International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.177
Volume 7 Issue V, May 2019- Available at www.ijraset.com

A Framework for Real Time Communication on Web using with WebRTC

Dr. Abid Hussain[1], Dr. Praveen Kumar. Sharma[2]
[1, 2]*Asistant Professor ,School of Computer Applications, Career Point University, Kota 'Vardhman Mahaveer Open Universit, Kota*

***Abstract:** In the current era of IT,there are many kinds of communication tools & services available. We use different kind of online communication services including social websites, messaging, email, VoIP etc. One of the last major challenges for the web is to enable human communication via voice and video: Real Time Communication, RTC for short. RTC should be as natural in a web application as entering text in a text input. Without it, we're limited in our ability to innovate and develop new ways for people to interact.[1] Web is becoming more and more popular. Web Real-time Communication (WebRTC), as one of the new technologies based on Web, makes real-time communication easy to be used through the Web. With Web Real-Time Communication (WebRTC), modern web applications can easily stream audio and video content to millions of people. WebRTC is a powerful tool that can be used to infuse Real-Time Communications (RTC) capabilities into browsers and mobile applications. In this paper, we will discuss about the working architecture and its various services based on a traditional communications techniques and how you can use WebRTC to set up peer-to-peer connections to other web browsers quickly and easily ?.*

***Keywords:** Web RTC, Multimedia, Browser, HTTP, NAT, STUN, Media Stream API, Video Conferences, P2P, HTML, JS, Signaling.*

I. INTRODUCTION

The Web is no more a stranger to real-time communication as Web RTC (Web Real-Time Communication) comes into play. With the appearance of Web RTC, modern web applications can easily stream audio and video content to millions of people. Web RTC is a free and open project that provides browsers and mobile applications with Real-Time Communications (RTC) capabilities via simple APIs. The Web RTC components have been optimized to best serve this purpose.

Web RTC stands for web real-time communication. It is a very exciting, powerful, and highly disruptive cutting-edge technology and standard.[2] Web RTC leverages a set of plug-in and free APIs that can be used in both desktop and mobile browsers, and is progressively becoming supported by all major modern browser vendors. Previously, external plug-ins were required in order to achieve similar functionality as is offered by Web RTC. Web RTC leverages multiple standards and protocols, most of which will be discussed in this article. These include data streams, STUN/TURN servers, signaling, JSEP, ICE, SIP, SDP, NAT, UDP/TCP, network sockets, and more.

With Web RTC browsers and apps learn to talk to each other instead of just to web servers. They can share audio and video streams from your microphone and camera, exchange files and images or just send and receive simple messages the fastest possible way: peer-to-peer.

II. USE OF WEB RTC

Web RTC is great technology for enabling real time communication over the website using with internet. In my opinion, Web RTC (Web Real-Time Communication) is definitely the next big thing in the tech world.[3] It is a free-to-use and open-source innovation that allows website browsers to communicate with each other in real-time, anytime, anywhere, using any device. [5] This cutting-edge technology can be used via the web and mobile applications, and is compatible with multiple browsers. I believe it's no surprise that more and more businesses want to get their hands on it.

1) *Multichannel Communication:* WebRTC works on multiple platforms; it helps businesses smoothen their multichannel communication by allowing customers to interact with them with a single click.
2) *Screen Sharing:* this feature is especially suitable for customer service reps, who are constantly trying to explain and resolve customers' technical issues. By allowing reps to see what callers are seeing, they can provide help more efficiently.
3) *Video Chat:* there's no doubt that video chat is on the rise, and it's definitely one of the best uses of WebRTC on the web. Users can simply contact each other via video chat, again with a single click.

4) *Video Conferencing:* WebRTC is very often used as a video conferencing tool, and for good reason: it allows businesses to set up video conference meetings and easily communicate with their clients/partners.
5) *File Sharing:* Suppose you want to send a massive file to a colleague while working on a project. Instead of emailing the file or uploading it to a third party cloud storage system (and waiting several minutes for the transfer to complete), you could send it directly through your web browser using WebRTC's data channel, with very low-latency, with the benefit of full encryption between the two endpoints.
6) *Embedded Endpoints:* ATMs. Vending machines. Bus stops. Retail store kiosks. All of these endpoints can be embedded with WebRTC engines. It's an easy way to connect customers with live agents while they are on-the-go.
7) *Sales Enablement:* Websites and applications are key tools for sales enablement. Customers rarely make important purchases on impulse. Decisions are often made after speaking with a sales associate. Outfitting a website or application with a WebRTC audio or video contact channel is a great way to provide ongoing assistance throughout the purchasing process
8) *Simple Collaboration:* It can be frustrating to have to open an account or download a platform to engage in a business meeting. WebRTC removes this barrier, providing a seamless, non-invasive way to connect and collaborate. Using WebRTC when communicating with colleagues, clients and business partners is easier, simpler and often more convenient.
9) *Emergency Response:* In some cases, WebRTC is being used to increase public safety. SaferMobility streamlines real-time interactions with authorities by enabling video, audio, text communications while also utilizing location based awareness. This use of the WebRTC data channel allows responding personnel to have deeper insight and better information by circumventing previously existing communication barriers when responding to emergency calls.
10) *Patient Management:* Many health clinics are now using WebRTC based solutions to reduce in-office patient visits. Doctors can now perform checkups over Web browsers. This allows them to allocate more time to higher priority patients. WebRTC is also a great way for clinic staff to communicate with patients in between visits as all the patient needs is a web browser and a URL.

III. BENEFITS OF WEB RTC

WebRTC provides ample value add for those businesses looking to build off of its potential, meaning there is certainly room for ingenuity. But its ability to be a creative lens is just one advantage of WebRTC.[4] WebRTC brings many benefits to the user, integrator, and developer that have been less available in past communications and collaboration platforms. With all the hype around WebRTC, it must offer capabilities that are leaps and bounds beyond our present communications and collaboration environment [5]. There are advantages to WebRTC that propel it past today's VoIP and video systems. Even if you dismiss some of the following nine advantages, many are well worth considering [6].

1) *It's Free:* WebRTC is an open-source application programming interface (API) first introduced by Google in 2011. Google's goal for WebRTC is to deliver a standard-based, real-time media engine that will be free and resident in all available browsers.
2) *Platform and Device Independence:* Any WebRTC-enabled browser with any operating system and a web services application can direct the browser to create a real-time voice or video connection to another WebRTC device or to a WebRTC media server. The browser operating system is not relevant. This is accomplished by implementing standard APIs from the W3C and protocols from the IETF. Developers can write HTML5 code that can work on desktop and mobile devices.
3) *Secure Voice and Video:* WebRTC has always-on voice and video encryption. The Secure RTP protocol (SRTP) is used for encryption and authentication of both voice and video. This is especially beneficial over WiFi networks. This prevents eavesdropping and recording of the voice and video.
4) *Advanced Voice and Video Quality:* WebRTC uses the Opus audio codec that produces high fidelity voice. The Opus codec is based on Skype's SILK codec technology. The VP8 codec is used for video. These selections ensure interoperability and avoid the need for codec downloads that may contain malicious code.
5) *Reliable Session Establishment:* WebRTC supports reliable session establishment. This is true for Network Address Translators (NAT), something that hinders and may block other communications and collaboration protocols. The reliable operation avoids server-relayed media and thereby reduces latency and increases quality. It also reduces the server load.
6) *Adaptive to Network Conditions:* WebRTC supports the transaction of multiple media types and endpoints. WebRTC is a browser-based technology that allows to adapt to a wide array of network conditions. The important competitive advantage produces a skilled use of bandwidth conveys the best possible video and voice and communications.

7) *Rapid Application Development:* Developers will experience a streamlined development process reducing the time for application implementation. Detailed knowledge of WebRTC will not be necessary because of the standardized APIs. Finally, the voice and video codecs are license-free.
8) *Interoperability with VoIP and Video:* The biggest value of WebRTC is its promise of interoperability with existing voice and video systems. This includes devices using SIP, Jingle, XMPP, and the PSTN. What may hinder the global interoperability will be the upgrades necessary in exiting devices. Alternately, gateways can be the solution to interoperability. Some are already on the market. If the existing voice and video devices using standard protocols, they will probably work with WebRTC-based devices.

IV. WORKING ARCHITECTURE OF WEB RTC

A WebRTC Architecture WebRTC extends the client-server semantics by introducing a peer-to-peer communication model between browsers. WebRTC offers web application developers the ability to write rich, realtime multimedia applications (think video chat) on the web, without requiring plugins, downloads or installs.[7] It's purpose is to help build a strong RTC platform that works across multiple web browsers, across multiple platforms. ARCHITECTURE The classic web structural design are based on a clientserver model, where browsers send an HTTP (Hypertext Transfer Protocol) request for content to the web server, which replies with a response containing the information requested. The server provides resources that are closely associated with an entity known by a URI (Uniform Resource Identifier) or URL (Uniform Resource Locator).[8]
In the web application situation, the server can embed some JavaScript code in the web page and sends back to the client. Such code can interact with browsers through standard JavaScript APIs and with users through the user interface.[9]
The overall architecture looks something like this:

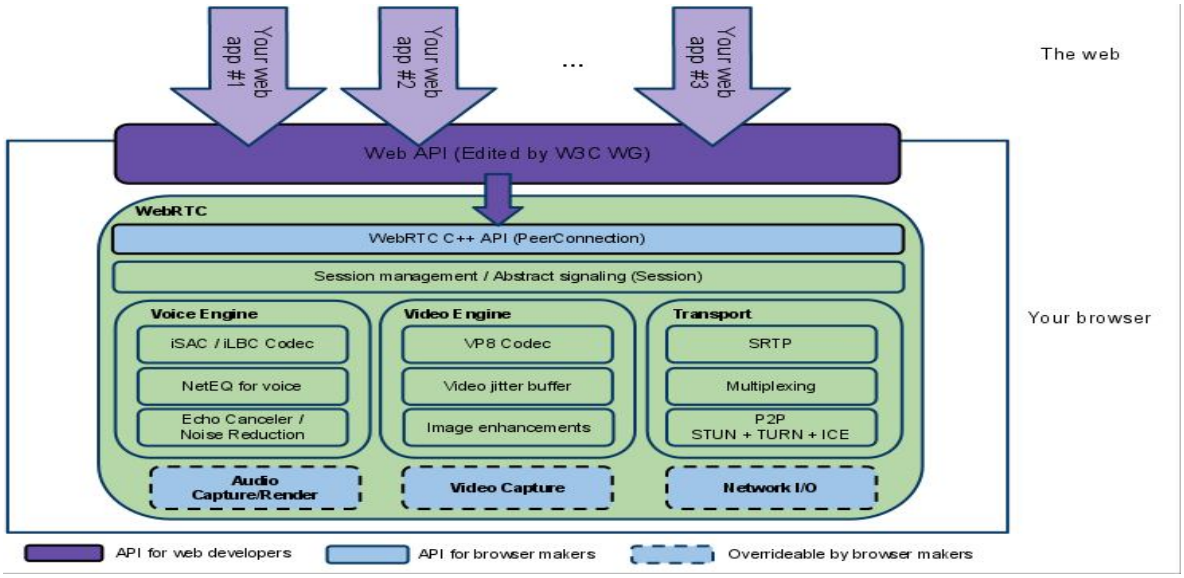

Figure 4.1 Working Architecture of Web RTC

1) *Your Web APP:* A third-party developer web based application with video and audio chat capabilities powered by the web API for real time communications.
2) *Web API:* An API to be used by third party developers for developing web based videochat-like applications.
3) *WebRTC Native C++ API:* An API layer that enables browser makers to easily implement the Web API proposal.
4) *Transport / Session:* The session components are built by re-using components from libjingle, without using or requiring the xmpp/jingle protocol.
5) *RTP Stack:* A network stack for RTP, the Real Time Protocol.
6) *STUN/ICE:* A component allowing calls to use the STUN and ICE mechanisms to establish connections across various types of networks.
7) *Session Management:* An abstracted session layer, allowing for call setup and management layer. This leaves the protocol implementation decision to the application developer.
8) *Voice Engine:* Voice Engine is a framework for the audio media chain, from sound card to the network.

9) *iSAC / iLBC / Opus : iSAC:* A wideband and super wideband audio codec for VoIP and streaming audio. iSAC uses 16 kHz or 32 kHz sampling frequency with an adaptive and variable bit rate of 12 to 52 kbps.
10) *iLBC:* A narrowband speech codec for VoIP and streaming audio. Uses 8 kHz sampling frequency with a bitrate of 15.2 kbps for 20ms frames and 13.33 kbps for 30ms frames. Defined by IETF RFCs 3951 and 3952.
11) *Opus:* Supports constant and variable bitrate encoding from 6 kbit/s to 510 kbit/s, frame sizes from 2.5 ms to 60 ms, and various sampling rates from 8 kHz (with 4 kHz bandwidth) to 48 kHz (with 20 kHz bandwidth, where the entire hearing range of the human auditory system can be reproduced). Defined by IETF RFC 6176. NetEQ for Voice
12) A dynamic jitter buffer and error concealment algorithm used for concealing the negative effects of network jitter and packet loss. Keeps latency as low as possible while maintaining the highest voice quality.
13) *Acoustic Echo Canceler (AEC):* The Acoustic Echo Canceler is a software based signal processing component that removes, in real time, the acoustic echo resulting from the voice being played out coming into the active microphone.
14) *Noise Reduction (NR):* The Noise Reduction component is a software based signal processing component that removes certain types of background noise usually associated with VoIP. (Hiss, fan noise, etc…)
15) *Video Engine:* Video Engine is a framework video media chain for video, from camera to the network, and from network to the screen.
16) *VP8:* Video codec from the WebM Project. Well suited for RTC as it is designed for low latency.
17) *Video Jitter Buffer:* Dynamic Jitter Buffer for video. Helps conceal the effects of jitter and packet loss on overall video quality.
18) *Image Enhancements:* For example, removes video noise from the image capture by the webcam.

V. WEB RTC DEVELOPMENT ENVIRONMENT

Before we start building our WebRTC applications, we should set our coding environment. First of all, you should have a text editor or IDE where you can edit HTML and Java Script. The other requirement for common WebRTC applications is having a server to host the HTML and Java Script files.[10] The code will not work just by double-clicking on the files because the browser is not allowed to connect to cameras and microphones unless the files are being served by an actual server. This is done obviously due to the security issues. There are tons of different web servers, but in this tutorial, we are going to use Node.js with node-static −

A. Visit https://nodejs.org/en/ and download the latest Node.js version.
B. Unpack it to the /usr/local/nodejs directory.
C. Open the /home/YOUR_USERNAME/.profile file and add the following line to the end − *export PATH=$PATH:/usr/local/nodejs/bin*
D. The you can restart your computer or run source*/home/YOUR_USERNAME/.profile*
E. Now the *node* command should be available from the command line. The *npm* command is also available. NMP is the package manager for Node.js. You can learn more at https://www.npmjs.com/.
F. Open up a terminal and run *sudo npm install -g node-static*. This will install the static web server for Node.js.
G. Now navigate to any directory containing the HTML files and run the *static* command inside the directory to start your web server.
H. You can navigate to http://localhost:8080 to see your files.

VI. TOOLS TO IMPLEMENT WEB RTC ON MOBILE

But yes the biggest challenge for developers can be that Apple browser Safari does not support WebRTC so you may have to go NATIVE way instead of web browser based way in app development. Let's now move into a resolution finding mode. [11] Here are some popular third party WebRTC SDKs that provide iOS as well as Android support.

1) *Telefonica Tokbox :* Among the most used and most common WebRTC SDKs is Tokbox and its API platform OpenTok. More than 80,000 users are currently using WebRTC SDKs. Major Brands like Universal Music Group, Bridgestone, Golf and many others are prominent users of Tokbox. The platform is fully featured, robust and scalable platform which comes with value addition of huge plug-ins. Algoworks holds niche in using Tokbox for developing many WebRTC mobile applications.
2) *Open Clove:* Open Clove offers WebRTC support with large number of demos and sample codes. FreeMAD offers free source code for two-way communications and peer-to-peer communication with video the default voice and messaging options. OpenClove's pay platform includes support for multi-party conferencing, streaming, recording and other advanced features.

3) *Hook Flash:* Hook Flash is among the only WebRTC SDK which was first to provide support for iOS App Development. It was then followed by Android and JavaScript framework.
4) *AddLive WebRTC Platform:* AddLive is a very powerful voice and video communication API. AddLive is used by over 5000 business and provides support for native iOS WebRTC application development. It is among the awarded webRTC tool in 2013.
5) *Bistri:* Bistri is a video calling solution fully based on WebRTC standard. It is available on web, Android and on iOS too. The solution allow users to get in touch with one click. Bistri.com provides everyone with their own link, like an online phone number. Bistri provides a disruptive way of communicating and share things online with privacy, confidentiality and simplicity.

VII. LIMITATIONS OF WEB RTC

Currently WebRTC suffers from a number of limitations which are outlined in the following. However, while we were able to implement a prototype there were still multiple restrictions involved [12]:

A. There is currently a interoperability issue between browser. This led to implementation difficulties among browsers. For example the library PeerJS currently supports the Google Chrome browser only, because WebRTC is implemented in Mozilla Firefox differently.
B. The browser implementations are currently in alpha or beta status and as a result have a number of bugs and may terminate unexpectedly.
C. The WebRTC API does not yet offer functions for connection management and establishment. Instead, a second communication channel is necessary to establish a connection. We were using XmlHttp Requests and Web Sockets to overcome this limitation.
D. Another problem for WebRTC technology is the list of essential codecs. At the instant all the participating companies have come to the agreement only on one thing – WebRTC needs one main codec which will be supported by all browsers and thus will be cross platform.

VIII. CONCLUSION

WebRTC will help us to minimize the use of third party plugins, improve the security of the contents in browser. WebRTC is as innovatory a market disruption for telecom as HTML was for the internet. Companies that are willing to adopt this technology will have plentiful of business opportunities. Although, no browser has completely implemented the current WebRTC draft, it is likely that it will be implemented in future releases [13]. With the many peer-to-peer applications and solutions that exist to date, WebRTC could greatly empower the web applications of the future

REFERENCES

[1] L. L. Fernandez, M. P. Diaz, M. R. Benitez, F. J. Lopez, J. A. Santos, J.A. "Catalysing the success of WebRTC for the provision of advanced multimedia real-time communication services," Intelligence in Next Generation Networks (ICIN), 2013 17th International Conference on , pp.23,30, 2013.
[2] W. Elleuch, "Models for multimedia conference between browsers based on WebRTC," Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on , pp.279,284, 2013
[3] A. Zeidan, A. Lehmann, U. Trick, "WebRTC enabled multimedia conferencing and collaboration solution," WTC 2014; World Telecommunications Congress 2014; Proceedings of , pp.1,6, 2014.
[4] P. Edholm. The Benefits of WebRTC. [Online]. Available at http://www.webrtcworld.com/topics/fromtheexperts/articles/319037-benefits-webrtc.htm [Accessed on: 11 December 2014].
[5] K. Singh, A. Johnston, J. Yoakum, "Taking on webRTC in an enterprise" in Communications Magazine, IEEE, April 2013, ISBN 0163-6804.
[6] Jennings, C., Narayanan, A., Burnett, D., Bergkvist, A.: WebRTC 1.0: real-time communication between browsers. W3C Editor's Draft, W3C, March 2015. http://w3c.github.io/webrtc-pc/. Accessed April 2015
[7] http://www.html5rocks.com/en/tutorials/webrtc/basics/
[8] http://www.yorktel.com/wpcontent/uploads/2012/06/3-Things-You-Need-To-Know AboutWebRTC_w_Links_Final.pdf
[9] Hitendra Patil, Amar Buchade, "Review and Study of Real Time Video Collaboration Framework WEBRTC " url http://www.ijcsit.com/docs/Volume%205/vol5issue01/ijcsit2014050123.pdf
[10] H. Heitkötter, S. Hanschke, T. A. Majchrzak, "Evaluating cross-platform development approaches for mobile applications" in Web information systems and technologies, Berlin Heidelberg:Springer, pp. 120-138, 2013.
[11] https://www.algoworks.com/blog/webrtc-mobile-app-development/
[12] S. Loreto, Romano, S. Pietro, "Real-Time Communications in the Web: Issues, Achievements, and Ongoing Standardization Efforts," Internet Computing, IEEE , vol.16, no.5, pp.68,73, 2012
[13] Kudan Singh, John Buford"Developing WebRTC-based team apps with the cross-platformmobile framework",13th IEEE Annual consumer communications and Networking conference,pp.236-242, Year:2016.