International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 1, February 2021, pp. 567~574
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i1.pp567-574

Multi-user media streaming service for e-learning based web real-time communication technology

Naktal Edan, Sanabil A. Mahmood
College of Computers Sciences and Mathematics, University of Mosul, Iraq

Article Info

Article history:

Received Dec 24, 2019
Revised Jun 15, 2020
Accepted Jun 28, 2020

Keywords:

4 generation (4G)
Explorer
Firefox
Google chrome
Internet
Mesh topology
Opera
Quality of experience (QoE)
Web real-time communication (WebRTC)

ABSTRACT

Web real-time communication (WebRTC) standards do not define precisely how two browsers establish and control their communication. Therefore, a signalling mechanism/protocol has not specified in WebRTC. The essential goal of this research is to create and apply a WebRTC bi-directional video conferencing based on mesh topology (many-to-many) using Google Chrome, Firefox, Opera, and Explorer. This experiment involved through Ethernet and Wireless of the Internet and 4G networks in e-learning. The signalling mechanism of this experiment has been created and implemented using JavaScript language along with MultiConnection libraries. In addition, an evaluation of quality of experience (QoE), resources, such as bandwidth consumption, and CPU performance was done. In this paper, a novel implementation was accomplished over e-learning using different networks, different browsers, many peers, opening one or many rooms concurrently, defining room initiator, sharing the information of the new user with participants, using user identification (user-id), and so on. Moreover, the paper also highlights the advantages and disadvantages of using WebRTC video conferencing.

Corresponding Author:

Naktal Edan,
College of Computers Sciences and Mathematics,
University of Mosul,
Mosul, Iraq.
Email: edannaktal@gmail.com, naktal.edan@uomosul.edu.iq

1. INTRODUCTION

The internet engineering task force (IETF) (wire protocol) and World Wide Web consortium (W3C) (browser API) have been developed a modern collection of standards, JavaScript libraries, and open source called Web real-time communication (WebRTC) [1, 2]. WebRTC provides interactive multimedia and data communications [3]. Also, it offers several benefits such as no fees, no license, no requirement for external software, and simplicity of use [4]. On the other hand, no agreement has been reached regarding the W3C and IETF as a full protocol to examine WebRTC [5]. Thus, the standard of the signalling channel was not specified in WebRTC [6]. In other words, WebRTC standards do not define precisely how two browsers control their communication [7]. The reason behind that is signalling presumes to be left to the developers to create an application or to choose from the existing protocols (e.g. extensible messaging and presence protocol (XMPP)), or to create their protocol [8-10]. Signalling is crucial to determine users and sets audio and video communication among them; it also enables exchanging data among users via channels [1]. Furthermore, signalling supports browsers to connect each other through a server [11].

In [12], mentioned that bidirectional videos between a server and a client could be supported by Socket.io (API). Besides, Socket.io is a concept for WebSockets with XMLHttpRequest (XHR), Flash and JSONP, and it permits developers to utilise WebSockets and determine different synchronised communication techniques managed by the client's browser [13, 14]. Accordingly, this signalling mechanism using JavaScript language and MultiConnecation libraries provide the follows:

- Bi-directional multimedia communication for many-to-many (mesh topology), so audio and video communication can share between a teacher and students, and among all students.
- Various type of communication, so all peers able to be an initiator or participant.
- Keep a session productive even other students leave.
- Connect current session or establish a new one.
- Stop own/remote transmissions.
- Using user-id to prevent the unknown peer joins the room, etc.
- It can be applied to different communications, for example, e-Health (between doctors and patients), and e-learning (between teacher and students) and so on.

The organisation of this paper is as indicates: section 2 discusses related work. In section 3, is a presentation of the methodology, application and analysis. Section 4 elaborates the evaluation. To conclude, the conclusion and future effort have mentioned in section 5.

2. RELATED WORK

It has been specified in [4, 6, 13-15], that signalling has been depicted as the core part of the application, which has not been stated in WebRTC between browser and server. Therefore, the signalling protocol has to be provided in the application level by the developer [7, 16]; and it must be agreed on between the two entities either with the central node or with the other user [17]. Primarily, it can be emphasised that the protocol between browsers and server has not yet standardised in WebRTC [3, 18]. Not only that but also, the architecture of client-server does not appear to be a reasonable solution in WebRTC [19].

In [20, 21], indicated that videoconferencing using WebRTC technology is required as long as, different hospitals, economic and social levels, language practice, and companies have needed videoconferencing to interact with people [22-24]. However, WebRTC lakes high-end of video conferencing [25]. Therefore, many types of research have designed WebRTC video calls using XMLHttpRequest (XHR). Nevertheless, using this protocol led to consuming bandwidth and high delay [26]. Also, XHR does not support bidirectional communication [12]. Likewise, various developers tried to apply the session initiation protocol (SIP) to WebRTC for gaining video calls, but the installation of plugins is still required for SIP [20]. Besides, SIP has no protocol that can work with WebRTC [9, 27]; and SIP requires some enhancements and leads to delay and bandwidth consumption [27, 28]. On the contrary, in [29], proposed a centralised architecture based on the multipoint control unit (MCU) to support WebRTC video conferencing.

On the other hand, the authors have not debated the signalling instead of explaining how the proposed structure offers resolutions like stream processing. In addition, in [30] illustrated that using MCU is very expensive; and uses as a rented device during conferences. Adding to that, [31] confirmed that MCU does not support a large number of participants and supports heterogeneous endpoints.

In [7], suggested a videoconferencing system to offer a P2P connection using XMPP server and WebRTC. However, the user was not able to access the signalling without installing external software, and the system enables two peers to communicate with each other. Furthermore, another application of WebRTC videoconferencing was designed to support a video call for multi peers, but, it supports only two peers [32]. What is more, [33] deliberate a system of video chatting for seniors utilising WebRTC, the system was applied to video chatting between two peers. Besides, in [34] implemented a suggestion based on JavaScript language and mesh topology; nevertheless, it offers communication between two users. Consequently, WebRTC videoconferencing for multi-users is required and should be designed and implemented.

3. METHODOLOGY, IMPLEMENTATION, AND ANALYSIS

3.1. Methodology

This implementation has been designed and tested using HTML5, CSS, JavaScript code, and MultiConnection libraries. Also, Wireshark analyser (to find out the bandwidth consumption), a task manager (to evaluate a CPU performance), and access point-NetComm wireless (to provide 4G) were used. Furthermore, Google Chrome, Firefox, Opera, and Explorer were utilised as a client-side. Additionally, many computers of different CPU cores as (CPU i5, i7 & different RAMs) linked via various locations through (Ethernet and Wireless) of the Internet and 4G networks as described below: (a) one computer connected through the 4G network, (b) two computers connected to Internet through Ethernet network, and (c) five computers associated to the Internet via a wireless network.

3.2. Implementation

A network lab was generated to test WebRTC multimedia communication between different students and teachers via (Ethernet & Wireless) of the Internet and 4G set up in a mesh topology. In this achievement, a signalling mechanism has been created based on JavaScript language and MultiConnection libraries, also have divided into two parts:

- Build and adjust the main browser.
- Utilise MultiConnection libraries in order to set up, establish and end a communication among students and teachers.
- Create "room-id", which appears by default as a mixture of numbers, and letters where it uses as channel name or label that should be controlled by the initiator.

According to the main web browser, the critical web browser of this test has many features such as create and send SDP-offer/answer, permits a user to enable the camera and microphone, open or join a room, mute-audio or video and utilise full-screen. It was designed with one button to allow peers to open or enter the room. Also, MultiConnetion library was used to initialise a constructor in order to: (a) Set a session type as video conferencing, (b) Setup SDP video directions as bi-directional streaming, (c) Add quick event handler (On-click) from the mentioned buttons, and (d) Link JavaScript code to be used as a signalling mechanism that handles communication between participants.

The MultiConnection library has been used to initialise and set up a new session for video conferencing. Also, it has added "onstream", "function" and "event" object to offer native and remote streams using a unique stream-id. More importantly, after applied this signalling mechanism, it has been found that it depends on a different number of techniques, such as send an offer, receive a response, exchange data, open and close channel, using a socket in order to determine which communication technique is suitable for each user. For instance, Socket.io can offer several communication techniques instead of JSON. In [14, 28], stated that using Socket.io leads to deal with server file and client file at the same time, including it can decide if the connection establishes with AJAX long polling, Flash or WebSocket. In like manner, server-side and client-side components can be provided using Socket.io [29].

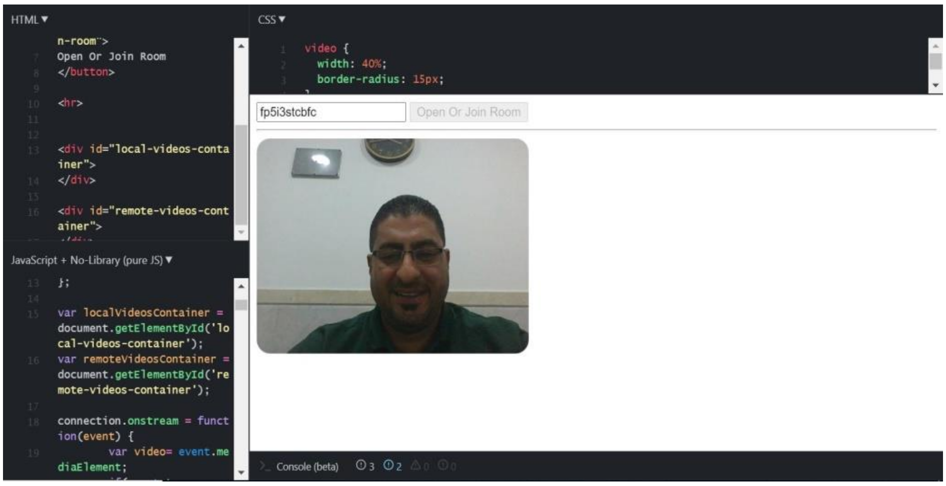
Room-id is a very significant aspect for initiating and joining the room; it is used to make sure relevant-messages are shared with relevant participants, and these messages are not accessible by other students. Therefore, all students must possess the same "room-id" to send a participation request and enter the same room. Otherwise, they will open a new room or join another one. In this implementation, communication has one user to start the session who specifies "room-id" and various participants who must use it. Moreover, many rooms can be opened but by using different "room-ids". When the initiator creates a new room, it needs to create a new socket. A new socket can be used to obtain various things, such as SDP-offer and an ICE gathered by the initiator and other participants. In other words, each peer should transmit their details over default-channel, create and exchange "SDP-offer/answer". At the same time, they are expected to play the role of “offeror" for any new participant after opening a new peer connection. On the contrary, each participant joining the room is expected to play the role of “answerer".

When an initiator opened the room after determining room-id, it will show multimedia “MediaStream”. A “MediaStream” can be gained through utilising “navigator.getUserMedia” method in order to use the camera and microphone for capturing screen. Upon granting the accession; a camera initiates streaming, and then the execution can be ready for joining. All participants need to invoke "getUserMedia" and share cameras and microphones. Figure 1, shows the main browser using Google Chrome, Figure 2 presents the main browser using Firefox, Figure 3 shows the main browser using Opera, and Figure 4 shows the main browser using Explorer. Also, Figure 5, shows the Pseudocode of this experiment.

3.3. Analysis

3.3.1. The created signalling mechanism

According to the network analysis inspecting components of Google Chrome, Firefox, Opera, and Explorer at the actual communication, this signalling mechanism was individually analysed among ten students and teachers depend on the delay for various thoughts, the first is rely on the delay to be prepared and the second based on sending a request and receiving a response from users. Consequently, it consumes between a minimum and maximum consumption that was 119 milliseconds (ms) to 185 (ms) to be ready. It also consumes 280 (ms), and 690 (ms) exhibits to transmit and receive a request and a response. The signalling mechanism can set up, establish and end a communication simultaneously between students and teacher. The variation of delay between using Google Chrome, Firefox, Opera, and Explorer was slightly different. However, CPU and bandwidth were impacting on the quality of audio and video. Besides, a long delay was revealed using the created signalling mechanism.



HTML ▾

```
n-room">
Open Or Join Room
</button>

<hr>


<div id="local-videos-container">
</div>

<div id="remote-videos-container">
```

CSS ▾

```
video {
  width: 40%;
  border-radius: 15px;
```

fp5i3stcbfc Open Or Join Room

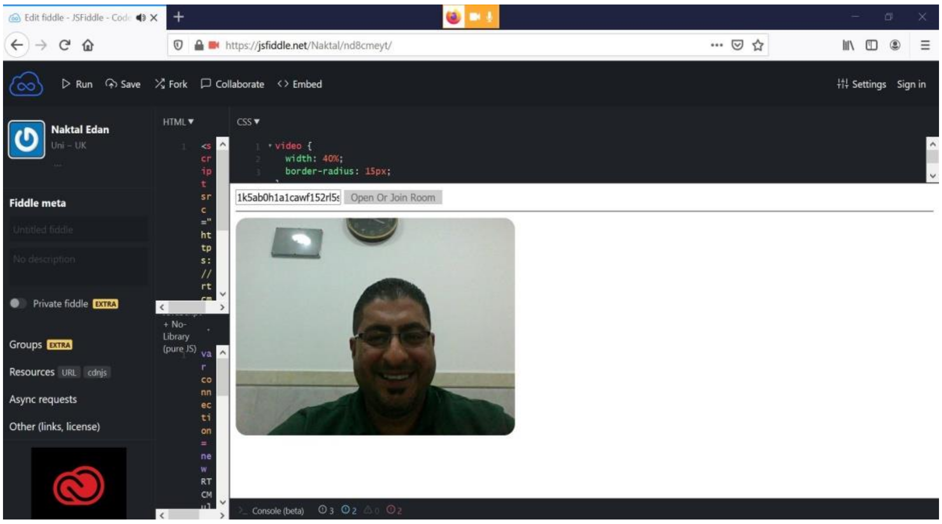
JavaScript + No-Library (pure JS) ▾

```
};

var localVideosContainer = document.getElementById('local-videos-container');
var remoteVideosContainer = document.getElementById('remote-videos-container');

connection.onstream = function(event) {
        var video= event.mediaElement;
```

Figure 1. Screenshot of the main browser using google chrome



HTML ▾

```
<script src="https://rtcm
```

CSS ▾

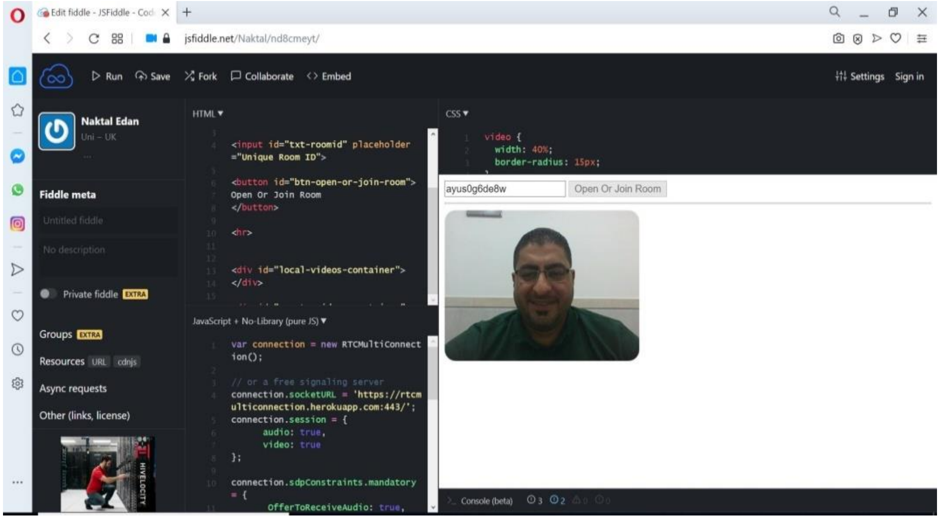
```
video {
  width: 40%;
  border-radius: 15px;
```

1k5ab0h1a1cawf152rl5s Open Or Join Room

+ No-Library (pure JS)

```
var connection = new RTCMul
```

Figure 2. Screenshot of the main browser using firefox



HTML ▾

```
<input id="txt-roomid" placeholder="Unique Room ID">

<button id="btn-open-or-join-room">
Open Or Join Room
</button>

<hr>


<div id="local-videos-container">
</div>
```

CSS ▾

```
video {
  width: 40%;
  border-radius: 15px;
```

ayus0g6de8w Open Or Join Room

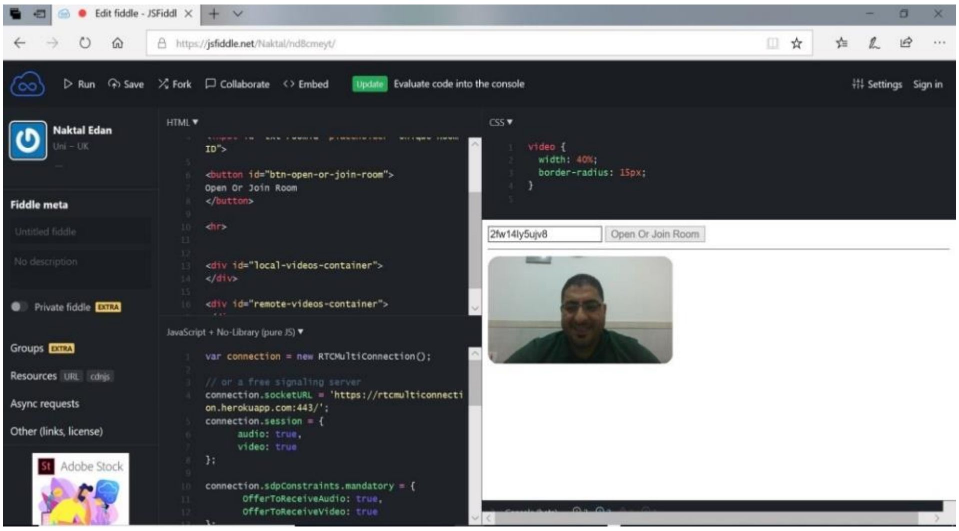
JavaScript + No-Library (pure JS) ▾

```
var connection = new RTCMultiConnection();

// or a free signaling server
connection.socketURL = 'https://rtcmulticonnection.herokuapp.com:443/';
connection.session = {
      audio: true,
      video: true
};

connection.sdpConstraints.mandatory = {
      OfferToReceiveAudio: true,
```

Figure 3. Screenshot of the main browser using opera



```
ID">

<button id="btn-open-or-join-room">
Open Or Join Room
</button>

<hr>


<div id="local-videos-container">
</div>

<div id="remote-videos-container">
```

```
video {
  width: 40%;
  border-radius: 15px;
}
```

```
var connection = new RTCMultiConnection();

// or a free signaling server
connection.socketURL = 'https://rtcmulticonnecti
on.herokuapp.com:443/';
connection.session = {
      audio: true,
      video: true
};

connection.sdpConstraints.mandatory = {
       OfferToReceiveAudio: true,
       OfferToReceiveVideo: true
```

Figure 4. Screenshot of the main browser using explorer

```
SET UI = New User-Id;
SET E.UI = Exist Current User-Id;
SET NU = New Userr;
SET SI = Session Initiator;
  SWITCH Start Connection;
  CASE1: open a new room;
  STEP1: open a room;
    IF SI = UI;
      THEN open a new session;
     ELSE ask for UI;
  STEP1: send offer;
     IF SI = created SDP-offer;
       THEN send SDP-offer to NU;
   STEP2: join the room;
     IF NU = E.UI && accepted SDP-offer;
       THEN NU = create SDP-answer;
     ELSE IF NU = unaccepted SDP-offer;
       THEN NU = disconnect;
     ELSE IF NU = UI;
     THEN NU = open a new session && create SDP-offer;
  STEP3: ICE candidate;
     IF SI && NU = exchanged ICEs;
       THEN start bi-directional video;
     ELSE re-connect;
    CASE2: Disconnect;
     IF SI = disconnect;
       THEN move control to the next peer;
     ELSE IF any peer= disconnect;
     END
```

Figure 5. Implementation pseudocode

3.3.2. Quality of video conferencing

The quality of video and audio was done by separate implementation among ten students and teachers on different networks, such as the Internet and 4G. Therefore, the quality of audio and video between two students was excellent. When the third peer joined the room, the quality of video and audio was excellent. Nevertheless, once the fourth participant joined the room; the quality of the video was not constant, so its quality was increasing and decreasing with a delay as well. Additionally, it was noticed that one peer was getting in and out with clear audio but frozen image for one minute. The same quality was also revealed when the fifth participant joined the room.

On the contrary, when the sixth, seventh and more peers had participated in the room, most of the connected peers presented a lousy quality of audio and video as unclear audio and frozen video. Not only that but also, it was not able to allow more than seven peers to join the room, whereas more than seven peers demolished the communication. Concerning to the peer who was using the 4G network, almost it was not visible or displayed in a frozen image. As a result, using this mechanism for many-to-many communication is efficient for three peers as a maximum number.

3.3.3. CPU performance and memory usage

In [35], emphasised that using mesh topology in sending and receiving data between students can influence the CPU load. By contrast, memory usage in real-time was not affected by the quality of multimedia communication [35-37]. Figure 6, shows the percentages of CPU performance and memory usage in a mesh topology.

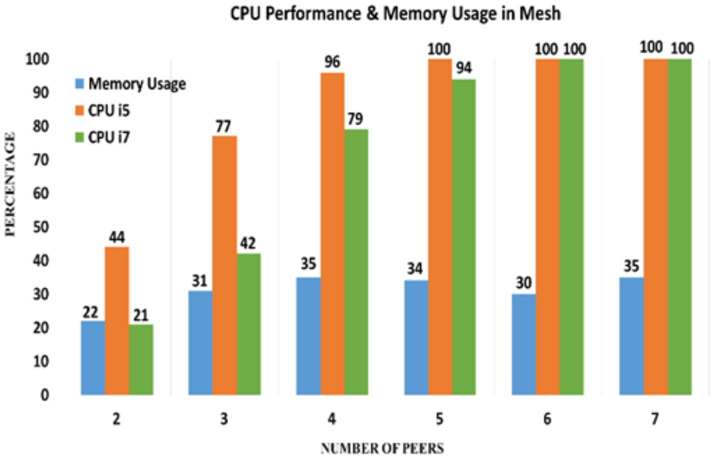


Figure 6. CPU performance and memory usage between students over the internet

3.3.4. Bandwidth consumption

Bandwidth consumption was measured using Wireshark analyser, so the analysis has shown that each client requests a minimum of 2Mb/s bandwidth on the video and 65kb/s bandwidth on the audio.

3.3.5. Quality of experience (QoE)

Actual students and teachers have involved in this test to elaborate their views on the recognised user knowledge by the use of questionnaires, as displayed in Table 1. This implementation confirmed an excellent quality of audio and video, in particular among three students via the Internet and 4G networks. However, the response of some peers was not effective while it takes more than three minutes on some occasions. According to evaluation, it can consider the bandwidth consumption as a source of reducing the quality of audio and video; and also the CPU performance as a source of allowing or controlling the number of peers. Accordingly, when a peer waits for other members until the CPU settles down, the quality gets better. Alternatively, this signalling has given enormous delay once sent a request until received a response.

Table 1. QoE from 10 students via the internet and 4G networks

Questions	Very Bad	Bad	Fair	Good	Excellent
	Very annoying	Annoying	Slightly annoying	Perceptible	
Rate the quality of audio during the session in e-learning	3	2		3	2
Rate the quality of the video during the session in e-learning	3	3	1	1	2
Rate the echo during the session in e-learning	3		2	3	2
Rate this application using Chrome					10
Rate this application using Firefox					10
Rate this application using Opera					10
Rate this application using Explorer					10

4. EVALUATION

It has proved that this mechanism can be considered to support communication among different browsers in a mesh topology. This implementation using a novel signalling mechanism can set up, established, and close connection over the Internet and 4G networks. Moreover, it offers bi-directional videos, and it retains the conference dynamic even if any client departs, prevents an unknown user from participating in the room. Whereas, it has been created without using any external devices and commercial cloud/server. This experiment can be considered as the first one that achieved a WebRTC bidirectional video conferencing using the 4G network. Nevertheless, it does not support Opera browser.

A critical problem during audio and video conferencing was a CPU performance and bandwidth consumption, while multimedia conferencing requires high CPU core and bandwidth. This can be known as CPU pressure, as long as it relies on various components, e.g. the used codecs. According to the specified specifications, it can confirm that CPU affects the number of users and bandwidth impacts on the quality of audio and video. On the contrary, this signalling mechanism can not support more than three peers in order to offer an excellent variety of audio and video. Also, it consumes a massive delay while it takes a long time to establish communication between peers. The QoE confirms that this testbed works applicably so that it can enhance these experiments in the future.

5. CONCLUSION AND FUTURE EFFORT

In this paper, WebRTC bi-directional multimedia conferencing in mesh topology through the Internet and 4G was designed and implemented in real-time execution. Moreover, a novel signalling mechanism was created and tested to set up, establish and end communication between participants. Also, a profound clarification of CPU presentation, memory performance, signalling usage, and QoE in a real test was completed. This state is beneficial as long it offers real demo over different browsers and networks with a user that needs physical communication. Besides, this experiment has been applied among three to five peers in e-learning between teacher and students. In the future, we have an idea to develop this achievement for more scalable video in e-learning.

REFERENCES

[1] A. A. Lozano, "Performance analysis of topologies for Web-based Real-Time Communication (WebRTC)," Aalto University, 2013.

[2] M. A. A. Suliman and A. Babiker A., "UMTS VoIP Codec QoS Evaluation," *IOSR Journal of Electronics and Communication Engineering*, vol. 10, no. 2, pp. 7-12, 2015.

[3] B. Y. Julian, et al., "WebRTC-based video conferencing service for telehealth," *Computing*, vol. 98, no. 1-2, pp. 169-193, 2016.

[4] M. Phankokkruad and P. Jaturawat, "An Evaluation of Technical Study and Performance for Real-Time Face Detection Using Web Real-Time Communication," in *International Conference on Computer, Communication, and Control Technology (I4CT)*, pp. 162-166, 2015.

[5] M. L. Giuliana. Et al., "A Performance Evaluation of WebRTC over LTE," in *12th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pp. 170-175, 2016.

[6] L. O. D. N. E. Fosser, "Quality of Experience of WebRTC based video communication," Norwegian University of Science and Technology, 2016.

[7] H. V. C. Cristian, "On multi-user web conference using WebRTC," in *18th International Conference on System Theory, Control and Computing (ICSTCC)*, pp. 430-433, 2014.

[8] A. Johnston, et al., "Taking on webRTC in an enterprise," *IEEE Communication Magazine*, vol. 51, no. 4, pp. 48-54, 2013.

[9] A. P. González, "Definition of a Mena Opinion Score for VP8 Over Real-Time Connections," Universida de Vigo, 2017.

[10] J. H. Paik and D. H. Lee, "Scalable signaling protocol for Web real-time communication based on a distributed hash table," *Computer Communications*, vol. 70, pp. 28-39, 2015.

[11] E. Bash, "Javascript Session Establishment Protocol," in PhD Proposal, vol. 1, pp. 1-85, 2016.

[12] E. E. I. Lajtos and D. O'Byrne, "WebRTC to complement IP Communication Services," *GSM Assciation*, 2016.

[13] S. Rajab, "Comparing different network topologies for WebRTC conferencing," Kthroyal Institute of Technology, 2015.

[14] M. Grinberg, "Socket.IO Documentation," 2017.

[15] R. Rai, "Socket.IO Real-time Web Application Development," Birmingham-Mumbai, PACKT, 2013.

[16] M. Nebra, "Socket.io: let's go to real time," *OPENCLASSROOMS*, 2017. [Online]. Available: https://openclassrooms.com/courses/ultra-fast-applications-using-node-js/socket-io-let-s-go-to-real-time.

[17] M. Rob, "Getting Started with WebRTC," Birmingham, *Packt Publishing Ltd*, 2013.

[18] B. Sredojev, et al., "WebRTC technology overview and signaling solution design and implementation," in *38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO-Proceedings*, pp. 1006-1009, 2015.

[19] M. Deshpande and S. P. Mohani, "Integration of WebRTC with SIP-Current Trends," *International Journal of Innovations in Engineering and Technology*, vol. 6, no. 2, pp. 92-96, 2015.

[20] M. Pasha, et al., “Analysis of challenges faced by WebRTC videoconferencing and a remedial architecture,” *International Journal of Computer Science and Information Security*, vol. 14, no. 10, pp. 698-705, 2016.

[21] S. Petrangeli, et al., “A scalable WebRTC-based framework for remote video collaboration applications,” *Multimedia Tools and Applications*, vol. 78, no. 6, pp. 7419-7452, 2019.

[22] A. S. Rosas and J. L. A. Martínez, “Video conference System Based on WebRTC With Access to the PSTN,” *Electronic Notes in Theoretical Computer Science*, vol. 329, pp. 105-121, 2016.

[23] M. Sağlam, “Revolutionizing Thailand’s Rural Healthcare and Mental Health Illness Treatments Through Telemedicine-Video Conferencing,” Siam University, 2019.

[24] A. Andujar and C. Medina-López, “Exploring new ways of eTandem and telecollaboration through the WebRTC protocol: Students’ engagement and perceptions,” *International Journal of Emerging Technologies in Learning*, vol. 14, no. 5, pp. 200-217, 2019.

[25] N. Edan and S. A. Mahmood, “Design and implement a new mechanism for audio, video and screen recording based on WebRTC technology,” *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 10, no. 3, pp. 2773-2778, 2020.

[26] N. M. Edan, et al., “Performance evaluation of QoS using SIP &amp; IAX2 VVoIP protocols with CODECS,” in *Proceedings of 2016 SAI Computing Conference (SAI 2016)*, 2016.

[27] W. C. K. Ng, et al., “A P2P-MCU Approach to Multi-Party Video Conference with WebRTC,” *International Journal of Future Computer and Communication*, vol. 3, no. 5, pp. 319-324, 2014.

[28] “Video Conferencing,” *Daitan GROUP*, 2013. [Online]. Available: http://www.daitangroup.com/video-conferencing-what-is-an-mcu/.

[29] S. Potthast, “Point to Point and Multipoint,” *Jisc community*, 2016. [Online]. Available: https://community.jisc.ac.uk/library/janet-services-documentation/point-point-and-multipoint.

[30] D. Walsh, “WebSocket and Socket.IO,” *Media Temple*, 2010. [Online]. Available: https://davidwalsh.name/websocket.

[31] N. Chhetri, “A Comparative Analysis of Node.js (Server-Side JavaScript),” St. Cloud State University, 2016.

[32] H. Shane, “Video-to-Video Using WebRTC,” in S. Hudson (ed), *JavaScript Creativity: Exploring the Modern Capabilities of JavaScript and HTML5*, *Apress*, p. 184, 2014.

[33] C. Y. Chiang, et al., “A video conferencing system based on WebRTC for seniors,” in *International Conference on Trustworthy Systems and their Applications*, Taichung, pp. 51-56, 2014.

[34] K. Bissereth, et al., “An Interactive Video Conferencing Module for e-Learning using WebRTC,” in *Internetional Conferences*, pp. 1-4, 2014.

[35] N. M. Edan, et al., “WebNSM : A Novel Scalable WebRTC Signalling Mechanism for Many-to-Many Video Conferencing,” in *3rd IEEE International Conference on Collaboration and Internet Computing (CIC)*, vol. 2, pp. 1-7, 2017.

[36] G. A. Abed, et al., “Integrated approaches to enhance TCP performance over 4G wireless networks,” *IEEE Symposium on Computers and Informatics*, pp. 154-158, 2012.

[37] N. M. Edan, et al., “WebNSM: A Novel WebRTC Signalling Mechanism for One-to-Many Bi-directional Video Conferencing,” in *Proceedings of 2018 SAI Computing Conference*, pp. 1-6, 2017.