Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 329 (2016) 105–121

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Videoconference System Based on WebRTC With Access to the PSTN

Alfonso Sandoval Rosas[1,2] José Luis Alejos Martínez[3]

Mobile Computing Laboratory, Telematics Academy, Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas (UPIITA)
Instituto Politécnico Nacional (IPN)
Mexico City, Mexico

Abstract

The research presented in this paper introduces Rendez-Vous, which is a system oriented to provide its users with a greater communication availability through the convergence of the conventional telephony and the real-time multimedia on the web browsers. The system consists on a web application that provides a videoconference room for multiple users without having to download any additional software. For the cases in which a user with no internet connection might need to participate in a conference, it is possible to dial and answer phone calls to/from the PSTN directly on the web browser, allowing the telephonic user to interact with the others using his/her voice. The aim of the present project is to propose a unified communications system that differs from others mainly in the interaction with the telephone network directly from a web browser while having an active videoconference, allowing real-time exchange of media streams between these two technologies. The system Rendez-Vous was implemented with the use of WebRTC (Web Real-Time Communications) for the transmission of audio and video on real-time, Node.js as a web and signaling server, as well as the software Asterisk for providing telephonic access, along with jsSIP, which is a JavaScript library for implementing a SIP User Agent.

Keywords: Videoconference, WebRTC, WebAudio, PSTN, SIP, Asterisk

1 Introduction

Currently, various agencies like companies and hospitals are increasingly opting for available videoconferencing systems to allow interaction with people in geographically distant locations, since communication through video brings benefits that voice or messages cannot meet themselves alone, such as body language, gestures,

[1] The authors wish to thank sincerely the National Polytechnic Institute (IPN) and the Professional Interdisciplinary Unit of Engineering and Advanced Technologies (UPIITA) for their support, especially the Mobile Computing Laboratory, belonging to the Telematics Academy for having provided since the beginning the installations and equipment required for the development of the project. Special acknowledgments to CLEARCOM COMUNICACIONES, S.A.P.I. DE C.V. for having sponsored the SIP trunk used during the project for academic purposes.

[2] Email: asandovalros@gmail.com

[3] Email: joseluis0791@gmail.com

http://dx.doi.org/10.1016/j.entcs.2016.12.007

and demonstration of emotions, finally resulting in a more clear and concise communication that achieves the objectives of the meetings with the same effectiveness as a classroom event [12].

However, in the cases where they have to participate in the videoconference but some of the participants may not have an Internet connection, or it is of a limited bandwidth, their participation in the videoconference would not be possible or it would have a deteriorated quality. There is thus the need to integrate other technologies to these multimedia systems to enable an acceptable interaction using a means of better access to an Internet connection: the PSTN (Public Switched Telephone Network). It is therefore necessary to design and implement a conferencing system that allows participation of a user's voice with a call from a landline or mobile phone.

The research presented in this paper introduces a system that consists on a unique videoconference room inside which three users can share audio and video using their web browsers without having to download any additional plugin. Another user can also participate with his/her voice in the active videoconference through a phone call from the PSTN. This solution makes use of the WebRTC (Web Real-Time Communications) API, which is part of the HTML5 specifications [9] and its main purpose is to enable two users gaining access to a web application to share voice, video and data between them, resolving network issues such as NAT (Network Access Translation) and establishing media types and formats with a negotiation prior to the real-time media exchange [19]. This faculty empowers the potential of the web-based systems allowing the integration with other real-time communication sources such as the telephonic networks.

The remainder of this paper is organized as follows. In Section 2 the background of this project is mentioned, plus gold projects with similar characteristics. The principles of our approach are explained in Section 3. The operation of the system is shown in Section 4, while in Section 5 the paper is concluded outlining some perspectives on future work.

2 Related Work

Prior to the existence of WebRTC, there were already many videoconferencing systems available on the market. One of the most comprehensive and focused on the business sector is Cisco WebEx [4], which is a robust remote collaboration solution composed of various services such as video conferencing rooms with file sharing features, desktop, presentation and dissemination. Unlike this system, the project in this document is based on WebRTC, so the users will not need to download any software to run it in the web browser. In addition, Cisco uses its own infrastructure to handle the telephone service [4]. The project of this document contemplates the usage of a SIP trunk, which means that an end user can use his own trunk if he already has one, and not rely on Cisco infrastructure and therefore their rates.

Outside of the corporative usage, many applications allow video conferencing. Popular examples include Skype [18], Firefox Hello [17] and Google Hangouts [8].

Skype is owned by Microsoft and uses a proprietary protocol for transmission of multimedia streams, plus it requires the installation of a mobile application or desktop to access the service. It also allows making phone calls to a subscriber balance, but it is not possible to integrate a call with an active videoconference. Firefox Hello is an application for video calls exclusively. It is based on WebRTC and is used from the Mozilla Firefox browser only. After the emergence of WebRTC, some companies have designed solutions based solely on this technology. A notable example is Talky [21], which allows the user to create video conferencing rooms in a simple way, as the URL is created anonymously and sent to those who want to participate. UberConference [22] is another example of integral solutions, as consisting on a voice-only conference in which the user can participate from a web browser or making a phone call to a number assigned by the system to log in.

The main difference between these solutions with the proposal presented in this document is the integration with the PSTN to participate by telephone with a voice session in an active videoconference. The objective of the project is to provide a unified communications solution for the cases in which the internet connection of a conference participant might not be available or not suitable for a proper session. For these situations, it is proposed to substitute the usage of the internet with a conventional phone call that connects to the web application.

3 Methodology

From the conception of the problem, taking into account today's usage of technology (which is agnostic to the kind of device or operative system), it was determined to develop the project as a web application for allowing the access to the videoconference room with the aid of a compatible web browser and nothing more.

Having the requirement to communicate the peers in this room between them with voice and video in real-time, the project was based on the WebRTC technology, making use of the client-side functionalities regarding the access to the local camera and microphone, as well as the handshaking process and consequent sharing of multimedia streams peer to peer [14]. WebRTC is better supported by the web browsers Google Chrome and Mozilla Firefox [20], and with each release more features are added in order to reach a common support for this technology.

The next requirement was to establish a voice communication between the web browsers and the PSTN, having as a main resource a SIP trunk with access to the telephone network. Since the WebRTC features of the browsers are controlled with JavaScript, it was possible to incorporate a SIP User Agent written in the same programming language, allowing direct interaction with an IP PBX server with the SIP trunk configured. Due to the popularity of WebRTC, IP PBX systems such as Asterisk have been incorporating full compatibility in their latest releases [6]. Therefore, it was only required to configure Asterisk properly to enable full interoperation between the web browser and the PSTN, having the IP PBX in the middle.

The implementation of the framework of this project is described next: The user

side of the system consists on a web application which has the WebRTC functions written in JavaScript required for the connection and real-time transmission of multimedia streams between users. The application is provided by a web server built with Node.js.

The users establish multimedia sharing sessions after having exchanged negotiation messages with the aid of a signaling server. The way these messages are exchanged and the structure itself of them follows a WebRTC standard. The API requires the construction of SDP (Session Description Protocol) messages which contain information about the media types to share, the IP addresses of the pair of users involved, the codec that will be used, and characteristics of the streams (amount of channels, amount of tracks, etc) [2][13].

For the connection with the PSTN, the access is obtained with an IP PBX server running the software Asterisk, which has a SIP trunk registered provided by a telephony carrier. For making or answering a phone call, an audio mixing and distribution module was designed. Its purpose is to collect and mix the voices of the users in the videoconference room with the aid of WebRTC sessions and the HTML API Web Audio. The mixed audio is sent to the IP PBX with a registered SIP User Agent programmed with jsSIP. Once the external audio is received, it is distributed to the users in the room. It is important to mention at this point that the prototype is limited for the moment to only three web users in the videoconference room as well as only one simultaneous phone call.

Figure 1 shows the main framework of the system.

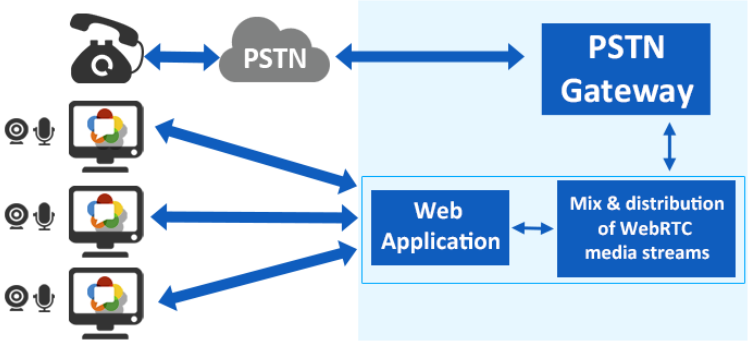

Fig. 1. Main framework of the system

The operation of the system goes as follows:

(i) A user logs in the web application using a compatible web browser and activates the videoconference room. In that moment, the user starts to share voice and video with everyone entering.

(ii) If another user logs in, its multimedia streams begin to be shared with everyone in the room as well.

(iii) All the users have in their web page a softphone with which they can make

phone calls to the PSTN. If a user dials a number and presses the Call button, the others are notified and their softphones are blocked.

(iv) An external user can dial a number associated to the telephonic services provider SIP trunk, placing a call in the system's IP PBX server. When a phone call is received this way, it is automatically answered and all users present in the room are notified.

(v) When the phone call's voice is received, it is incorporated in the room so the users can hear the caller, same which hears the voices of the users in return.

(vi) The videoconference room is deactivated until the last web user logs off. The room cannot be activated with a phone call.

Figure 2 shows the architecture of the system. The technologies and operation of each module are developed further in the document.

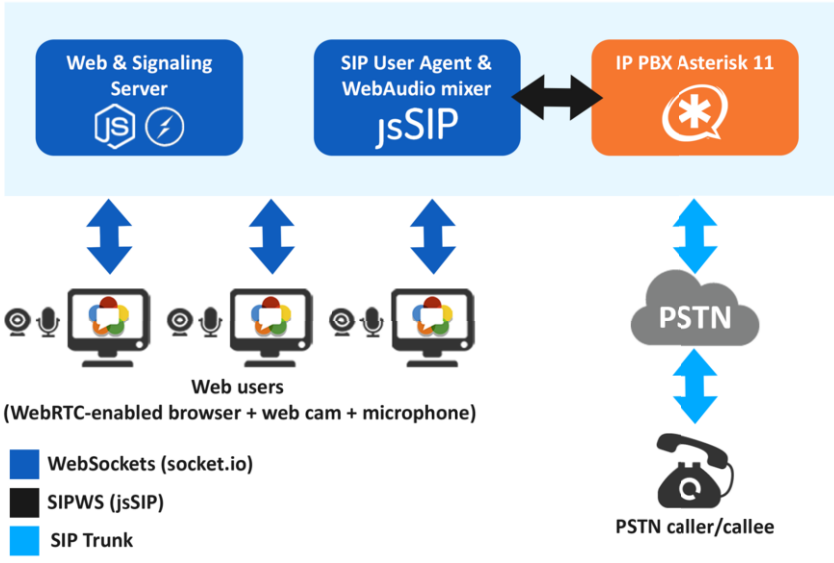

Fig. 2. Architecture of the system

The users in the videoconference room share voice and video from their device's microphone and web camera, while they only share the voice with the audio mix and distribution block. This module provides in response the external voice from the phone call, so the users perceive this module as another user in terms of negotiation messages by using WebRTC. The exchange of multimedia streams is shown in the **Figure 3**.

WebRTC introduces support for the codecs OPUS [16] for audio and VP8 [11] for video, and it defaults them in order to make use of their superior quality in comparison to other codecs, as well as their adaptability when changes in the bandwidth occur. Nevertheless, WebRTC supports other codecs such as G.711, PCMA, PCMU and so on. Another WebRTC feature regarding the media streams is the security. The streams are required to incorporate some kind of encryption before being sent to the other peer [10]. Web browsers supporting WebRTC use the DTLS mechanism

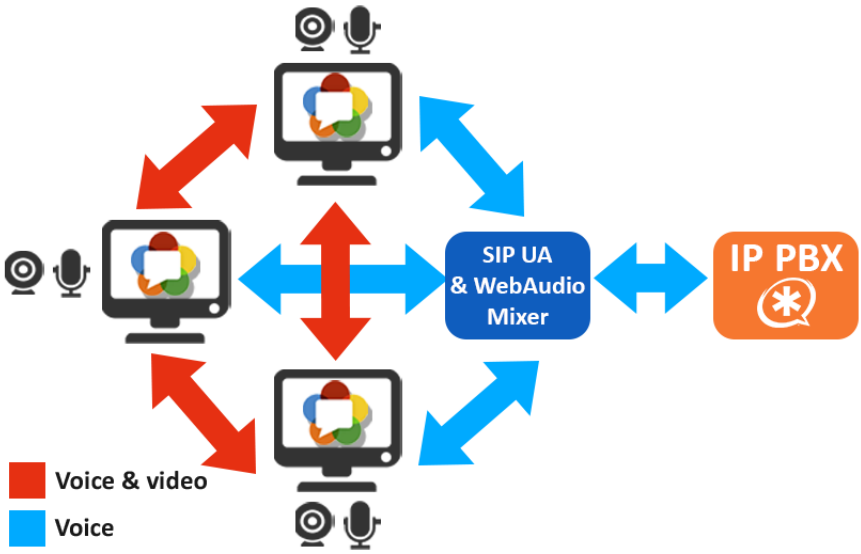

Fig. 3. Exchange of multimedia streams within the system

by default for counting with the specification [14].

Web and Signaling Servers

The web server provides the application for the users by entering a URL in a web browser compatible with WebRTC, while the signaling server is the link between the users for allowing them to send and receive messages prior to the real-time multimedia sharing. It is important to mention that once a couple of users is done negotiating the WebRTC with SDP messages, the streams are transmitted in a P2P (Peer to peer) fashion, given that both users know their location in the network already by having negotiated [19].

Audio mix and distribution module

This module works as an intermediary between the videoconference room and the IP PBX Server, which means that it manages the outgoing phone call requests of the users and the incoming calls from the PSTN. Before dialing or answering a phone call, this module must collect the voices of all users in the room. Afterwards, the voices have to me mixed in real-time, sample by sample, in order to be sent with the aid of a SIP User Agent on the browser to the IP PBX server and then to the SIP Trunk. This module of the system uses the WebAudio API for the voice mixing. This API is part of the HTML5 specifications [23] and includes audio processing capabilities in a modular fashion with inputs/outputs that can be used to design complex processing tasks.

The schematic diagram of **Figure 4** shows the entire design. In the diagram of the figure it can be seen that the input samples are divided in half for compensating the volume when performing the addition in the subsequent blocks. Since the WebAudio API blocks (apart from personalized scripts) do not receive more than a channel as input, it was required to split the stereo streams with the Merger

blocks. Finally, the blocks depicted as script are the ones that perform the mix of the input samples by receiving a stereo stream, storing the samples in the buffer aforementioned, and adding the value pair by pair. This part of the module converts the input voices to mono for processing them in personalized nodes called ScriptProcessor nodes in order to add each sample with a buffer of 16384 samples. The output is converted into the MediaStream format which is the one managed by WebRTC. After this, the module sends to the users in the videoconference room an "incoming call" or "dialing" DTMF tone to be played in the user interface while managing an incoming or outgoing call, given the case. When the picking of voices and mixing are done, and the external audio is received, these tones are replaced with the actual audio streams of the phone call, so users in the room shall reproduce then on the user interface, while the caller can hear and speak with all of them.

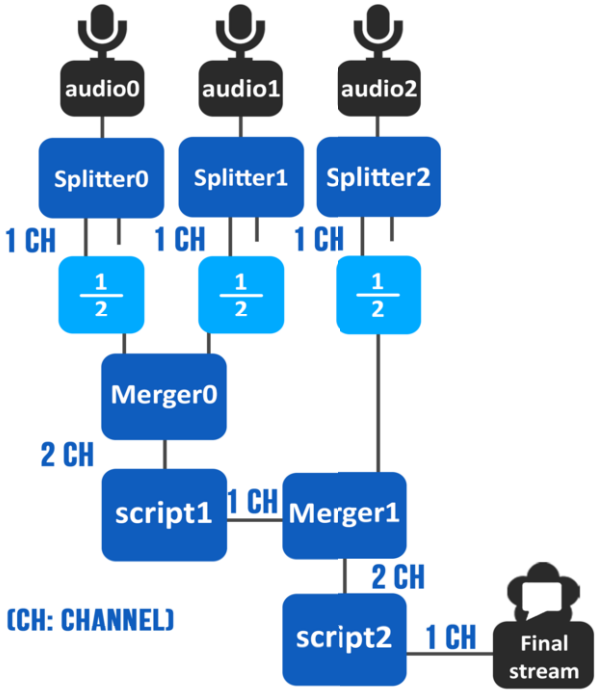

Fig. 4. Schematic diagram of Web Audio mixing system

Regarding the implementation of the SIP User Agent, it was developed as a script with the JavaScript library jsSIP, which contains the SIP dictionary required for registering and operating a SIP User on the client side of a web application [15]. The SIP User Agent registers in the IP PBX server with its own extension, so the dialplan identify it with its username and IP address, and may redirect the incoming/outgoing voice streams when a call is made or received.

In the system, the computer hosting the servers opens a web browser window which navigates to a web page as soon as the servers start running. This page is the

responsible for running this module since these are technologies that operate on the client-side [15] [23].

IP PBX Server

The open-source IP PBX server software Asterisk is used for providing the web application with a connection to the PSTN. Since its version 11, Asterisk incorporates WebRTC functionalities which allow it to send and receive multimedia streams having established communication via SIPWS (SIP over Websockets)[6]. This means that Asterisk can treat a SIP Agent based on web technologies the same as if it was a hardware IP phone or a softphone desktop application. For enabling the WebRTC features in Asterisk, it is necessary to install the libsrtp module, since its function is to add a security layer to the media streams by encrypting them after having generated keys and certifications [1]. This is necessary due to the imposition of WebRTC that regards the security, and it establishes that all media streams shall be encrypted using SRTP (Secure Real-Time Protocol) [1].

The IP PBX server requires certain ports to be open and accepting requests in order to enable inward and outward dialing. With the Asterisk configuration files, the port 5060 was set to receive and transmit SIP over TCP messages, while the port range from 10000 to 20000 was set for RTP exchange. The web SIP User Agent was built and incorporated into the web page. It is registered in the IP PBX Server and is the only extension which can dial and receive phone calls. These calls are connected with the PSTN making use of a SIP trunk provided by a telephonic carrier. There is a limitation of only one active call at a given moment since the SIP trunk had only a DID (Direct Inward Dial) number available.

4 Tests and Results

Testing scenario

The main purpose of the tests is to demonstrate the integration of the two communication technologies: the PSTN and the Internet real-time media streaming, into a unified solution, making use of the powerful features of modern web browsers, and opening a proposal in terms of communications services at a regular customer and enterprise level. Whether a user logs in the web application, or makes/receives a phone call with any telephonic device, the system aims to demonstrate the ability to establish a functional and real-time media interaction between all the participants, regardless of their location or means of access.

The test scenario includes a network installation on a (LAN) local environment, which connects the Web and Signaling Server, the Asterisk IP PBX with the SIP trunk configured, as well as the web clients. An outlet for Internet connection in the IP PBX server is required for registering and interacting with the SIP trunk. The connection to the servers and between the users is done via Ethernet connection to a switch.

The complete scenario is shown in **Figure 5**. It is necessary that the computers functioning as web users in the videoconferencing room have a webcam, microphone

and a web browser that supports WebRTC, regardless of the operating system. The computer acting as a web / signaling server has port 3000 open to serve HTTP requests. On that computer the aforementioned servers are running simultaneously using Node.js on a Fedora operating system.

The computer equipment that functions as IP PBX server has two Ethernet cards to connect simultaneously to both the rest of the network via the central switch, such as accessing the Internet of the Mobile Computing Lab to gain access to the SIP trunk. This computer hosts the Asterisk server on a Linux CentOS 6 operating system.

Since this project has been developed in the Mobile Computing Laboratory facilities, belonging to the Academy of Telematics UPIITA, it is necessary to have a public IP address with a number of open ports to enable the realization / receive phone calls using the SIP trunk. Ports that are required are the 5060 SIP over TCP, in addition to the 10,000 to 20,000 range for multimedia transport by Asterisk.

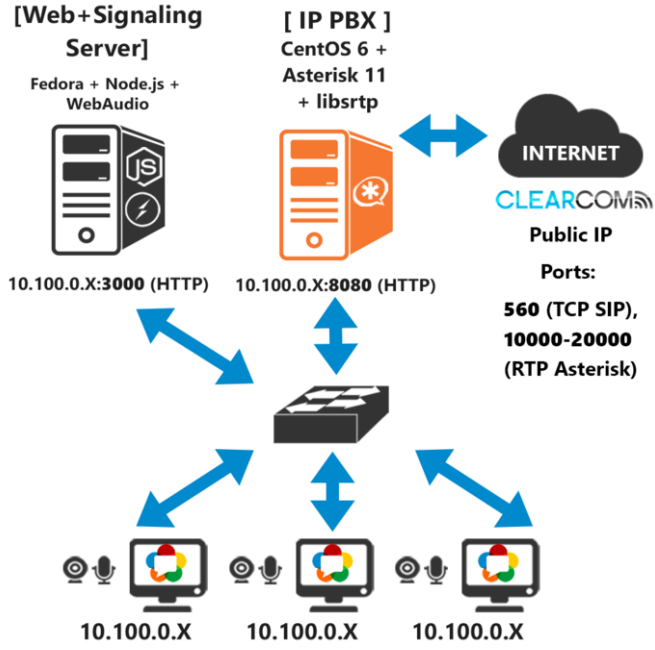

Fig. 5. Testing scenario

Results

In order to put the system up and running, the Asterisk server needs to be initiated before any other element. It also has to be registered in the SIP trunk. Having verified the aforementioned, it is necessary to turn on the web and signaling server using a command prompt.

On the server screen, an administrator panel in a web browser will show up as soon as the services start running. **Figure 6** shows this panel, where three log

consoles can be distinguished. The Server logs contain messages about incoming requests and closed sessions. The PBX Client logs show information about requests for making phone calls, dialed numbers and incoming calls from the PSTN. Finally, the SIP log shows all the SIP messages delivered by the IP PBX server during calls dialing, establishment and ending.

A user shall access the web application using a compatible web browser. A login screen is displayed requesting for a username, which will be displayed once entering the videoconference room. If the given username hasn't been used by anybody else in the room, the user is redirected to the main panel. **Figure 7** shows the elements that compose it. The username is displayed in the upper part along with a button for ending the session. The blue, red and gray mosaics will display the video of the user itself and the other users in the room. There is a log in the middle showing messages about the current events.

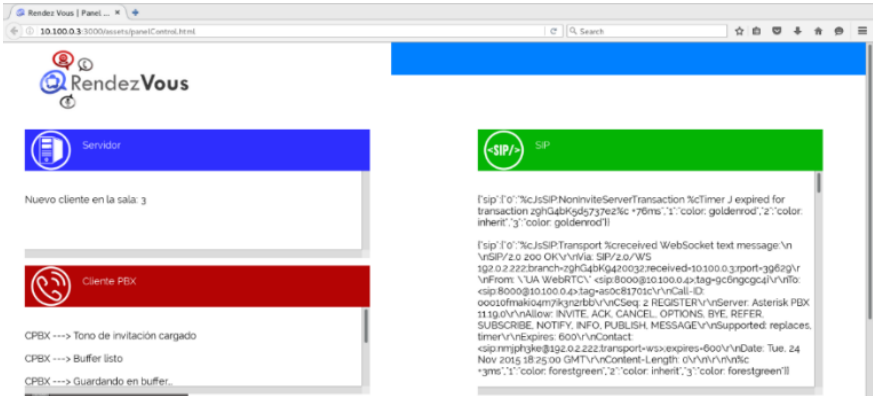

Fig. 6. Administrator panel on the Web/Signaling Server

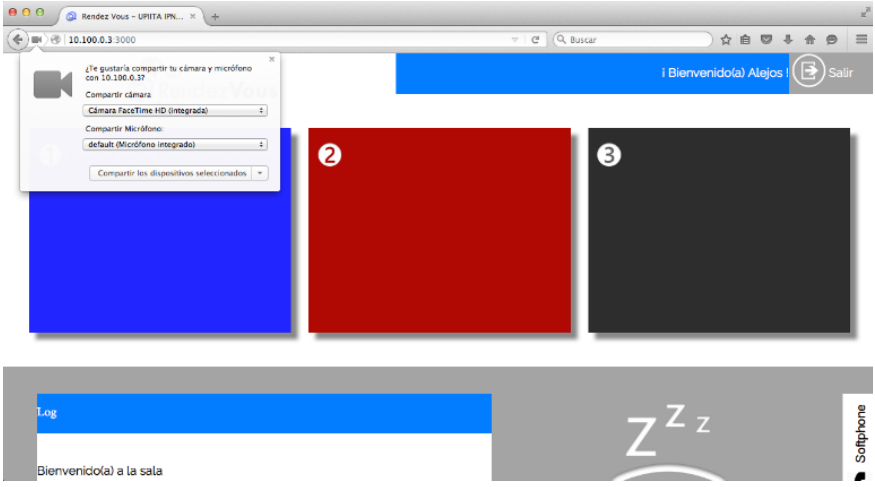

Fig. 7. Videoconference room web page

As soon as the user is redirected, the web browser requests for access to the computer's microphone and web cam. If the user grants the access, the video of the

web cam will immediately appear on the mosaic that corresponds to his order of appearance. **Figure 8** shows the local video displayed on the first mosaic.

Given that the prototype is restricted to a maximum of three users, if a user tries to login when the videoconference room is full, the error message "The room is full. Please try later" will be displayed. If the username has already been used by another member in the room, the screen displays the message "The input name is already on use. Please try with a different one". Once in the main page, when other users login, their media streams are automatically shared between each other. That scenario is shown in the **Figure 9**.

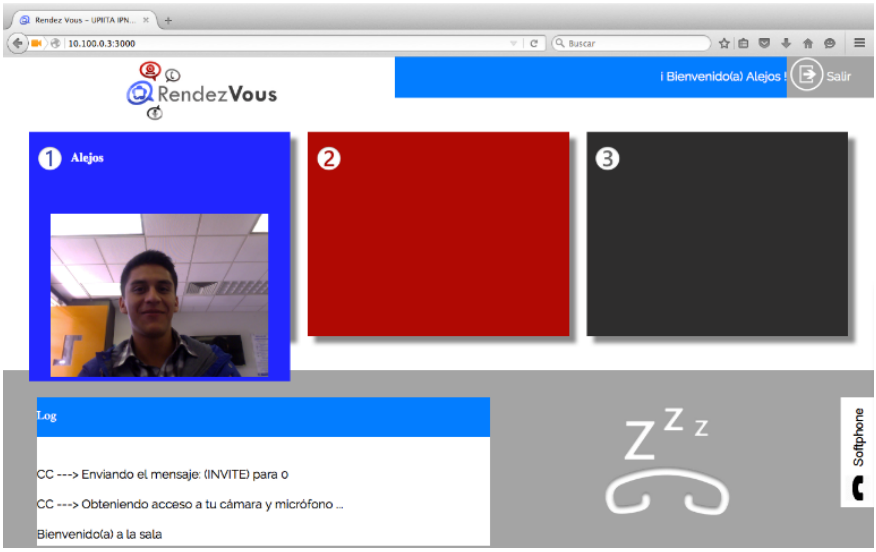

Fig. 8. Local video displayed on a mosaic

If one of the users wants to dial a phone number, he can use the tab on the right side of the page for showing a softphone. **Figure 10** shows the input of a telephone number. While the call is being stablished, the softphone tab of all users except the one who dialed is locked. The telephone icon in the right side of the pages of all users changes in order to notice that a call is being established. The dialed phone number is also displayed, and the DTMF dialing tone is played. **Figure 11** shows the main page during this process.

Once the call is established, the DTMF tone is over, giving step to the remote voice stream on each user's browser, so they can speak between them and with the telephonic user, and he can hear and speak in return to all of them. The call can

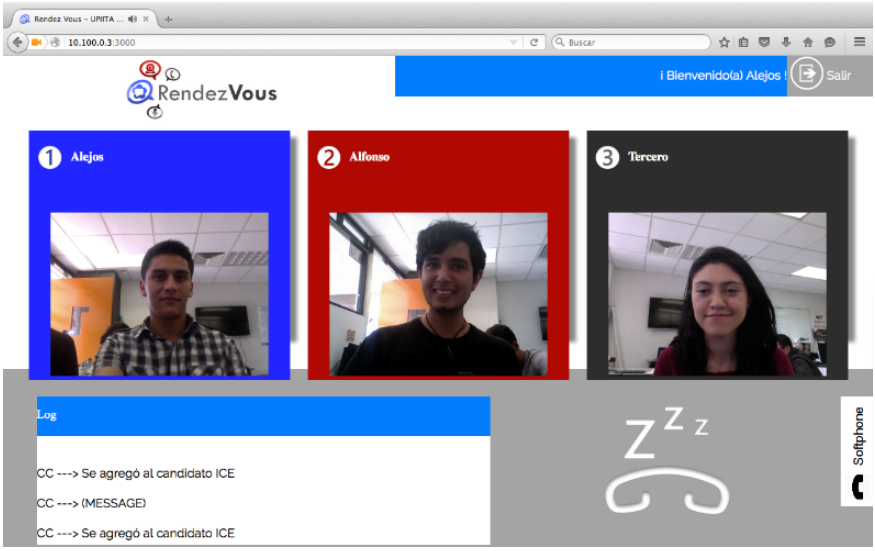

Fig. 9. Three web users in the videoconference room

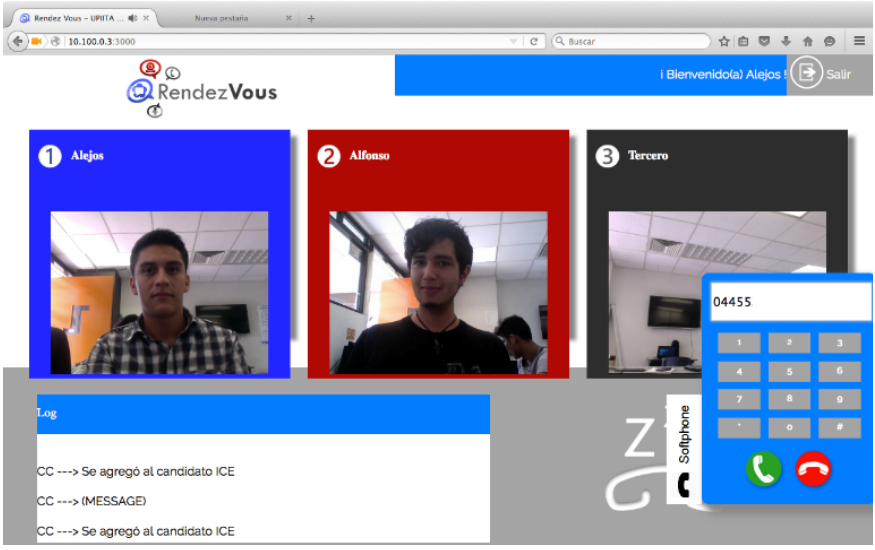

Fig. 10. Dialing of a phone number from within the web app

only be hung up either by the PSTN callee or by the web user who dialed it, since he will be the only one with the softphone unlocked.

When a remote telephonic user dials the DID of the SIP Trunk, the web users are requested for access to their microphones. This is requested by the Mix and Distribution block for collecting all the voices and sending them to the IP PBX server. The telephone icon on each user's page shows the number of the caller. **Figure 12** shows the event.

This call can be hung up by anyone in the room. Finally, **Figure 13** shows the main page when a user leaves the room, either by clicking on the logout icon in the right upper corner, or by quitting the web browser window. The mosaic shifts to a

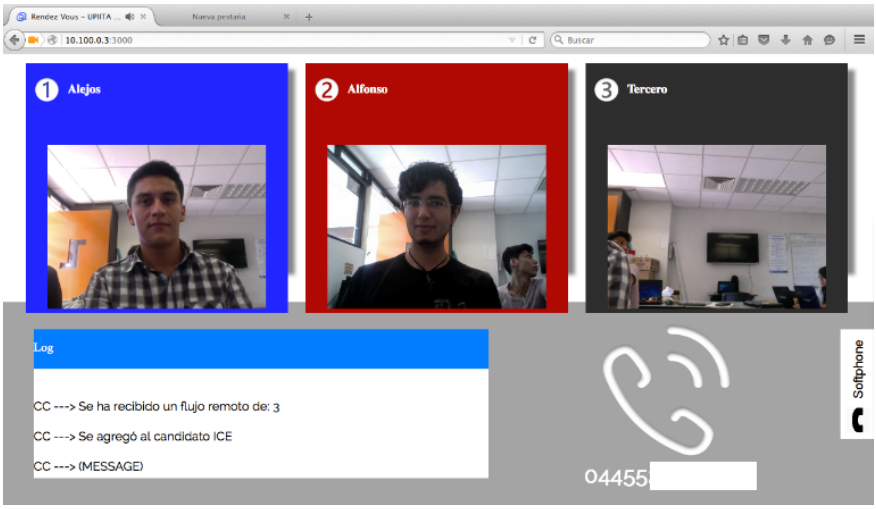

Fig. 11. Main page while establishing a dialed phone call

solid color.

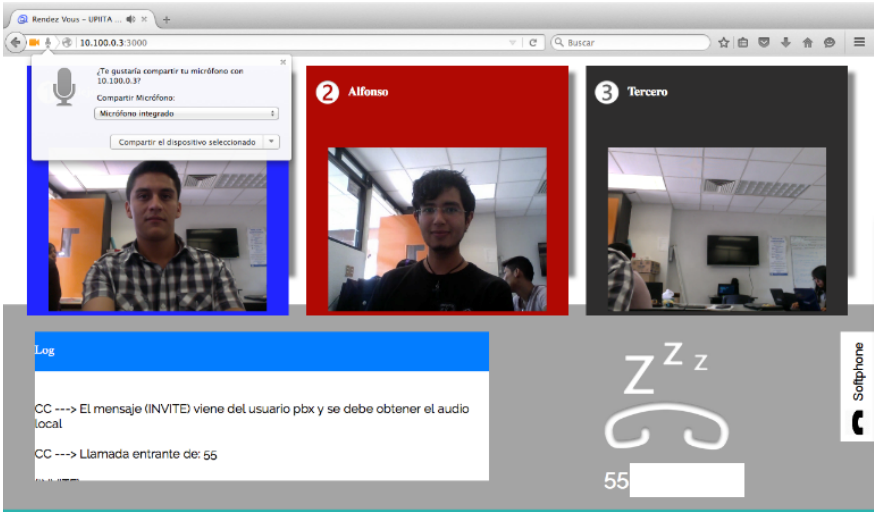

Fig. 12. Inbound PSTN call in the web app

The SDP messages exchanged between the users when establishing a WebRTC session showed the collection of codecs used for audio and video. The audio codecs supported by the web browsers were opus, G722, PCMU and PCMA. The corresponding collection of video codecs supported by the web browsers limits to VP8 and H264.

In order to examine the behavior of the web browsers when transmitting and receiving media streams from different users with WebRTC, certain values were consulted with the aid of the developer mode of Google Chrome and Mozilla Firefox. **Table 1** shows the performance of each web browser in terms of sent and received packets, jitter and coding/decoding rates when having 2 web users along with a PSTN active call:

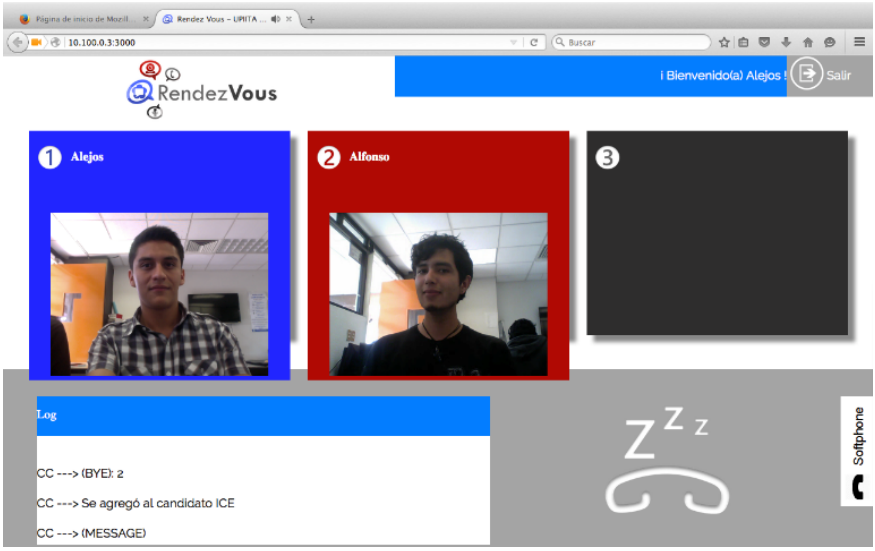

Fig. 13. User logged out

Outgoing video: coder	**Outgoing audio**
Average bitrate: 0.35 Mbps **Frame loss rate**: 25.11 fps	**Sent packets**: 433 (29.64 Kb)
Incoming video: decoder	**Incoming audio**
Average bitrate: 0.31 Mbps **Frame loss rate**: 24.78 fps **Received packets**: 471 (391.19 Kb) **Lost packets**: 23 **Jitter**: 0.778	**Jitter-buffer delay**: 48 ms, **Received packets: 406 (23.72 Kb)**

Table 2 shows the performance on the same scenario but when having 3 web users along with a PSTN active call:

Outgoing video: coder	**Outgoing audio**
Average bitrate: 0.65 Mbps **Frame loss rate**: 24.83 fps	**Sent packets**: 1114 (75.62 Kb)
Incoming video: decoder	**Incoming audio**
Average bitrate:0.72 Mbps **Frame loss rate**: 27.04 fps **Received packets**:(1740.87 Kb) **Lost packets**: 24 **Jitter**: 1.113	**Jitter-buffer delay**: 120 ms, **Received packets**: 1112 (61.59 Kb)

Given that WebRTC implements a mechanism for avoiding NAT issues, the IP addresses it determines as alternative paths in case of NAT presence (formerly known as ICE candidates, which means Interactive Connectivity Establishment) are encapsulated in UDP datagrams, thus generating the presence of jitter. Comparing the two performance tables, it can be highlighted that phenomena such as the network congestion increment the jitter. The adaptability of the codecs employed can be also verified since the coding/decoding rates are different depending on the status of the network. The web browsers also make decisions regarding the jitter value by

assigning higher jitter buffer values.

Project Scope

As a result of the tests performed on the integrated system, there are the following limitations:

- Customers can only use the Mozilla Firefox browser from version 42.0 on. The development, implementation and system integration were compromised by the maturity of the WebRTC API. Since the W3C consortium is responsible for issuing recommendations on how should be the attributes, classes and methods of this API, they are manufacturers of web browsers responsible for implementing them. Therefore, given the relative youth of WebRTC, not all recommendations have been implemented yet by all browsers. Since this system requires the use WebRTC renegotiation functions, it is necessary to use Firefox because it is the only browser that so far the implemented.
- To make phone calls having more than one user in the room, the web / signaling server must open its control panel with the Firefox browser in beta (Nightly). Firefox in version 42.0 has some incompatibilities when sending SIP messages.

5 Conclusions and Future Work

In this project, the design and implementation of a web application that would allow the establishment of a videoconferencing room with the ability to make and receive calls to / from the PSTN, being primary the usage of WebRTC was set as a target. It was also intended to demonstrate the scope of WebRTC to unify two media: the telephone network and the Internet. In summary, it is concluded that the overall objective and the specific objectives of this project have been successfully completed, since the functional implementation of a videoconferencing room was achieved, along with a link to the PSTN.

It was concluded that the mesh architecture was sufficient to implement the system, being necessary to integrate a web / signalling server and an IP PBX server. Regarding the aforementioned elements, the choice of Node.js and socket.io for the web / signalling server, the SIP User Agent with jsSIP and Asterisk 11 for the IP PBX server was adequate. It was possible to establish a clear communication in video conferencing and telephone audio among four users, the first three were web users and the last one a telephone user.

Being necessary to have installed and configured an IP PBX server, it was discovered that there is still some lack of information and support on how to enable support of Asterisk with WebRTC. In the final stage of testing, it was necessary to use Asterisk 11 in a server with CentOS 6 operating system because only in this way managed to use the SIP User Agent from a web browser. The problems mentioned related to the web browsers pose a lack of maturity in the implementation of the recommendations of WebRTC. However, given the huge support from the community computer and telecommunications industry, academic and scientific level, in each

new version of the browsers new deployments and greater compatibility between them is incorporated. Naturally, the point at which all browsers implement all the recommendations in the WebRTC draft maintained by the IEFT will be reached.

For future work, it is required to take into account the following considerations: To scale the number of users supported by the system to an enterprise scale, it is necessary to incorporate an MCU server (Multimedia Control Unit) [5], or implement a proprietary solution to mix video streams and voice into one stream, so that the connection establishment shall not require a point-to-point connection for each new user.

The dynamic creation of various videoconferencing rooms would also be necessary. Although Node.js supports multiple socket identifiers, which would allow only certain users to communicate with each other, the real problem relies on the telephonic services in terms of amount of digital lines, DID numbers and a robust dialplan in Asterisk, which would involve IVR (Interactive Voice Response) dynamic menus for pointing the user to the adequate room.

Following the actual tendency of software availability in the cloud, the project of this document could be scaled to a SaaS (Software as a Service) level since it is web-based and today's cloud platforms provide all the tools required to mount all the servers that compose the system, transforming it into a UCaaS (Unified Communications as a Service) [3]. To allow client access to the system on a larger scale like the Internet, it would be necessary to host the application and the Asterisk server in a cloud computing service. Vendors such as Windows Azure, Amazon Web Services and Rackspace allow the installation and operation of Asterisk on its platform, in addition to the execution of the application on a server Node.js.

Finally, since security regarding the web systems is an actual concern nowadays, it would be necessary to take action in terms of safe transport of the media streams and user logging mechanisms [7]. The WebRTC specification incorporates security rules in response to this concern, but additional techniques such as a higher level of payload encrypt shall be required.

References

[1] Altanai, "WebRTC Integrator's Guide", Packt Publishing, 2014.

[2] Antón, R., "Anatomy of a WebRTC SDP", WebRTC Hacks, 27 January 2015. https://webrtchacks.com/sdp-anatomy/.

[3] Bertin, E. and Beltran, V., *Unified communications as a service and WebRTC: An identity-centric perspective*, Computer Communications, vol. 68, pp. 73-82, 2015.

[4] Cisco Systems, "Cisco WebEx Web Conferencing, Online Meetings, Desktop Sharing", http://www.webex.com/.

[5] Cola, C., *On multi-user web conference using WebRTC*, System Theory, Control and Computing (ICSTCC), pp. 430 - 433, 2014.

[6] Colp, J., "Asterisk WebRTC Support", March 8th 2014. https://wiki.asterisk.org/wiki/display/AST/Asterisk+WebRTC+Support.

[7] Georgios Karopoulos, K., *Security and privacy in unified communications: Challenges and solutions*, Computer Communications, vol. 68, pp. 1-3, 2015.

[8] Google Inc., "Hangouts - Google", http://www.google.com/+/learnmore/hangouts/?hl=es-419.

[9] IEEFT, W3C, "IEFT and W3C draft about WebRTC", https://tools.ietf.org/html/draft-ietf-avtcore-rtp-topologies-update-04.

[10] IEEFT, W3C, "WebRTC Use Cases and Requirements," http://tools.ietf.org/html/draft-ietf-rtcweb-use-cases-and-requirements-10.

[11] IEFT, "VP8 Data Format and Decoding Guide", November 2011. http://tools.ietf.org/html/rfc6386.

[12] Johnston, A., *Taking on webRTC in an enterprise*, IEEE Communications Magazine, vol. 51, n° 4, pp. 48-54, 2013.

[13] Loreto, Salvatore., "Real-Time Communication with WebRTC", O'Reilly, 2014.

[14] Mason, R., "Getting Started with WebRTC", Packt Publishing, 2013.

[15] Millán, J. L., Baz Castillo,I. and Ibarra Corretgé, S., "jsSIP: The Javascript SIP library", 2011. http://jssip.net/.

[16] MIT, "Opus Interactive Audio Codec", http://opus-codec.org/.

[17] Mozilla Foundation, "Firefox Hello", January 2016. https://www.mozilla.org/es-MX/firefox/hello/.

[18] Skype, "Skype | Llamadas gratis a familiares y amigos", www.skype.com/es/.

[19] Sredojev, B., *WebRTC technology overview and signaling solution design and implementation*, Information and Communication Technology, Electronics and Microelectronics (MIPRO), pp. 1006 - 1009, 21 May 2015.

[20] Talky, "Is WebRTC ready yet?", May 2016. http://iswebrtcreadyyet.com/.

[21] Talky, "Talky", https://talky.io/.

[22] Uberconference, "Uberconference", https://www.uberconference.com/.

[23] W3C, "Web Audio API, W3C Editor's Draft", 2012. https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html.