Optimized Video Conferencing System using Web Real Time Communication

Group Members

W. A. S. De Silva **13000241**

R. R. Liyanagamage **13001043**

K. A. I. Thiunuwan **13001221**

Supervised by

Mr. G. K. A. Dias

Dr. Kasun De Zoysa

Submitted in partial fulfillment of the requirements of the

B.Sc. (Hons) in Software Engineering 4th Year Project (SCS4123)

University of Colombo School of Computing

Sri Lanka

January 02, 2018

Declaration

We certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of our knowledge and belief, it does not contain any material previously published or written by another person or ourselves except where due reference is made in the text. We also here by give consent for our dissertation, if accepted, be made available for photocopying and for the title and abstract to be made available to outside organizations.

Candidate Name	**Signature of Candidate**	**Date**
W. A. S. De Silva		
R. R. Liyanagamage		
K. A. I. Thiunuwan		

This is to certify that this dissertation is based on the work of

Mr. W. A. S. De Silva

Mr. R. R. Liyanagamage

Mr. K. A. I. Thiunuwan

under my supervision. The dissertation has been prepared according to the format stipulated and is of the acceptable standard.

Supervisor' (s) Name: Mr. G. K. A. Dias

……

Signature of Supervisor(s) Date:

Declaration

We certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of our knowledge and belief, it does not contain any material previously published or written by another person or ourselves except where due reference is made in the text. We also here by give consent for our dissertation, if accepted, be made available for photocopying and for the title and abstract to be made available to outside organizations.

Candidate Name	**Signature of Candidate**	**Date**
W. A. S. De Silva		
R. R. Liyanagamage		
K. A. I. Thiunuwan		

This is to certify that this dissertation is based on the work of

Mr. W. A. S. De Silva

Mr. R. R. Liyanagamage

Mr. K. A. I. Thiunuwan

under my supervision. The dissertation has been prepared according to the format stipulated and is of the acceptable standard.

Supervisor' (s) Name: Dr. Kasun De Zoysa

……

Signature of Supervisor(s) Date:

Abstract

Video conferencing is gaining the attention of all internet users through rapid gains in performance over the past few years thanks to increasing computation power and better internet connectivity. It is easy to perform peer-to-peer video conferencing without the need of installing dying out proprietary plugins like Adobe Flash or applications, with the introduction of Web Real Time Communication (WebRTC). WebRTC is widely adopted for building up browser based video conferencing systems with the support of models like Multipoint Control Unit (MCU), Selective Forwarding Unit (SFU) and Mesh model.

The goal of this research project is to demonstrate how a video conferencing system can be built using SFU model and how the number of participants in a particular conference can be increased under the given conditions. The motivation for this is that the video conferencing systems that are available only supports a limited number of users per video conference. If a client wants to upgrade the number of participants more than the freely supported number a considerable fee is charged by the system. In order to tackle this problem two approaches have been explored: use audio mixing method and use changing the video quality method.

The solutions for the recognized problems have been addressed by using Jitsi as the main technology and Prosody, Nginx are used as servers. The main contributions of the system were the development of webinar mode using Jitsi media server, finding the possibility of increasing the number of participants for a video conference under the given conditions by using the selected approaches. The evaluation process of webinar mode was conducted by using two user groups: expert users and nominal users. Both nominal users and expert users have responded positively to the given solution.

Acknowledgement

We are grateful to the University of Colombo School of Computing for providing us the opportunity to conduct this final year research project under Software Engineering stream.

First, we would like to express our sincere gratitude to our project supervisor Mr. G. K. A. Dias for his tremendous encouragement and constant support throughout this research project. This thesis wouldn't be so steady without his valuable feedback and support.

We would also like to thank Dr. Chamath Kappetiyagama and Mr. H. D. L. Madhumal a Technical Lead at Thinkcube Systems (Pvt) Ltd for guiding us by sharing their expert knowledge on research areas with us, without their participation this research could not be completed successfully.

We would also like to acknowledge Mr. Buddhika Jayawardhana, CEO at Siplo (pvt) Ltd and his team for guiding us with their expert knowledge and helping us to evaluate the testing process of our system.

Finally our deepest gratitude goes to our dearest parents for their unconditional support, love and encouragement extended towards us throughout all the ups and downs in our lives, specially when we needed them the most. Last but not the least we as a team appreciate the team spirit of the colleagues for the collaboration and support which greatly affected the successful end result of this project.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

UDP - User Datagram Protocol

TCP - Transmission Control Protocol

TLS - Transport Layer Security

DTLS - Datagram Transport Layer Security

PSTN - public switched telephone network

SFU - Selective Forwarding Unit

MCU - Multipoint control unit

VoIP - Voice over Internet Protocol

DSI - Dominant Speaker Identification

CPU - Central Processing Unit

XMPP - Extensible Messaging and Presence Protocol

JVB - Jitsi Videobridge

JiCoFo - Jitsi Conference Focus

COLIBRI - Conferences with Lightweight Bridging

LAN - Local Area Network

WLAN - Wireless Local Area Network

RTP - Real-time Transport Protocol

IQ - Intelligence Quotient

IQR – Interquartile Range

Chapter 1: Introduction

1.1 Overview

Video conferencing is a convenient way of having communication between two parties in different locations, involving both audio and video. With the advancement of modern technology people move away from time wasting things and looking forward to use the easier ways of solving day-day problems. Using internet for the communication purpose is rather easier than using telecommunication network facilities like text-messaging and voice-calling (teleconference). The major difference between Video conferencing and Teleconferencing is, video conferencing is a live link communication which involves both audio and video streaming, while teleconferencing is a live link involving audio streaming only [1]. There are many definitions available for the term "video conferencing", but as Polycom white paper [2] says "*Video conferencing is an online meeting that takes place between two parties, where each participant can see an image of the other, and where both parties are able to speak and listen to the other participants in real time.*" The components that need to develop a basic video conferencing system can be classified like this. A microphone, webcam and a speaker, a display to convey the video conference, a software program that is responsible for capturing audio streams from the microphone, encodes them, transmits to the server and then to the each participant in the conference, and suitable audio and video codecs, a software program that is responsible for managing the exchange of audio and video streams between participants and a suitable model to control the behavior of receiving and transmitting audio and video streams [2].

Video conferencing systems can be categorized into two distinct categories according to the way of happening, point-point or multi-point. Point-point is the simplest way of having a video conference while multi-point needs some advanced procedure to maintain the conference mode. Conventional video conferencing systems lack the capabilities of adapting into the usage of CPU, memory, disk and bandwidth. Therefore the technologies used to enable video conferencing are constantly evolving with respect to the required quality, latency, security, cost and the scale size. Video conferencing is widely supported and available through many platforms and applications like Skype, Hangouts, and Zoom.

1.2 Problem Statement

Most of the available video conferencing systems are flash based ones (e.g. Skype, Hudlmo, Adobe Connect). The problems with flash-based systems are, flash has become deprecated now and people don't like to install additional plugins or frameworks when they are using some services. If someone is using a flash based video conferencing system, it is necessary to install adobe flash player and other required plugins in that relevant machine. Therefore it needs to find an alternative solution to replace the flash based system. With the advancement of modern technologies, people move forward to use web based solutions rather than using the installable software.

The other major problem with currently available systems is that, those systems are proprietary ones which support a limited amount of features for a free account. People who are looking for more features have to buy the premium version of those systems or software. The source code of proprietary software is not available for free and open source for the further development, thus the real users are restricted to participate in code level changes as the contribution for the development of the system.

A webinar is an online seminar, a presentation or a workshop that is broadcasting live through the internet. It is much popular among people to use online webinars than auditory seminars which hold in large halls from the participants' perspective as well as the webinar coordinators' perspective. Therefore develop a webinar mode with some additional features (screen sharing, file sharing) is a benefit for the people who are using these kinds of services regularly.

1.3 Aims and objectives

1.3.1 Goal of the project

The main goal of the project is to demonstrate how a video conferencing system can be built using SFU model and how the number of participants in a particular conference can be increased under the given conditions.

1.3.2 Objectives

Objectives of the project describes the outcome of the project. The following factors are the main objectives that are going to cover in this research project.

- Find an alternative solution to replace flash-based video conferencing systems.
- Develop a webinar mode with an additional feature called screen sharing.
- Use audio mixing method to maximize the number of participants per a conference
- Use changing video quality method to maximize the number of participants per a conference.

1.4 Research Questions

Research questions are based on the problems that are discussed in this research project. The primary objective of research questions is focusing on the issue that is going to cover in a particular problem.

- How to select a media server having a limited CPU consumption level, which can be an alternative solution to replace flash based systems?
- How changing bitrate in videos, affects increasing number of participants in a conference?
- How is audio multi-channel mixing affect the increase of number of participants in conference as well as reducing the echo problem?

1.5 Methodology

As mentioned in section 1.3 four main objectives are going to address throughout this thesis. A literature survey should be done to find out the suitable model that is going to address the first objective, then leave rest of the models. Developing a webinar mode is a process which will be implemented via modifying the meeting mode.

The main objective that is highlighted throughout this thesis is trying to maximize the number of participants per a conference under a given set of conditions. Finding out the reasons that are acting as the bottlenecks for achieving this objective is the initial step and the next one would be searching for ways that can be used to handle those bottlenecks. Transmitting video bit rate and the number of channels streaming from the server are two of the bottlenecks that can be recognized. Therefore it needs to find out what are the attributes that need to be modified

to change the bitrate of streaming video and the methods that can be used to change the relevant attributes. Reducing the number of audio streams transmitting through the media server could be gained by applying a mixing method for those streams. Therefore it needs to be found out the ways of mixing audio streams together into one single stream and evaluate the process.

1.6 Challenges

WebRTC is still under development stage and there are a very few research papers and white papers are available. Lack of available testing tools is another major problem that has to be considered with WebRTC and the limitation in Bandwidth and the packet loss (Jitter) causes to create connectivity issues while having a conference.

1.7 Outline of thesis

Chapter 2 discusses about the background description of the research and the related technologies, WebRTC architectures, related work, similar systems and mainly about Jitsi media server. In chapter 3 problem analysis, the approaches which are related to the problems and the justification for the selection of available solutions are described. A detailed description on how the problems have been resolved with the selected solutions is discussed in chapter 4. Chapter 5 includes experimental tests and usability evaluation of webinar mode. This thesis concludes in chapter 6 where all findings are summarized and recommendations are provided.

Chapter 2: Background Study

This section includes about the background description of the research and the related work regarding the work done. When using video conferencing for the communication purpose there are several options available to have a video conference.

1. A pre-packaged application
2. A cloud service
3. A full blown unified communication platform
4. A Web Real Time Communication (WebRTC)

Options	Pre-packaged applications	Cloud services	Unified communication platforms	WebRTC
Description	Designed to be standalone services	Being a total video collaboration service delivered from the cloud	Framework for integrating various asynchronous and real-time communication tools, with the goal of enhancing business communication, collaboration.	A free, open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs
Technologies	Skype	Go-to-Meeting, lifesize	Microsoft Lync, Avaya, Siemens, Jabber, Zoom	Talky.io, Appear.in, AppRTC, Bluejeans
Benefits	Already buildup systems are available, Supports more user attraction,	High scalability and mobility, Accommodating growth and expansion	Provides high robustness and reliability with in the usage area.	Supports real time communication Plugin free, Supports file and desktop sharing, Easy to integrate and deploy, Supports more secure platform, Cost effective, Free and open-source
Drawbacks	Systems are not designed for integration	High implementation cost and	Expensive to implement, Hard to integrate, Need to	Still under development stage,

	purpose, High cost and need to buy to enable full features.	maintenance cost is high,	purchase both software and hardware, Require specialized knowledge on this area	Incertitude about future codecs

Table 2. 1: Comparison among the available options for creating a video conference

2.1 Motivation to WebRTC

Web Real-Time Communication (WebRTC) is an effort that was started in 2011 [3] to enable direct real-time communication between two browsers without needing to install any browser plug-ins or platform-specific applications.

Neither does WebRTC place any restrictions on the type of devices like computers, mobile phones nor TVs. Usually applications in a browser communicate with a web server to receive and send data, but when using WebRTC a direct communication channel between two peers is opened without using any servers for the transmission of data.

2.1.1 Usage of WebRTC

WebRTC exposes ECMAScript[1] APIs that allow a participant to use a web application that retrieves audio and video streams from cameras or microphones and also it allows to establish a peer-to-peer connection using standardized protocols between two compatible browsers.

The current standard defines different types of WebRTC-compatible devices [4]. The most important ones are:

[1] JavaScript is one implementation of [ECMAScript] as defined by ECMA-262

First a WebRTC device, it conforms to the protocol specifications, meaning it supports the needed UDP based protocols for the peer connection and the TCP based ones for the data channel, including encryption for both packet types using TLS and DTLS.

A WebRTC browser is the second type, which is a WebRTC device that also supports the full ECMAScript API.

The third one is a WebRTC gateway, which is a WebRTC device that mediates media traffic to non-WebRTC devices and may not conform to all protocol specifications.

A WebRTC gateway might be used to enable audio-only communication between a PSTN telephone and a WebRTC browser or one-way communication between a participant using a WebRTC browser and another device that only shows the first participants stream, but cannot send anything on its own.

2.1.2 Transport Protocol

Transmission Control Protocol (TCP) is used to transport information between the browser and server (e.g. HTTP/HTTPS). TCP is known for its reliability and error-correction which ensures packets get delivered in the same order in which they are sent with retransmissions when packet loss occurs. In WebRTC speed is preferred over reliability. WebRTC transmits audio, video, and data between browsers over the User Datagram Protocol (UDP) [5]. WebRTC over TCP is also possible and use as a last option, when all UDP ports are blocked which can be the case in heavily shielded enterprise networks.

Packets are sent to the recipient with UDP without knowing whether they arrive or not, because there are no retransmissions, congestion control or acknowledgements, UDP is noticeably faster than TCP [5]. UDP is chosen for WebRTC because low latency and high throughput are more important than reliability.

2.2 Multipoint conferencing with WebRTC

Several architectures have proven usable to achieve multipoint communication. The communication format of WebRTC is roughly divided into two types: one that solves encryption and one that does not involve a server that interprets media [6].

(I) Solving encryption or not interpreting media via a server

- P2P - peer to peer

(II) Through a server that deciphers encryption or interprets media

- Through **MCU (Multipoint control unit)**
- Via **SFU (Selective Forwarding Unit)**

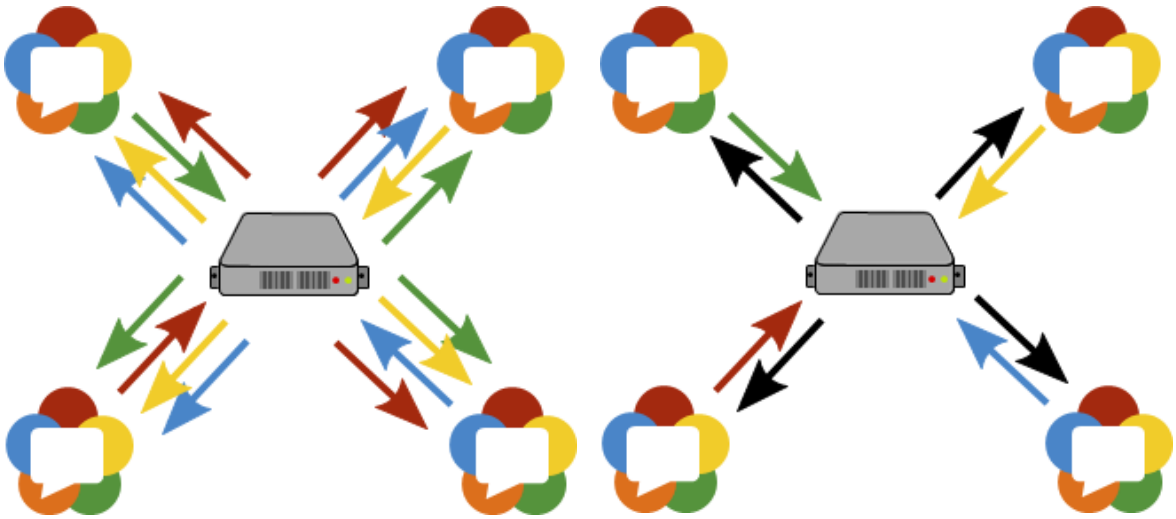


Figure 2. 1: SFU Model (Left) and MCU Model (Right) [7]

As WebRTC is a technology that is used to create direct peer-to-peer connections between two entities, the most natural way of connecting multiple participants would be to use SFU or MCU network as displayed in Figure 2.1 for a four-way call.

There are three main models of deploying a multiparty video conference.

2.2.1 Mesh

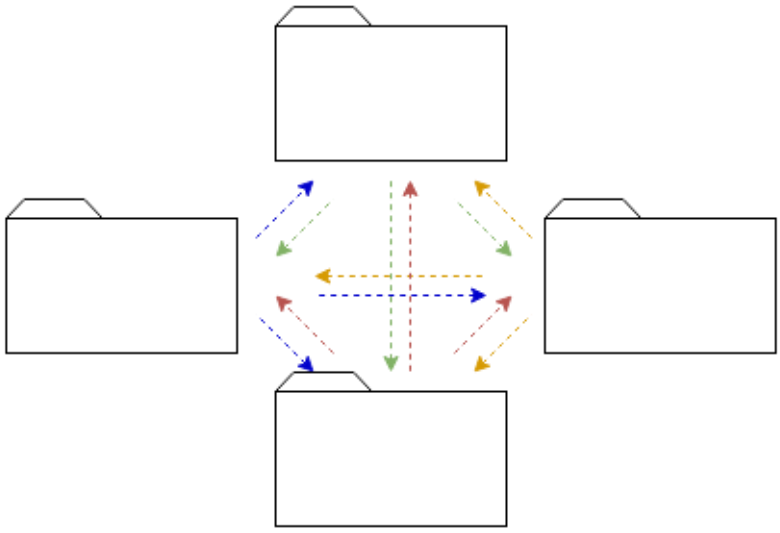


Figure 2. 2: Mesh Architecture [8]

In mesh topology each user will need to send its media to all other users in the session as well as receive all the media streams from them. A full mesh network, every peer establishes a connection with every other peer in the network, hence there are n*(n-1) number of connections where n is the number of peers as shown in Figure 2.2. (For example, a full mesh network with 4 users has 12 connections) [9].

No servers are needed for this to work, which makes this an inexpensive option to use. The downside with this is that as the number of participants increases, a lot more bandwidth and CPU processing will be needed. As a result, this architecture is unsuitable for a large network.

2.2.2 Multipoint Control Unit (MCU)

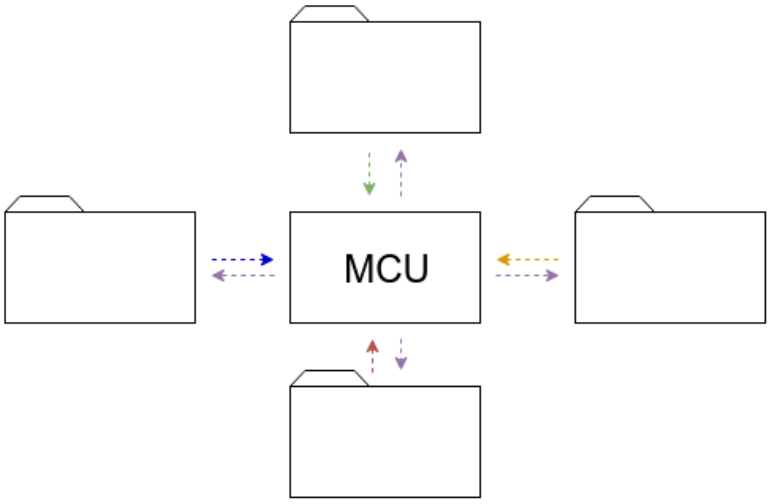


Figure 2. 3: MCU Architecture [8]

MCU means that each browser sends a single video stream. The MCU takes all these video streams and composes them into a single video stream that is then sent to each participant separately as shown in Figure 2.3.

The concept of video mixing is that of creating composite images. In other words, if users A, B, C, and D were ready to participate in a mixed video conferencing call, then they would each start a regular one-to-one session with the mixer and send their video streams to it as usual. In return, they would receive a single video stream that would happen to contain everyone else's content even if a little scaled down (Figure 2.4).

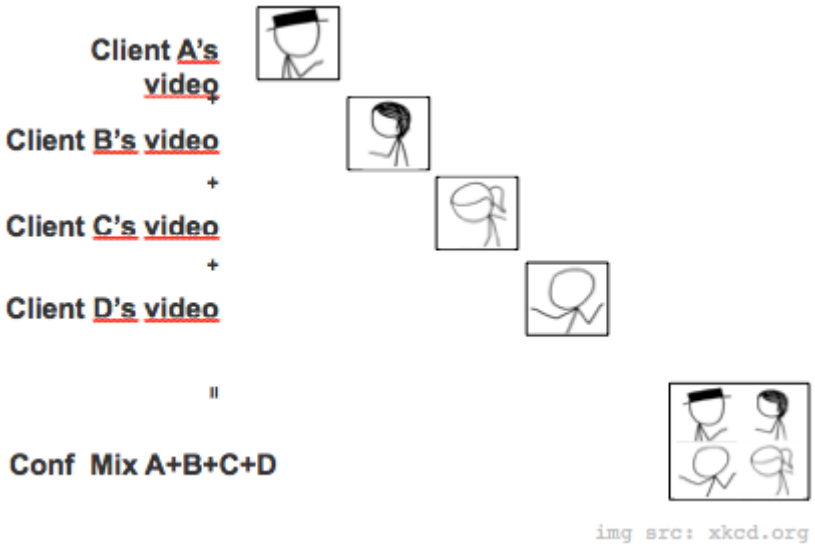


Figure 2. 4: Video Mixing [10]

One needs to decode all incoming frames (one per participant), scale down each one of them, create composite images and then re-encode them once again when performing video mixing. In addition to the cost of processing resources, video content mixing also implies substantial compromises in terms of quality and usability and every single frame received by a participant has undergone lossy encoding twice, rather than once. Images are scaled down. The video layout is fixed. Also, central content mixing is bound to add at least 200ms of latency [10]. MCU was the famous architecture before SFU model was introduced.

2.2.3 Selective Forwarding Unit (SFU)

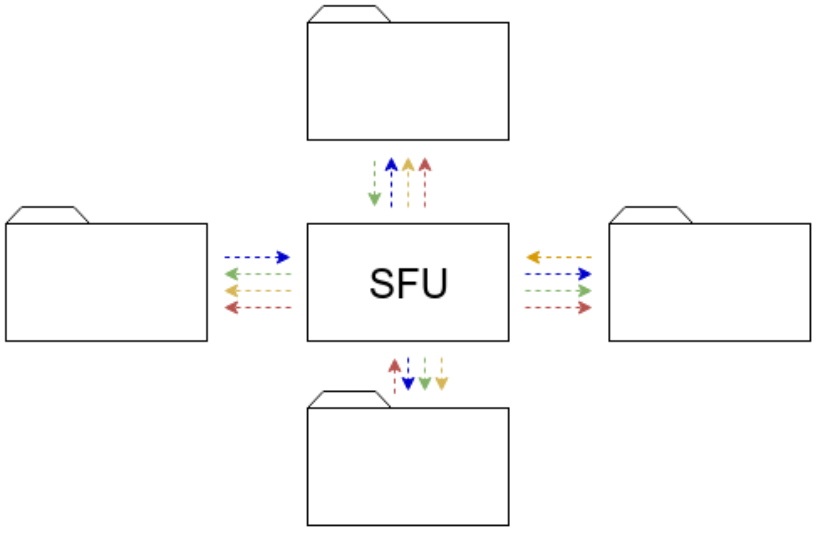


Figure 2. 5: SFU Architecture [8]

Time went by, the Internet evolved, bandwidth prices dropped and the game changed. After broadband becoming a commodity, downloading a stream of three to five megabits per second was no longer a problem and an alternative video conferencing architecture quickly became obvious. This is the time SFU architectures begin to evolve.

In SFU design each participant forwards his media stream to the central unit, and it relays the suitable streams to all participants without mixing the streams as shown in Figure 2.5. An SFU model is capable of receiving multiple media streams and then decide which of these media streams should be sent to which participants. The advantages of this approach are various. Users can render them anyway or they can choose their receivers by using these separate streams. Quality is better as video streams have undergone encoding only once. Latency is not increased by the additional encodings, scaling, and decoding [10]. If clients have a low bandwidth connection, then SFU might not be a good design because it consumes a lot of bandwidth for streaming.

As discussed, no topology is perfect, and each come with distinct advantages as well as disadvantages. Multipoint Control Unit architectures are ideal for when computation and bandwidth are limited and there is a need for interoperability with disparate networks, but come at the cost of high server load and limiting video layout. On the other hand, Selective Forwarding Unit topologies are ideal for high server performance and maximum flexibility for the client UI but come at the cost of requiring all connecting clients to share the same codec, frame-rate, and resolution profile. The tough decision is which to use for your application. Launching an application into real-world scenarios requires on-demand access to both the capabilities of an MCU and the capabilities of an SFU. This complementing feature set has covered the way for a new, next-generation hybrid-SFU/MCU architecture [11]. Figure 2.6 shows the comparison of above discussed three architectures.

	Server		Client		Advanced Functionality			
Conference size = N	CPU encoders: decoders	Bandwidth uplink: downlink	CPU encoders: decoders	Bandwidth uplink: downlink	Video layout	Latency	Record/ playback	Transcoding
Peer-to-peer mesh	NA	NA	High (N:N)	High (N:N)	Client decides	Lowest	Client-side	No
Multipoint Control Unit (MCU)	High (N:N)	High (N:N)	Normal (1:1)	Normal (1:1)	Server decides	Acceptable	Server-side	Yes
Selective Forwarding Unit (SFU)	Moderate (1:N)	Moderate (1:N)	Medium (1:N)	Medium (1:N)	Client decides	Low	Server-side	No

Figure 2. 6: Comparison of Architectures [11]

2.3 Media Server

Conceptually, a WebRTC media server is just a kind of “multimedia middleware" where media traffic pass through when moving from source to destinations. Media servers are capable of preparing media streams and offering different types including group communications, mixing (transforming several incoming streams into one single composite stream), transcoding (adapting codecs and formats between incompatible clients), recording (storing in a persistent way the media exchanged among peers), etc. In above MCU, SFU architectures media server act as the central unit [12].

The main purpose of media servers is handling of many media streams while increasing the delay of the stream as little as possible and media server act as a WebRTC device and create a peer connection a WebRTC browser.

2.3.1 Comparison of existing media server solutions

Many different possibilities were evaluated and a short listing of that evaluation is located in Appendix B: Comparison of available Media servers. The conclusion that can be obtained by referring appendix B, C is Janus and Jitsi are popular and related to the problems that are covered in this research project under SFU based systems and Kurento is good under MCU model-based systems.

2.4 Preliminaries in Jitsi (The main components of Jitsi)

Jitsi is a collection of Open-Source projects which have been developed aiming to provide highly secured video conferencing solutions [13]. There are many features available in Jitsi such as audio and video conferencing, supports recording and simulcasting. Apart from the given features, there are some specialties in Jitsi with compared to other video conferencing solutions.

- Jitsi has three main components named as Jitsi Videobridge (JVB), Jitsi Meet and JiCoFo (Jitsi Conference Focus) [13]. JVB passes everyone’s video and audio streams to every participants without mixing them together (SFU model)
- Jitsi is based on WebRTC solutions.
- Jitsi supports some advanced video conferencing techniques like simulcasting, bandwidth estimations and scalable video recording.

2.4.1 Jitsi Meet

Jitsi Meet is the front-end Open-Source application of Jitsi, which provides large-scale video conferencing by using a WebRTC supported web browser developed by Atlassian. The base of the Jitsi Meet is JavaScript and the responsibilities of Jitsi Meet can be categorized as below.

- Creating a suitable layer for XMPP signaling.
- Initiating the connection with the relevant peer.
- Accepting messages, requests, files and media components.

Jitsi Meet is available as a mobile application and there are no artificial restrictions on the number of participants in a conference. The only factors that should be considered are the server power and the bandwidth usage [14].

The conferencing process of Jitsi Meet is handled by Jitsi Video Bridge (JVB) and also Jitsi Meet provides a very flexible way of embedding it into external applications by using Jitsi Meet API [15].

2.4.2 Jitsi Videobridge

Jitsi Videobridge commonly known as JVB is the heart of Jitsi which acts as a media server of Jitsi infrastructure. The media distribution method of JVB is SFU model which does not mix all the streams coming from participants into a single composite stream (Both audio and video). But JVB relays the received audio and video channels to all the participants in the meeting [16]. There are many more features available in JVB such as featuring call encryption with DTLS/SRTP, provides more scalability, higher flexibility, easy to control through XMPP or through an HTTPS connection.

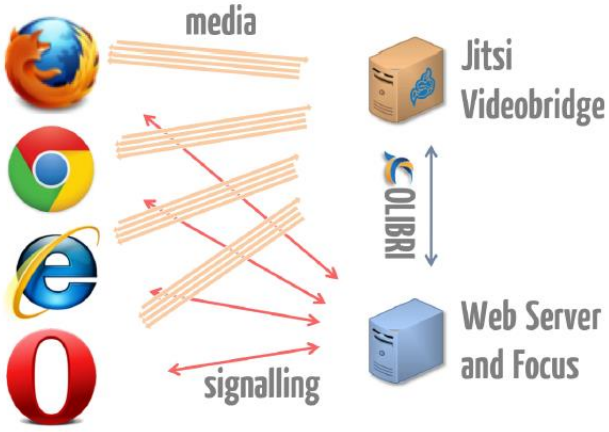


Figure 2. 7: JVB with WebRTC [10]

2.4.3 JiCoFo (Jitsi Conference Focus)

Jitsi Conference Focus is a server-side focus component that used in Jitsi Meet conferences that manage media sessions between each of the participants in a meeting and the video-bridge. The module works in mainly room and member management in Jitsi. JiCoFo has some responsibilities like

- Managing conferences by deciding who joins the room and who leaves the room.
- JiCoFo is the load balancing unit belongs to JVB.
- Managing Colibri channels for the participants in a meeting and establishing media flow to and from JVB [17].
- JiCoFo works as the central signaling component of the system.

2.4.4 XMPP (Extensible Messaging and Presence Protocol)

XMPP is a communications protocol for message-oriented middleware based on XML. XMPP gives a general framework for messaging across a network. It is pretty much the same piece of technology as the one Google uses for Hangouts [18]. XMPP is defined in an open standard and uses an open systems approach of development and application, by which anyone may implement an XMPP service and interoperate with other organizations' implementations. There are a large number of XMPP servers available, from proprietary to open source, with communities of varying sizes [19].

- **Openfire** - A hugely popular Java-based server with a large community supporting a large number of plugins and that is actively developed.
- **Tigase** - A popular Java-based server with active development and a great community.
- **MongooseIM** - An Erlang-based server forked from a previous XMPP server implementation and actively developed by Erlang Solutions.
- **Prosody** - A fast and resource-light Lua-based server with a great core development team and an active community.

2.4.4.1 Prosody

Prosody is a modern XMPP communication server. Prosody server is the default communication server used in Jitsi. Some useful improvement can be done by changing prosody configurations in Jitsi system like authorization creations, creating specific user accounts. Prosody is based on Lua language, Lua is a powerful, efficient, lightweight,

embeddable scripting language. It supports procedural programming, object-oriented programming, functional programming, data-driven programming, and data description [20].

2.5 Related Work

This section includes the facts already discussed in other research papers which are technically relates to the areas that are covered by this research project.

1. Relevance-Based Selectivity for Forwarding Video in Multimedia Conferences [21]

The proposed system is based on the technology called Jitsi. Jitsi Videobridge is an SFU implementation. The main intention of using SFU model is to maximize the number of participants in a meeting by minimizing the CPU consumption level. In Jitsi there is a special technique called Last N, which orders endpoints according to their audio activity by using an algorithm for dominant speaker identification adapted to work solely with audio-level information and which operates without decoding the audio streams. This allows the Last N scheme to be used in the context of an SFU. SFUs enabling the Last N scheme perform Dominant Speaker Identification (DSI) and traditionally DSI is performed using raw audio streams [22], but an SFU (e.g., Jitsi Videobridge) forwards audio streams without decoding. Last N feature is tested by using Jitsi Video Bridge [23]. Figure 2.8 illustrates the last N feature.

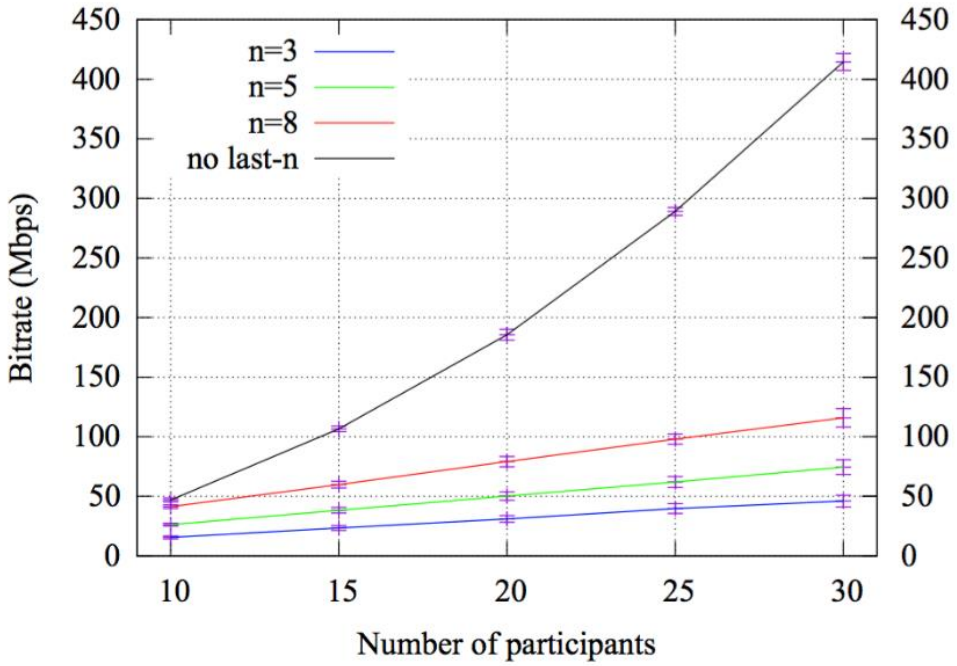


Figure 2. 8: Change of bit rate according to the number of participants in last-N [21]

2. WebRTC multipoint conferencing with recording, using a Media Server [7]

The system uses Jitsi media server for the implementation purpose, but there are few other media servers available which have been used to develop video conferencing systems, such as Kurento [24]. Kurento media server uses MCU as the core model and the media server supports some valuable features like group communication, recording and playing videos, media filters for augmented reality and computer vision filters. Since MCU handles mixing mode for both audio and video, CPU consumption is much higher than the SFU approach.

When the video conferencing is happening within the same network, there is no need of providing a public IP address. WebRTC supports two kinds of servers called STUN and TURN which are used for streaming purpose. If the STUN server uses the same IP address as the media server, it is even more unlikely that a TURN server is needed further. But the best option is to use a server that is both a TURN and a media server [7].

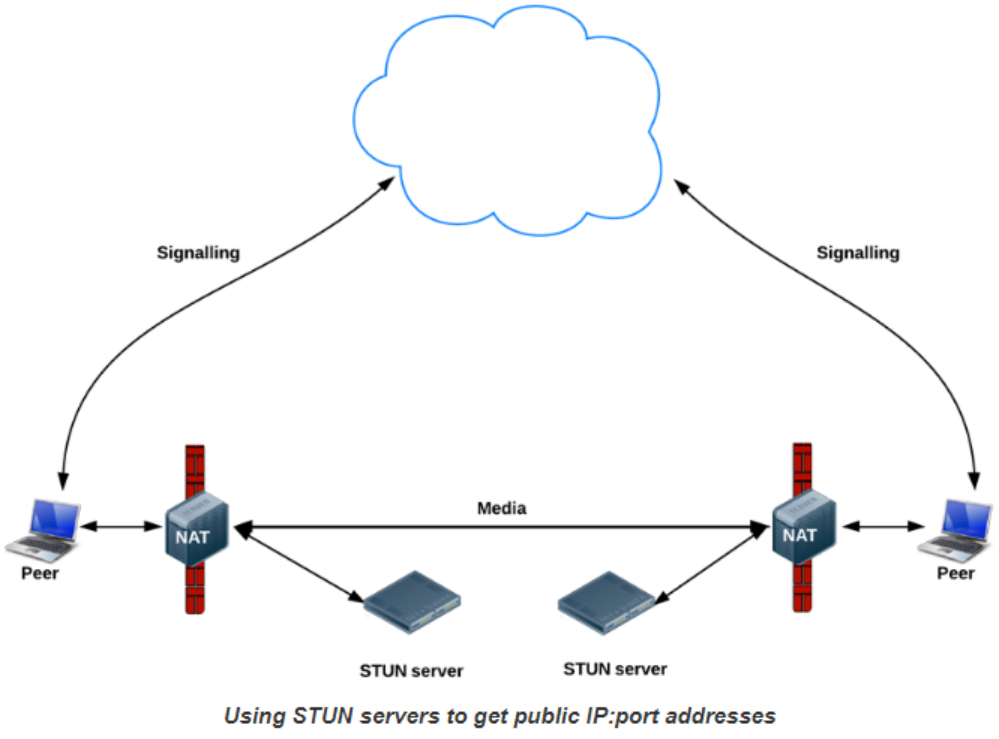


Figure 2. 9: Using STUN servers to get public IP: port addresses [25]

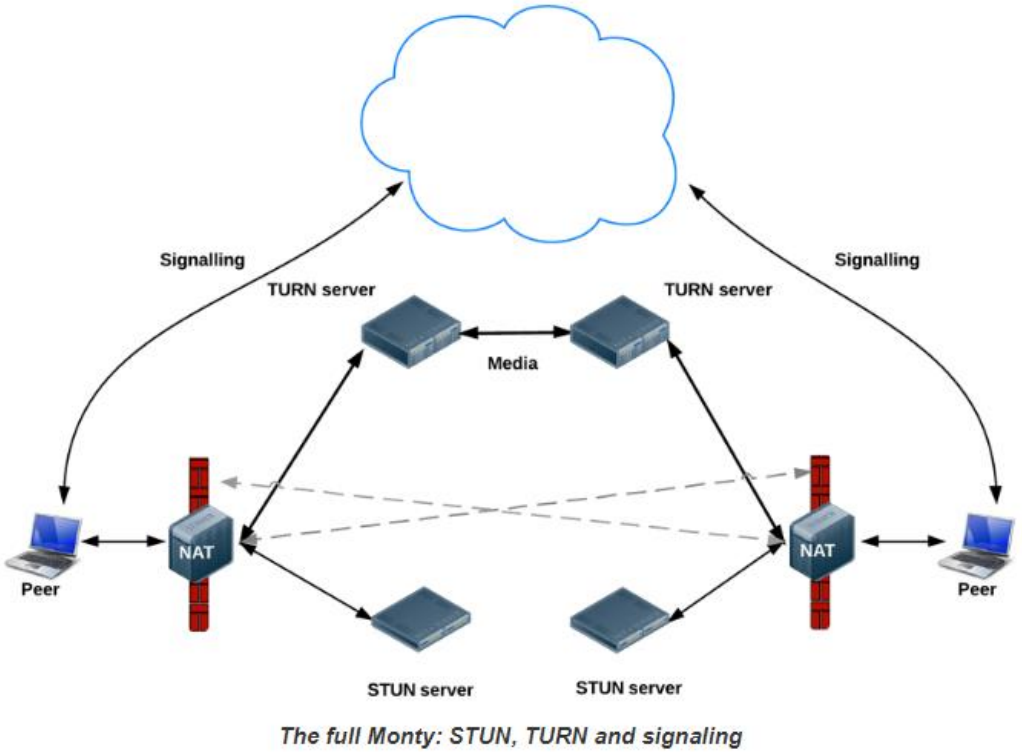


Figure 2. 10: The full Monty: STUN. TURN and signaling [25]

2.6 Similar Systems

This section includes a brief introduction about the systems which are related with the system that has been proposed by this research project. The following sections describes about the available systems like AppRTC, Appear.in.

2.6.1 AppRTC

AppRTC is a video conferencing system which was originally developed by Google. It allowed for two conference participants and optimized for quick session setup [26]. Running AppRTC locally requires Google app engine SDK for python, nodeJS, and Grunt. AppRTC integration is available as a free and open source code which supports both UNIX and Windows platforms. But there are some cons in AppRTC such as a need to pre-install nodeJS and Grunt for the integration of AppRTC for selected applications, background noise while having a video conference is a serious matter that should have to handle [27].

2.6.2 Appear.in

Appear.in is a video collaboration tool which uses WebRTC as the core technology and supports video conferencing up to 8 participants. It supports screen sharing which allows

showing presentations, videos and photos [28]. Appear.in supports interactive Trello boards directly while having video conferencing. When someone claims a room, need to register by entering a valid email address or phone number and enter a display name to be used in the service [29]. Chat room owner has permissions to modify the room structure at any time and also can unblock users that have been blocked, kick out users from the relevant conversation. But there are some significant cons can be seen in Appear.in compared with other available systems. The number of participants in a free account is limited to 8 participants and have to spend a lot of money for a premium account, Echo and the noise problem should be more considerable than other available systems, Every individual user has to type the URL of the conversation in a browser window to join in to the conversation (This is not a good approach when the number of participants are increasing) [30].

But as the pros of the Appear.in system, there is no need of installation of additional plugins or frameworks to work with Appear.in, provides high security(with the locked room facility and encrypted messages) and the chat messages will be purged when all participants leave the room, no transcripts are saved on appear.in servers.

2.6.3 Skype

Skype is a VoIP service which uses internet to allow people to communicate using free audio and video calls. It is a proprietary software that supports up to five users for a particular video conference under a free version. But if a user wanted to maximize the number of participants in a video conference, he/she has to upgrade the current free version of software into the premium version. Premium version supports up to 25 participants in a single video conference. Skype provides a desktop version of software which works fine in every platform like Windows/Unix and also it supports a mobile application.

Skype supports a good audio quality by using its own codecs for the selected number of participants. But the free version doesn't allow to connect more than predefined number of users. This becomes a significant issue that can be seen in Skype, also it needs to be installed in the device which is going to be used for video conferencing [31]. Skype needs a good bandwidth coverage for video conferencing, so the Skype developers recommend to close the applications that use internet especially those playing music or video and cancel any file transferring process before start a video conversation using Skype [32].

2.6.4 Google Hangouts (Google Talk)

Google Hangouts supports audio and video conferencing within Google Apps domain. It enables the users to form teams, enhance teamwork and also improve teamwork. But Google Hangouts supports only up to ten (10) users per a single video conference under the free edition and twenty five (25) participants under a premium account [33]. The basement of the Google Hangouts is depended on WebRTC, but it uses a completely different platform called "Vidyo" for the development process [34]. All the mobile devices support Hangouts mobile application and it should be installed in a device to use for a video conference.

2.6.5 Comparison between similar systems

Table 2.2 summarize the available systems features that described above.

AppRTC	Appear.in	Skype	Google Hangouts
Needs NodeJS, Python and Grunt to run locally.	No need to install additional plugins or frameworks to work.	Need to install Skype software to use the service.	Need to have a gmail account to use the service.
Need to install AppRTC mobile app to use the service by mobiles.	No need to install additional app for using the service.	Need to install skype mobile app to use the service by mobiles.	Need to install Google Hangouts app to use the service by mobiles
Supports up to 2 participants in a conference.	Supports upto 8 participants in a meeting in a free account.	Supports up to 5 participants in a meeting in a free account.	Supports up to 10 participants in a meeting for a free account.
A free software.	A free system	A proprietary software.	A proprietary software.
Do not support screen sharing	Supports screen sharing	Supports screen sharing	Supports screen sharing

Table 2. 2: Comparison between similar systems

2.7 Summary

Chapter 2 includes the background study of the proposed system. The main technologies, mechanisms, related work, similar systems and the preliminaries in Jitsi are the topics that are covered under this chapter. The main technologies chapter includes the technologies that are going to use in the development process of the research project (WebRTC, Prosody, and Jitsi). Media servers section is related to the description of available media servers like SFU, MCU and Mesh models. Descriptions of similar systems as Skype, Google Hangouts, Appear.in and AppRTC are discussed in similar systems section. The main components in Jitsi are explained under the chapter Preliminaries in Jitsi.

Chapter 3: Analysis and Design

3.1 Problem Analysis

Video conferencing is a major form of communication media in modern world. There are plenty of free and proprietary video conferencing systems available with many features. But as the requirements of users vary from user to user, the intention of the proposed system is to provide solutions for the problems like given below.

3.1.1 Alternative solutions for a Flash based video conferencing systems

Flash-based video conferencing systems have been developed with the requirement of installation of Flash player on a particular computer and other Flash plugins on web browsers. However, in the past, the popularity of these systems was in a higher scale due to the ease of use and web conferencing systems offer enhanced interaction capabilities like embedded multipoint and conference recording in both audio and video. But lately the Flash-based systems became deprecated, and people began to find alternative solutions to replace the flash-based systems.

3.1.1.1 Approaches

There are few alternative solutions available to replace Flash-based video conferencing systems such as WebRTC based video conferencing systems, Pre-packaged applications like Skype, Unified communication platforms like Microsoft Lync and some of the cloud-based systems like ezTalks [35]. But the cloud services like Adobe connect are flash-based systems [36]. The proposed solution for this is to use a WebRTC based video conferencing techniques.

3.1.1.2 Justification

System	Performance and utility features
WebRTC based systems	• Does not need to install additional plugins or frameworks to work with specific system. • Supported by every browser including Safari for iOS 11 [37]. • Supports real time communication [38]. • Supports newest audio and video codecs like OPUS, VP8, VP9 [38] • Needs least amount of resources for a video conference [38] • Supports high security, ease of integration and deploy [38]

Pre-packaged applications	• Most of the pre-packaged softwares are proprietary ones. • Does not support integration into external applications. • Supports new audio and video codecs such as OPUS and VP8.
Unified communication platforms	• Provides good time efficiency, needless effort to build up applications. • Hard to integrate into external applications

Table 3. 1: Designs of Video Conferencing Systems

Apart from the comparison shown in Table 3.1**,** there is another popular video conferencing system is available called Zoom. But Zoom has some significant weaknesses like

- Needs to install a launcher application called Zoom Launcher for every user.
- Limited amount of features are available under the free version, needs to upgrade into premium version to experience full benefits.

The proposed system is going to use WebRTC as the core technology to develop a video conferencing system since it has much suitability to replace the flash-based system with compared to other available options.

3.1.2 Find suitable architecture with a lower CPU consumption level

CPU consumption level is a critical topic that has to be concerned while developing a video conferencing system. Therefore it needs to find an appropriate way of minimizing CPU consumption level for the better performance of a particular system [39].

3.1.2.1 Approaches

Apparently, there are few architecture models available in WebRTC which are suitable for developing a video conferencing system, Mesh model, MCU model and SFU model. Mesh is a model that is used basically for the implementation of peer-peer architecture while MCU model is used for mixing all the channels into one single channel and stream towards every participant. Kurento is one such example for MCU model. SFU model does not mix all the channels together into a one single channel but streams all the channels separately towards each participant in a conference. Jitsi media server is an example of the SFU model.

3.1.2.2 Justification

As the comparison shown in the Figure 2.6 between Mesh, MCU and SFU models, SFU is the most suitable approach for minimizing the CPU consumption level. MCU model requires more CPU power as it uses a different processing method for the mixing procedure. The following graphs show the comparison between resource usage of SFU and MCU models.

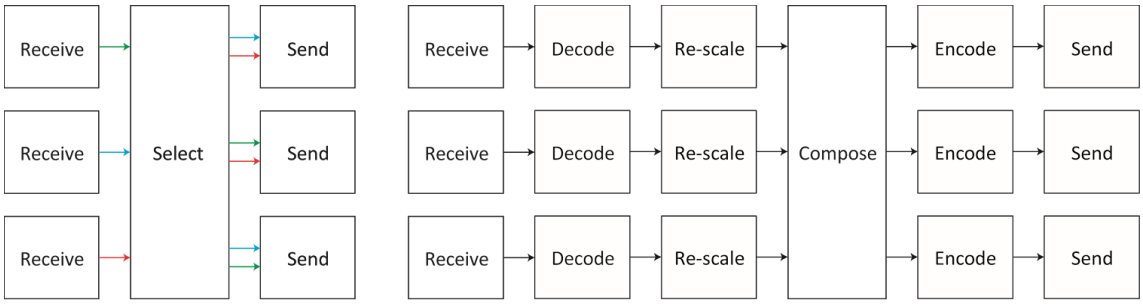


Figure 3. 1: SFU processing (Left) and MCU processing (Right) [5]

Figure 3.1 illustrate the processes of SFU and MCU models.

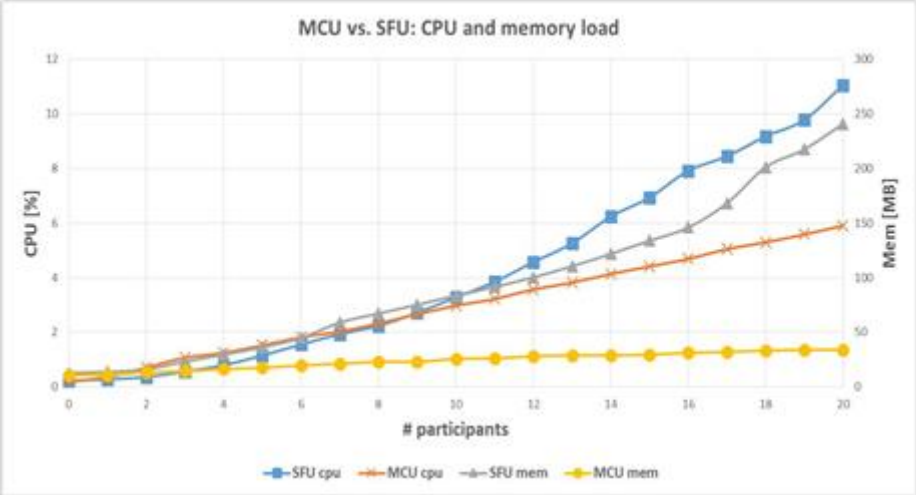


Figure 3. 2: MCU vs. SFU (CPU and memory) [40]

Figure 3.2 illustrates the memory consumption of SFU and MCU models. The comparison concludes that SFU model needs more memory than SFU model.

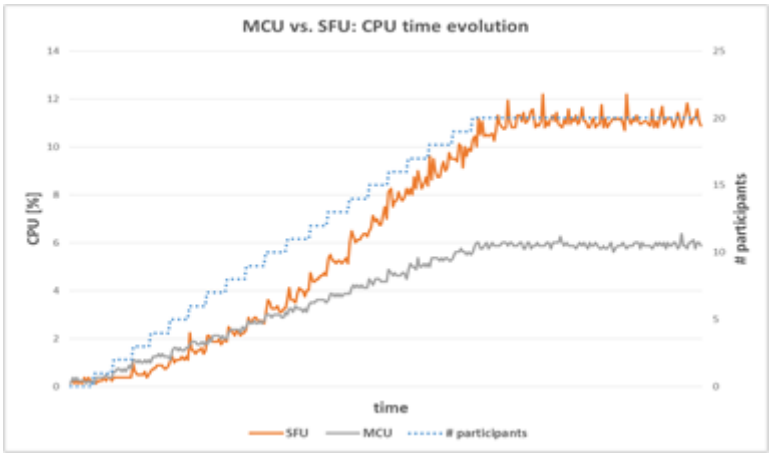


Figure 3. 3: MCU vs. SFU (CPU time evolution)

According to the Figure 3.3 MCU model consumes more CPU power than SFU model. Therefore SFU model is suitable for the available conditions mentioned in section 4.4.3.1 Conditions for the test

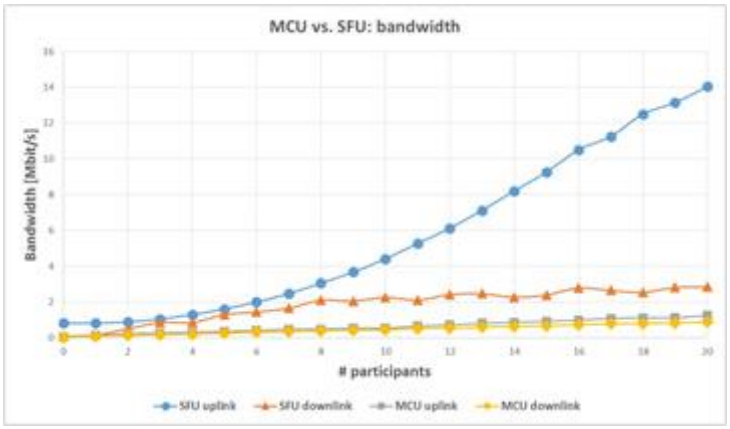


Figure 3. 4: MCU vs. SFU (Bandwidth) [40]

As shown in Figure 3.4, the bandwidth consumption of SFU model is greater than MCU model.

3.1.3 Develop a webinar mode

Webinars play a major role in this era as well as the meeting mode. Webinar tends to be the medium of choice for professional events to deliver presentations to large audiences. A webinar is a live web-based video conference that uses the internet to connect the individual hosting the webinar to an audience, the viewers and listeners of the webinar from all over the world. Hosts can show themselves speaking, switch to their computer screens for slideshows or demonstrations, and even invite guests from other locations to co-host the webinar with them.

Professionals use webinars to give educational presentations associated with their businesses and connect with their audiences in a much closer way. It could be a webinar where one person simply hosts a lecture or seminar to teach something or it could be a promotional presentation to sell a product, or it could be both. Webinars are also helpful tools for conducting live interviews with other professionals, which are often compelling aspects that draw more people in to attend webinars.

3.1.3.1 Approaches

There are two ways to perform the webinar mode.

1. By Streaming video into YouTube and share the live streaming link to participants.
2. Built the webinar mode inside the proposed system by modifying the meeting mode.

3.1.3.2 Justification

The second approach is suitable for required system and live streaming is an additional option for most systems. Jitsi already uses the first feature for webinars but does not provide an inbuilt system for webinars. The problem of YouTube streaming is, the video conference is propagating a delayed video due to the time that needs to upload first into YouTube.

3.1.4 Increase the number of participants

The proposed video conferencing system expects to maximize the number of participants in a single conference without decreasing the video quality less than 360p (The conditions that apply for this scenario is mentioned in section 4.4.3.1 Conditions for the test)Many techniques that have been used by different conferencing systems to maximize the number of participants more than 10(Hangouts supports 10 participants in free version), but those systems are commercial systems which allows a limited number of connections per a meeting. People have to buy the premium version of such products to experience the full benefits.

3.1.4.1 Approaches

Few approaches that are being used by different video conferencing systems to maximize the number of participants in a meeting. Those methods can be divided into two main categories.

3.1.4.1- (I) Hardware-based

Increasing hardware resources is one such option that can be used to achieve the required goal, but there are some limitations in this solution due to the factors like cost, connection issues. In the proposed solution the intention is to focus on solutions which are related with a software-based approach [41].

3.1.4.1- (II) Software-based

The software-based approach is the most popular and common way of increasing the number of participants. There are three main approaches under this category as enable Last- N feature, use audio channel mixing method and use changing the video quality method. Each of these methods is described below.

LastN

Last N feature is available in SFU model to choose only a subset of streams to forward at any time and the SFU model forwards a fixed number of a set of video streams to each endpoint in a conference and controls the set of forwarded audio and video streams dynamically according to the audio activity that happens [21]. Last N scheme only applies to the forwarding video streams and not for the audio streams; all the audio streams come from each endpoint are always forwarded. According to a user experience perspective, a video which is transmitted from only a subset of all endpoints is displayed, but if the Last N method is used as soon as a member of the video conference starts to speak, their video is going to display automatically. In order to use this method, the SFU model has to identify the dominant speaker first and then it maintains a list of all the endpoints in the conference order by the time that an endpoint is identified by the dominant speaker. The endpoint currently identified as the dominant speaker by the SFU model always comes to the top of the list. There is a specific algorithm available for the dominant speaker identification (DSI) [21]. The implementation of this algorithm needs the speech activities of each endpoint from time intervals of different lengths.

Changing the video quality

Video conferencing is a very bandwidth sensitive way of communication [42]. Therefore if the available bandwidth is too much lesser than expected, packets will be lost in the network and the whole conference will be disrupted [43]. Changing the bitrate of a streaming video is an alternative solution that can be used to optimize the bandwidth usage and have a good quality in the video stream as much as possible during a video conference. This method is a conceptual mathematical model that is depended on the frame rate and the spatial resolution of the video

[42]. The second method is applying an algorithm like Kush-Guage Algorithm. All these methods intend to minimize the bandwidth consumption level of down video streams for each participant in the conference.

Audio channels mixing

The mixing procedure of channels together into a single channel needs a higher level of CPU power than streaming those channels separately. This is the approach which is used in MCU model for both audio and video channels. The mixing method is suitable for a conference which holds under a limited bandwidth and this is a kind of solution for the procedure of maximizing the number of participants in a conference by optimizing the available bandwidth. The output channel of mixing mode needs a lesser bandwidth consumption compared to the total bandwidth consumption of sending each channel separately [41]. This proposed method intends to mix all the audio channels and relays all the video channels as they are to save the bandwidth consumption aiming to connect more people to the conference (Without losing the essential quality of remaining audio and video streams). There are four types of audio mixing methods available.

- End-point mixing - All audio streams are delivered from the audio source to the participants without mixing or modifying them together. A recipient might get more than one audio stream at a time by this method and those streams can be mixed and played. The disadvantage of this approach is the necessity of more bandwidth to transmit audio streams separately.
- Distributed mixing - This approach is as same as the endpoint mixing method except it handles audio streams automatically in the server side. There is only one mixer for each participant in the conference and therefore every participant will receive exactly one audio stream. This mechanism is good to reduce the bandwidth requirement for the streams transmission and the advantage is, the users are given an opportunity to select an option to choose the audio format they prefer. However, distributed mixing is more flexible than endpoint mixing [42].
- Hybrid Model - This is a combination of endpoint mixing and distributed mixing. Some participants have dedicated mixing mechanism for them, and some participants will receive audio streams separately as they are transmitted directly from the server [42].
- Centralized mixing - In this mechanism, there is only one audio mixer in the server which is used to mix all the audio streams and send the output audio to all the

participants in the meeting. Central mixing is less computationally intensive as compared to all the other available mixing mechanisms, and this is the most widely used audio mixing mechanism [42].

3.1.4.2 Justification

Table 3.2 illustrates the comparison between the available audio mixing methods.

Endpoint mixing	Distributed mixing	Hybrid mixing	Centralized mixing
No mixing method is used in server side.	A mixing method is used in server side.	A mixing method is used only for some participants	Use only one central mixer for the mixing process.
A client gets more than one streams at once.	Each participant receives exactly one stream.	Some participants get more than one streams while others get exactly one stream.	All the participants get exactly one same stream.
Minimal transmission delay occurs	There is some delay in transmission.	There is some delay in transmission	Less delay in transmission
Need more bandwidth to send streams separately.	Require less bandwidth compared to End point mixing.	Require less bandwidth compared to Distributed mixing.	Require less bandwidth compared to other methods.
Easy to implement Dominant speaker Identification.	Difficult to implement	Difficult to implement.	Uses silent suppression techniques to remove the noises.

Table 3. 2: Comparison between available audio mixing methods [44]

3.2 Environment setup

The meeting mode of the proposed system is tested in a machine with an Intel ® Xeon ® CPU E5-2680 v3 @ 2.5GHz processor and RAM capacity of 16GB. All tests are performed by using Glances tool (Section 4.4 Experiment), htop, netstat and ntop tools. The downstream rate and the upstream rate of server are both 1Gbps. The used version of Jitsi media server is "*jitsi-videobridge_953-1_i386.deb*", JiCoFo version "*jicofo_1.0-357-1_i386.deb*" and Jitsi meet version "*jitsi-meet_1.0.2098-1_all.deb*".

The Webinar mode of the proposed system is tested in a machine with Ubuntu 16.04.3 LTS, Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz and RAM capacity 1GB. The versions of Jitsi media server, JiCoFo and Jitsi meet are same as the meeting mode.

3.3 System Architecture

3.3.1 Meeting Mode

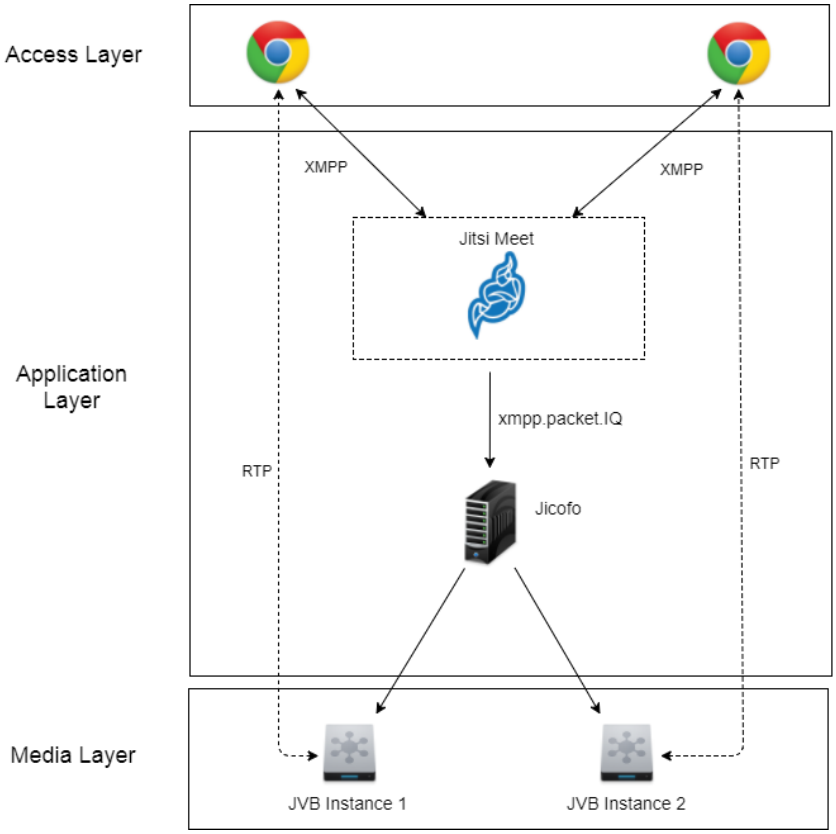


Figure 3. 5: Architecture for meeting mode

The Figure 3.5 shows the architecture of the meeting mode designed by using Jitsi components. The access layer consists of the browsers which are used for the front end application of Jitsi, Jitsi meet application. Participants can use any available browser to connect in a meeting. When the user is trying to connect to a meeting through a browser, it sends an XMPP request to the application layer. The application layer is responsible for the initialization of a meeting and it generates an XMPP packet with the details of the user, which is relevant for the XMPP request came through the web browser [45]. The XMPP packet comes from the Jitsi meet issues an IQ query to JiCoFo and then the conference room is created for the moderator by JiCoFo. The RTP protocol is used to transfer data between the browser and the JVB instance. JVB instance acts as the SFU unit of the system and the connection is a peer-to-peer one with the browser.

Then the users who want to connect to the same room are connected through the RTP protocol automatically by JiCoFo. This is the basic process happening in meeting mode.

3.3.2 Webinar mode

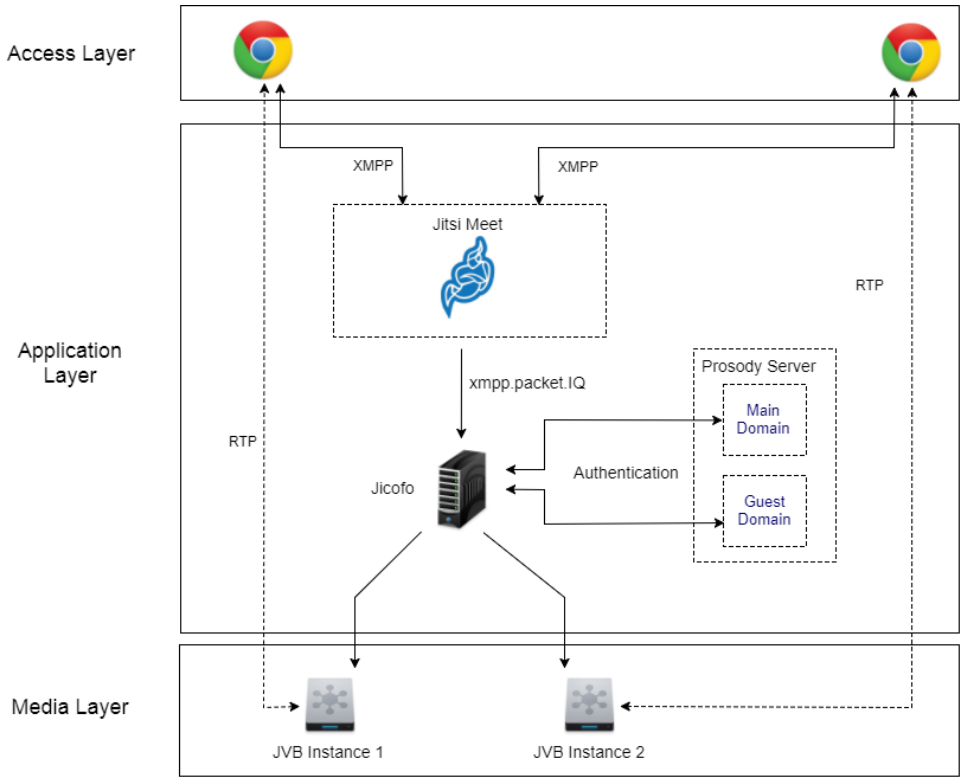


Figure 3. 6: Architecture for webinar mode

Figure 3.6 shows the architecture of webinar mode. Prosody server needs to be configured as a special component here to enable the required authentication. There are two types of users in a webinar called registered users and guest users. Registered users are acting as the hosts in webinar mode while guest users are acting as the participants of webinars. As mentioned in section 3.3.1 Meeting Mode JiCoFo checks whether the request is belonging to a host or guest users. If the request is a host-based one, then the privileges are given to the particular host to create a room. Guest users can connect into a room only after the initialization process of a room is completed. If a user knows the name of the created room, he can connect to that room automatically and this is the working procedure of webinar mode. The participants are only allowed to see the video stream transmitting from the moderator.

3.4 Summary

Chapter 3 describes the problems that will be addressed during this thesis and the approaches can be used to solve those problems. Mainly, there are three main problems have been discussed in this chapter and the approaches that can be used to solve those problems are described briefly. The environment setup and the system architecture are the other subchapters included under chapter 3.

Chapter 4: Implementation

This chapter includes the implementations of the proposed system, the approaches which have been used with the theoretical analysis and the practical scenarios that are linked with chosen approaches. The graphs which are related to the practical scenarios are shown under the section 4.4 Experiment.

4.1 Media server selection

Media server selection can be categorized as a multi-criteria optimization problem that requires a full description of the system before taking a decision. According to this research project, it needs a media server which is WebRTC supported and SFU based.

4.1.1 SFU based media servers

Under this section, the following media servers are discussed with their performance, advantages, disadvantages and the features already available.

4.1.1.1 Janus

Janus is an Open Source WebRTC based gateway which was originally developed by Meetecho [46]. The video conferencing solution that is based on Janus is called as Jangouts [47]. Jangouts is as similar as Hangouts, but it is a JavaScript application runs on client-side and all the server side WebRTC is handled by Janus Gateway. Currently Jangouts supports some features like audio, video, screen sharing and textual chatting for limited amount of participants in a single conference. Figure 4.1 shows the modular architecture of Janus media server.

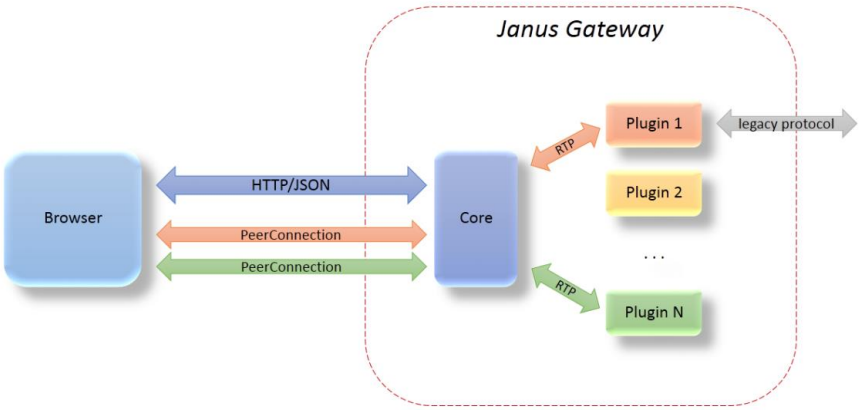


Figure 4. 1: Janus modular architecture [46]

4.1.1.2 SwitchRTC

SwitchRTC is a WebRTC based SFU platform which runs on Amazon AWS and other private and public clouds, specially designed for large-scale multi-party audio, video conferencing and screen sharing purpose [48]. It supports with collaboration features and offers a commercial license with dedicated support for the customers' support for their needs in audio and video conferencing. SwitchRTC is JavaScript based and consists of a native mobile client SDK which is easy to integrate with users' applications.

4.1.1.3 Comparison between SFU based media servers

Janus	SwitchRTC	Jitsi
Supports audio and video recording	Supports audio and video recording	-
Supports real time video and audio streaming.	Supports real time video and audio streaming.	Supports real time video and audio streaming.
Designed for multiple purposes.	Designed specially for business purpose.	Designed for multiple purposes.
-	-	Provides encrypted password storage.
-	-	Supports simulcasting, bandwidth estimation.
Supports latest audio and video plugins.	Supports latest audio and video plugins.	Supports latest audio and video plugins.
Compatible with all platforms.(Linux, Windows)	Compatible with all platforms.(Linux, Windows)	Compatible with all platforms.(Linux, Windows)
Regular updates are available.	Regular updates are not available.	Regular updates are available.

Table 4. 1: Comparison between SFU based media servers.

According to the comparison mentioned in Table 4.1, Jitsi supports more features than the other available SFU based media servers.

4.2 Increase the number of participants

There are two approaches which have been selected, changing the video quality and the audio channels mixing to increase the number of participants in a meeting. The mechanisms and the techniques used under these approaches are explained further in below sections.

4.2.1 Changing the video quality

The needed bit rate to transmit a single video stream has been calculated by using Kush Guage under the section 4.2.1.1. The minimum required bandwidth is calculated for both upstream and downstream of the media server and the client is mentioned under the section 4.2.1.2.

4.2.1.1 The Kush Guage for required bandwidth estimation

Kush Guage is a rule of thumb to calculate the needed bitrate for H.264 encoded video.

The thesis [49] says,

''to estimate the optimal H.264 bit rate value that would give what is considered "good quality" results for a given video, you could multiply the target pixel count by the frame rate; then multiply the result by a factor of 1, 2 or 4, depending on its motion rank; and then multiply that result by 0.07 to get the bit rate in bps (divide that by 1,000 to get a kbps estimate or by 1,000,000 to get a Mbps estimate)''

The Kush Guage formula;

Final bitrate in bps = width x height x frame rate x motion rank x constant

Multiplying width by height of the video frame gives the number of pixels in a frame. It can be vary for different video qualities.

Video Quality	No. of pixels
240p (SD)	352 x 240
360p	480 x 360
480p	858 x 480
720p (Half HD)	1280 x 720
1080p (Full HD)	1920 x 1080
2160p (Ultra HD/ 4K)	3860 x 2160

Table 4. 2: Resolutions for different video qualities [49]

The frame rate is taken immediately as number of frames per second (fps). A high fps rate video will play seamlessly, while a low fps rate will result in jumpy and sketchy-looking videos [50].

The amount of motion of the video is the motion rank. The motion rank has been ranked into three groups as high, medium, and low. These ranks has given the numerical values as following.

- **Low motion** is a video that has minimal movement. For example, a person talking in front of a camera without moving much while the camera itself and the background is not moving at all. (Low = 1)
- **Medium motion** would be some degree of movement, but in a more predictable and orderly manner, which means some relatively slow camera and subject movements, but not many scene changes or cuts or sudden snap camera movements or zooms where the entire picture changes into something completely different instantaneously. (Medium = 2)
- **High motion** would be something like the most challenging action movie trailer, where not only the movements are fast and unpredictable but the scenes also change very rapidly. (High = 4)

According to the thesis [49], a constant value has been introduced to produce the real word bitrate estimates as following.

"I sought to develop a base number from which these multipliers can produce real-world bit-rate estimates. After numerous experiments, I noticed a certain pattern of what could be considered a "constant" or base value (for most commonly used video frame-size and frame-rate ranges). When rounded off, that value is 0.07 bps per pixel, per frame, per motion rank value."

The Kush Gauge constant value should be changed to calculate video bitrate for a codec different from H.264 accordingly to the codec used. The constant value for H.264 codec is 0.07 and it is changed for different video codecs as following.

Video Codec	Value
H.264	0.07
VC-1	0.075
HEVC	0.042
VP8	0.075

Table 4. 3: Constant values for different video codecs

The constant value for VP8 codec is similar to the value of VC-1 [51].

4.2.1.2 Application of Kush Guage for SFU Model

The bit rate can be estimated for a single VP8 video stream of 720p quality with low motion (a person talking in front of a camera without moving much) and with 30 fps frame rate.

> No of pixels in a 720p video frame = 1280 x 720, frame rate = 30 fps, constant value for VP8 codec = 0.075, motion rank = 1
>
> Bit rate = 1280 x 720 x 30 x 1 x 0.075 bits/s = 2073600 bits/s = 1.978 Mbps

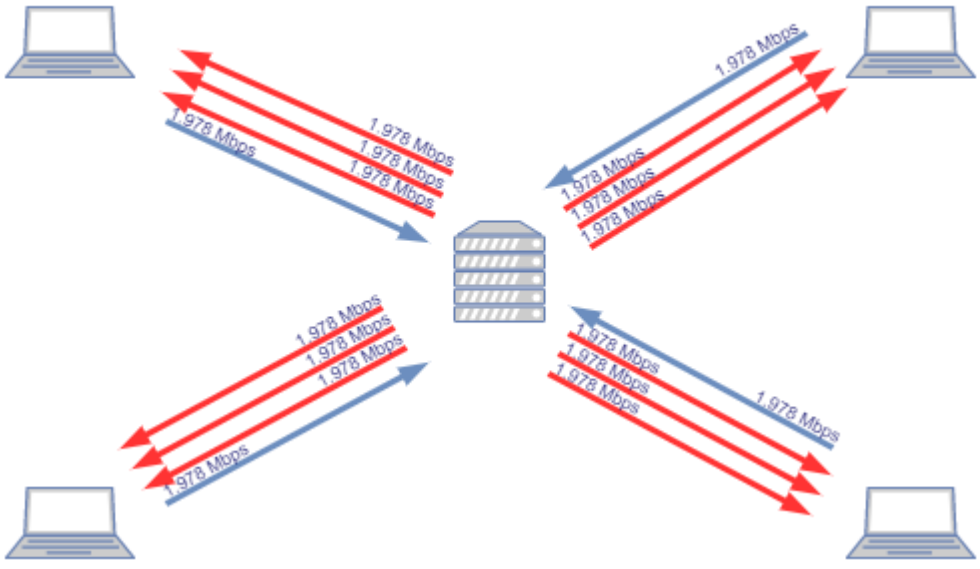


Figure 4. 2: SFU model 4 participants streaming video with 1.978 Mbps

The minimum required bandwidth is calculated for both upstream and downstream of the media server and the client.

The minimum upstream bandwidth for a participant = 1.978 Mbps

The minimum downstream bandwidth for a participant = 3 x 1.978 Mbps = 5.934 Mbps

The minimum upstream bandwidth for the server = 4 x 3 x 1.978 Mbps = 23.736 Mbps

The minimum downstream bandwidth for the server = 4 x 1.978 Mbps = 7.912 Mbps

Considering the Figure 4.2**,** if the number of participants is **n** and the bit rate of a single video stream is **q Mbps**, the required bandwidth can be defined as follows.

The minimum upstream bandwidth for a participant = ***q*** Mbps

The minimum downstream bandwidth for a participant = ***(n-1) q*** *Mbps*

The minimum upstream bandwidth for the server = ***n (n-1) q*** Mbps

The minimum downstream bandwidth for the server = ***nq*** Mbps

The bit rates for different video qualities with 30 fps frame rate and with low motion can be calculated as follows.

Bit rate = resolution x 30 x 1 x 0.075 bit/s

video quality	240p	360p	480p	720p	1080p
bit rate (Mbps)	0.181	0.370	0.884	1.978	4.449

Table 4. 4: Bitrate values for different video qualities for a single video stream

Therefore the bit rate is decreased when the video quality is being reduced. The graph which is related with the practical scenario is shown under Figure 4. 13 According to the Kush Guage equation, video frame rate is affected for the bit rate in the same manner. Therefore the video quality and video frame rate are directly proportional to the bit rate.

4.2.2 Mixing audio channels

Audio channels mixing is one such way that can be used to increase the number of participants in a meeting by reducing the bandwidth consumption as mentioned in section 3.1.4 Increase the number of participants. In Jitsi, all audio-based activities are controlled by the codec OPUS, since Jitsi media server uses SFU model for both audio and video packets transmission it does not support streams mixing by default. According to the section 3.1.4 Increase the number of participants, there are four types of audio mixing methods available. Centralized mixing is the suitable approach among those methods, as it has many advanced features with compared to other available methods.

4.2.2.1 Centralized mixing for Jitsi

Currently, Jitsi transmits audio and video streams separately. The mixing method should be applied to JVB since the whole transmission procedure is handled by Jitsi Videobridge. As soon as someone participates in the created conference, audio and video streams are starting to transmit towards the JVB separately. Centralized audio mixing intends to apply a mixing algorithm to mix all these audio streams come to JVB into one single stream, while all the video streams are transmitting as they are. There are some facts have to be considered before applying centralized mixing for Jitsi. It is essential to suppress the silence of each incoming audio stream as silence suppression[2] is particularly important because when audio streams are mixed, silence packages make an unwanted noise in output audio stream [39]. There are various silence detection algorithms available such as HAM algorithm, Exponential algorithm, Absolute algorithm, Differential algorithm [52].

[2] The capability to stop sending RTP packets during silent periods

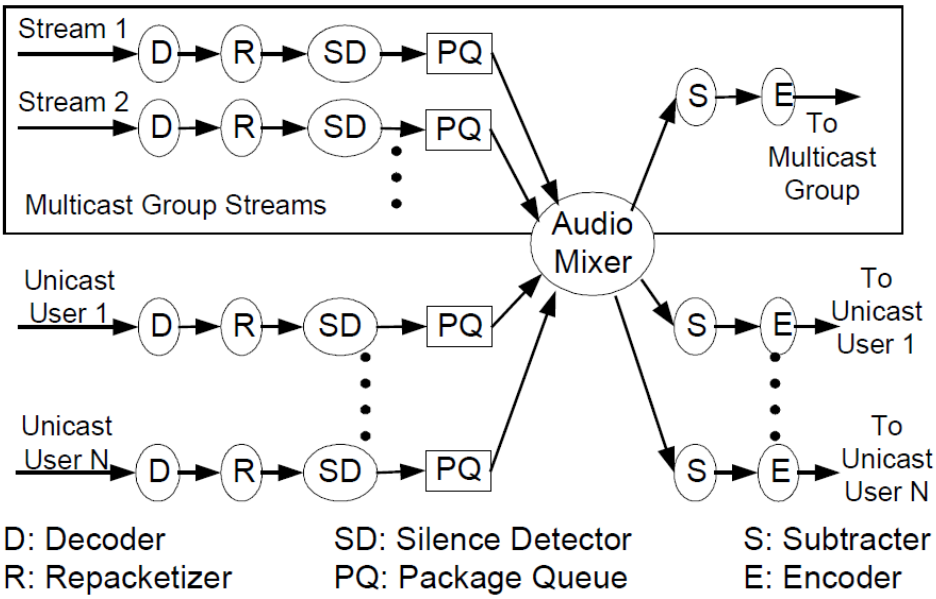


Figure 4. 3: Centralized audio mixing [52]

Figure 4.3 shows the mechanism of centralized audio mixing method which can be applied to mix all the audio streams into a single stream. Silence detector identifies and passes the silence packages to the package queue and the packages will wait in the queue to be picked up by the audio mixing algorithm. Packages are buffered in the queue to avoid missing the late-arriving packages due to the jitter happens in transmission time. Audio mixer polls all queues regularly and passes a copy of mixed audio data to subtractors and just adds the values of all data and store the result in a short array to avoid overflow or underflow. If there is any, then the subtractors subtract the data stored in an array, from the received mixed data, store the result in a byte array and pass it to the encoder. Sometimes the mixed audio sample value is out of range compared to the expected value for byte type, thus: the maximum or the minimum byte value is assigned accordingly to solve the deviation.

4.2.2.2 Audio mixing using COLIBRI

As reference [53] says, in Jitsi there is a way of enabling audio mixing by using COLIBRI which is responsible for providing focus agents with a way of using remote mixers as they were available locally. The important role of this process is the possibility to allocate ports on the mixer interface and then use these ports when establishing Jingle sessions with connecting participants for the conference. Every participant in the conference is assigned one port for RTP data and one port for RTCP data and the combination of both these ports are called as a channel (RTP/RTCP).

Channels are used to transmit data from the video bridge to participants and from participants to the bridge. Every channel has an attribute called "*rtp-level-relay-type*" which has two possible values called "mixer" and "translator" indicating how the video bridge is going to deliver the data on a specific channel/channels to the participants in the conference. The mixing procedure can be enabled by adding the following piece of code for the file "*channels.java*", which can be originally found out as a COLIBRI file.

"rtp-level-relay-type = mixer"

The graph which is related with the practical scenario of audio mixing is shown under Figure 4. 14. For further details refer the Appendix D: Experts' Comments on Audio Mixing and Appendix E: Code Level Change for Audio Mixing

4.3 Implementing Webinar mode

Another Major contribution of this thesis paper is webinar mode system with Jitsi media server. As mentioned before the method used to develop the webinar mode is, modifying the meeting mode. In a webinar there are two major roles.

Host/Moderator - The one who create (initialize) the conference

Participant - The one who attend to the conference after conference create by host.

4.3.1 The working process of the webinar mode

If a user wants to initialize a conference (e.g. https://webinar.meetrix.io/testRoom) room called "*testRoom*", the system notifies the user that "if you are the host, then please authenticate". If this particular user is the host, he can create the room by entering his account details or otherwise the user can wait until the completion of the initiation of the particular room. This is the high-level process happening in the system. The specific users require to have their accounts to achieve this authentication and the participants of a webinar do not need their accounts to use the webinar mode.

4.3.2 The value of authorization in a webinar

If there are no specific accounts created, the unauthorized users also can create conferences. This could be a problem. (For an example in online teaching course if this webinar mode used without authorization then students also can create conferences and that's become

unnecessary). This is the reason that we need specific accounts for whom will be conducting the webinars.

The prosody server (Section 2.4.4.1 Prosody is used as the default communication server in Jitsi. Prosody implementation is used to create and register users in the system by entering username and password. In all platforms except for Windows (currently), prosody has a command-line utility called prosodyctl. This can be used to add a user account like below [54]:

"prosodyctl register me example.com mypassword"

Following Figure 4.4 and Figure 4.5 shows how system appears when a user wants to create a conference.

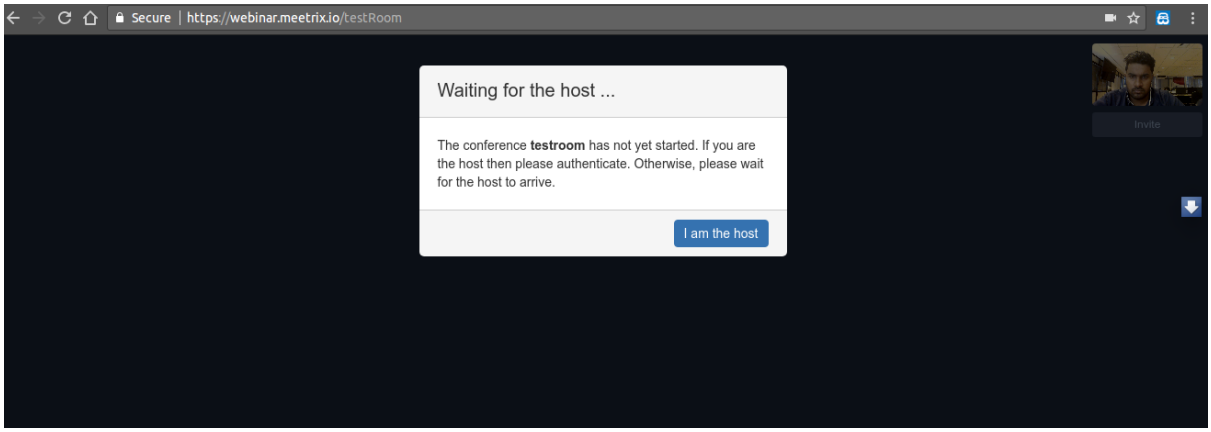


Figure 4. 4: Create Conference testRoom

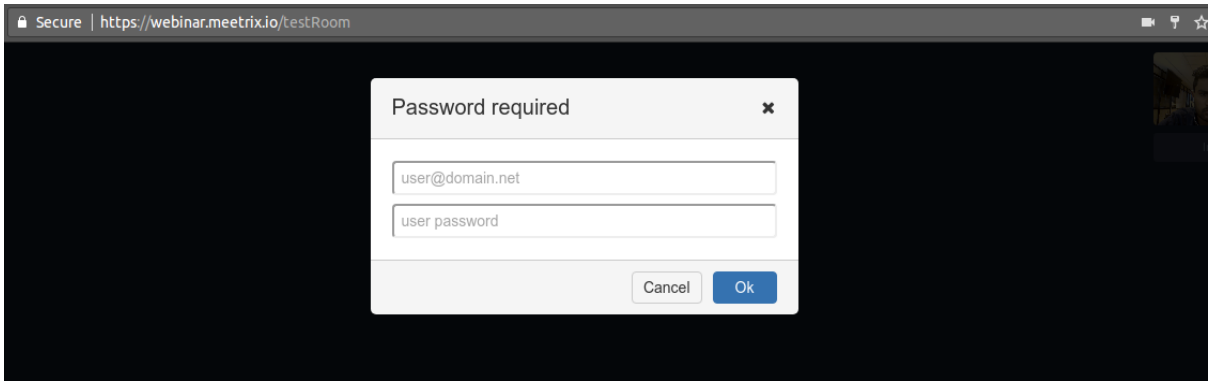


Figure 4. 5: Conference login window for host

This implementation is far different from the Meeting mode. In here only host's video and audio streams are sent through the JVB to the participants. Since that CPU consumption is very low regarding to meeting mode. The architecture used in webinar mode is one to many (1-n)

architecture. The following Figure 4.6 shows how the streaming is happening in the webinar and there is no option called streaming for the participants.

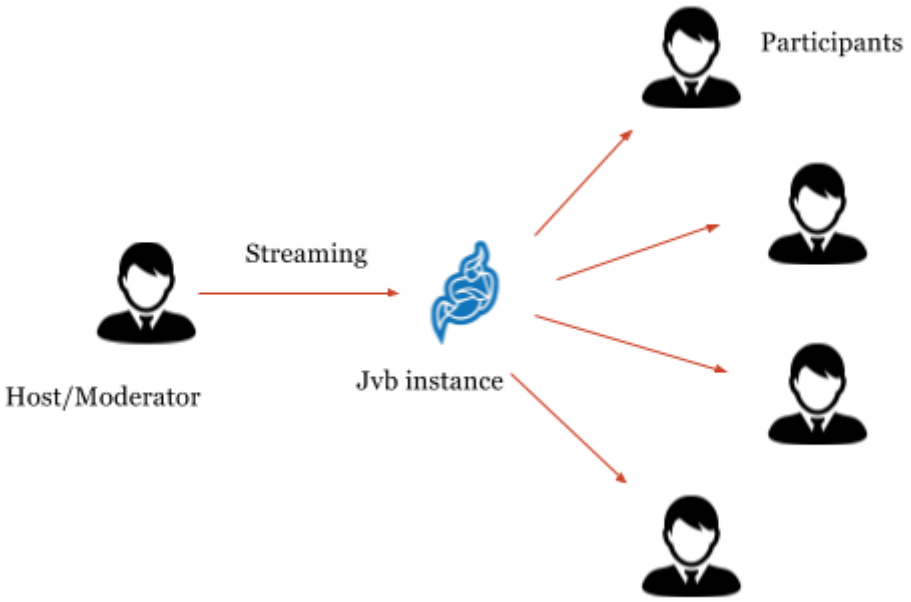


Figure 4. 6: webinar mode Streaming visualization

As mentioned above prosody is used to control the user management. Prosody configuration can be divided into two parts. The first part is known as the "global" section. All the settings here are applied to the whole server and are the default for all virtual hosts. Virtual hosting is a method for hosting multiple domain names on a single server (or pool of servers). This allows one server to share its resources, such as memory and processor cycles, without requiring all services provided to use the same hostname.

The second half of the configuration is a series of VirtualHost and Component definitions. Settings under each VirtualHost or Component line apply only to that host. Components are extra services your server can provide, usually on subdomains of the main server. They provide functionalities such as Chatrooms and transports/gateways to other networks and protocols [55].

Prosody supports authentication-provider plugins, by using these plugins users can have different authentication access into the system. There are four authentication plugins provided by the prosody server.

Name	Description
internal_plain	Plaintext passwords stored using built-in storage
internal_hashed	Hashed passwords stored using built-in storage
cyrus	Hashed passwords stored using built-in storage
anonymous	random username, requires no credentials

Table 4. 5: Prosody Authentication Provider Plugins [56]

In the developed webinar system, one virtual host is created for guest users and main domain is for moderators. “Internal_plain” provider is used for the main domain and “anonymous” is used for guest domain. Only registered users can login as a host by doing this. Therefore moderators need to be registered on the server by providing their usernames and passwords. The following table shows how the configurations are developed.

Main Domain	Guest Domain
VirtualHost "webinar.meetrix.io" authentication = "internal_plain"	VirtualHost "guest.meetrix.io" authentication = "anonymous"

Table 4. 6: Two types of domains of webinar

System will only get the host’s video/audio streams and relays to all the participants in the conference. Participants will only have downlink and host has uplink to the videobridge instance. The configurations are needed to restrict the upload video and audio streams of participants. This is the way how implementation has done in webinar mode of the system.

4.4 Experiment

This section includes the analytical statistics of the approaches which have been used for the implementations of the system mentioned in sections 4.1, 4.2 and 4.3. Glances system monitoring tool is the tool that has been used to get the statistics from server side. Glances is a Python-based resources-friendly monitoring tool which is used for monitoring CPU, memory, load average, network interfaces, disk I/O, processes list, file system spaces utilization [57]. C3.js, D3-based reusable chart library was used to plot the graphs.

4.4.1 Experiment 01: Jitsi media server bandwidth testing

The initial state of meeting mode was created by configuring the components of Jitsi (JVB, JiCoFo, Jitsi meet). The intention of this experiment is to identify how Jitsi media server performs under LAN and WLAN connections under the server specifications which are mentioned in section 3.2 Environment setup.

Before starts the experiment a meeting should be started by a user (e.g. https://domainName/testRoom). After that each user connects to the meeting in every 50 seconds of time period. Glances tool is used to get the statistics of the meeting and plot the graphs for the relevant data.

- LAN Specifications: A Leaf line Fibre connection with Uplink speed: 50 Mbps, Downlink speed: 50 Mbps.
- WLAN Specifications: A 4Mbps up and downlink speed with Dual band 2.4GHz 5Ghz 802.11/a/b/n controller based network.

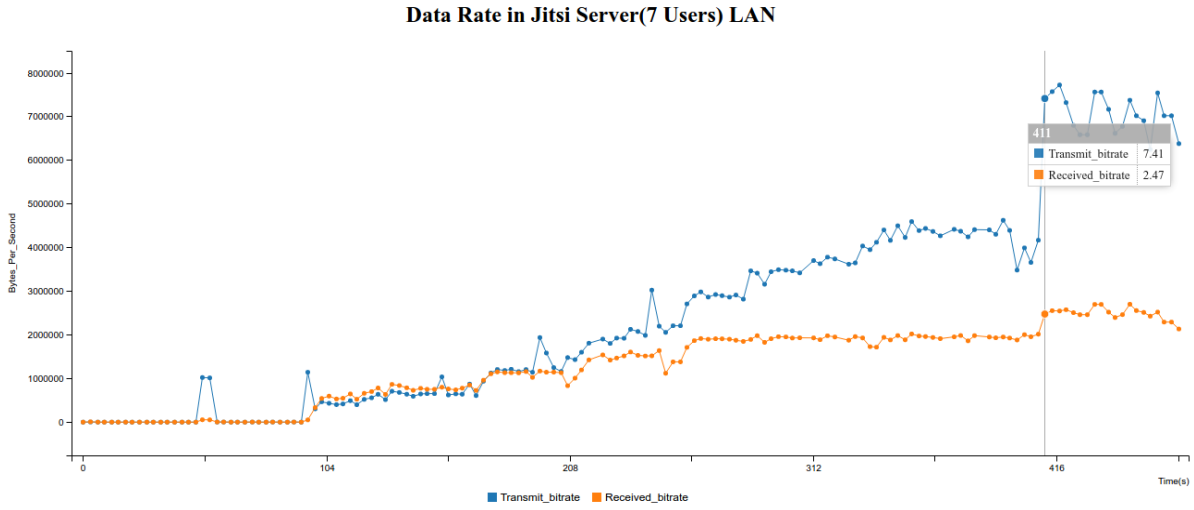


Figure 4. 7: Transmit rate/Received rate of Jitsi in LAN

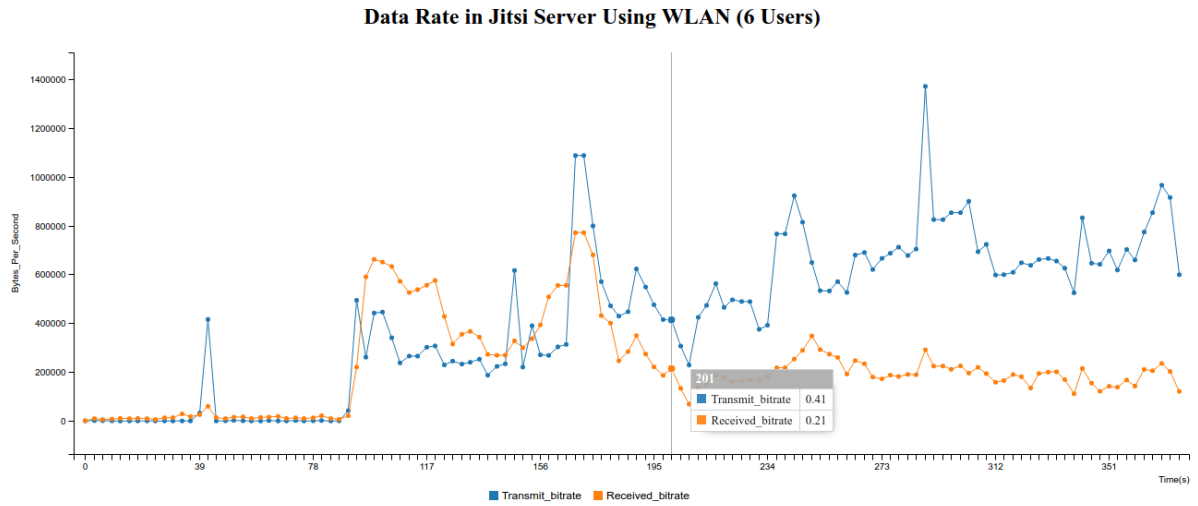


Figure 4. 8: Transmit rate/Received rate of Jitsi in WLAN

Figure 4.7 illustrates the test which has done to check the bandwidth usage against the number of users in server side while Figure 4.8 illustrating the test which has done to the same scenario under WLAN connection. The conclusion that can be obtained by referring those figures is that Jitsi behaves as a SFU model, since the transmit rate is increasing while the number of participants are increasing in a conference.

4.4.2 Experiment 02: Jitsi media server CPU consumption

Figure 4.9 illustrates the CPU consumption of Jitsi media server for 7 users under LAN connection while Figure 4.10 illustrates the CPU consumption of Jitsi media server for 6 users under WLAN connection.

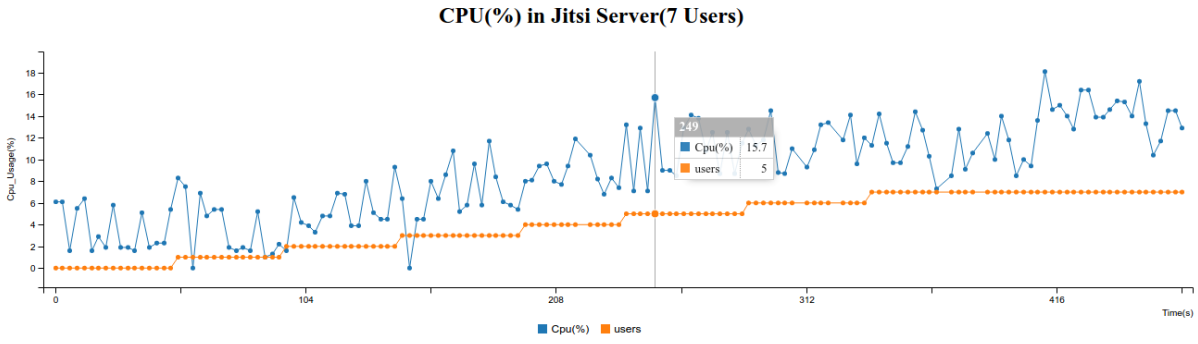


Figure 4. 9: CPU (%) in Jitsi Media Server in LAN

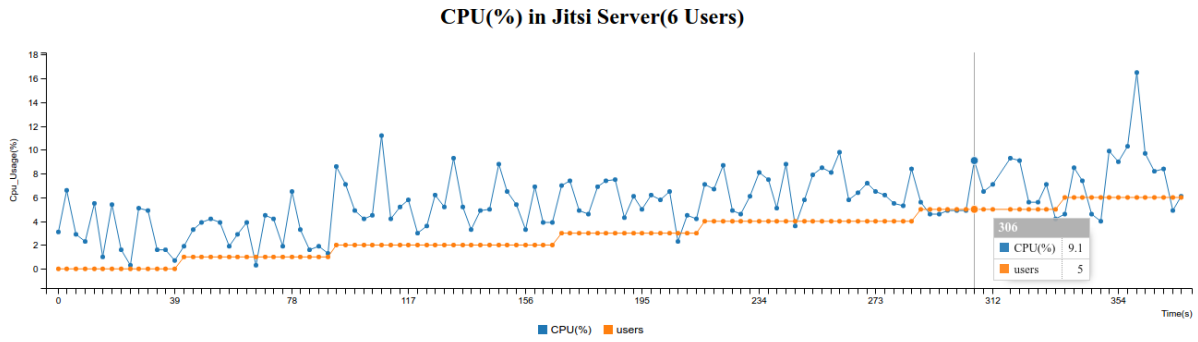


Figure 4. 10: CPU (%) in Jitsi Media Server in WLAN

The conclusion that can be obtained by referring Figure 4.9 and Figure 4.10 is that Jitsi has a lesser amount of CPU consumption.

4.4.3 Experiment 03: Check the maximum number of users that can be connected to single conference without cutting down the video - 480p

The expectation of doing this experiment is to find out the maximum number of users that can be connected to a created meeting, without cutting down the video under the following conditions (The number of stable users).

4.4.3.1 Conditions for the test

All the users connected to the meeting, by using 4G routers which has downlink speed: 4 Mbps - 40Mbps, uplink speed - 1 Mbps -3 Mbps [58] and the video quality of 480p which is supported default by Jitsi.

The number of test users for this test is twelve. When someone started a meeting, each user connected to the meeting after every 50 seconds of time gap and the users are advised to inform immediately about the connectivity issues they have to experience. When a user reported about a connectivity issue that will be the stop point of the testing attempt. After similar five attempts took the most frequent number of users connected to the meeting without having connectivity issues. The mode value for this test was 9 and therefore 9 users would be the outcome value of the test.

The following table includes the average values for uplink and downlink speeds of twelve users that had been used for the above test. The testing results were calculated by using the site https://speedof.me/ [59] (In following table P implies the participant's number).

Mbps	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12
Down Speed	2.1	2.5	2.4	2.6	3.2	1.8	2.4	2.4	2.3	2.1	2.6	2.7
Upload Speed	0.7	0.8	0.9	0.9	0.8	0.7	0.6	0.9	0.7	0.8	0.9	0.8

Table 4. 7: Average Bandwidth of Test Users

Figure 4.11 shows the transmit/receive statistics for 9 users. The conclusion of this test is that the number of stable users in a meeting mode is 9 under the given conditions in section 4.4.3.1 Conditions for the test.

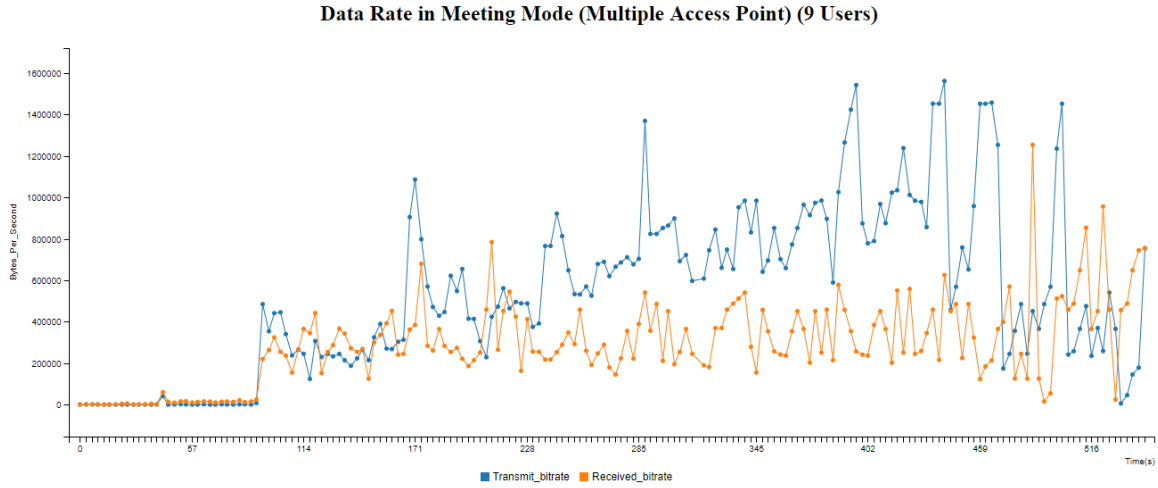


Figure 4. 11: Transmit Rate/Received Rate in Developed System for 9 users

4.4.4 Experiment 04: Check the CPU consumption of the meeting mode for 9 users

This section contains the graph which shows the CPU consumption level of the meeting mode after connecting 9 users for a meeting. The number of participants against the CPU usage (%) was plotted in Figure 4.12.

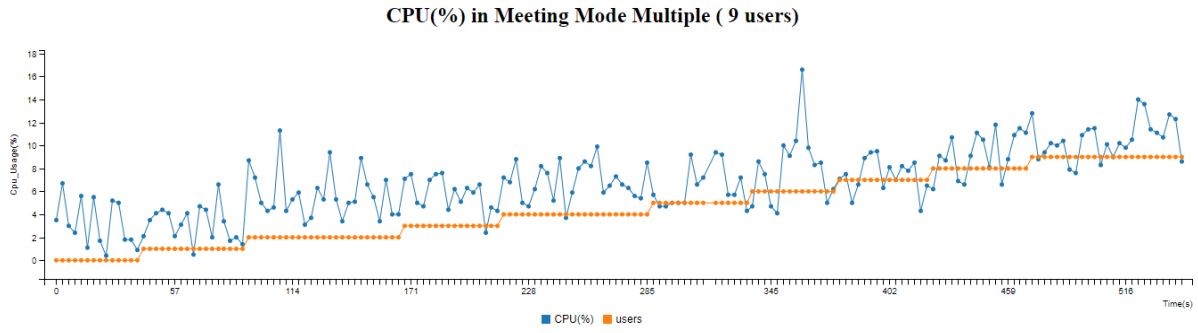


Figure 4. 12: CPU (%) in developed System for 9 users

4.4.5 Experiment 05: Check the maximum number of users that can be connected to single conference without cutting down the video - 360p

According to the results obtained after applying Kush Guage algorithm, the next step is reducing the resolution of the video up to 360p to find out whether the number of participants can be increased or not. The testing procedure is as same as the testing procedure in section 4.4.3 Experiment 03: Check the maximum number of users that can be connected to single conference without cutting down the video - 480p. After analyzing the test results, the number of participants for a stable video conference remains as same as the results of the test which was done with the video quality of 480p.

The Figure 4.13 shows the graph which was plotted for 9 users against the transmit/receive rate under 360p video quality. There is no increasing in number of stable participants, though some decrease can be seen in receive/ transmit rate in Figure 4.13 compared with transmit/receive rate in Figure 4.11. The opinion of experts is that this incident can be happened due to the packet loss while transmitting audio and video streams. The section 4.4.8 Experiment 08: Analyze WebRTC dump file to check the packet loss describes the experiments which are related with packet loss problem.

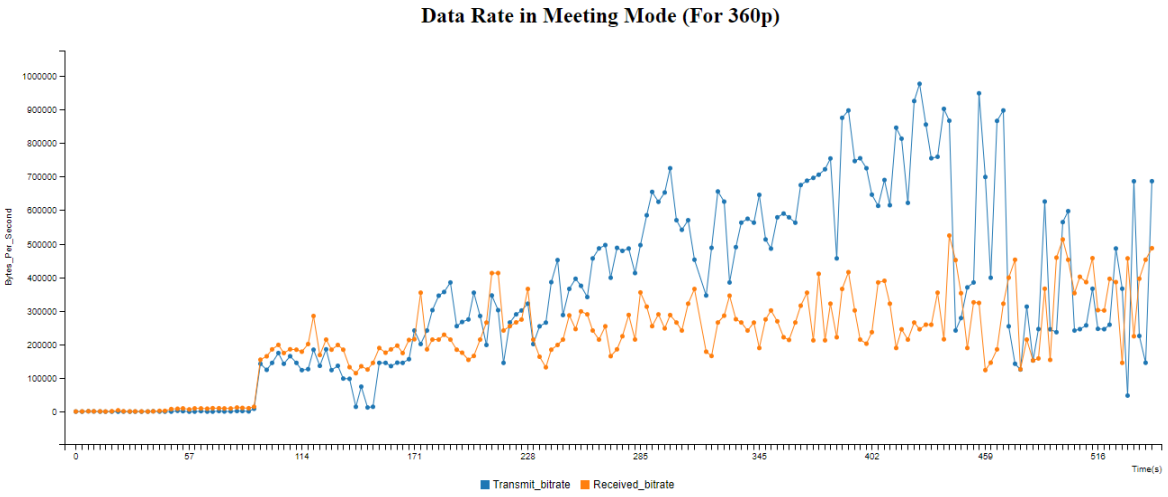


Figure 4. 13: Transmit Rate/Received Rate in Developed System for 9 users - 360p

4.4.6 Experiment 06: Check the maximum number of users that can be connected to single conference without cutting down the video - After Audio Mixing

The intention of this experiment is to identify that, is there any possibility of increasing the number of participants for a meeting by mixing the audio streams together in JVB. The Figure 4.14 shows the graph which illustrates the change in bandwidth after mixing the audio streams in JVB. This graph indicates that there is no big effect for the bit rate by using audio mixing mechanism. Therefore this experiment proofs that audio mixing is not a good solution for increasing the number of participants in a meeting, but the feedbacks got from users clearly indicates that the mixed audio has a good quality compared to the audio before mixing.

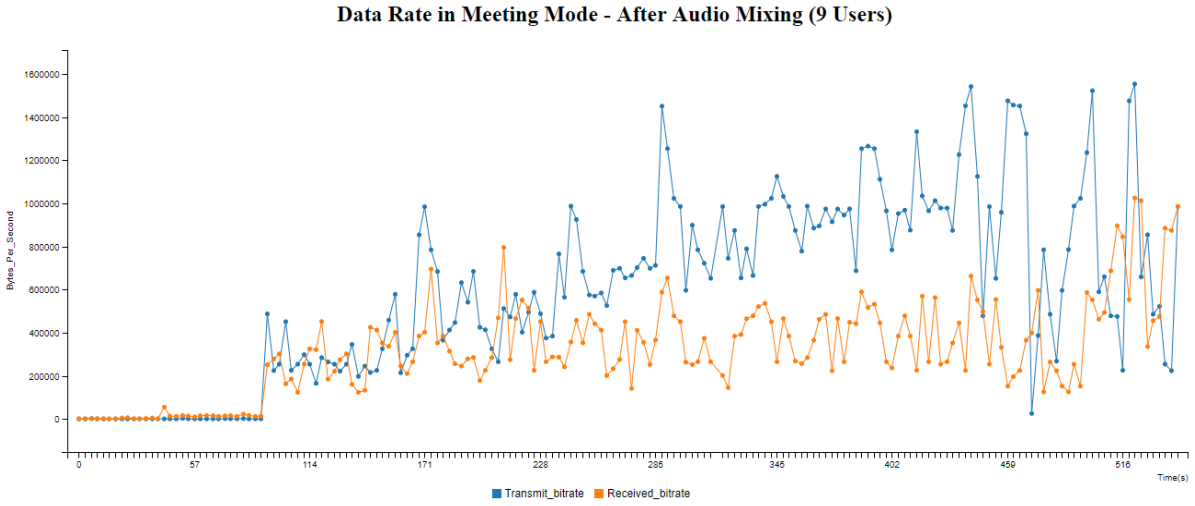


Figure 4. 14: Transmit rate/ Received rate in developed system after audio mixing

4.4.7 Experiment 07: Bandwidth and CPU consumption testing in Implemented webinar mode

The implementation of the Webinar mode is described under the section 4.3 Implementing Webinar mode. There are 11 participants have connected to the webinar mode including the moderator (host). The Figure 4.15 in the below shows the transmit/receive rate to respect to the number of users. The stress test which is used to clarify the maximum number of participants that can connect to a webinar is needed to be done. Figure 4.16 shows the CPU consumption level of this test.

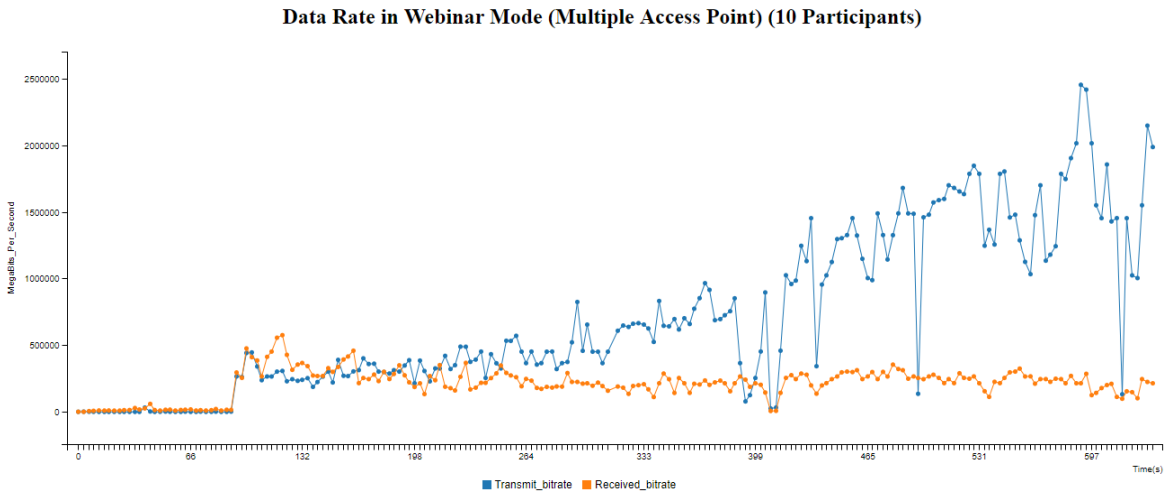


Figure 4. 15: Transmit/ received rate in implemented webinar mode for 10 users

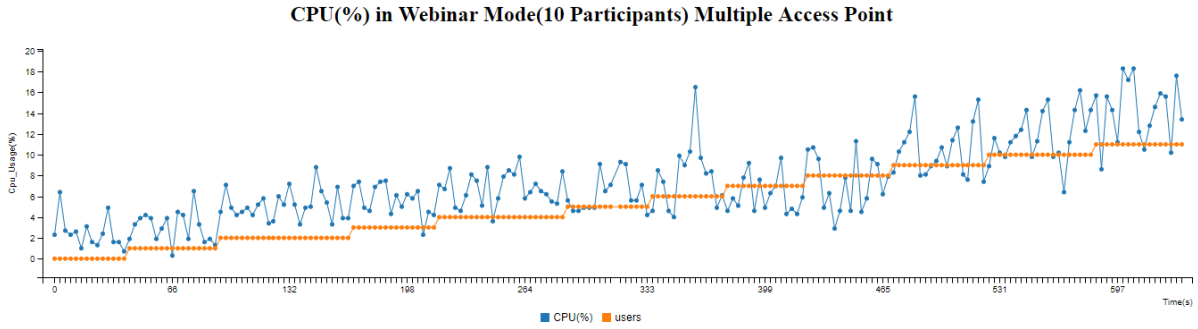


Figure 4. 16: CPU (%) for 10 users in webinar mode

4.4.8 Experiment 08: Analyze WebRTC dump file to check the packet loss

This experiment intends to understand how packet loss happens while users are connecting to a conference (Appendix C:). Chrome WebRTC internals is used to get statistics of client-side, and used “testRTC” which is a web-based application to analyze that raw data.

4.4.8.1 Chrome WebRTC internals

Chrome://webrtc-internals is an internal Chrome tab that provides statistics about ongoing WebRTC sessions. It can be used to debug the flow of WebRTC sessions to determine issues during development or deployment.

4.4.8.2 TestRTC

This application is hosted in https://testrtc.com need to upload a WebRTC dump file which is get from chrome WebRTC internals [60]. After upload the dump file it analyzes the data and plot the bandwidth details, packet loss percentage details video jitter like these parameters.

4.4.8.3 Method

As mentioned in the section 4.4.3 the meeting mode was tested for 9 users. The percentage of packet loss was checked by analyzing the dump files taken from WebRTC internals by increasing the connected participants one by one sequentially. Table 4.8 includes the packet loss percentage of each user that collected from dump files in each test attempt.

In each and every attempt the meeting time was 10 minutes and all the dump files which received during the 10 minutes of time period were analyzed to get the maximum packet loss of each dump file.

	P1	P2	P3	P4	P5
Test 1	1%	1.8%	-	-	-
Test 2	2.4%	4.2%	3.8%	-	-
Test 3	7.8%	12.4%	9.3%	8.2%	-
Test 4	16.7%	19.5%	18.9%	14.2%	15.6%

Table 4. 8: Maximum packet loss measure of test attempts

After collecting all the values, the average packet loss value was calculated and stored in the Table 4.9.

Users in Meeting	Average of Maximum Packet loss percentage (%)
2	1.4
3	3.47
4	9.43
5	16.98

Table 4. 9: Average packet loss for each test attempts

The conclusion that can take after looking at the values of table y is the percentage of packet loss increases with the number of participants of a meeting. However, finding the connection between the increments of packet loss percentage concerning the number of users is hard since this leads to another research area which is not included in this research scope.

4.5 Summary

The implementation chapter includes everything which has been done throughout the development process of the research project. It has consisted of four main sections: media server selection, increase the number of participants, implementation of the webinar mode and experiment. The first section, Media server selection describes how a suitable media server has been selected for the further development by using a comparison of available SFU based media servers: Janus, SwitchRTC, and Jitsi. Then Jitsi has been chosen as the core media server of the research project. The second section, increase the number of participants explains the two methods: change video quality and audio channels mixing, which have been justified to use under the Chapter 3: Analysis and Design with more details. Implementing webinar mode section includes the implementation steps and a detailed description of the webinar mode. The final section discusses the experiments which are related to all the above sections under Chapter 4: Implementation, and the conclusions came up at the end of each experiment.

Chapter 5: Evaluation

5.1 User Evaluation for Webinar Mode

The user evaluation for webinar mode is based on 10 nominal users and 5 expert users who are experts in the video conferencing domain. A feedback form is shared among the users to collect user experience after participating in ten minutes of a webinar. The same feedback form is given to both nominal users and expert users. The statements of the feedback form are created to map the features of the conferencing system such as video jitter, video quality and video/ audio synchronization. Five choices are given for a statement with numbers from 1 to 5 as below.

1. Strongly Disagree
2. Disagree
3. Neutral
4. Agree
5. Strongly Agree

Maximum one choice is picked up by the user according to the experience gained. The instructions and the explanation about the statements were given before filling in the feedback form.

5.1.1 Nominal user evaluation

The data was collected by using twenty nominal users of two webinars with ten participants of each. Each statement has twenty choices selected by the different twenty users. The values of these choices are in between 1 to 5 as described in previous section (5.1). The mode, median, mean and interquartile range were calculated for each statement from the collected data as shown in Table 4.10.

S. No	Statement	Mode	Median	Mean	IQR (Q3-Q1)
1	Video was smooth and plays without lagging or stuttering (video jitter)	3	3	3.1	1.25
2	Both audio and video plays at the same time (real time audio video)	4	3.5	3.45	1

3	It was possible to recognize clearly the person who conducts the webinar.	4	4	3.55	1
4	It was possible to recognize the facial expressions of him/her (The conductor)	3	3	3.1	1
5	There were no connectivity issues during the webinar	3	3	2.75	1
6	Gained an understanding of the features of this video conferencing system	4	4	3.65	1
7	There is an advantage using this system over other video conferencing systems such as Skype and Hangouts	4	3.5	3.45	1
8	I have an understanding of the limitations of this video conferencing system	4	4	3.7	1
9	I will use this system in future for webinars or online teaching	3	3	3.3	1

Table 4. 10: Statistics of collected data from nominal users

The majority of users have given their opinion as neutral for the experience of video jitter in the webinar mode as seen in Table 4.10 (Mode=3, Median=3). The interquartile range is 1.25, meaning that the data is not dispersed much. Therefore the nominal users have not seen a difference in the smoothness of the video compared to their prior experience on the shuttering videos in video conferencing systems. The majority of users have given their opinion as agree to the audio/ video synchronization in the webinar mode as seen in the table Z (Mode=4, Median=3.5). But the mean is 3.45, meaning that some users are not agreed with the audio/ video synchronization. The interquartile range is 1, meaning that the data is not dispersed much. Therefore the nominal users have agreed for the audio and video synchronization. The most of the users have agreed to the third statement (Mode=4, Median=4). The mean value is 3.55, meaning that nominal users have recognized clearly the person who conducts the webinar. Therefore the nominal users have received a video quality which sufficient to recognize the host. The majority of users have given their opinion as neutral for the facial expressions of the host (Mode=3, Median=3). The IQR is 1, meaning that the users may or may not see the facial expressions sometimes during the webinar. The reason for this might be the background light of the host and the video jitter. The most of the users have given their opinion as neutral for having connectivity issues during the webinar (Mode=3, Median=3). However, some are

disagreed to the connectivity issues (Mean=2.75). The reasons might be the packet loss of the network channels, the lack of upstream bandwidth of the host and the downstream bandwidth of the users. The users have good understanding about the features and have a good interaction with this conferencing system (Mode=4, Median=4, Mean=3.65). The majority of users have agreed to that there is an advantage of using this system compared to other existing video conferencing systems(Mode=4, Median=3.5) and there might be some users which have not a prior experience of a webinar with an existing video conferencing systems and they might have a neutral opinion(Mean=3.45). The users have gained understanding about the limitations about this conferencing system (Mode=4, Median=4, Mean=3.7). The majority of users have given their opinion as neutral for the need using this system in future (Mode=3, Median=3). The need of using this system depends on the prior experience of a webinar and the interest of a webinar for their work.

The nominal user evaluation was done for the small number of users due to the lack of resources. The number of users for the evaluation should be increased to maximize the accuracy of the evaluation. The summary of collected data from nominal users are appended to the Appendix F.2 Test Data Summary of Nominal Users.

5.1.2 Expert user evaluation

The expert user evaluation is based on twelve users who are working in a company called ‘Siplo’, and they are developing online tutoring platform for an education institute. The data was collected from the users by using one webinar with twelve participants. The same feedback form has been given to the expert users with the same statements and choices as nominal user evaluation. The only difference with the nominal user evaluation is the category of the user.

Each statement has twelve choices selected by the different twelve expert users. The values of these choices are in between 1 to 5 as described in the previous section (5.1). The mode, median, mean and interquartile range were calculated for each statement from the collected data as shown in Table 4.11.

S. No	Statement	Mode	Median	Mean	IQR (Q3-Q1)
1	Video was smooth and plays without lagging or stuttering (video jitter)	4	3	3.16	1.25

2	Both audio and video plays at the same time (real time audio video)	3	3	3.41	1
3	It was possible to recognize clearly the person who conducts the webinar.	4	4	3.58	1
4	It was possible to recognize the facial expressions of him/her (The conductor)	3	3	2.66	1
5	There were no connectivity issues during the webinar	2	2	2.25	1
6	Gained an understanding of the features of this video conferencing system	4	4	3.75	1
7	There is an advantage using this system over other video conferencing systems such as Skype and Hangouts	4	4	4.08	0.25
8	I have an understanding of the limitations of this video conferencing system	4	4	4.08	1
9	I will use this system in future for webinars or online teaching	3	3	3.25	1

Table 4. 11: Statistics of collected data from expert users

The majority of expert users have given their opinion as agreed to that video was smooth and plays without shuttering (Mode=4) as seen in the Table 4.11. However, some users did not agree to the first statement (Median=3, Mean=3.16). The interquartile range is 1.25, meaning that the data is not dispersed much. Therefore the expert users have agreed to the smoothness of the video available in a webinar. The expert users have a neutral opinion for the audio and video synchronization (Mode=3, Median=3). Therefore the audio/ video synchronization has a less difference compared to the experience of expert users with other conferencing systems. The expert users have been able to recognize the person who conducts the webinar (Mode=4, Median=4). Therefore the received video quality has been sufficient to recognize the conductor of the webinar according to the expert's eye. The users have given a neutral opinion for the recognition of facial expressions of the conductor of the webinar (Mode=3, Median=3). Some users did not agree with the statement (Mean=2.66). Therefore the video quality of this system has not been sufficient to recognize the facial expressions. The most of the expert users have disagreed for having connectivity issues during the webinar (Mode=2, Median=2, Mean=2.25). The reasons might be the packet loss of the network channels, the lack of upstream bandwidth

of the host and the downstream bandwidth of the users. The expert users have a good understanding on the features of this conferencing system (Mode=4, Median=4, Mean=3.75). The expert users have agreed to that there is an advantages to use this system with compared to the other existing systems (Mode=4, Median=4, Mean=4.08). The interquartile range is 0.25, means that the data is not dispersed much. Therefore the features of this system can be interacted easily to the user with compared to the other existing systems. The expert users have a good understanding about the limitations of this system (Mode=4, Median=4, Mean=4.02). The reason for this might be the expert users know the factors of video conferencing. The majority of expert users have given a neutral opinion for the need of using this system in future (Mode=3, Median=3). The mean is 3.25, meaning that some users have agreed to use this system in future for the online webinars. Therefore the expert users have not seen much differences with compared to the other systems.

The evaluated system is in acceptable level and some improvements has to be done according to the both nominal/expert user evaluations and the comments from the expert users. It will be better to increase the number of users for the evaluation, to maximize the accuracy of this evaluation. The summary of collected data from nominal users are appended to the appendix F.1 Test Data Summary of Expert Users.

5.2 Summary

Evaluation chapter describes how the evaluation process of the research project has been achieved. User evaluation is the evaluation method that was used in this research project since the lack of the availability of testing tools. Webinar mode was evaluated by using this technique, and the evaluation criteria that has been used is described according to the results got for the questionnaire, and the conclusions made for each question are explained further in separately.

Chapter 6: Conclusion

Video conferencing is a field which is growing rapidly due to the higher demand of internet users. WebRTC comes in the first hand as an emerging technology which can be used to develop video conferencing systems with the support of real-time communication, installation free from external plugins or frameworks and availability in all web browsers and platforms. There are three main models in WebRTC called Multipoint Control Unit (MCU), Selective Forward Unit (SFU) and Mesh model, which are used for building video conferencing systems. Each model has different ways of handling audio, video streams and also each model has the different architecture based on streaming process.

The goal of this thesis is to demonstrate how a video conferencing system can be built using SFU model and how the number of participants in a particular conference can be increased under the given conditions in section 4.4.3.1 Conditions for the test . Four objectives have been used to achieve the mentioned goal, which is considered as research work. Finding an alternative solution to replace flash based video conferencing systems, using audio mixing mechanism to increase the number of participants per a conference, using video quality change method to increase the number of participants per a meeting and developing a webinar mode are the main objectives that were covered throughout this thesis. Choosing WebRTC as the core technology is the approach that has been taken as the solution for the first objective. SFU model has been selected as the core architecture of the system since it needs a less CPU consumption.

As mentioned in the goal of this thesis in above paragraph, there two approaches have been selected as the solutions for increasing the number of participants in a meeting, using audio mixing method and using video quality change method. Centralized mixing mechanism was the solution that had been chosen for audio mixing method. The conclusion that has been taken after having a few experiments was, the audio mixing method was not suitable to achieve the goal since there was no significant difference in transmitting/ receive bit rate compared to the original streams before applying mixing method. However, another conclusion of the experiments was the mixed audio stream has a good listening quality compared to the original audio stream.

Before applying video quality change method the difference between the bit rates with different video qualities were calculated and compared theoretically by using the Kush Guage equation.

The Kush Guage equation represents that the video bitrate is directly proportional to the video quality and the frame rate of the video stream. The calculated bitrate values for low video qualities were less than the high video quality bitrates. The bitrate values are reduced for low framerates as same as the impact video quality. This intention was lead to change the video quality and compare the bitrates practically. The bit rates were reduced in average for low video qualities rather than the higher qualities according to the experiments done. But this was not sufficient to increase the number of participants in video conference due to the other facts which are not in this research scope. One such fact is a packet loss, and the packet loss is increased when the number of participants is increased in a conference concerning the experiments done. The impact of packet loss for the video conference is another research area. But the conclusion that took was this approach cannot be applied to achieve the mentioned goal above.

Another main contribution of this thesis paper is the implementation of webinar mode. The method that has been chosen to develop the webinar mode was enabling authentication of Jitsi components. The evaluation process of the webinar mode was conducted by using two types of users, expert users, and nominal users. As the conclusion of this research project, it is proven that audio mixing method and the changing video quality methods cannot be applied to increase the number of participants under the provided conditions. The implementation of webinar mode was successful.

The challenges that went through this research project were the lack of the availability of testing tools, the limited amount of resources and WebRTC is still in the development stage. The main deliverable of this research based project is that the used methods cannot be used to achieve the goal of the project. Implementation of Webinar mode is the second deliverable of the project.

Chapter 7: References

[1] "SpringerReference," 2017. [Online]. Available: https://sctt.org.uk/wp-content/uploads/2016/05/VC-Intro-v10.pdf. [Accessed 30 December 2017].

[2] P. Inc, "An Introduction to the Basics of Video Conferencing," 2017. [Online]. Available: http://www.polycom.com/content/dam/polycom/common/documents/whitepapers/intro-video-conferencing-wp-engb.pdf. [Accessed 30 December 2017].

[3] R. Manson, "Explore WebRTC for real-time peer-to-peer," 2013.

[4] "draft-ietf-rtcweb-overview-19 - Overview: Real Time Protocols for Browser-based Applications," ietf.org, [Online]. Available: https://tools.ietf.org/html/draft-ietf-rtcweb-overview-19. [Accessed 01 January 2017].

[5] B. Jansen, "Performance Analysis of WebRTC-based Video Conferencing," 2016.

[6] "WebRTC MediaServer(SFU/MCU)の情報まとめ," Iwashi.co, [Online]. Available: http://iwashi.co/2016/09/03/webrtc-sfu-mcu-summary. [Accessed 01 January 2018].

[7] M. Walter, "WebRTC multipoint conferencing with recording using a Media Server," 2015.

[8] "How Different WebRTC Multiparty Video Conferencing Technologies Look Like on the Wire • testRTC," testRTC, [Online]. Available: https://testrtc.com/different-multiparty-video-conferencing/. [Accessed 31 December 2017].

[9] S. Rajab, "Comparing different network topologies for WebRTC conferencing," 2015.

[10] E. Ivov, "Hangout-like Video Conferences with Jitsi Videobridge and XMPP," 2013.

[11] Dialogic, "Conquering Scalable WebRTC Conferencing," 2016.

[12] "What's WebRTC and what's a media server » FIWARE," Fiware.org, [Online]. Available: https://www.fiware.org/devguides/real-time-processing-of-media-streams/whats-webrtc-and-whats-a-media-server/. [Accessed 01 January 2018].

[13] "What is Jitsi? | Jitsi," Jitsi.org, [Online]. Available: https://jitsi.org/what-is-jitsi/. [Accessed 30 December 2017].

[14] J. Meet, "Jitsi Meet on the App Store," App Store, [Online]. Available: https://itunes.apple.com/us/app/jitsi-meet/id1165103905?mt=8. [Accessed 30 December 2017].

[15] "jitsi/jitsi-meet," GitHub, [Online]. Available: https://github.com/jitsi/jitsi-meet. [Accessed 31 December 2017].

[16] "jitsi/jitsi-videobridge," GitHub, [Online]. Available: https://github.com/jitsi/jitsi-videobridge. [Accessed 31 December 2017].

[17] F. Rupin, "Web(RTC) Conferencing as a Service".

[18] "Demystifying JITSI | TO THE NEW Blog," TO THE NEW BLOG, [Online]. Available: http://www.tothenew.com/blog/demystifying-jitsi-2/. [Accessed 31 December 2017].

[19] "XMPP | XMPP Servers," Xmpp.org, [Online]. Available: https://xmpp.org/software/servers.html. [Accessed 31 December 2017].

[20] "Lua: about," Lua.org, [Online]. Available: https://www.lua.org/about.html.

[21] Boris Grozev, Lyubomir Marinov, Varun Singh, Emil Ivov, "Last N: Relevance-Based Selectivity for Forwarding Video in Multimedia Conferences," 2016.

[22] Ilana Volfin and Israel Cohen, "Dominant speaker identification for multipoint videoconferencing," 2013.

[23] "jitsi/jitsi-videobridge," GitHub, [Online]. Available: https://github.com/jitsi/jitsi-videobridge/blob/master/doc/last-n.md. [Accessed 31 December 2017].

[24] Luis López , Miguel París, "Kurento: the WebRTC Modular Media Server".

[25] "WebRTC in the real world: STUN, TURN and signaling - HTML5 Rocks," HTML5 Rocks - A resource for open web HTML5 developers, [Online]. Available: https://www.html5rocks.com/en/tutorials/webrtc/infrastructure/. [Accessed 31 December 2017].

[26] V. Singh, "AppRTC — callstats.io," Callstats.io, [Online]. Available: https://www.callstats.io/integrate/apprtc/. [Accessed 31 December 2017].

[27] "1716 - Audio feedback on apprtc - webrtc - Monorail," Bugs.chromium.org, [Online]. Available: https://bugs.chromium.org/p/webrtc/issues/detail?id=1716.

[28] "appear.in," appear.in, [Online]. Available: https://appear.in/information/team/. [Accessed 31 December 2017].

[29] "appear.in," appear.in, [Online]. Available: https://appear.in/information/tos/. [Accessed 31 December 2017].

[30] "appear.in Reviews | G2 Crowd," G2 Crowd, [Online]. Available: https://www.g2crowd.com/products/appear-in/reviews. [Accessed 31 December 2017].

[31] "How much bandwidth does Skype need? | Skype Support," Support.skype.com, [Online]. Available: https://support.skype.com/en/faq/FA1417/how-much-bandwidth-does-skype-need. [Accessed 31 December 2017].

[32] "What is Skype?," Lifewire, [Online]. Available: https://www.lifewire.com/what-is-skype-3426903. [Accessed 18 December 2017].

[33] "Google Hangouts," udel.edu, [Online]. Available: https://www1.udel.edu/edtech/GoogleAppsResources/hangouts.html. [Accessed 18 December 2017].

[34] "Google Groups," google, [Online]. Available: https://productforums.google.com/forum/#!topic/hangouts/ELhQw_ZJkII. [Accessed 18 December 2017].

[35] João Luís Gonçalves Domingos, "Website file download acceleration using WebRTC," 2015.

[36] "Top 5 Cloud Based Video Conferencing Services | ezTalks," Eztalks.com, [Online]. Available: https://www.eztalks.com/video-conference/top-5-cloud-based-video-conferencing-services.html. [Accessed 19 December 2017].

[37] "Apple Announces Support for WebRTC in Safari 11," Peer5 Blog, [Online]. Available: https://blog.peer5.com/apple-announces-support-for-webrtc-in-safari-11/. [Accessed 19 December 2017].

[38] Maruf Pasha1,Furrakh Shahzad2, Arslan Ahmad3, "Analysis of challenges faced by WebRTC videoconferencing and a remedial architecture," 2016.

[39] Ahmet Uyar, Hasan Bulut, Geoffrey C. Fox, Wenjun Wu, "An Integrated Videoconferencing System for Heterogeneous Multimedia Collaboration".

[40] Lorenzo Miniero , Simon Pietro Romano, "Performance analysis of the Janus WebRTC gateway".

[41] M. Luo, "HORIZONTAL SCALING OF VIDEO CONFERENCING APPLICATIONS IN VIRTUALIZED ENVIRONMENTS," 2016.

[42] Pedram Pourashraf,Farzad Safaei, Daniel Franklin, "Minimisation of video downstream bit rate for large scale immersive video conferencing by utilising the perceptual variations of quality," 2014.

[43] Marcus Carlberg, Christoffer Stengren, "Video Conferencing - Session and Transmission Control," 2015.

[44] Kundan Singh, Gautam Nair and Henning Schulzrinne, "Centralized Conferencing using SIP".

[45] "Authorizing Creation/Entrance of a Conference in Jitsi | TO THE NEW Blog," TO THE NEW BLOG, [Online]. Available: http://www.tothenew.com/blog/authorizing-creationentrance-of-a-conference-in-jitsi/. [Accessed 24 December 2017].

[46] A. Amirante, T. Castaldi, L. Miniero, S. P. Romano, "Janus: a general purpose WebRTC gateway".

[47] "jangouts/jangouts," GitHub, [Online]. Available: https://github.com/jangouts/jangouts. [Accessed 21 December 2017].

[48] "Start-Up Nation Finder - Israeli startup network," Start-Up Nation Finder, [Online]. Available: http://finder.startupnationcentral.org/company_page/switchrtc. [Accessed 22 December 2017].

[49] K. AMERASINGHE, "H.264 FOR THE REST OF US".

[50] "How to Increase the FPS (Frames per Second) on a Webcam | Techwalla.com," Techwalla, [Online]. Available: https://www.techwalla.com/articles/how-to-increase-the-fps-frames-per-second-on-a-webcam. [Accessed 25 December 2017].

[51] "Kush Gauge," Different Video Codecs, [Online]. Available: http://blog.sporv.com/andreas-corollary-to-the-kush-gauge/. [Accessed 26 December 2017].

[52] Claypool M., Riedl J, "The Effects of Silence Deletion on the CPU Load of an Audio Conference," 1994.

[53] "XEP-0340: COnferences with LIghtweight BRIdging (COLIBRI)," Xmpp.org, [Online]. Available: https://xmpp.org/extensions/xep-0340.html. [Accessed 27 December 2017].

[54] "Creating accounts - Prosody.im," Prosody.im, [Online]. Available: https://prosody.im/doc/creating_accounts. [Accessed 28 December 2017].

[55] "Configuring Prosody - Prosody.im," Prosody.im, [Online]. Available: https://prosody.im/doc/configure#adding_componentsservices. [Accessed 29 December 2017].

[56] "Authentication providers - Prosody.im," Prosody.im, [Online]. Available: https://prosody.im/doc/authentication. [Accessed 30 December 2017].

[57] "Glances - Python-based system monitoring tool — Quintagroup," Quintagroup.com, [Online]. Available: http://quintagroup.com/cms/python/glances. [Accessed 30 December 2017].

[58] "Help Home Broadband Speed," Dialog.lk, [Online]. Available: https://www.dialog.lk/home-broadband-help-data-speed. [Accessed 31 December 2017].

[59] "How to find a reliable network speed test," CNET, [Online]. Available: https://www.cnet.com/how-to/how-to-find-a-reliable-network-speed-test/. [Accessed 31 December 2017].

[60] "What do the Parameters in webrtc-internals Really Mean? • testRTC," testRTC, [Online]. Available: https://testrtc.com/webrtc-internals-parameters/. [Accessed 31 December 2017].

Chapter 8: Appendix

Appendix A: Individual Contribution

This section includes the individual contribution of each member in the research group. There are three main research parts covered in this research project.

W. A. S De Silva

I studied about the Jitsi media server on how it is working, what is the architecture that has been used to implement the media server and After getting to know about the structure of Jitsi, I explored on how SFU model can be applied into our research project by testing it using in a local environment. I could find a tool called glances to collect the data which were collected during the experiments we did and then used those data to plot the graphs by using the c3.js library. The research part that I covered was implementing a webinar mode by using Jitsi as the core media server. As I found out, Jitsi already performed a way of having a webinar by using YouTube. Therefore I searched a way to implement the webinar mode differently. The major problem that I had to face was handling the authentication procedure. The solution that I could find out about this problem was using prosody server. The testing procedure was conducted as mentioned in section 4.4 and the graphs which were based on the test results were plotted.

R. R. Liyanagamage

I studied about the systems like EasyRTC, Big-blue button, OpenVCX, OpenTok and Licode on how they are working, used architectures to develop the systems. The research part I took care was, find a way to increase the number of participants per a meeting by changing the video bit rate. The approach I used was, referred an algorithm called Kush Guage to find out the connection between video resolution and the bandwidth. It proved that the video resolution is directly proportional to the bandwidth. Therefore I experimented to find out whether there is a possibility to increase the number of participants in a conference by saving the bandwidth. I wanted to find out how much bandwidth is needed to Jitsi with the increment of some participants in a meeting. So I looked for a particular way to perform the objective. I used Glances tool to find out how the video transmission bit rate is changed against the time in server side. But as the conclusion of the research, it proved that this method cannot use to increase the number of participants of a meeting, without addressing the mentioned factors in this

thesis.. According to the expert opinions, we could get to know that packet loss or the jitter caused for not increasing the number of participants in a meeting.

K. A. I Thiunuwan

I studied about Kurento media server during the literature survey of our research project. I focused on the architecture of the Kurento media server and how it performs as a media server. Then I explored about MCU model and got to know that it was not suitable for our research project since the CPU consumption is too high due to the mixing process. The research area I covered was using audio channels mixing method to increase the number of participants in a meeting. Therefore first I had to study about the audio mixing mechanisms, which are available. Among those methods, I chose Centralized audio mixing. Not only this mechanism also I tried to modify the codes in JiCoFo to mix the audio channels. The problem that I had to face was the lack of testing tools. Therefore I used webrtc-internals to collect a dump file and tested it by using TestRTC to find out whether the mixing process was successful or not. Then used glances tool to collect data on both occasions, without audio mixing, and with audio mixing to plot graphs and compare the bandwidths in two situations. After analyzing the plotted graphs, the conclusion that I had to take was there was no much difference in bandwidth usage after applying mixing method compared to the original audio stream, and therefore the method I used was not suitable to increase the number of participants in a meeting. But after the user evaluation, it gave a result that the listening quality of the mixed audio stream is better than the original stream. According to the expert opinions, we could get to know that packet loss or the jitter caused for not increasing the number of participants in a meeting.

Appendix B: Comparison of available Media servers

H1 - Technology

H2 - Supported Browsers

H3 - Need plugins

H4 - Core Languages

H5 - Architecture types

H6 - Mobile Compatibility

H7 - Third party algorithms or protocols

H8 - License

H9 - WebRTC compatibility

H10 - Open Source

H11 - Security

H1	**H2**	**H3**	**H4**	**H5**	**H6**	**H7**	**H8**	**H9**	**H10**	**H11**
Kurento	Opera, Firefox, Google-Chrome	No	C++, JavaScript	P2P	yes	Speech Recognition, Face recognition, sentiment analysis	LGPL v2.1	yes	yes	Encrypted
Jitsi	Opera, Firefox, Google-Chrome	No	Java, React Native(Mobile App)	P2P	yes	IRC, MSNP, OSCAR, SIP/SIMPLE, XMPP/Jingle, YMSG	Apache 2.0	yes	yes	Encrypted
BigblueButton	Opera, Firefox, Google-Chrome	No	Java, Action Script, JavaScript, Groovy, CSS, Scala	P2P	yes	-	LGPL v3.0	yes	yes	Encrypted
OpenVSX	Opera, Firefox, Google-Chrome	No	Java(Java Platform JRE 6), Implemented in C	P2P	yes	MCU supports		yes	yes	Encrypted

Self hosted solutions - Comparison table

This list was composed in October and November 2014, and some informations in it might be already outdated.

Media Servers		Doubango Telepresence	FlashPhoner	Jitsi	Kurento Media Server	Licode	Medooze and Mobicents	NG Media Server	Powermedia XMS
		doubango		Libjitsi, Video Bridge, JitMeet		Lynckia, Erizo	Medooze MTU and Mobicents AppServer (SIP)		Dialogic
short	**blocker**	no	audio only recording	no	no	no	no	no	no
	pros	- transcode between VP8 and H.264 - Commercial license available	Works together with Wowza (already in use)	- OpenFire plugin (magnocall interoperability) - 10 persons chatting simultaneously on 1.6Ghz 1 Core	- Kurento Application Server is supplied - uses JSR309 (Media Server Control API) - Pluggable pipelines (filter, post-process and transcode media streams) - connect with SIP using Kurento Application Server - Receive-only streaming with http	- MIT license, changes are allowed and don't need to be published - Allows to record each participant stream - Allows to send messages over data stream - Erizo is split up in 3 parts and may be run on different machines (Controller, Agent, JS) - play recorded files into conversation	- JSR309 (Media Server API) - Commercial License available	-	- JSR309 (Media Se API) - REST Interface - Audio features: Voi activity detection, sil suppression, comfor noise generation
	cons	- No information about benchmarks - GPL - codecs need to be built on/for destination machine (licensing?) - transcoding uses ffmpeg (Doubango discourages using ffmpeg due to licensing issues) - one encoder per codec, transcoding for low bandwidth is not supported (one setup for all encoders)	- No information on benchmarks - expensive	- LGPL - Application Server needed - only works on Chrome	- LGPL - only Media Server, Application Server is needed - No direct connection with SIP	- No mixing functionality - No transcoding of streams for low bandwidth - No SCTP - No direct connection with SIP clients - Application Server is needed	- Needs Application Server - GPL - MCU Media Server is only available in GPL	-most information is either vague or in french (most are both)	- Needs Application Server
Goals									
record		yes to local file, using ffmpeg	audio only	yes	yes VP8 and H.264	Yes, each participating stream is recorded separately on the server (as mkv with VP8 and Opus). Recording is started by the participant.	yes	Yes, but no further information	yes
8+ simult. participants			no information	24+ possible		yes			yes, see benchmark
ICE (stun, turn, negotiation, none)		negotiation	STUN, TURN, negotiation	negotiation	negotiation	negotiation	STUN, negotiation	no?	negotiation
Signaling		yes (TLS and Websockets with TLS)	SIP 2.0 rfc3261 RTP rfc3550 SDP rfc4566	yes (Websockets)	no (v5), yes (v4)	Yes, Erizo API also publishes Signaling API	yes	SIP	SIP (RFC3261) WebRTC JavaScrip RTSP

Rooms, User auth		yes, with passwords	no	yes	no (needs Application Server)	yes	no (needs Application Server)	no information	yes (API)
Connect with SIP Clients		yes	yes, with SIP/RTP	yes	no (needs Application Server)	no	no (needs Application Server)	yes	yes
Connect directly to PSTN		no	no	no	no (needs Application Server)	no	no (needs Application Server)	no	no
Costs and License		GPLv3 Or commercial license available	$56,675 (256 simult. calls) online Configurator One license for multiple Servers	free LGPL 2.1	free LGPL 2.1	MIT license	Mobicents: free (GPL) Medooze MTU: free (GPL) and commercial	Fully featured evaluation version license based on channel and feature set	Commercial license
Other Information									
Benchmark	**framework**	C++	Java and C++	Java and C++	Media Server: C++ Application Server: JEE 7	C++	none		Video: up to 450 unidirectional stream (one conference = 1 stream (mixed))
	hardware		QuadCore 2.5Ghz 4-16 GB RAM JDK 1.7 Dedicated Server (but VPS also supported)	1 Core 1.6Ghz, 4GB Ram, 4 upstreams full size, rest is mixed in (10 persons)	8GB RAM 16GB HDD Media Server: Ubuntu Linux 13.10 or newer (needs gstreamer 1.2.0) Application Server: OpenJDK 7	only works on unix based systems	none	Recommended hardware: OS: Windows Virtualization: VMWare ESXi 5.0 or higher, Microsoft HyperV Processor: QuadCore at 3GHz or higher Memory: 4 GB Disk: 500 Gb or higher Network: Ethernet 1 Gb	Minimum: < 500 cha (~100 video) Intel Xeon E5420 Quad-Core (2.50 GH 1333 MHz FSB, 80W 8 GB RAM 250 GB HDD Recommended: up t 2000 channels (~45(video) Intel Xeon X5650 D Hex-Core (2.66 GHz 1333 MHz FSB) > 16 GB RAM > 2 TB HDD
Special		- this software needs to be built on destination machine - codecs need to be installed during build (except for G.722 and G.711) - One encoder per codec (no extra encoder for Cell phone) - Usable Web-SIP-Client: http://conf-call.org/ - Presentation sharing (Screenshots of Powerpoint/LibreOffice are streamed in) - Video: Full HD (1080p) and Ultra HD (2160p), up to 120fps	No information on how the stream is recorded and in the configurator it is greyed out, but only audio is listed in the developer documentation	- Selective Forwarding Unit (many user supported, but only talking users are mixed into stream) - Only works on Chrome (multiple Tracks in one connection) - Audio is always mixed	- Works with Application Server (e.g. Kurento Application Server), which needs to implement JSR309 and Signaling - Communicates with Application Server using Kurento API (Java or JS) or Kurento Protocol (Websockets and JSON-RPC)	- Stream other resources into conversation: H.264 (RTSP), VP8+Opus (from file) - Architecture - Supports Erizo API (nodeJS) and Nuve API (python, ruby, js)	2 different Applications communicating over SIP on Websockets JSR309		Each service may be started/restarted/shu e.g. webrtc signalin msml video codecs allow parameters
			Tutorial exists on using flashphoner with Wowza	Uses XMPP for signaling and is available as an OpenFire plugin (which magnocall also uses)	Pluggable Pipelines filter, post-process and transcode media streams play recorded files	Programmed by University of Madrid	Needs a SIP application server to run, using Mobicents SIP Application Server is recommended Bitrate adaption algorithm		REST; JSR 309; J2E Converged Applicati Server (SIP Servlets MSML, NetAnn (Voil LTE)
Codecs	**Audio**	G.711, G.722, Opus, G.711, AMR, Speex	G.711, G.729, Speex, Opus	Opus, SILK, G.722, Speex, ilbc, G.711 (PCMU, PCMA), G.729	OPUS, AMR, SPEEX	Opus	G.722, Speex, Opus	G711 (PCMA/PCMU),G729, AAC DTMF: in band (RTP-NTE rfc2833) and out of band (SIP-NOTIFY/INFO	G.711u/a, G.723, G. G.729a, G.729b, GSM-FR, GSM-EFR AMR-NB, iLBC Opus, G.722
	Video	H.264, VP8	H.263, H.263+, H.264, VP8	H.263, H.264, VP8	VP8, H.264	VP8	VP8, H.264, H.263	H.264, VP8	H.264, VP8, H.263, H.263+, H.263++ Ba Profile

Cloud based solutions - Comparison table

Contains solutions that only work on vendor-owned hardware

Media Servers		Bistri	OpenClove (ex OCX)	OpenTok	Requestec (Zenon platform)	SightCall (ex weebo)	Twilio
			OpenClove Video Exchange	TokBox	Saypage Developer API		
short	blocker	no recording, only 4 participants	no	no	competing solution	no	no video
	pros	Plugin for IE and Safari	- Flash fallback for IE and Safari - Multiplexes all streams to one for phones - view-only streams (RTSP)	- intelligent quality control (traffic shaping) - relay session (direct connection between clients may be used to reduce costs)	- Big meeting rooms - Whitboard, Screen sharing - Optimization for mobile - Web interface	- Calls from and to non-users supported - Screen sharing	- Well document REST API - WebRTC and Flash client APIs available
	cons		- not licensable right now	- recording is only supported for routed sessions (needs to be decided when starting a session)	- competitor	- Pay per user (yearly or monthly)	- no video
Goals							
record		no	yes, download link is displayed after ending session	Recommended maximum: 5 streams; Up to 9 streams are supported (but quality might degrade). Archived streams may automatically be uploaded to e.g. amazon S3 Maximum length: 90 minutes	Yes (no info, though)	yes to cloud storage (buy separately)	10k minut free, then $0.0005 p min / mon Recorded is availabl via http download authentica needed)
8+ simult. participants		maximum of 4	9, upgradeable to 16	yes, save stream: max 9	Meeting rooms with up to 100 persons		limit: 40
ICE (stun, turn, negotiation, none)		STUN, TURN, negotiation	STUN, TURN, negotiation	STUN, TURN, negotiation	STUN, negotiation	negotiation	STUN, negotiatio
Signaling API		Proprietary (uses WebSocket)	Proprietary	Session ID and user token must be generated by external client	yes	yes	yes, base on xml
API to create rooms		yes	yes		yes	yes	yes, base on xml
API for User auth		no	yes	yes	yes	yes	no
Connect with SIP Clients		no	no		yes	no	yes
Connect directly to PSTN		no	yes (dial in costs may occur)	no	yes	no	yes

Costs and License		$30/month: 1500 calls/month $0.058 per additional call http://developers.bistri.com/webrtc-sdk/#sdk-pricing	To be announced http://developer.openclove.com/learn	pay per minute (minute is counted for every minute over turn-server), signaling or stun is for free. E.g. If 3 persons talk on their server for 5 minutes, 15 minutes will be billed. $50 per month (includes 10k minutes - next 90k minutes: 0.475Cent / minute - prices drop to $0.004/m) Additionally: Saving media costs ~200$ per simultaneous connection (flatrate), or $0.035 per recorded minute.	???	$2 per user per month, additionally $0.10 per call from non-users Recording: $0.10 per call (on own storage)	$0.0025 p minute
Other Information							
Benchmark			9 participants 16 participants max	for archive: best quality 5 streams (max 9 - might degrade quality)	Transcoding, etc. : C/C++		
Special		Plugin for IE and Safari	- Flash fallback for IE and Safari - Multiplexes all streams to one for phones	- intelligent quality control (traffic shaping) - https://tokbox.com/#iqc (fallback: audio only and last image) - relayed sessions are cheaper than routed sessions	Mixes streams for Users with low bandwidth	- Weemo driver is downloadable if WebRTC is not supported in Browser (IE 9+, Safari 6+)	- Dynamic phone number generation - Transcript ($0.05 per minute) (Speech t text) - WebRTC and Flash client APIs available
Codecs	Audio	Browser	Opus, G711	Opus	Browser	Browser	Browser
	Video	Browser	VP8, H.264	VP8	Browser	VP8	Browser

Appendix C: Experts' Comments on Connectivity Issues

to Jitsi

Hi All,

We are trying to connect to a meeting in https://meet.jit.si . But we are getting connectivity issues even for 3 participants in the meeting.

What is the reason for having connectivity issues. ?

What is the minimum required bandwidth for a participant if bandwidth is the problem?

Currently we have configured jitsi meet manually in our local server. We have same issues on that server also.

Then we checked by changing the MAX_INACTIVITY_LEVEL in video bridge. But still we are getting these connectivity issues.

Please let me know if you know something on this regard.

Saúl Ibarra Corretgé scorretge@atlassian.com via jitsi.org
to Jitsi

Hi Ravindu,

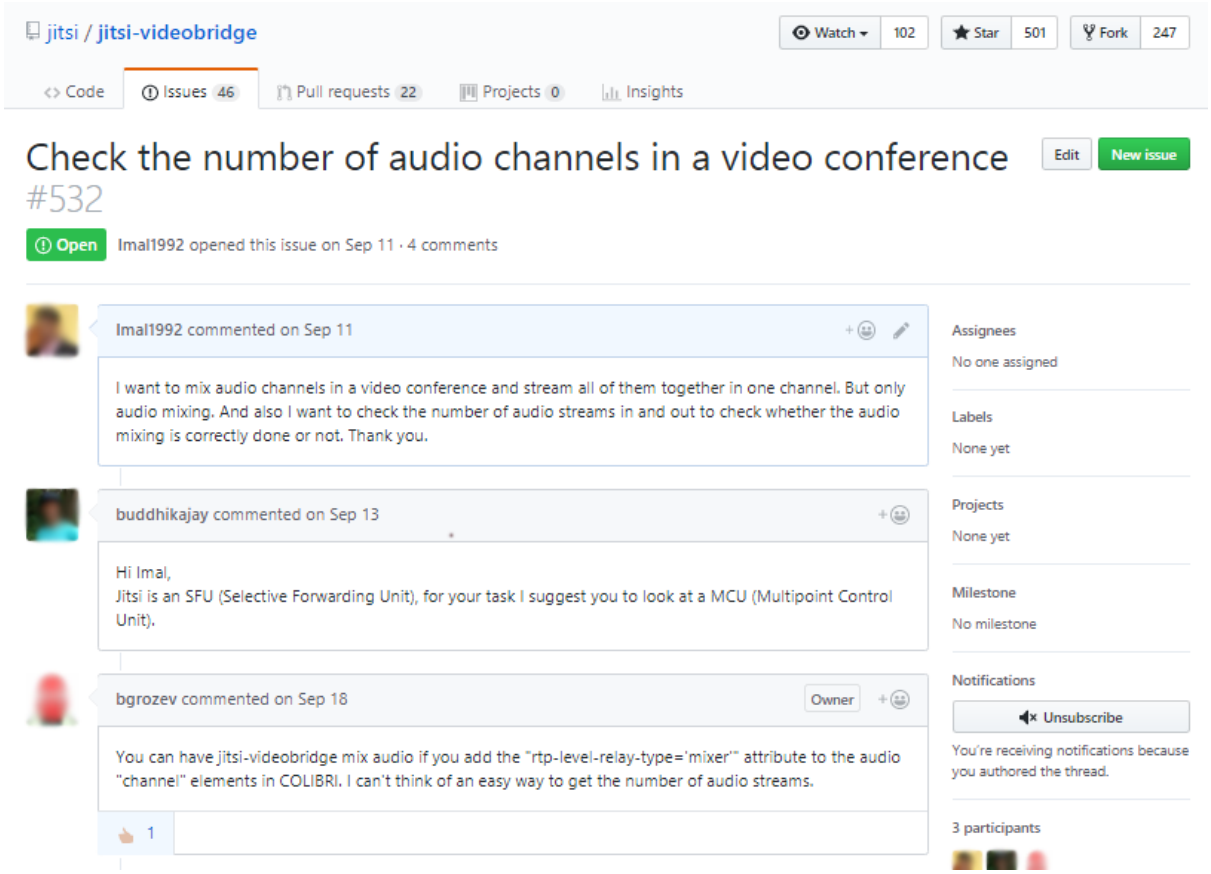
> On Oct 26, 2017, at 08:43, Ravindu Roshan <ravinduroshan.rr@gmail.com> wrote:
>
> Hi All,
>
> We are trying to connect to a meeting in https://meet.jit.si . But we are getting connectivity issues even for 3 participants in the meeting.
>
> What is the reason for having connectivity issues. ?
>

Usually packet loss. You can check the perceived packet loss by hovering the "signal bars" for the local participant.

> What is the minimum required bandwidth for a participant if bandwidth is the problem?
>

We will adapt to the detected bandwidth, so unless there is packet loss you shouldn't see any connectivity issues, the video quality will just degrade.

Appendix D: Experts' Comments on Audio Mixing



Appendix E: Code Level Change for Audio Mixing

```
<content name='audio'>
  <channel rtp-level-relay-type='mixer' direction='recvonly'
      initiator='true' id='c6a142b7cf728fd0' expire='60'>
    <source xmlns='urn:xmpp:jingle:apps:rtp:ssma:0' ssrc='3716204482'/>
    <transport xmlns='urn:xmpp:jingle:transports:ice-udp:1'
        pwd='5d0mj' ufrag='amiq'>
      <fingerprint xmlns='urn:xmpp:jingle:apps:dtls:0' hash='sha-1'>
        99:...:F6
      </fingerprint>
      <candidate type='host' protocol='udp' id='ca' ip='10.0.1.1'
          component='1' port='5144' foundation='3' generation='0'
          priority='2113932031' network='0'/>
      <candidate type='host' protocol='udp' id='ga' ip='10.0.1.1'
          component='2' port='5145' foundation='3' generation='0'
          priority='2113932030' network='0'/>
    </transport>
  </channel>
  ...
</content>
```

Appendix F: Usability Test Statistical Data

F.1 Test Data Summary of Expert Users

Questions	Strongly Disagree 1	Disagree 2	Neutral 3	Agree 4	Strongly Agree 5
Video was smooth and plays without lagging or stuttering (video jitter)	0	3	4	5	0
Both audio and video plays at the same time (real time audio video)	0	0	7	5	0
It was possible to recognize clearly the person who conducts the webinar.	0	3	5	4	0
It was possible to recognize the facial expressions of him/her (The conductor)	0	5	6	1	0
There were no connectivity issues during the webinar	1	7	4	0	0
Gained an understanding of the features of this video conferencing system	0	0	6	6	0
There is an advantage using this system over other video conferencing systems such as Skype and Hangouts	0	0	2	7	3
I have an understanding of the limitations of this video conferencing system	0	1	1	6	4
I will use this system in future for webinars or online teaching	0	2	5	5	0

F.2 Test Data Summary of Nominal Users

Questions	Strongly Disagree 1	Disagree 2	Neutral 3	Agree 4	Strongly Agree 5
Video was smooth and plays without lagging or stuttering (video jitter)	0	5	9	5	1
Both audio and video plays at the same time (real time audio video)	0	1	9	10	0
It was possible to recognize clearly the person who conducts the webinar.	0	1	7	12	0
It was possible to recognize the facial expressions of him/her (The conductor)	0	4	10	6	0
There were no connectivity issues during the webinar	0	8	9	3	0
Gained an understanding of the features of this video conferencing system	0	0	7	13	0
There is an advantage using this system over other video conferencing systems such as Skype and Hangouts	0	1	9	10	0
I have an understanding of the limitations of this video conferencing system	0	1	7	9	3
I will use this system in future for webinars or online teaching	0	2	10	8	0