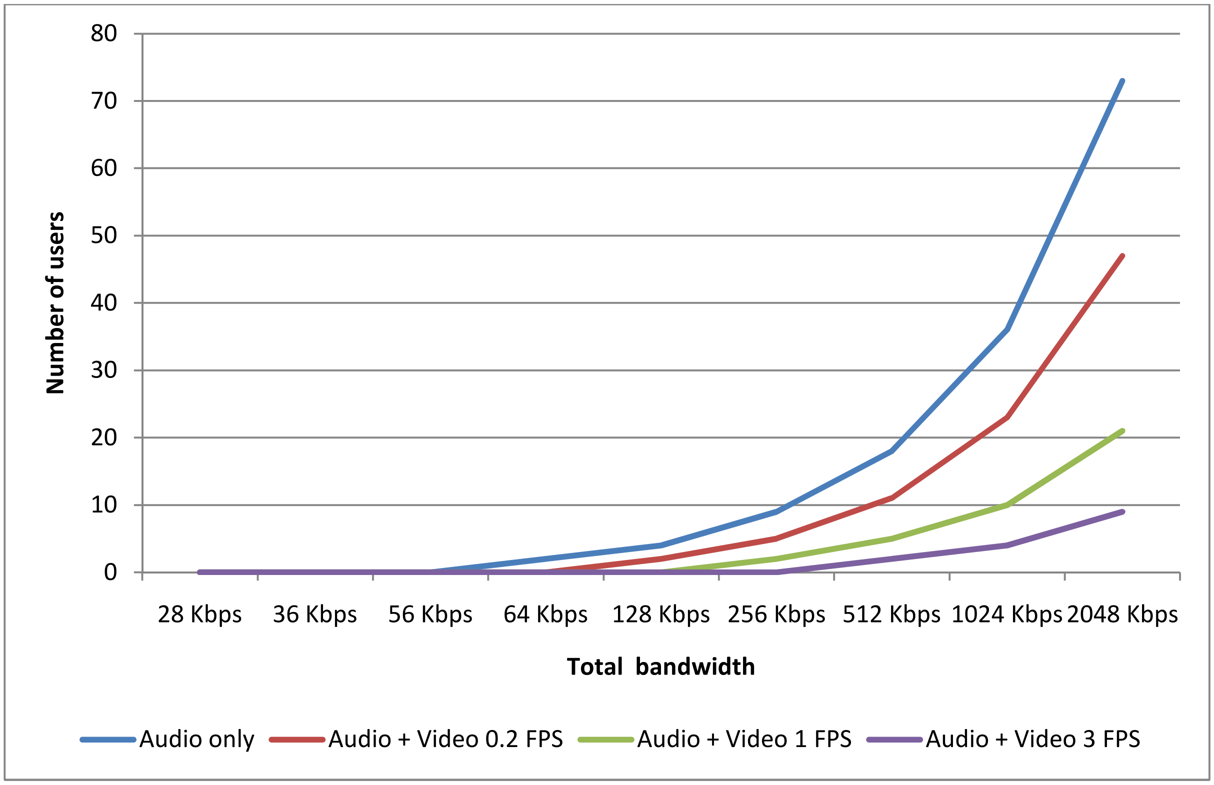
	Client	Server			
		2 Users	5 Users	10 Users	20 Users
Audio Only	56	56	140	280	560
Audio + video (0.2 FPS)	86	86	215	430	860
Audio + video (1 FPS)	192	192	480	960	1 920
Audio + video (3 FPS)	418	418	1 045	2 090	4 180

For the client, the half of bandwidth is needed for upload link and the other half for download link. Since the communication uses half duplex mode, only one link (or half of the total bandwidth required) is used at any time for either sending or receiving stream.

The server requires a broader upload link. With half duplex mode, the download link is used to receive one stream which is uploaded to the rest of client. Table 4.14 and Graph 4.17 summarize the expected number of users that the server can handle for different configurations.

Table 4.14: Expected maximum number of participant at the server

	28 Kbps	36 Kbps	56 Kbps	64 Kbps	128 Kbps	256 Kbps	512 Kbps	1024 Kbps	2048 Kbps
Audio only	-	-	-	2	4	9	18	36	73
Audio + Video 0.2 FPS	-	-	-	-	2	5	11	23	47
Audio + Video 1 FPS	-	-	-	-	-	2	5	10	21
Audio + Video 3 FPS	-	-	-	-	-	-	2	4	9



Graph 4.17: Expected number of participant at the server

4.4 User based system evaluation

The objective of this evaluation is to assess the quality of user experience for the audio-video sub-system. For the audio system, the evaluation should reveal if the sound quality provided is satisfactory conveying a good user experience during meetings. The second part of the evaluation assesses the video quality as perceived by users. The 3 video settings (low, medium and high frame rate) are tested in order to get a user based rank of how the video enhances the sense of presence during the meeting.

4.4.1 Test organization and logistics

A total of 13 users have accepted to participate in the experiment; which consisted of participating in a virtual meeting using the prototype developed. To create a realistic condition, two separated rooms were used for remote meetings. Due to technical problems (webcam drivers not supported) and logistic limitation (only two rooms where available); it was not feasible to organize meetings with more than 2 participants. In the whole, 7 meeting sessions were organized.

4.4.2 Test process

Before starting the evaluation, the users were:

- Warmly thanked for their willingness to participate to the experiment;
- Quickly brief the overall project and team members;

- Quickly describe the prototype;
- Present a detailed description of the test process;
- Given the user consent, for reading and signing (a copy of the consent is on Appendix A);
- Given time for questions;

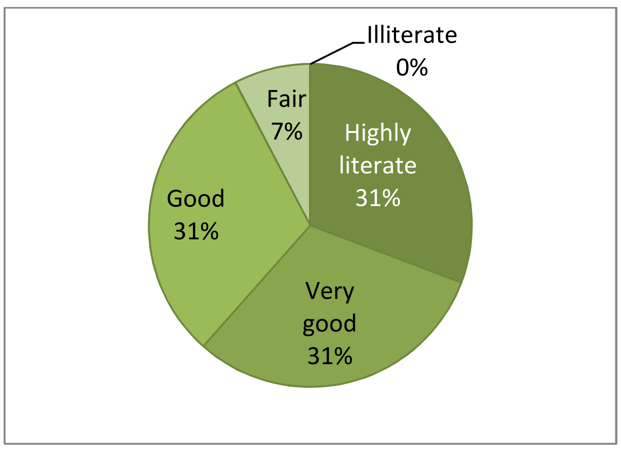
After the briefing, 2 users where invited to attend an audio-video meeting session. Each meeting took around 5 minutes. At the end, the users were invited to fill out a feedback questionnaire (a copy of the questionnaire is on Appendix B).

Before leaving, the participants were once again warmly thanked for their participation in the experiment.

4.4.3 User background

4.4.3.1 Computer literacy

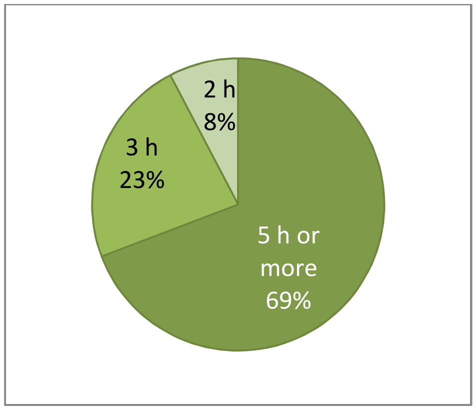
On a scale of 1 to 5 (with 5 = highly computer literate), the average level is 3,8. Graph 4.18 summarizes the computer literacy of users who participated in the experiment.



Graph 4.18: User computer literacy

4.4.3.2 Internet usage

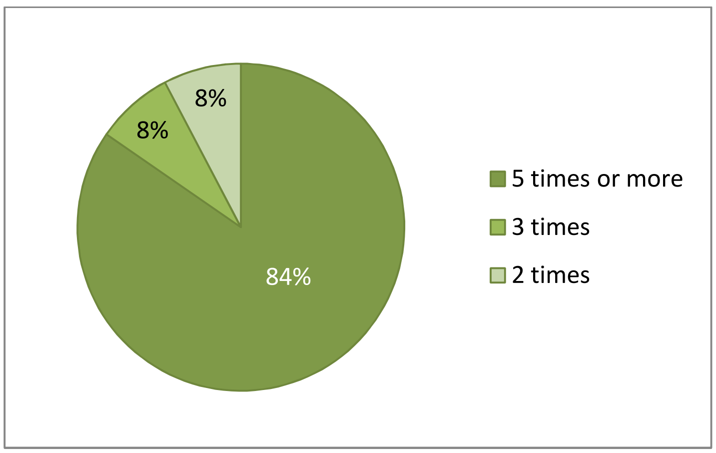
9 users out of 13 spend 5 hours or more online on a typical weekday. Graph 4.19 summarizes the average time spent online by users per day.



Graph 4.19: Number of hours spent daily online

4.4.3.3 Previous experience using audio-video Internet conferencing tools

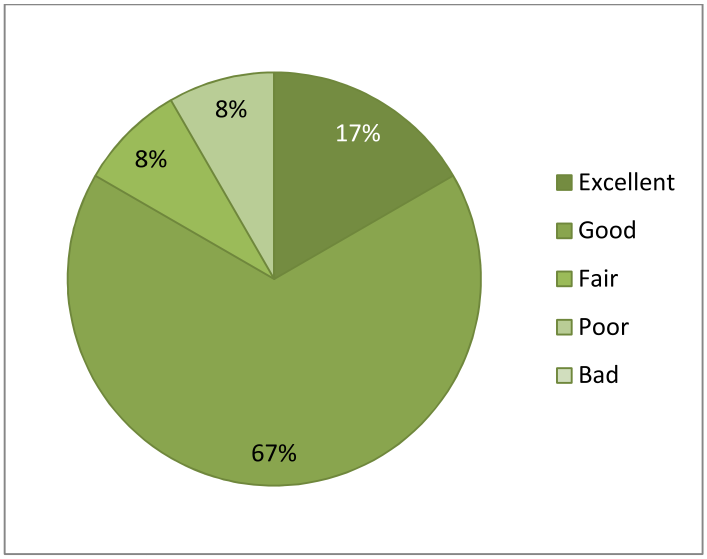
As showed on Graph 4.20, 84% of users used 5 times or more an audio-video Internet conferencing tool during last 10 months.



Graph 4.20: Testers previous experience using audio-video Web conferencing tools

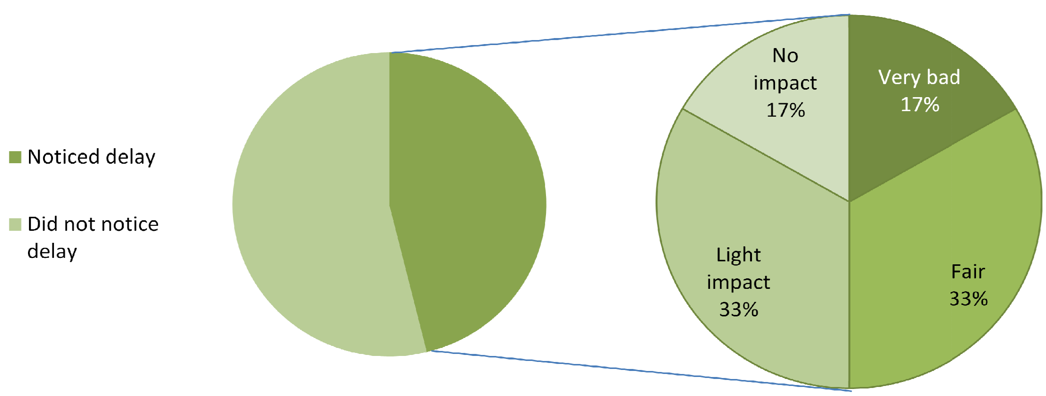
4.4.4 Audio evaluation and results

During testing, the sound was activated for all meetings. On a scale of 1 to 5 (5 = Excellent), the sound quality was ranked at 4 on average. Graph 4.21 gives the users perception of audio quality.



Graph 4.21: Sound quality as perceived by users

Due to buffering and transmission, the sound is played with a delay of 1,5 second. 46% of users reported that they noticed that delay during the meeting. On average testers who noticed the delay reported it as fairly disturbing the audio meeting. Graph 4.22 summarizes user perception of the delay and it impact on the meeting.



Graph 4.22: User perception of the sound delay and its impact on the meeting

The overall audio experience was reported to be good. One participant reported an echo effect: "I was hearing my own voice when speaking".

4.4.5 Video evaluation and results

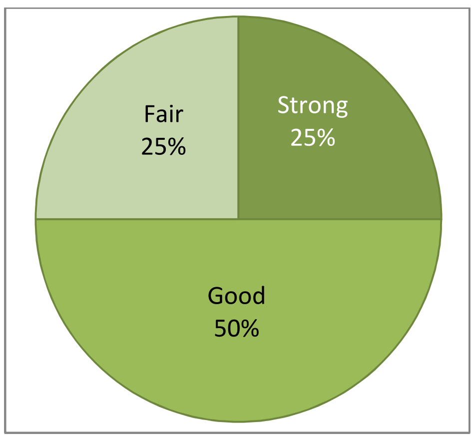
The video stream was evaluated in 3 different settings: low frame rate (0.2 FPS), average frame rate (1 FPS) and relatively high frame rate (3 FPS).

4.4.5.1 Video evaluation with low frame rate (0.2 FPS)

4 users participated in 2 video meetings run at low frame rate, with image updated every 5 seconds. This video setting is perceived as “not good” by most of the users. The following comments were collected: “I think it is a very interesting product, even if the video quality if not very good”; “Good job, but the video quality needs improvement”.

The image differentiation algorithm sends a key frame after 9 middle frames to cope with image degradation produced by accumulation of differences. At low frame rate, these intermediate degradations get more noticeable. This fact can explain the poor perception reported. A user commented: “The audio quality is good, but the video quality is not good, there is a delay and the image is not clear”.

As expected, a video delay was reported by all testers. The quality of the video is ranked as “Fair” by all the users. But more importantly, 3 testers out of 4 estimate that the video is conveying a certain sense of presence during the video meeting. Despite the quality of video, 75 % of users reported a good or strong sense of presence and participation to an actual meeting. Graph 4.23 gives the user rank of their feeling of presence and participation to an actually meeting.



Graph 4.23: Feeling of presence and participation to an actual meeting with a low frame rate video stream (0.2 FPS)

4.4.5.2 Video evaluation with medium frame rate (1 FPS)

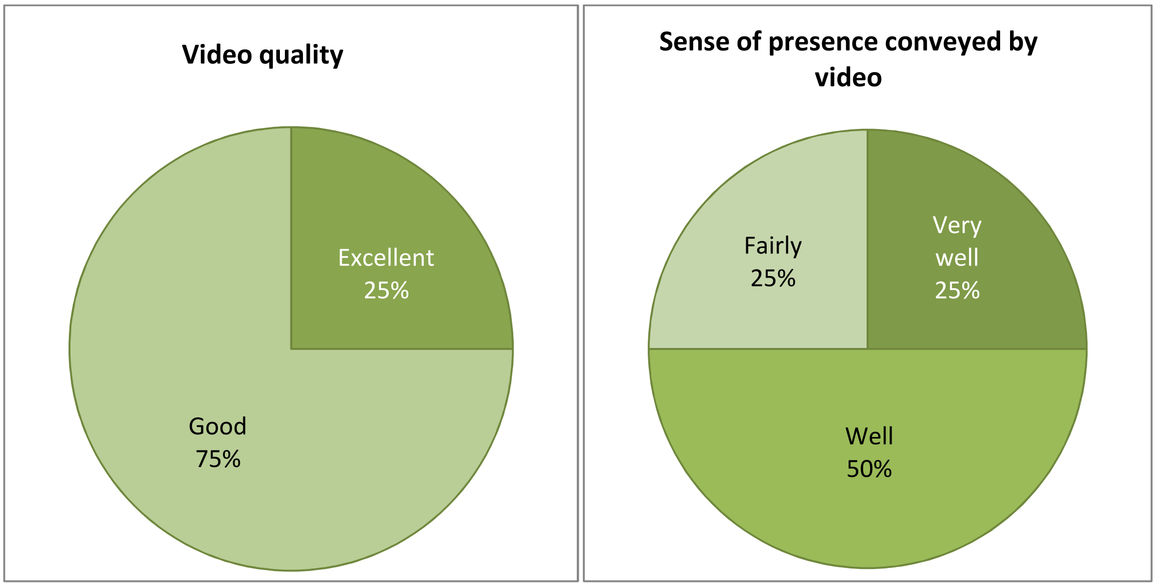
5 users participated in 3 meetings with video streamed at 1 FPS. One tester attended 2 meetings. The overall appreciation of video quality is controversial. For example, one tester commented: “Audio delay noticeable, very good video stream”, while another observed: “… the image quality is not very good … But we could see each other, which is great”.

With images updating every second, all testers noticed the video delay but they all reported that the video is conveying “well” a sense of presence. The video quality is perceived as “good” by 60% of users and as “fair” by 40% of users. 3 testers estimate that the sense of presence during the experience is “good” while 2 testers rank it as “very good”.

4.4.5.3 Video evaluation with higher frame rate (3 FPS)

4 users participated in 2 meetings with video playing at 3 FPS. The global video quality is perceived as “good”, while now the sound is perceived as “average”! Observations collected from users: “The image quality was good but the audio during the conversation was average”; “The video is excellent, but the audio is good”. No user reported to have noticed a delay on video playback.

As described on Graph 4.24, the video quality is perceived as “good” by 3 users and “Excellent” by one user. On average, the sense of presence conveyed by the video is ranked as “well”.



Graph 4.24: User quality perception for 3 FPS video stream

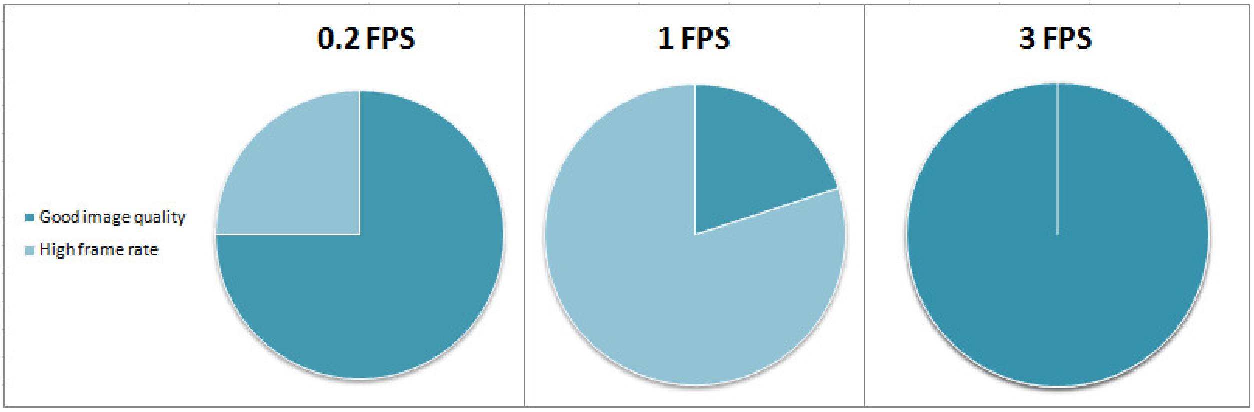
Curiously enough, the overall meeting experience is ranked as “Good” (3 on a scale of 5). This result may be explained by the fact that audio is perceived as “average”.

4.4.5.4 Best video compression approach for users

The users were asked on which of the two below video compression approaches is the best:

- Keep good image quality with very low frame rate;
- Privilege a higher frame rate to the detriment of image quality;

Graph 4.25summarizes the result obtained from users grouped by video frame rate they tested. 62% of all users would prefer to have a good image quality with lower frame rate.



Graph 4.25: Best video compression approach according to users

4.5 Evaluation results and discussion

The analysis of bandwidth usage shows that half duplex communication is more effective than full duplex. This fact is particularly true for a meeting context, where communications are less interactive compared to phone call, for example. In a meeting with 2 participants, half duplex uses 76% of the bandwidth required by full duplex. But this difference drastically increases with the number of participant: for 8 users, half duplex uses only 13% of bandwidth required by full duplex.

Image differentiation helps to reduce bandwidth usage by sending across the difference between images. The experiment indicates an average gain of 28% in bandwidth usage when image differentiation is used. This gain increases when higher frame rates are used, as still images are closer to each others.

Combining image differentiation with low frame rates can result in a very light video stream. When the image is updated every 5 seconds, the resulting video stream requires only 15 kbps (~ 2 KB/s), which is almost half of audio stream bandwidth.

The audio compression scheme adopted reduces the sound stream to 44% of its initial size without quality degradation. The resulting stream is evaluated as good or excellent by 86% of users.

The video was evaluated in 3 different settings: low frame rate (0.2 FPS), medium frame rate (1 FPS) and high frame rate (3 FPS). For low frame rate video stream, as expected, the quality of the video is perceived as “not that good” by users. But more importantly, the video stream helped to convey a sense presence ranked as either “good” or “strong” by 75% of users. This fact demonstrates that even with limited bandwidth, it is possible to supply a minimalistic video stream that can significantly enhance the meeting experience. This appreciation of video contribution to the meeting naturally increases when to frame rate raises.

The prototype developed requires only 64 kbps at the server to reliably host an audio meeting with 2 participants. When a low frame rate video is used with audio, 96 kbps is needed to host a consistent meeting with 2 participants. A server with 512 kbps bandwidth can host meeting with: 18 participants using audio only, 11 participants using audio and low frame rate video, 5 participants using audio and medium frame rate video or 2 participants using audio and high frame rated video.

CHAPTER 5 – FUTURE WORK

During experiments, it has been noticed that the background noise level has a direct effect on the compression. Microphones featuring noise cancellation systems provide a stream with better compression. For future work, this result can be enhanced by implementing software-based noise cancellation to reduce background noise.

The experimental system records, compresses and sends sound packets every second. But observations show that when a speaker is silent, the resulting audio stream uses half of the bandwidth required by a normal speech. Implementing an efficient silence detection system could reduce to null the bandwidth usage when no one is talking.

There is a fairly good penetration of mobile devices and telecom networks in African and other developing countries. Mobiles phones are getting more and more affordable while including new features and capabilities. A future work could study the possibility to implement a mobile client for audio video Web meetings.

CHAPTER 6 – CONCLUSION

There is currently a global demand and need for real time communication and collaborative tools. Talking to and seeing each other is now a common usage of the Internet. This need of Internet real-time communication is also true for Africa. The aim of this project was to investigate the feasibility of an audio-video Internet-based tool that can provide a satisfactory user experience with limited bandwidth and networking problems. Approaches used to address the preceding problems include:

- Prioritization of audio stream over the video;
- Use of half duplex mode instead of full duplex;
- Utilization of 3 different levels of frame rate to adapt the video stream to the actual bandwidth available;
- Implementation of an image differentiation algorithm to reduce the size of image transmitted;
- Compression of audio stream and video stream;

The above approaches guided the development of an experimental audio-video Web meeting prototype. The objective of the prototype was to deliver a good user experience when using the lowest bandwidth possible and copying with networking problems. The system developed can provide:

- A clear audio stream (radio quality) at 28 kbps (or 2.5 KB/s);
- An audio stream + low frame rate video (0.2 FPS) at 43 kbps (or 5.4 KB/s);
- An audio stream + medium frame rate video (1 FPS) at 96 kbps (or 12 KB/s);
- An audio stream + high frame rate video (3 FPS) at 209 kbps (or 26 KB/s);

A server with 512 kbps (or 64 KB/s) total bandwidth can handle a meeting with:

- 18 users using audio only;
- 11 users using audio and 0.2 FPS video;
- 5 users using audio and 1 FPS video;
- 2 users using audio and 3 FPS video;

The sound quality was perceived as either good or excellent by 84% of users who tested it. The feeling of presence and involvement in the meeting was estimated as good or strong by:

- 75% of users who tested the system with low frame rate (0.2 FPS); and
- 100% of users who tested with medium (1 FPS) and high (3 FPS) frame rate.

The above results show that it is possible to provide a fairly good Internet audio video meeting experience with constraining networking conditions.

REFERENCES

Ajuwon, G.A. & Rhine, L., 2008. The level of Internet access and ICT training for health information professionals in sub-Saharan Africa. *Health Information & Libraries Journal*, 25(3), pp.175-185.

Amir, E., McCanne, S. & Katz, R., 1997. Receiver-driven bandwidth adaptation for light-weight sessions. In *Proceedings of the fifth ACM international conference on Multimedia - MULTIMEDIA '97*. New York, New York, USA: ACM Press, pp. 415-426. Available at: http://portal.acm.org/citation.cfm?doid=266180.266395.

Anon, 2006. Impact of delay on perceptual quality. *Management*, (January), pp.1-7.

Anon, 2011a. Top 5 Web Conferencing Solutions. Available at: http://www.webconferencing-test.com/en/webconferencing_top5/webconference_compact_results.html [Accessed October 1, 2011].

Anon, 2011b. Wikipedia. Available at: http://en.wikipedia.org/.

Brakmo, L. & O'Malley, S., 1994. TCP Vegas: New techniques for congestion detection and avoidance. *ACM SIGCOMM Computer*. Available at: http://portal.acm.org/citation.cfm?id=190317 [Accessed October 25, 2011].

Chen, M., 2002. Achieving effective floor control with a low-bandwidth gesture-sensitive videoconferencing system. In *Proceedings of the tenth ACM international conference on Multimedia - MULTIMEDIA '02*. New York, New York, USA: ACM Press, p. 476. Available at: http://portal.acm.org/citation.cfm?doid=641007.641109.

Claypool, M. & Tanner, J., 1999. The effects of jitter on the peceptual quality of video. In *Proceedings of the seventh ACM international conference on Multimedia (Part 2) - MULTIMEDIA '99*. New York, New York, USA: ACM Press, pp. 115-118. Available at: http://portal.acm.org/citation.cfm?doid=319878.319909.

Davidson, J.D. & Coward, D., 1999. *Java servlet specification, V2. 2*, Available at: http://webs.hufs.ac.kr:8989/HUFS/jsp1.1/pdf/servlet2_2pr.pdf [Accessed October 25, 2011].

Deering, S.E. & Cheriton, D.R., 1990. Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computer Systems*, 8(2), pp.85-110. Available at: http://portal.acm.org/citation.cfm?doid=78952.78953.

Edigo, C., 1988. Videoconferencing as a Technology to Support A Review . of its Failure Group Work : *Computer Supported Cooperative Work (CSCW)*, pp.13-24.

Foo, S., Hui, S.C. & Yip, S.W., 1999. Enhancing the quality of low bit-rate real-time Internet communication services. *Communications*, 9(3), pp.212-224.

Gill, P. et al., 2008. Scalable on-demand media streaming for heterogeneous clients. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(1), pp.1-24. Available at: http://portal.acm.org/citation.cfm?doid=1404880.1404888 [Accessed April 27, 2011].

Gray, V. & Minges, M., 2008. *African Telecommunication/ICT Indicators 2008: At a Cross-Roads*, Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:African+Telecommunication+/+ICT+Indicators+2008+:+At+a+Crossroads#0 [Accessed October 24, 2011].

Guan-Ming, S., 2005. Efficient bandwidth resource allocation for low-delay multiuser video streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(9), pp.1124-1137. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1501880.

Gurhan, G.F. et al., Audio Video Conferencing. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24.3422. [Accessed April 27, 2011].

Hargreaves, D.M.G. & McCown, B.R.L., 2008. Low-cost, low-bandwidth online meetings between farmers and scientists. In *Proceedings of the 20th Australasian Conference on Computer-Human Interaction Designing for Habitus and Habitat - OZCHI '08*. New York, New York, USA: ACM Press, p. 271. Available at: http://portal.acm.org/citation.cfm?doid=1517744.1517776.

Jain, R., Kim, P. & Li, Zhao, 2003. Experiential meeting system. *Proceedings of the 2003 ACM SIGMM workshop on Experiential telepresence - ETP '03*, pp.1-12. Available at: http://portal.acm.org/citation.cfm?doid=982484.982486.

Marshal, D., 2001. Digitization of sound. Available at: http://www.cs.cf.ac.uk/Dave/Multimedia/node145.html [Accessed October 1, 2011].

McCanne, S., Jacobson, V. & Vetterli, M., 1996. Receiver-driven layered multicast. *ACM SIGCOMM Computer Communication Review*, 26(4), pp.117-130. Available at: http://portal.acm.org/citation.cfm?doid=248157.248168.

Ou, Y. et al., 2008. Modeling the impact of frame rate on perceptual quality of video. *Image Processing, 2008.*, pp.1-5. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4711848 [Accessed October 30, 2011].

Scholl, J. et al., 2005. Designing a large-scale video chat application. In *Proceedings of the 13th annual ACM international conference on Multimedia - MULTIMEDIA '05*. New York, New York, USA: ACM Press, pp. 71-80. Available at: http://portal.acm.org/citation.cfm?doid=1101149.1101160.

Suduc, A.-M., Bizoi, M. & Filip, F.G., 2009. Exploring Multimedia Web Conferencing. *Informatica Economica Journal*, 13(3), pp.5-17.

Watson, A. & Sasse, M.A., 1996. Evaluating audio and video quality in low-cost multimedia conferencing systems. *Interacting with Computers*, 3(3), pp.255-275.

Watson, A. & Sasse, M.A., 1997. Multimedia Conferencing via Multicast : Determining the Quality of Service Required by the End User. In *Proc. 1997 International Workshop on Audio-Visual Services over Packet Networks (AVSPN)*. pp. 1-6.

Xia, J. et al., 2011. Modeling and Compressing 3D Facial Expressions Using Geometry Videos. *Circuits and Systems for Video Technology, IEEE Transactions on*, (99), p.1–29. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5782944 [Accessed October 30, 2011].

Xin, J. & Lin, C., 2005. Digital video transcoding. In *Proceedings of the IEEE*. pp. 84-97. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1369700 [Accessed October 30, 2011].

APPENDIX A – USER AGREEMENT

1. Project presentation

A Web meeting system is an Internet-based tool offering a virtual environment for remote meeting and collaborative work among geographically dispersed participants. Remote conferencing can avoid travel expenses and time required by face-to-face meetings. This explains the worldwide spreading of Web based conferencing tools.

However, low bandwidth and unstable Internet connections make most of Internet conferencing solutions unreliable for South Africa and developing countries in general. This project aims to design a developing-country-aware application to host online meetings where multiple participants can share audio, video, presentations and screen. The project is realized by a team of 3 members:

Flora Kundaeli: responsible for Screen sharing and floor control
Tresor Mvumbi: responsible for audio and video communications
Zafika Manzi: responsible for presentation sharing and text chat

2. Evaluation of audio and video sub-system

The purpose of this evaluation is to test audio-video module and assess the quality of user experience. The proper test will consist of participating in an audio video meeting and reflect the experience by responding to a feedback questionnaire. The whole test will take 30 minutes. Should needed, each participant is completely free to leave the experiment at anytime.

3. Test course

Task	Time
Brief presentation of the project and experiment	5 min
Quick question-response session before the experiment	3 min
Experiment: participation in an audio video Web meeting session	10 min max
Feedback: filling the questionnaire	10 min
Thanksgiving and handshaking	2 min
Total time	30 min

4. Confidentiality and personal data

The information recorded on the consent document is solely for the sake of ethical clearance requirement. These data will not be published on the final report or anywhere else. The feedback questionnaire is completely anonymous and most of the information collected will be presented using aggregated graphs and tables.

5. User agreement

I, ____, acknowledge that I read and agree with this consent; and I would freely like to participate to the experiment.

Signature

Date

UNIVERSITY OF CAPE TOWN DEPARTMENT OF COMPUTER SCIENCE	HONOURS PROJECT 2011 Subject: “AfriMeet: An Internet meeting tool designed for low bandwidth and unstable network conditions”

APPENDIX B – EXPERIMENT FEEDBACK

Note: stick the right case

Question 1: How would you value your computer literacy? (1 = Illiterate, 5 = highly literate)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Question 2: How much time do you spend on the Internet in a typical weekday?

☐ 1h or less ☐ 2 h ☐ 3h ☐ 4h ☐ 5h or more

Question 3: How many times have you used audio or video communication software on the Internet this year?

☐ never ☐ 1 to 3 times ☐ 4 to 6 times ☐ 7 to 10 times ☐ 10 times or more

Question 4: how would you rate the overall sound quality during the meeting?

☐ Bad ☐ Poor ☐ Fair ☐ Good ☐ Excellent

Question 5: Did you notice a sound delay when participating in the meeting?

☐ YES ☐ NO

Question 6: If you answered yes to Question 5, how badly the sound delay impacted the meeting experience?

(1 = no impact, 5 = very bad impact)

☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Question 7: Did you notice a video delay when participating to the meeting?

☐ YES ☐ NO

Question 8: how would you rate the feeling of presence and participation to an actual meeting?

☐ Poor ☐ Fair ☐ Good ☐ Strong ☐ Very strong

Question 9: how would you rate the overall video quality during the meeting?

☐ Bad ☐ Poor ☐ Fair ☐ Good ☐ Excellent

Question 10: how well is the video streaming conveying the sense of presence in the meeting?

☐ Not at all ☐ Poorly ☐ Fairly ☐ Well ☐ Very well

Question 11: which approach for video compression would you prefer?

A: Good image quality with very low frame rate (that is the number of images displayed per second)

B: Poor image quality displayed at a higher frame rate (fluent video stream)

☐ A ☐ B

Free comments and feedbacks on the audio-video meeting experience

____

____

____

____

____

____

THANK YOU !