EENA Technical Committee Document

WebRTC and Emergency Services

Title:	WebRTC and Emergency Services		
Version:	1.0		
Revision Date:	02-11-2016		
Status of the document:	Draft	For comments	**Approved**

EENA
info@eena.org - www.eena.org

Authors and contributors to this document

This document was written by members of EENA:

Authors	Country / Organisation
Andrew Hutton	UK, Unify Enterprise Communications Ltd.

Contributors	Country / Organisation
Markus Bornheim	Germany / Avaya Deutschland GmbH
Ian Colville	Aculab
Razvan Craciunescu	Teamnet
Gunnar Hellström	Omnitor
Cristina Lumbreras	EENA
Christian Repaski	Eurofunk

Table of contents

1 Executive Summary 4
2 Definitions 4
3 Introduction to WebRTC 5
3.1 Why is WebRTC Disruptive? 6
4 WebRTC Architecture. 7
4.1 Security. 8
4.2 WebRTC on Mobile. 8
4.3 Current standardization and implementation status. 8
4.4 WebRTC Platform as a Service – PaaS. 9
5 WebRTC and Emergency Services. 9
5.1 Emergency Applications. 10
5.2 Incident Reporting and Enhanced Situation Awareness. 10
5.3 Non-Emergency Applications. 11
5.4 Control Room to First Responder Communications. 11
5.5 Emergency Services Interoperability and Shared Situation Awareness. 11
6 Conclusion. 12
7 References 12

EENA
info@eena.org - www.eena.org

1 Executive Summary

This paper provides an overview of the current status of Web-based real-time communications technology (WebRTC), which is under development in the IETF and W3C standards bodies. It looks at the potential impact of this technology on both the way the public interacts with the emergency services and internal communications within emergency service organisations, including providing support to first responders.

In essence, WebRTC is an API that allows Web application developers to build real-time communications (voice, video, and data) into their Web applications, in much the same way as other Web APIs provide access to location or other services provided by the Web platform or browser.

WebRTC is already in use today, with Google reporting that they are aware of over 800 companies developing WebRTC-based applications. It is almost certain that you will have already used WebRTC-based applications without even knowing it. It is also probably true to say that the majority of mobile applications being built today, which provide real-time voice and video capabilities including those focused on public safety, are already making use of WebRTC.

The technology is very disruptive to the real-time communications industry, because it provides Web developers with the tools to integrate high quality secure voice, video, and data communications capabilities with their Web applications, including those related to emergency scenarios.

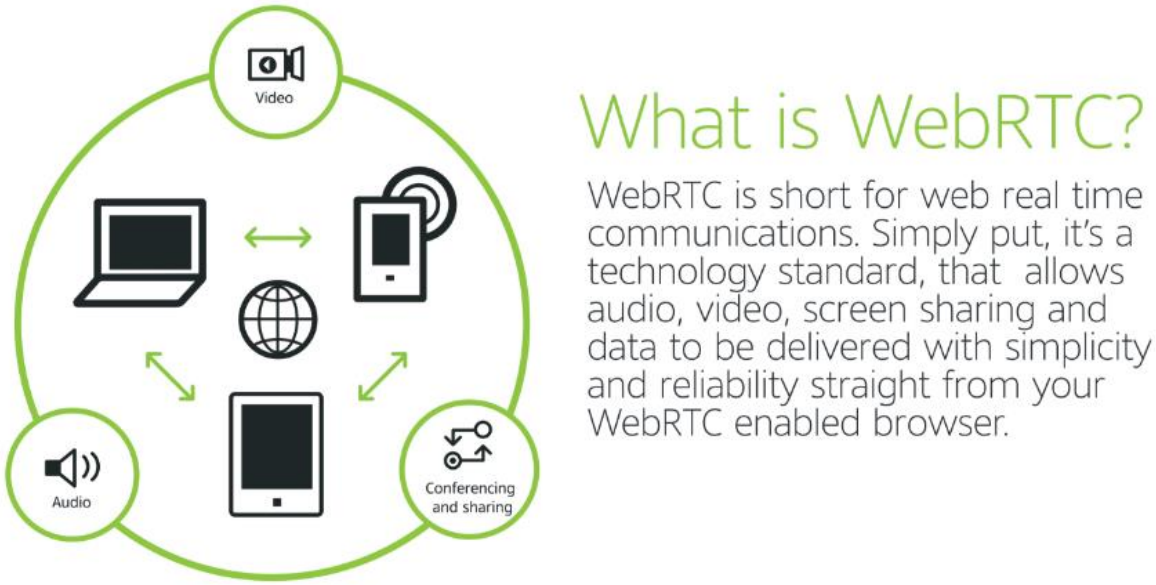
The fact that WebRTC provides the full media stack, including voice, video and data, and uses strong security, makes WebRTC very well suited for use in public safety applications. It is easy to see how WebRTC technology can be used to enhance the way in which the public communicates with emergency service organisations, for example, by adding high quality, secure video capabilities to emergency calling applications. WebRTC applications can also be used to enhance operational effectiveness and situational awareness, by providing rich, secure and contextual-based communications for use between emergency responders and the control room.

Whilst initially WebRTC will most likely supplement existing primary emergency calling mechanisms, it is relatively easy to envisage WebRTC causing such disruption that it becomes the technology behind the primary means of contacting the emergency services. It is possible that WebRTC will enable a full transition from current phone-centric PSAPs to a fully Web-based approach.

2 Definitions

Term	Definition
API	Application Programming Interface
IETF	Internet Engineering Task Force (www.ietf.org)
W3C	World Wide Web Consortium (www.w3c.org)
WebRTC	Web Real-Time Communications.
OTT	Over the Top – Mobile applications providing services over the public internet.
JESIP	Joint Emergency Services Interoperability Program (JESIP) http://www.jesip.org.uk/home
PaaS	Platform as a Service.

3 Introduction to WebRTC



Web-based real-time communications (WebRTC) is already causing major disruption within the real-time communications industry, even though the first phase of standardization is not yet finalised. Google, who initiated the project, estimates that in late 2015 there were already over 800 companies developing real-time applications based on WebRTC technology. It is almost certain, therefore, that you have already used WebRTC-based applications.

WebRTC was initiated in late 2010 by the Google team working on Google Hangouts, who realised that to achieve true, web-based real-time communications, standardization of protocols and APIs across all browsers was going to be needed.

The Google team invited the relevant standardization experts from both the Internet Engineering Task Force (IETF) and the World Wide Web Consortium (W3C) to a workshop at their Mountain View Headquarters to as we say in standardization "to get a feeling of the room". The feedback they got from the industry was very positive, for a number of reasons.

In 2010, many companies in the communications industry were already looking to deploy unified communications services to the cloud, and deploy services to their customers through web browsers. Another factor that was discussed at length during the IETF80 meeting, in March 2011, was the need to increase the speed of innovation in the telecoms industry to match that of the Web, and to do that, a different approach to VoIP standardization was needed.

3.1 Why is WebRTC Disruptive?

There has been a lot said about WebRTC being a technology that will be highly disruptive to both the consumer and enterprise communications industries, including mobile. That is big talk, however, there is no doubt that WebRTC is already the cause of much disruption.

One reason is simply that WebRTC makes real-time communications, and especially real-time video conferencing, simply another tool available to the millions of Web application developers. The WebRTC APIs provide an abstraction layer, which removes the need for Web developers to understand the complexities of real-time communications. Web application developers now have an API, written in a language they understand, that can be used to embed real-time voice, video, and data in any Web application.

Having such capabilities in the browser means that real-time communications can easily be embedded in any Web application, be it a complex business process type application, social media application, or an emergency calling application. This means that you can initiate your real-time conversations within the context of whatever you are doing, rather than having to leave that tool and use something else; that is what is termed contextual communications.

EENA
info@eena.org - www.eena.org

4 WebRTC Architecture.

The essence of the WebRTC architecture is that WebRTC clients transmit and receive audio, video, and data in real-time to their peers. Audio and video use standard codecs, which are transported over an encrypted media connection. In basic terms, the client-side application script (JavaScript) invokes a W3C specified browser API [3] that instructs the browser to transfer audio or video between local input/output devices (microphone / speaker or camera / display) and a remote endpoint via IETF defined protocols [2]. However, there are other considerations, such as session control signalling, connectivity checking, NAT traversal, security and codec negotiation and media transport. The basic architecture is shown in Figure 1.

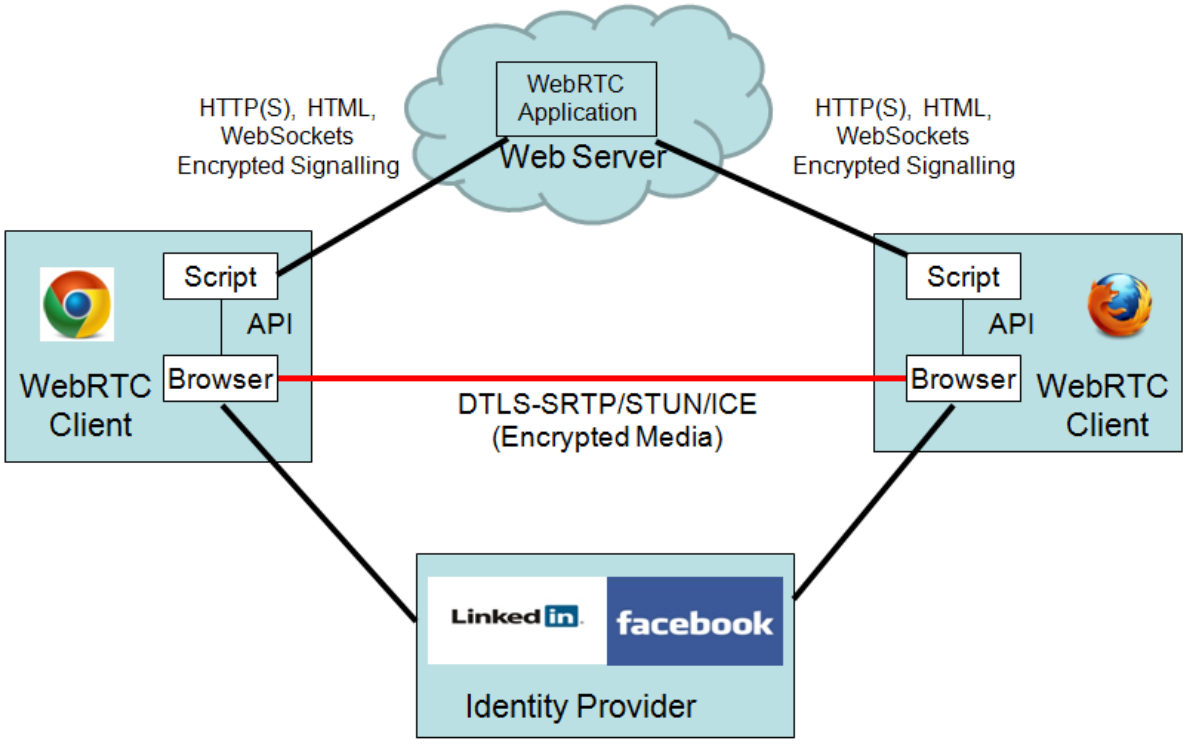


Figure 1 - WebRTC Architecture

Figure 1, "signalling" is shown between the client and the server. Some form of signalling is required, so that the application script running on the server can coordinate the two (or more) clients involved in a communication. However, as WebRTC follows a Web model in which only the basic building blocks are standardized, the actual signalling protocol for session establishment is not specified for WebRTC, and which signalling protocol to use for that purpose is a design decision for the WebRTC application developer to make.

The fact that the session control signalling between the WebRTC client and server is not specified has been the source of much discussion and confusion. Many people have interpreted that as a weakness, but actually it is a strength and a deliberate design decision made by the IETF working group [7]. It is a strength, because it means that the signalling protocol is left to the application developer who can make it as simple or as complex as necessary, and standardization is not needed to add new functionality. After all, why would the most basic communication application need to be burdened with the same protocol complexity as the most complex application?

The media plane is another matter, because WebRTC provides the possibility of direct media connections between browsers. This has to be the same across all browsers and, therefore, needs to be standardized, which is the role of the IETF. The fact that WebRTC provides the possibility of direct connections between browsers is a revolutionary step for the Web as this was not previously possible; it also has major security implications, which have had to be addressed by the standards bodies.

The other component of the WebRTC architecture that had to be standardized is the browser API. This was necessary to enable WebRTC applications to be browser independent. The WebRTC API [3] is the responsibility of the W3C WebRTC working group [6].

WebRTC applications may optionally make use of an interface between the browser and a web-based identity provider (IdP) that is accessed via the WebRTC API [3]. This provides a means for the media channels established using WebRTC to be authenticated by an identity provider with which the user has a relationship.

4.1 Security.

Security and privacy are at the heart of everything that happens in Internet related standardization. In the case of WebRTC, this has the highest priority in both the IETF and the W3C.

WebRTC media is always encrypted using either DTLS-SRTP (RFC 5764) [1] in the case of audio and video, or by using SCTP (RFC 4960) [4] encapsulated in DTLS (RFC 6347) [5] in the case of data. The increased use of encryption for communications has been the subject of debate, even at government level, but in the WebRTC standardization bodies the decision to always encrypt was not controversial. Previous experience with the Session Initiation Protocol (SIP) proved that the complexity caused by making encryption optional makes it very difficult to deploy as an add-on. Therefore an 'always on' policy for encryption was an easy decision to make for WebRTC.

4.2 WebRTC on Mobile.

At the beginning of the WebRTC project both the IETF and W3C scoped their work to cover browser to browser interoperability. However, it soon became clear that WebRTC on mobile devices was a major requirement, and therefore a solution to help the developer community build WebRTC into native applications was needed.

Google realised early on that WebRTC could not be successful without a robust solution for mobile devices, be they iOS- or Android-based, and therefore they have put much effort into supporting the industry by providing support on those platforms. WebRTC is, therefore, well supported on Android-based devices, with both browsers (Chrome, Firefox and Opera) and native apps implementing WebRTC.

Apple's current lack of support for WebRTC in iOS (WebKit) means that browsers running on iOS-based devices are unable to provide WebRTC support. Apple has been typically quiet regarding its support for WebRTC, but it is widely expected that Apple will at some point in time provide support. The lack of support in iOS has been a concern to many, but actually this has not held back WebRTC deployment, as WebRTC on mobile devices is primarily based on native applications rather than browsers and Google have provided support for iOS based application development.

Apple's PC operating system, Mac OS X, does not have the same limitations as iOS, and browsers such as Chrome, Firefox, and Opera, running on MacOS, support WebRTC.

4.3 Current standardization and implementation status.

Is has been over five years since the WebRTC project started and still the WebRTC 1.0 standards are not finalised. However, in the Web world, nobody waits for the ratification of standards. It is the agile model of development that is followed, meaning that experimentation, implementation, and standardization run in parallel as an iterative process, enabling Web speed innovation. Having said that, the IETF and W3C standards groups are keen to finalise WebRTC 1.0, and hopefully this will happen by the end of 2016.

With regard to browser support, clearly Google Chrome has been leading the pack, with both Firefox and Opera trying hard to keep pace, which they have done reasonable well. WebRTC 1.0 should be finalised by the end of

2016, and following that, we can expect to see standard compliant implementations in all the major browsers. As always, Apple has not made its plans public and has so far not released WebRTC support for the Safari browser, however, as WebRTC 1.0 is approaching completion, it is widely expected that Apple will provide support for the WebRTC API.

Microsoft has, until recently, been following a different track for WebRTC standardization, and participated in work to define an alternative API called Object Real-Time Communications (ORTC) [11], however, most of the ORTC API has now been merged with the WebRTC 1.0 API [3] and Microsoft has announced that its Edge browser will align with the other browsers and support WebRTC 1.0 [12].

The implementation of WebRTC in browsers presents a unique challenge to the browser vendors, because it is the first time that browsers have been required to communicate directly with each other, rather than via a web application. To promote interoperability between browsers, the International Multimedia Telecommunications Consortium (IMTC) has established a WebRTC Interoperability group [10] and holds test events for vendors.

Many WebRTC-based applications are already deployed and providing the basis for robust commercial collaboration services. For example, in 2014, Unify launched a WebRTC-based collaboration application [8], as did Avaya, in 2016 [9].

4.4 WebRTC Platform as a Service – PaaS.

The W3C WebRTC API [3] makes it straight forward for a skilled Web developer to write an application that establishes a peer-to-peer multi-media connection between browsers, and Google's WebRTC project makes it possible to extend that application to mobile platforms.

However, it is much more complex to build applications that are highly robust, scale to a very large number of users, and provide complex, multi-party video conferencing features. Interworking with existing systems, such as legacy telephony systems, also requires expert knowledge, software, and hardware.

One way of removing these barriers to deploying WebRTC-based applications is to use a WebRTC platform and API built by a vendor who already has these capabilities, and who has made the service available to others. Such a service is known as a WebRTC platform as a service (PaaS), and there are a number of vendors providing such platforms.

Building an application using a WebRTC PaaS means the application developer can make use of APIs developed by the PaaS vendor, which provide an even higher level of abstraction than the W3C WebRTC API, and removes much of the complexity in building a back-end platform that is secure, scalable, and provides the complex functionality needed by many applications.

5 WebRTC and Emergency Services.

There is little doubt that WebRTC will be the dominant technology for real-time collaboration applications deployed in the cloud over the next few years [13]. It also has some unique properties that make it particularly suitable for use within emergency service related applications, including the built-in encryption and identity features, and the promise of being widely available via your browser, without any installation.

The following sections will explore some use cases for the use of WebRTC in the context of emergency services applications, which provide enhanced services for collaboration between the public and emergency service organisations, and also within and between the different emergency service organisations themselves.

5.1 Emergency Applications.

There are an increasing number of emergency calling related applications being developed, which mostly are aimed for use within a particular public safety context, such as motorcycle safety (https://www.realrider.com/) or personal safety (http://www.onescream.com/). These currently use the native phone application of the mobile device on which they are hosted to make any necessary emergency call. Whilst at least for the foreseeable future this may remain the case for the voice part, WebRTC provides an opportunity to add richer media capabilities, such as video and text, without having to wait for the whole PSTN based emergency calling infrastructure to be upgraded to support video and other rich media.

There is even potential for using WebRTC for the audio part of the emergency call, for example, when there is no cellular coverage, but, for example, a Wi-Fi hotspot is available.

All these emergency calling applications have to be pre-installed on the user's device and are therefore not limited by the fact that WebRTC is not currently supported on all mobile browsers.

5.2 Incident Reporting and Enhanced Situation Awareness.

The fact that WebRTC is intended to be a Web application, available through any Web browser without any additional installation being required, promises to make every mobile device capable of providing a high quality, real-time video feed to the control room during an emergency.

Some emergency services have already recognized this opportunity and developed prototype WebRTC-based services, which enable a member of the public reporting an incident to stream video to the control room. For example, in the UK, the West Midlands Fire Service has developed and trialled 999eye (https://999eye.wordpress.com/). The 999eye application is a Web application that is invoked by the control room dispatcher sending a text message (SMS, Whatsapp, etc.) containing a unique URL for the 999eye application, which identifies the specific incident. When the user selects the URL, the Web browser connects to the 999eye application and is able to stream video directly within the context of the given incident to the dispatcher, potentially giving the dispatcher much improved situational awareness and, therefore, enabling him or her to make better informed decisions.

This type of application is reliant on WebRTC being available on the default browser of the device, without any additional download to the device. The fact that Apple does not currently make WebRTC available on Safari means this does not currently work on an Apple device. However it is widely expected that Apple will soon make the WebRTC API available in Safari, therefore, removing this barrier.

None of these current applications replace, or are even intended to replace, the existing primary means of contacting the emergency services, which in the vast majority of cases is via landline or more likely via a mobile telecoms provider. However, the rapid growth in over the top (OTT) voice applications means that within the next decade or so we might to see changes even in this area, and WebRTC may become a candidate for even the primary means to access the emergency services.

It is easy to envisage a future fully Web-based approach in a construct that starts off from a national Web page (e.g. www.112.yourcountry.org), offering a communicate-button that allows callers to get in touch with emergency services through text-based communications as well as real-time voice and video through WebRTC. This approach not only expands the concept of "Total Conversation" into the Web and thus allows equal access to emergency services with the inclusion of all citizens, but also would engage with citizens in a true "Omni-Channel" way, leaving the choice of channel to the citizen, based on what the situation requires and the capabilities available in that situation. In a multi-year transition scenario from the current, phone-centric PSAP world to full next generation emergency communications, WebRTC gateways would even allow at least the opportunity of bringing the voice communication element from a website to today's existing emergency calling model.

5.3 Non-Emergency Applications.

A number of services aimed at providing access to public safety and health organisations in non-emergency scenarios are available today. For example, the NHS 111 service in the UK which provides health advice in less urgent scenarios.

For these types of applications, WebRTC-based services are ideally suited, especially when the service already has Web pages that the service user is likely to browse before making the call for advice. This is an example of how WebRTC enables contextual communications by embedding the communication with the context of an existing interaction with the service user.

Since the WebRTC API is built-in to the browser itself and can be used by developers with very little knowledge of real-time communications, adding voice and video capabilities to these websites, and even mobile applications, becomes a reasonably simple task. When this is complemented by the use of a WebRTC PaaS, as described in section 4.4, then adding such capabilities to an existing website can be made to be even simpler.

5.4 Control Room to First Responder Communications.

In section 5.2, we discussed how WebRTC might be used by the general public to enhance situational awareness during conversations with a dispatcher, and the fact there are some limitations due to the fact that not all browsers support WebRTC today. However, when communicating within the closed user group of first responders and the control room, no such issues exist as it can be assumed that a WebRTC enabled application can be pre-installed on the relevant devices. If the same application is also installed across the emergency services, this can provide a very easy to deploy means to improve situation awareness and collaboration across services.

5.5 Emergency Services Interoperability and Shared Situation Awareness.

WebRTC communications, when combined with contextual communications and collaboration applications that can provide text, voice, video, screen and document sharing, which can be recorded within the context of an incident.

In the UK, the Joint Emergency Services Interoperability Program (JESIP) has done some work aimed at improving the way in which different emergency service organisations work together when responding to major incidents. One of the conclusions they made was that:

> *"Commanders arriving at the scene take too long or don't make contact with commanders from the others services. This leads to poor information sharing, lack of communication and no joint understanding of the unfolding emergency".*

Contextual communications applications with embedded WebRTC technology can quickly overcome interoperability issues between organisations, due to it being Web-based and, therefore, extremely easy to deploy across services.

Given that WebRTC provides 'always on' encryption and integrity protection for the media, it would seem that WebRTC is an ideal technology for enhancing the communications between first responders and the control room as well as between emergency service organisations.

6 Conclusion.

WebRTC is an open standards-based technology, encompassing all modes of real-time communications (voice, video, and data), which has been developed within the IETF and W3C over the past few years. It is already widely deployed and is the current technology of choice for most OTT / cloud-based applications that provide real-time communications capabilities, including those targeted at emergency services.

The use of the latest encryption technology and other features, such as the use of the high quality audio and video codecs, and options for authentication using Web-based identity providers, make WebRTC very well suited for use within emergency service related applications.

The current status of implementation in browsers, especially the lack of support in Safari, means it is not possible for all smart-phone users to access WebRTC services from their mobile device using their default browser, but this situation is likely to change. The use of native mobile applications with a built in WebRTC stack, however, means that WebRTC is already the technology of choice for mobile applications that include audio and video conferencing capabilities.

WebRTC-based contextual communications applications are likely to play a part in improving cross service interoperability and, therefore, help improve situational awareness during major incidents, and the tools to do that are already in existence.

7 References

[1] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <http://www.rfc-editor.org/info/rfc5764>.

[2] Alvestrand, H., "Overview: Real Time Protocols for Browser-based Applications", (work in progress), January 2016. https://tools.ietf.org/html/draft-ietf-rtcweb-overview-15

[3] Bergkvist, A., Burnett, D., Narayanan, A., and C. Jennings, "WebRTC 1.0: Real-time Communication Between Browsers", World Wide Web Consortium. https://www.w3.org/TR/webrtc/ (work in progress).

[4] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <http://www.rfc-editor.org/info/rfc4960>.

[5] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

[6] W3C Web Real-Time Communications Working Group, https://www.w3.org/2011/04/webrtc/.

[7] IETF Real-Time Communications in Web Browsers Working Group, https://datatracker.ietf.org/wg/rtcweb/charter/.

[8] Unify Web Based Business Collaboration Service - https://www.circuit.com/.

[9] Avaya Web Based Business Collaboration Service - https://www.zang.io/.

[10] IMTC – WebRTC Interoperability Activity Group - http://www.imtc.org/uc/imtc-webrtc-interoperability-activity-group-charter/.

[11] W3C Object Real-Time Communications Community Group, https://www.w3.org/community/ortc/

[12] Microsoft Edge Platform Status, WebRTC 1.0 - https://developer.microsoft.com/en-us/microsoft-edge/platform/status/webrtcwebrtcv10api/

[13] WebRTC World Article; Reports Show WebRTC Markets Set for Massive Growth - http://www.webrtcworld.com/topics/webrtc-world/articles/416424-reports-show-webrtc-markets-set-massive-growth.htm