International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

WebRTC Peer to Peer Learning

H. Fateh Ali Khan [1], A. Akash [2], R. Avinash [3], C. Lokesh [4]
[1,2,3,4] Department of Information Technology,
Valliammai Engineering College,
Chennai, India

***Abstract*— WebRTC is a peer to peer technology that provides web browsers with Real-Time Communications (RTC) capabilities. The WebRTC technology is built within almost all modern browsers using which the browsers are able to talk to each other instead of just to web servers. Once the connectivity is established, the browsers can share media streams such as audio and video from the microphone and camera, exchange files or send and receive messages through the fastest possible way: peer-to-peer. With this peer to peer technology as the foundation for the web application, the system is able to provide diverse learning capabilities. The learner discovers the expert and is able to establish a direct connection. WebRTC handles all the relevant traffic and relays the media streams among the peers. Additional functionalities are also supported that benefit the learner such as messaging, file sharing and screen sharing which helps the learner to actively interact with the expert for the clear understanding of a particular concept. The web application paradigm is being widely adopted due to the ubiquity of web browsers across PCs and mobile devices.**

Keywords*— *WebRTC; peer-to-peer; learning; browsers; communication; media.

I. INTRODUCTION

In recent education times, online learning is certainly the most transformative development. The advancements in computer technology and the internet in network education has wide acceptance. Internet combined with education will make teaching and learning activities much simpler, while teachers teach over the internet, students study in the internet, information flows, and the knowledge is gained, and eventually, all offline activities will become the supplement of online activities. Nowadays, students need not be constrained by the physical classroom they can use online learning tools to find material when and where it best suits them.

Virtual Learning is one of the most popular and effective tools that can provide a web-based learning environment. The only audio and video network education choices currently available to teachers and students are those proprietary systems. However, each of these systems either requires an additional plug-in or need a completely standalone application to be installed, such as the most commonly used plug-in Adobe Flash and the most famous VoIP application Skype, both are inconvenient for users. The installation and setup time for each of these systems is usually quite high, and some of them even require a registration fee.

WebRTC has resulted in virtual learning to become even more interactive and immersive. The power of such a virtual learning system can be harnessed by institutions or organizations to ensure that the product takes full advantage of WebRTC. The user experience is improved and can offer a wide range of advanced features, making it a useful tool for anyone interested in an immersive learning experience. WebRTC ("Web Real-Time Communications") is the most advanced and powerful technologies supporting virtual learning systems. The technology behind WebRTC has been standardized for the transmission of audio and video content. The WebRTC peer to peer learning system based on HTML5 has the necessary technology to allow real-time audio and video via a Web browser without the need for third-party software and additional plugins such as Java, Adobe Flash, and others. It helps to combine different services such as Instant Messaging, presence, audio/video conferencing and does not require any installation of plug-ins or setup software. The usability and accessibility of these education services surely will determine their wide adoption. WebRTC based online learning systems have a very broad development space and applications.

A. Overview

WebRTC makes it possible to establish peer-to-peer connectivity to other web browsers easily. Traditionally, building such an application from scratch requires a number of frameworks and libraries that deal with typical issues like packet loss, connection dropping, and NAT traversal. With WebRTC, all of this comes built-in into the browser by default. There is isn't any need for plugins or third-party software. It is open-sourced which simplifies development. WebRTC can be used for multiple tasks, but real-time peer-to-peer audio and video (i.e., multimedia) communications is the primary benefit. In order to communicate with another person (i.e., peer) via a web browser, each person's web browser must agree to begin communication, know how to locate one another, bypass security and firewall protection, and transmit all multimedia communications in real-time. WebRTC allows a desktop or mobile browser-based application to access the device's microphone and video camera. The browser typically informs the user that an application is requesting access to their computer's camera and microphone. Once the user allows access to use these devices, WebRTC can create individual streams of transmittable audio and video data from data generated by these input devices. This data is then transmitted via network data 'channels' established in a bidirectional stream to send the data between the peers.

B. Objective

The primary objective of the peer to peer learning system is to help students acquire the necessary knowledge by consulting with the experts by making use of the underlying WebRTC technology to simplify the process. With

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

WebRTC the overhead on end users in the installation of additional plugins such Adobe Flash and third-party software is reduced. Since WebRTC is a peer to peer technology, there is no need for handling of a centralized server due to decentralized connections among the peers. Since the system is decentralized the users are guaranteed of privacy as the traffic is sent directly between them rather than through a server. The system also aims to promote dynamic interactions through messaging, file sharing and screen sharing to enable effective communication between the participants.

II. RELATED WORKS

The standardization efforts proposed by W3C and IETF are given in [1] with discussions revolving around the real-time multimedia applications in the web. The RTCWEB working group defined an architecture with the complete set of protocols for the support of real-time multimedia communication directly between the browsers in a reliable manner. The ongoing efforts of the WebRTC 1.0 standard are proposed. This also introduced the concepts of bringing the ORTC (Object Real-Time Communication) concepts into WebRTC for the future standardization efforts.

The overall architecture of the WebRTC system is laid out in [2] and the information is depicted to flow in a peer to peer fashion directly between the two web browsers and the protocols used to transport and secure the encrypted media are specified. The details regarding the WebRTC data channels are laid out clearly. The issues associated with the transport layer are given which revolve around NAT (Network Address Traversal) and firewall traversal, multiplexing of data and media over one transport flow are also addressed.

The impact of WebRTC as an enterprise application is provided in [3] and some of the challenges associated with enterprise usage can be: firewall traversal, access control, and peer to peer data flow, other problems related to the enterprise can be attributed to integration and interoperation with existing communications infrastructure. WebRTC firewall traversal must work without a standardized signaling protocol and the integration issues since it changes the way enterprises operate either within the organization or outside with customers or other organizations.

The possibilities of packet loss within the network along the video communication path is considered in [4] and an adaptive scheme such as the hybrid NACK/FEC method is proposed which can be used to balance the video quality, smoothness of rendering and end to end delay. An offline simulation tool can be used to analyze the system behavior and result analysis which can simulate the various network conditions. It also employs a hybrid NACK/FEC method to get a better trade-off between temporal quality and spatial video quality.

The possible security vulnerabilities associated with WebRTC are provided in [5] and since this technology is an open standard it suffers from several network-based attacks and can have severe security implications. The security model which makes use of HTTPS is used for traditional applications but for applications that make use of complex connections a much more secure mechanism is necessary. With HTTPS, security guarantees can be made possible due to the trusted nature of the system. The browser can make use of TLS (Transport Layer Security) to establish a TLS session with a server whose credentials are valid. TLS is able to provide confidentiality and data integrity.

III. EXISTING SYSTEM

The existing system is implemented using document-based or web-page based user interfaces. Such systems are privately held within institutions and are proprietary. The system employed the use of a centralized server to establish the connectivity and coordination among the participants. However, due to this centralized approach it involved high maintenance and handling overhead and this resulting in increased costs. The system was not available for public utilization and made use of a pay-per-use model leading to increased expenses on the users' end. The development effort of such systems was very complex. The existing system required additional third party software to be installed on the client-side and made use of proprietary plugins leading to increased cost of usage with one of the commonly used plugin to be Adobe Flash. The setting time for each of these systems is usually quite high, and some of them even require a registration fee which had to be paid in order to utilize the system. Since the system required the installation of additional software client side storage requirements were also necessary. Such systems were difficult to set up and the learning curve for the users is pretty high and they also did not support the availability of dynamic interactions between the participants.

A. Disadvantages

The existing systems lacked direct interaction with the experts. They required a separate installation of plugins and third-party software. Such systems are only available privately within institutions and not for public consumption. The system utilized centralized servers to accomplish the communication which included additional handling overhead. They also lacked privacy and data was not secure since all the traffic that is relayed between the users is passed through a server so privacy is not guaranteed.

IV. PROPOSED SYSTEM

The purpose of the system is to use the underlying WebRTC technology to enable interaction between students and experts so that knowledge is imparted in an effective manner. The no plugin approach offered by WebRTC is helpful since the users don’t have to install any plugins like Adobe Flash or use third-party software such as Skype thereby minimizing the setup time required drastically. All the user needs to access the web application is to access it via the URL (Uniform Resource Locator) within the web browser. The application is fetched from the web application server which is in this case is the Heroku Cloud by making HTTP requests from within the browser. If the user is a student then he logs in and enters his details else if the user is an expert then he logs in and enters the details and his area of specialty.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

The users can be a student or an expert each of whom may be separated across the network. The student enters his details and is validated and then the student request the service of an expert and as soon as the expert is available the connectivity is established and the call can proceed which is followed by the streaming of multimedia data. The expert enters his details and his area of specialization and the details are validated then the experts gives a signal that he is available to proceed with the connection and connectivity is established if there is student presently waiting on the other end of the network, here the expert acts as the other peer in the network.

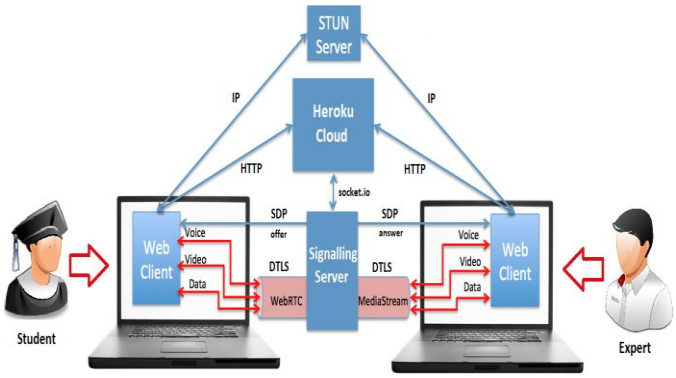

Fig. 1. System Architecture

The call establishment proceeds between the two peers and multimedia data is streamed between them. Additional functionalities are supported such as messaging, file transfer and screen sharing to provide dynamic interactions between the participants. The system has a user-friendly UI that can provide an easy understanding of the system and all the user requires to establish connectivity with the other peer is the link to the website. The proposed method allows the use of voice, video and messaging to point out specific things you are concerned about or to show the person on the other end exactly what you are getting at, which demonstrates the learning capability of WebRTC.

A. *Advantages*

1) Real time communication with experts: The system allows for direct real time communication with the experts which is lacking in the existing system, using which students can acquire knowledge and proceed accordingly.

2) Permits dynamic interactions: The system enables dynamic interactions between the participants through media data such as voice and video, and also supports advanced functionalities such as screen and file sharing for more active participation.

3) Readily available for public usage: The system is readily available for active use by the end users since the system is hosted on a public server

4) Decentralized approach: The existing system involves maintaining a separate server for handling the media traffic and maintaining it is not cost effective but the proposed method eliminates the need of server since the connectivity is directly established between the peers.

5) Plugin free mechanism: The system does not require any additional plugins and third party software for its operation thereby enabling the users a hassle free experience. Since no plugins are required the overall cost is signficantly reduced.

B. *Methodology*

Generally, there are two kinds of IP addresses: private and the public IP address. A private IP address is a non-Internet facing IP address that the router assigns to each device connected to it. A public IP address is an Internet facing IP address that is assigned to the device each time it connects to the Internet. The NAT (Network Address Translation) is a device that generally performs the translations between the public and private IP address spaces. This provides unique identification for devices that are within your home network. Since the peers need to know the each other's IP address to communicate this is done with the help of the STUN (Session Traversal Utilities for NAT). The web client first contacts the STUN server which sends back the public IP address back to the client. The same operation is performed by the other client and hence the connectivity between them is established behind the NAT. The web clients then contacts the Web Application Server (i.e. Heroku Cloud) where the application is stored by sending an HTTP request. Once the two peers are connected to the same "channel", the peers are able to communicate and negotiate session information through SDP (Session Description Protocol). The initiating peer sends an "offer" using SDP and the initiator waits to receive an "answer" from any receivers that are connected to the given "channel". Once the answer is received, local data streams and data channels are created by each peer, and multimedia data is finally transmitted. In this application, the student first enters his credentials and waits for the expert to join the connection. The expert on the other end enters his credentials and specialty and establishes the connection and the video conferencing session between them begins in a peer to peer manner.

V. MODULE DESCRIPTION

A. *Student details*

The student who wants to participate in the session first navigates to webpage and fills in the required details which are validated and requests the expert for communication. The students acts as one peer in the network.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

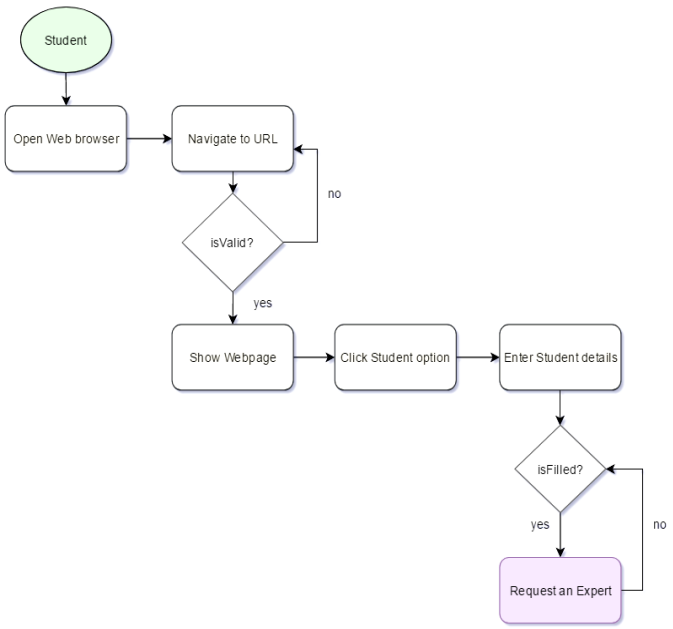

Fig. 2a. Student details

B. Expert details

The expert enters his details and his area of specialization and the details are validated then the expert gives a signal that he is available and the connection can proceed. Here the expert acts as the other peer in the network.

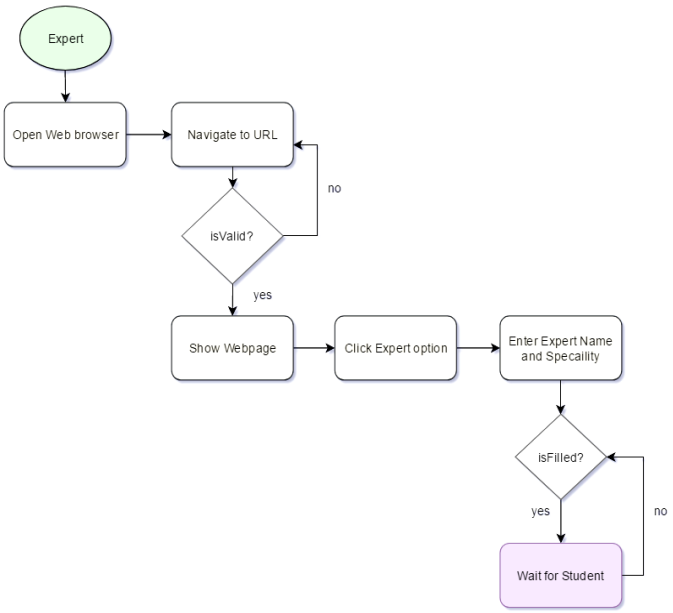

Fig. 2b. Expert details

C. Call establishment

Once both the student and expert on the either ends have entered their details the call establishment proceeds and the video stream of both participants is communicated in real time.

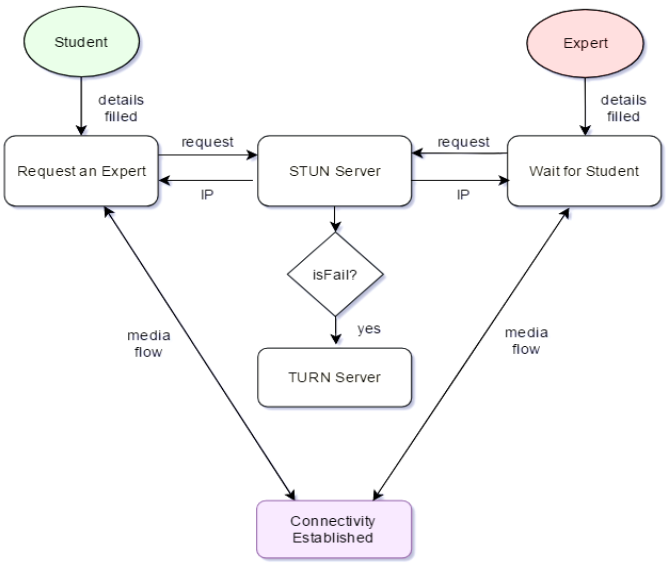

Fig. 2c. Call Establishment

The video connectivity between the student and expert is established instantaneously. The video quality is dependent on the speed of the internet connection higher the speed greater the quality of the video. Also if the public STUN server fails the TURN (Traversal Using Relays around NAT) server is used for routing the video traffic between the peers but maintaining the TURN server is very expensive.

VI. WORKING

The WebRTC Peer to Peer Learning system consists of the following components for its working -

A. STUN and TURN

The STUN server is primarily made use by clients to know their public IP address. Since most of the clients are located behind a NAT they are unable to determine their public IP, this is where the STUN server comes into play. STUN is a client-server protocol and can work across TCP and UDP connections. A STUN server can be defined as an entity that can handle the STUN requests made by the clients. The aim STUN is to overcome issues associated with lack of standardized behavior in NATs. STUN servers generally reside on the public internet and have a simple task: check the IP: port address of an incoming request and send the obtained address back to the requesting clients as a response. The STUN servers are generally publicly available with offerings from Google as well. Most of the WebRTC calls make use of the STUN server to know and communicate the IP between the peers so that the peers discover each other and the connectivity is established through the signaling mechanism, in order to set up a direct link. STUN can also provide protocol encryption through TLS (Transport Layer Security) which can guarantee message integrity and authentication.

The TURN server is used when peer to peer communication fails. TURN is advantageous over STUN since it has the ability to traverse symmetric NATs. The TURN server acts a fallback mechanism when the STUN server fails since STUN servers are not reliable in nature especially when it comes to the handling of heavy media traffic and due to low

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

bandwidth supported by them. However, the TURN server has a critical disadvantage when it comes to cost, maintenance, and huge bandwidth usage when HD video stream is being delivered.

B. Signaling

Signaling is used to exchange session control messages between the peers using the SDP (Session Description Protocol) and also network configurations as ICE candidates and media capabilities using same session control messages. This signaling process needs a way for clients to pass messages back and forth. The session exchange proceeds with one client making an offer and the other client send back the answer and the connection proceeds when the either of the clients accept the connection. The signaling mechanism is not implemented by the WebRTC APIs: it needs to be built separately. For the purpose of signaling Socket.IO is used and provides a good solution. Socket.IO is a JavaScript library that can provide a reliable and effective way to setup a real time connection between peers. Socket.IO acts as the signaling medium and as a fallback mechanism when the standard WebRTC peer to peer connectivity fails. Socket.IO can be added with a simple piece of middleware to enable this functionality so there is no need to setup own signaling exchange or deploy and scale new servers.

It only takes a few lines of code to set up the server and client.

Server: var io = require ('socket.io') (server);
Client: var io = require ('socket.io-client');

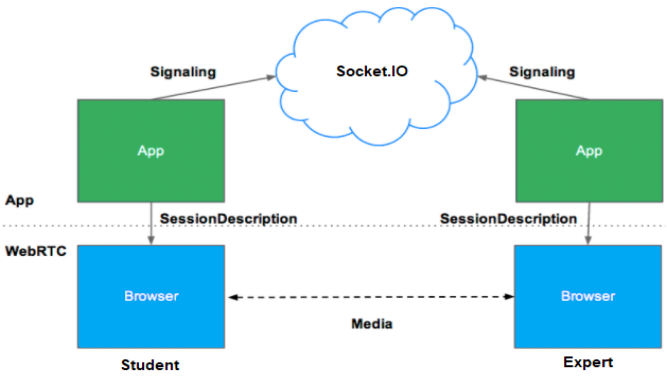

Fig. 3. Signaling

C. WebRTC APIs

1) MediaStream API: The MediaStream API is used to capture the media streams from the local camera and microphones. The API uses the getUserMedia () method which acts as the primary way of accessing the local streams. The MediaStream objects are each made up of several MediaStreamTrack objects. The streams (i.e. audio and video) are obtained from the input devices that are connected. The MediaStreamTrack object may also contain several channels (left and right audio channels), which are the smallest parts as defined by the MediaStream API.

2) RTCPeerConnection API: The RTCPeerConnection API is the primary component to establish peer-to-peer connectivity between the browsers. To create the objects for the API, we add the following:

var pc = RTCPeerConnection (config);

The config argument contains at least one key which can contain a list of ice servers. An array of URL objects are provided which can be added as arguments, each containing information about STUN and TURN servers.

3) RTCDataChannel API: The RTCDataChannel API provides a channel which is used for bi-directional peer-to-peer transfers of binary data. The peer that joins the channel to exchange data receives a data channel event to know that a data channel is added to the connection.

VII. IMPLEMENTATION

A. Main page

The main page of the web application is fetched from the Heroku cloud through HTTP requests made by the web browser.

Fig. 4a. Main page

B. Student details

The student navigates to the website and selects the student option. Then the student enters the details which are validated. On successful validation the student may be able to request the service of an expert by clicking the request expert option.

Fig. 4b. Student details

C. Expert details

The expert navigates to the website and selects the expert option. Then the expert enters the details which are validated. The expert also specifies the area of expertise or specialty within the form. On successful validation the expert may wait for a student to establish the connection.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 7 Issue 03, March-2018

Fig. 4c. Expert details

D. Call establishment

The call is established between the student and the expert when there is a student requesting on one end and an expert waiting on the other. The student and expert act as the peers in the network. Once the connectivity is established between them the student is displayed with the corresponding information as shown in fig 4.d. The student can proceed with the call by clicking on the call the expert option. Once clicked the interaction between them can take place.

Fig. 4d. Call establishment

E. Media Stream

Once the call is established the media begins to flow in a bidirectional manner between the student and the expert. This media sends the local video streams captured from each of the users web camera and sends it along the channel.

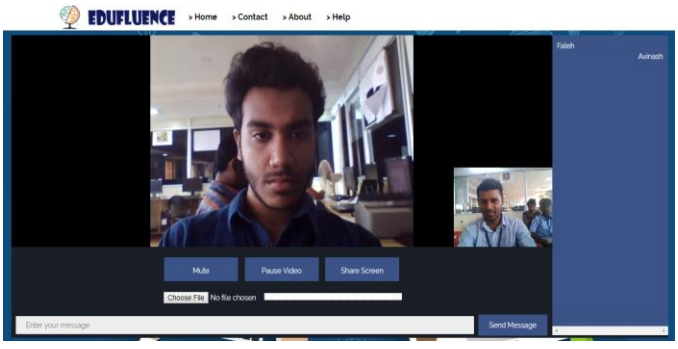

Fig. 4e. Media stream

F. Messaging, File and Screen Sharing

The WebRTC Peer to Peer Learning system also supports additional functionalities such as messaging, file sharing and screen sharing for a better interaction between the participants. These added features ensure that the student can gain knowledge with ease.

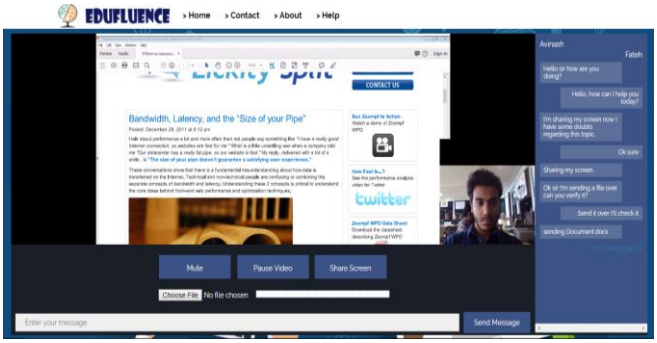

Fig. 4f. Messaging, File and Screen Sharing

VIII. CONCLUSION AND FUTURE WORK

By using the underlying WebRTC technology, a peer to peer learning system is designed and implemented. Such a system fosters collaboration and interaction, increasing engagement in and out of the classroom and increases accessibility and reach. The main goal is to make learning more affordable and cost effective which is possible through the plugin-free nature of WebRTC. Besides, the peer-to-peer nature of WebRTC may also lead to cost reductions of network and infrastructure. With WebRTC real-time applications can be deployed in an effective manner due to availability of simple JavaScript APIs to handle the stream of multimedia data. Due to higher bandwidth utilization and resource requirements the current system supports only one-to-one interactions. In the next stages it possible to support one-to-many interactions. The system can also be scaled to serve multiple users simultaneously. It is also possible to include a virtual whiteboard where the participants can draw diagrams and pictures which can be communicated in real time between them which can further help improve the learning capability of the system.

REFERENCES

[1] S. Loreto and S.P. Romano, "How far are we from WebRTC-1.0? An update on standards and a look at what's next", IEEE Communications Magazine, vol. 55, no: 7, May 2017, pp. 200-207, DOI: 10.1109/MCOM.2017.1600283.

[2] Felix Weinrank, Martin Becke, et al, "WebRTC Data Channels", IEEE Communications Standard Magazine, vol. 1, no. 2, July 2017, pp. 28-35, DOI: 10.1109/MCOMSTD.2017.1700007

[3] A. Johnston, J. Yoakum, and K. Singh, "Taking on WebRTC in an Enterprise," IEEE Communication Magazine, vol. 51, no. 4, April 2013, pp. 48–54, DOI: 10.1109/MCOM.2013.6495760

[4] S. Holmer, M. Shemer, and M. Paniconi, "Handling Packet Loss in WebRTC," 2013 IEEE Int'l. Conf. Image Processing, Melbourne, VIC, 2013, pp. 60–64, DOI: 10.1109/ ICIP.2013.6738383

[5] R. L. Barnes and M. Thomson, "Browser-to-Browser Security Assurances for WebRTC," IEEE Internet Computing, vol. 18, no. 6, Dec. 2014, pp. 11–17, DOI: 10.1109/MIC.2014.106

[6] Boni Garcia, Francisco Gortazar, Luis Lopez-Fernandez, Micael Gallego, Miguel Paris, "WebRTC Testing: Challenges and Practical Solutions", IEEE Communications Standard Magazine, vol. 1, no. 2, July 2017, pp. 36-42, DOI:10.1109/MCOMSTD.2017.1700005

[7] C. Jennings, T. Hardie, and M. Westerlund, "Real-time Communications for the Web," IEEE Communication Magazine, vol. 51, no. 4, April 2013, pp. 20–26, DOI: 10.1109/ MCOM.2013.6495756

[8] S. Loreto and S. P. Romano, "Real-Time Communications in the Web: Issues, Achievements, and Ongoing Standardization Efforts," IEEE Internet Computing, vol. 16, no. 5, October 2012, pp. 68–73, DOI: 10.1109/MIC.2012.115.