García, B., Gallego, M., Gortázar, F. & Bertolino, A. (2019) Understanding and estimating quality of experience in WebRTC applications. *Computing* 101(11), pp. 1585–1607.

DOI: 10.1007/s00607-018-0669-7

Noname manuscript No.
(will be inserted by the editor)

Understanding and Estimating Quality of Experience in WebRTC Applications

Boni García · Micael Gallego · Francisco Gortázar · Antonia Bertolino

Received: 22 March 2018 / Accepted: 24 September 2018

Abstract WebRTC comprises a set of technologies and standards that provide real-time communication with web browsers, simplifying the embedding of voice and video communication in web applications and mobile devices. The perceived quality of WebRTC communication can be measured using Quality of Experience (QoE) indicators. QoE is defined as the degree of delight or annoyance of the user with an application or service. This paper is focused on the QoE assessment of WebRTC-based applications and its contribution is threefold. First, an analysis of how WebRTC topologies affect the quality perceived by users is provided. Second, a group of Key Performance Indicators for estimating the QoE of WebRTC users is proposed. Finally, a systematic survey of the literature on QoE assessment in the WebRTC arena is presented.

Keywords WebRTC · Quality of Experience · QoE Management

1 Introduction

Multimedia applications and services are becoming the main force of the Internet. A recent forecast by Cisco [14] shows that IP video traffic will be 82% of all consumer Internet traffic by 2021. Among the diversity and multiplicity of multimedia technologies, in this paper we focus on Web Real-Time Communications (WebRTC), which is a set of emerging technologies and APIs with

Boni García, Micael Gallego, Francisco Gortázar
Universidad Rey Juan Carlos
E-mail: boni.garcia@urjc.es, micael.gallego@urjc.es, francisco.gortazar@urjc.es

Antonia Bertolino
Consiglio Nazionale delle Ricerche
E-mail: antonia.bertolino@isti.cnr.it

This is a post-peer-review, pre-copyedit version of an article published in Computing. The final authenticated version is available online at: https://doi.org/10.1007/s00607-018-0669-7

the purpose of adding real-time media communications directly between web browsers (and also mobile devices) [46]. A popular communication platform based on WebRTC is Google Hangouts. Moreover, nowadays WebRTC is more and more used in different conferencing systems with multiple participants.

WebRTC is a joint standardization effort between the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF). On the one hand, W3C defines the JavaScript APIs and the standard HTML5 tags to enable peer-to-peer (P2P) connections between web-enabled devices. On the other hand, IETF defines the underlying communication protocols for the setup and management of a reliable communication channel between browsers. WebRTC has come a long way since its inception in May 2011. Among its highlights, we can point to the interoperability between Chrome and Firefox browsers in 2013, and the support for Android mobile in 2014. Such market impetus is expected to continue growing. A recent analyst's report predicts that with Apple and Microsoft supporting WebRTC in their browsers, there might be 7 billion WebRTC-compliant devices by 2020 [55]. With such a strong growth rate, it is imperative for developers and practitioners to have a strategy in place to efficiently assess the quality of WebRTC-based applications.

Software quality is an "elusive target" [44], and since the early 70's there has been a wide debate in the literature on what it means and how it can be measured. One definition was proposed in 2005 in the ISO 9000 standard, stating that quality is the "degree to which a set of inherent characteristics fulfils requirements" [27]. However, conformance to requirements is only one of several possible views of a product's quality, the one that Garvin called manufacturing quality [19]. Another important view, the user's view [19], is related with user satisfaction: "Quality in use" was proposed by Bevan [4] to measure the extent to which a software system meets the user's needs in a working environment.

In recent years, the term "Quality of Experience" (QoE) has gained momentum, mainly with respect to media transmission systems and services. In parallel to the re-consideration of the importance of user satisfaction within the term "quality", the term QoE was coined in contrast with the widely used term of Quality of Service (QoS) to express the notion that users' perceptions be addressed. In this context, the management of QoE is becoming a key aspect for researchers and practitioners. As described in [24], QoE management requires three basic steps:

1. Understanding and modeling QoE: On the one hand, to understand QoE for a given application, an analysis of the effect of disturbances on the user's perceived quality should be carried out. On the other hand, QoE models should be specified using measurable parameters.
2. Estimating and monitoring QoE: The QoE is estimated by means of the models developed in the first step. Monitoring includes the retrieval of information about network conditions (e.g. available bandwidth, packet loss) or terminal capabilities (e.g. CPU power, resolution, etc.), among other factors.

3. Adapting and controlling QoE: The final step is the dynamic adjustment of the corresponding influential factors based on a knowledge of the underlying QoE model, so as to deliver the optimal QoE.

This paper focuses on the first and second of the above steps in the context of WebRTC applications. Hence, our first objective is to understand and model the specific aspects of WebRTC that affect QoE. Second, we aim to propose a set of measurable factors to estimate the QoE of a WebRTC application. To accomplish these two objectives, we first provide a comprehensive review of the historical and theoretical background of QoE measurement in Section 2. Then, we analyse the different WebRTC topologies and their impact on the QoE perceived by the end users in Section 3. Based on this analysis, we propose a set of system Key Performance Indicators (KPIs) for estimating the QoE for a WebRTC application in Section 4. Then, we present a systematic survey of the related literature in Section 5. The goal of this survey is twofold: on the one hand, to provide a status quo about the adoption of QoE assessment in the WebRTC arena; on the other hand, to evaluate the extent to which our proposed KPIs are used in the current state of the art. Then, Section 6 analyses the main contributions, challenges, and limitations of the research presented in this paper. Finally, the conclusions of this paper and possible future research are summarized in Section 7.

2 Methods to measure QoE

QoS is the most widely used way of measuring the performance of a service, and has been defined by the International Telecommunication Union, Telecommunications Standardization Sector (ITU-T) as "the totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service" [32]. QoS is used to quantify conditions in Service Level Agreements (SLAs) between providers and customers [39]. QoS is therefore particularly pertinent for applications that require a given minimum network connection to operate properly, such as Voice over IP (VoIP), video conferencing, and safety-critical applications, which may all require a good end-to-end connection.

Regarding distributed multimedia systems, the capability of monitoring and ensuring QoS for such systems is critical, and includes two parts: QoS provisioning from the network and QoS provisioning from the media application. On the one hand, the challenges facing network QoS provisioning include unreliable channels, bandwidth constraints, and heterogeneous access technologies. On the other hand, QoS provisioning from the media application includes advanced encoding schemes, error concealment, and adaptive streaming protocols [10].

Measures of media quality based on QoS parameters do not include user-related and contextual factors. As a result, they cannot represent the true user experience in multimedia systems. In 2007, Quality of Experience (QoE), a user-centric quality strategy, was formally proposed by ITU-T in order to

overcome the shortcomings of the conventional quality metrics. It defined QoE as "the overall acceptability of an application or service, as perceived subjectively by the end-user" [31]. This definition implies that QoE includes the effects of the complete end-to-end system. It also implies that the overall acceptability may be influenced by the users' expectations and context.

However, this definition has been criticized since it only includes the acceptability of QoE [58]. For that reason, a more comprehensive definition of QoE was presented in the context of the COST Action IC1003 European Network on Quality of Experience in Multimedia Systems (Qualinet). The Qualinet White Paper [6] states that "QoE is the degree of delight or annoyance of the user of an application or service." This definition has been included in Amendment 5 of the ITU-T Recommendation P.10/G.100 [36].

QoE assessment is difficult because user experience is hard to quantify. QoE assessment methods can be classified as *subjective* or *objective*. Subjective methods directly quantify QoE by soliciting users' evaluation scores. There are two main groups of subjective QoE assessment: i) Conversational tests (i.e. audio-only or audiovisual communication) are carried out to evaluate the QoE; and ii) Passive tests [16], in which users are given a series of audiovisual sequences, i.e. the original (unimpaired reference) and processed (distorted reference). After that, users are required to score the media quality [28], for example using Mean Opinion Score (MOS) [37].

While subjective tests provide the most valid way to measure QoE, they suffer from several drawbacks. First, subjective tests are time consuming and costly. In addition, subjective tests are typically conducted in controlled environments, under limited conditions. In order to overcome these issues, objective quality models have been developed. Objective quality models compute a metric as a function of QoS parameters and external factors. The output metric should correlate well with the subjective test results, which serve as the ground truth QoE. The objective quality measurement methods can be classified as follows [11]: i) Parametric packet-layer models: these models predict the QoE from the packet header information and do not analyse media signals. ii) Parametric planning models: these models use quality planning parameters for networks and terminals to predict the QoE. iii) Bit-stream-layer models: in these models, encoded bit-stream information and packet-layer information are used to estimate QoE. iv) Hybrid models: the inputs for these models are information about the signal, bitstream, and/or packet-headers. v) Media-layer models: the QoE is calculated using some reference information and the degraded audio and video signals.

Media-layer models are further divided into three types: i) Full Reference (FR): the degraded signal is compared with the original signal. ii) Reduced Reference (RR): these methods actually build upon representative parameters (typically statistical) that allow the quantification of the change of quality between the original and the distorted version. iii) No Reference (NR): stream analysis on receipt without comparing it to the original signal. In the NR methods, significant effort has been put into mapping the network statistics and application-specific factors to the quality estimation.

Table 1 Full Reference objective QoE assessment methods

Category	Method	Description
Pixel-based	MSE	Mean Squared Error (MSE) measures the average of the square of error between the distorted and reference signals
	PSNR	Peak Signal-to-Noise Ratio (PSNR) [26] is the proportion between the maximum signal and the corruption noise
Natural visual characteristics oriented	SSIM	Structural SIMilarity (SSIM) measures the difference of structure between the original and the distorted image in terms of luminance, contrast and structure [61]
	VQM	Video Quality Metric (VQM) [49] is calculated as a linear combination of several impairment parameters
	DVQ	Digital Video Quality (DVQ) [63] calculates the visual difference between the original and distorted video sequences using the Discrete Cosine Transform (DCT)
	VSNR	Visual Signal-to-Noise Ratio (VSNR) [9] quantifies the visual fidelity of natural images based on human thresholds
Perceptual oriented	PESQ	Perceptual Evaluation of Speech Quality (PESQ) [50] is a method for evaluating speech quality autonomously as the experience of a telephony system user
	POLQA	Perceptual Objective Listening Quality Assessment (POLQA) [34] evaluates perceptual audio quality addressing the weaknesses in previous models such as PESQ
	PVQM	Perceptual Video Quality (PVQM) [22] uses a linear combination of three indicators to measure perceptual video quality: edginess, temporal decorrelation, and colour error
	PEVQ	Perceptual Evaluation of Video Quality (PEVQ) [33] provides MOS scores of the video quality for IPTV, streaming video, mobile TV and video telephony

A wide range of FR methods have been proposed in the literature. These methods can be further divided into three categories [57]: i) Pixel-based: these metrics are computed by comparing the reference and the degraded signal only taking into account their physical magnitudes. ii) Natural visual characteristics oriented: the quality is calculated as perceived by the Human Vision System (HVS). iii) Perceptual oriented: the quality is predicted using Mean Opinion Score (MOS) ratings. Table 1 provides a brief summary of the relevant FR methods classified into the above-mentioned categories.

3 WebRTC topologies that affect QoE

WebRTC is a collection of standards, protocols, and APIs that enables secure P2P high-quality audio, video, and data sharing between browsers. Instead of

relying on third-party plug-ins or proprietary software, WebRTC turns real-time communication into a standard feature that any web application can leverage via a simple JavaScript API, namely *getUserMedia* (which gains access to the camera, microphone, or screen device), *RTCPeerConnection* (communication of audio and video data, encoding and decoding media, sending media over the network, NAT traversal), and *RTCDataChannel* (communication with low latency of arbitrary application data between browsers).

In order to coordinate a WebRTC session, a signaling channel between the clients is required. Signaling is the process of exchanging messages to support the media communication, such as control messages to open or terminate the media session, error messages, or network data. No signaling protocol is defined for WebRTC, making it suitable for a large number of use cases where the actual signaling protocol is selected by the developer.

Despite the fact that WebRTC has been conceived as a P2P technology, in practice it requires several infrastructure components (i.e. servers) which allow establishing WebRTC sessions between peers. A fine-grained consideration of this infrastructure is key to understanding the diverse possible scenarios in a WebRTC session, and its impact on the quality perceived by end users. This section provides a comprehensive analysis of those scenarios and discusses their impact on QoE.

3.1 Signaling

From the most basic point of view, a WebRTC-based application is a simple web application that uses the above-mentioned JavaScript APIs to access the user's media and establish real-time communication with remote peers. Therefore, in order to create a WebRTC-based application, first we need to host our application on a common web server, such as Apache, Microsoft ISS, Nginx, etc. Of course, the communication with this server is by means of the omnipresent HTTP protocol (see Fig. 1).

Moreover, we need a signaling channel to exchange the information needed to establish a media session between peers. As explained earlier, WebRTC does not define a specific signaling protocol to be used, giving the freedom to choose the most convenient one (e.g., SIP, REST, WebSocket, etc.). The only component within WebRTC dealing with signaling is SDP (Session Description Protocol) [21], which is a protocol used for describing multimedia session capabilities and negotiating them. The SDP negotiation happens based on the offer–answer exchange mechanism described in RFC 3264 [52]. An SDP offer contains information about the session, for example, whether the session is audio or video, and also the codecs to use. Regarding codecs, WebRTC peers can support any codec for audio and video, but some are Mandatory to Implement (MTI). For audio, those are Opus and G.711. Opus is an open format that provides excellent quality at the majority of bitrates. G.711 is included for compatibility with legacy systems. For video, VP8 and H.264 are MTI. H.264 is the industry standard hardware encoding and decoding, and is well

supported on mobile devices. VP8 is an open and royalty-free video codec suitable for real time.

Overall, the signaling infrastructure should provide a channel to exchange the above-mentioned messages. It is a common practice to use the same application server to provide both web and signaling for WebRTC-based services. For example, we can use a Node.js server (typically Express) or Java EE servers/containers (such as WildFly or Tomcat) to group the functionality of web and signaling server. Moreover, it is also quite common to use Software as a Service (SaaS) solutions to support the signaling channel, by means of cloud providers, such as Firebase, PubNub, Pusher, among others.

3.2 NAT traversal

In order to get media flowing from one browser to another, typically the media packets may need to pass through a firewall and NAT (Network Address Translators) devices. The selected solution to this issue in WebRTC is the use of the ICE (Interactive Connectivity Establishment) protocol [51], which in turn makes use of STUN (Session Traversal Utilities for NAT) [54] and TURN (Traversal Using Relay around NAT) [47]. STUN is a protocol that can be used by an endpoint to determine the IP address and port allocated to it by an NAT. STUN is implemented in a client–server architecture, in which a client (e.g. WebRTC peers) finds out its own public IP address simply by asking an external server (known as the STUN server), which must reside in the public Internet. With this mechanism in place, whenever two WebRTC peers behind NATs want to talk to each other, they first send binding requests to their respective STUN servers, and following a successful response from both sides, they can then use the established public IP and port tuples to exchange media. However, in practice, STUN is not sufficient to deal with all NAT topologies and network configurations [20]. In order to understand the problem, it is worth reviewing the different types of NAT devices. According to RFC 3089, there are four types of NAT devices [53]:

1. Full-cone NAT. First an internal address (*iAddr:iPort*) is mapped to an external address (*eAddr:ePort*) by the NAT device (in the so-called NAT table). After that, any external host can send packets to *iAddr:iPort* by sending packets to *eAddr:ePort*.
2. Restricted-cone NAT: These NATs work like the full-cone NAT, but with the difference that the external host can only send packets to *iAddr:iPort* (by sending packets to *eAddr:ePort*) if *iAddr:iPort* has previously sent a packet to that external host.
3. Port restricted-cone NAT: These NATs work like the restricted-cone NAT, except that in this case the restriction includes also the port numbers (not just the IP of the destination host).
4. Symmetric NAT: These NATs works like the port restricted-cone NAT, except with the further restriction that the external address (*eAddr:ePort*) changes for different destination hosts.

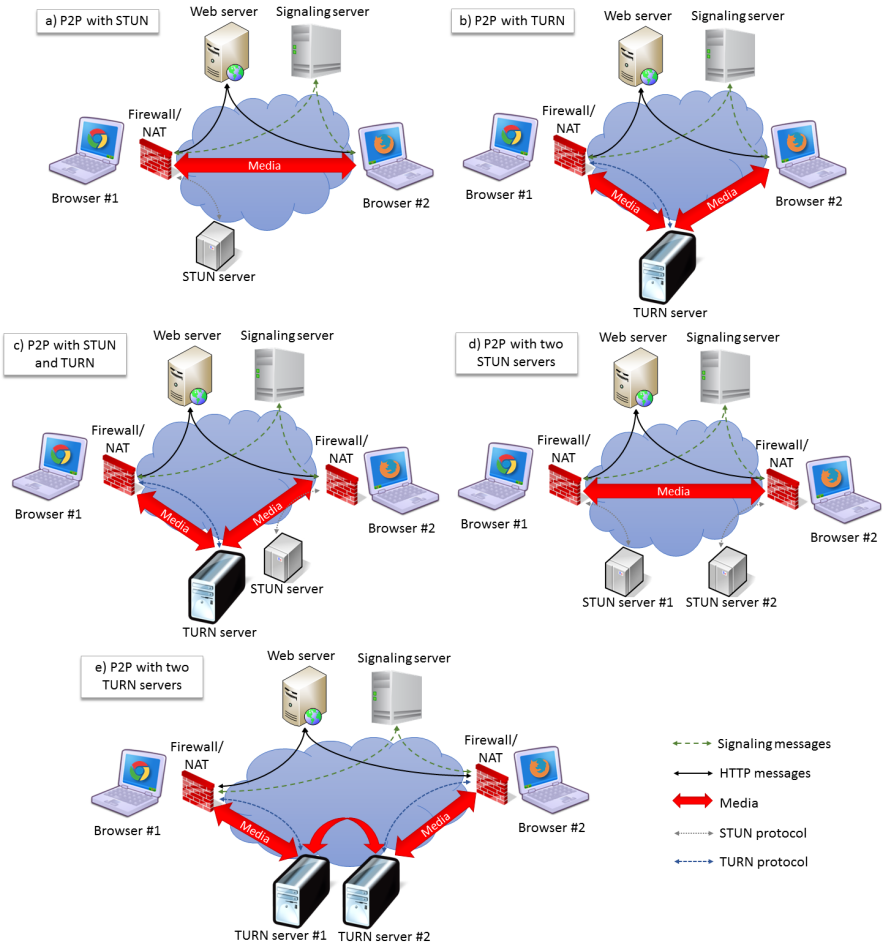

Fig. 1 WebRTC P2P scenarios with NAT traversal

Given the nature of symmetric NATs, it is not possible to establish a WebRTC media session using a STUN server for NAT traversal, due to the fact that STUN servers are not be able to determine the external address to establish media sessions (this address changes from request to request). To address this issue, whenever STUN fails, peers should use the TURN protocol as a fallback. The keyword in TURN is, of course, "relays". The TURN protocol is based on the presence and availability of a public relay server (called a TURN server) to shuttle the data between the peers. The downside in this exchange is that it is no longer P2P.

All this complexity leads to a series of different scenarios, which in the end affects the QoE of WebRTC-based services. Consider a WebRTC session between two peers when one of the peers is behind a NAT. If the NAT is non-symmetric (Fig. 1-a and Fig. 1-d), the application should know at least one

STUN server. In this case, the media flow is P2P. But in the case of a client behind a symmetric NAT (Fig. 1-b and Fig. 1-c), the WebRTC flow should be relayed through a TURN server, adding extra end-to-end latency due to the additional packet paths and the TURN server's processing time. The situation is even worse when the two peers use different TURN servers (Fig. 1-e), due to the fact that in this case the WebRTC media flow should cross both TURN servers.

Therefore, building an effective NAT traversal solution in the real world can be a difficult task. In order to simplify the process, the WebRTC stack includes an ICE agent to coordinate STUN and TURN to make a connection between peers. As described in the previous section, in order to establish a WebRTC session, peers need to exchange an SDP offer and answer. In addition to information about the session, an SDP offer also includes a list of ICE candidates. Each ICE candidate describes a method through which the originating peer is able to communicate. To build the list of ICE candidates, each peer first makes a series of requests to STUN. The server returns the public IP address and port pair that originated the request. This process is called ICE candidate gathering. Once the originating peer has finished gathering ICE candidates, it can return an SDP and the list of candidates to the destination peer through the signaling channel. That peer generates an SDP answer including its own ICE candidates. Once the peers have exchanged SDPs, they perform a series of connectivity checks ordering the ICE candidates from highest to lowest priority, looking for a valid pair. If a peer cannot find any address–port pair that achieves connectivity, it makes a request to the TURN server to obtain a media relay address. This relay address is then added to the candidate list and exchanged via the signaling channel.

The main bottleneck in this process is the time it takes to collect all the ICE candidates. This time can be considerable, as in tens of seconds to complete the gathering process. In order to optimize this process, an extension to the standard ICE protocol has been developed: Trickle ICE. This mechanism allows providing the ICE candidates incrementally, as they are discovered. Trickle ICE parallelizes the whole gathering process by providing the ability to send single or multiple ICE candidates asynchronously [38], allowing the anticipation of the connectivity checks looking for valid candidates.

3.3 Media servers

WebRTC has been conceived as a P2P architecture where browsers can directly exchange media (except in the case where a TURN server is needed). This model is sufficient for creating basic applications, but features such as group communications, media stream recording, media broadcasting, and media transcoding are difficult to implement on top of it. For this reason, many applications require using a media server. In the past few years, the expectations arising from WebRTC technologies have brought about a golden age for media server vendors. The common features of WebRTC media servers fall

into just three categories: i) Media bridging capabilities, referring to the attainment of interoperability between networks or domains having incompatible media formats or protocols. ii) Group communication capabilities, including mixing and forwarding. This type of media server includes: Multipoint Control Unit (MCU), in which each participant connects to the media server, which then mixes all inputs and sends out a single stream to each participant [62]; and Selective Forwarding Unit (SFU), in which the media server clones and forwards (i.e. routes) the received encoded media stream on to many outgoing media streams. iii) Media archiving capabilities deal with recording audiovisual streams as structured or unstructured repositories and recovering them later for visualization.

There are different media server implementations available nowadays, including Jitsi[1] (a videoconferencing system on an SFU), Janus[2] (a general purpose modular WebRTC gateway), Medooze[3] (a multiparty videoconferencing service based on MCU), Licode[4] (a videoconferencing system), and Kurento[5] (a modular media server and set of client APIs [18]).

When using a media server, the scenario is different from the traditional P2P schema. As can be seen in Fig. 2, the media server is a central component in which media are relayed. In addition, depending on the features requested by the clients, different processes should be carried out by the media server, such as transcoding, recording, and so on. Each media server provides a protocol (labeled as "control messages" in Fig. 2) for access and management.

The scenario is actually more complicated in a real-world environment, e.g. when the peers are behind different types of NATs. For instance, when one or both peers are behind a non-symmetric NAT (Fig. 2-a, Fig. 2-d) the STUN server is sufficient for establishing a WebRTC media session. Nevertheless, if one NAT is symmetric (Fig. 2-b, Fig. 2-c), media should hop two times, relying on the TURN and media server. If both NATs are symmetric (Fig. 2-e), the media flowing between the peers might have to cross up to two TURN servers and also the media server (four hops in all). This complexity leads to extra end-to-end latency, affecting the QoE for the end users.

3.4 Congestion control

In packet-switched networks, congestion occurs when the amount of data sent over the network is more than the path is able to carry. Congestion produces queuing delays, packet loss, inability to establish new connections, and, in the worst case, a network collapse. The goal of congestion control is to avoid these problems, providing robust and predictable application behaviour. For regular Internet applications, this is typically provided at the transport level by

[1] https://jitsi.org/
[2] https://janus.conf.meetecho.com/
[3] http://www.medooze.com/
[4] http://lynckia.com/licode/
[5] http://www.kurento.org/

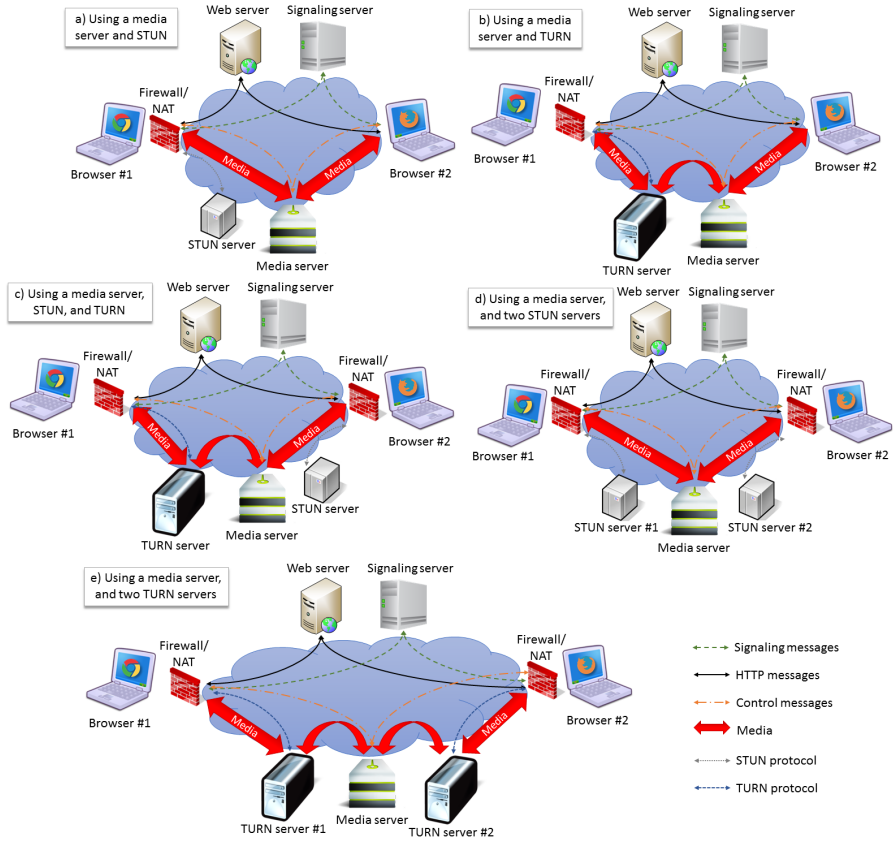

Fig. 2 WebRTC scenario with a media server and NAT traversal

the TCP protocol. WebRTC-based applications demand high and consistent bandwidth to maintain the low latency required for real-time communication, and therefore the transport protocol for media is UDP in most cases, so there are no acknowledgements or retransmissions [20]. Moreover, DTLS over UPD is used to secure media transfers between peers since encryption is a mandatory feature of WebRTC.

Since UDP does not provide a congestion control algorithm, WebRTC entities implement a custom congestion control protocol at the application layer. This protocol was first known as Receiver-side Real-time Congestion Control (RRTCC) [1] and then renamed to Google Congestion Control (GCC) [7]. GCC predicts congestion by analysing two parameters: the delay between packets and the packet loss. When a WebRTC receiver detects congestion, it sends REMB (Receiver Estimated Max Bitrate) messages to the sender. Then the sender uses that information to adapt the transmission bitrate accordingly.

WebRTC flows are highly affected by connection latency, which should be kept as low as possible to guarantee QoE. Thus, congestion control is

of paramount importance and new algorithms are being developed, such as Network Assisted Dynamic Adaptation (NADA) [66], developed by Cisco, or Self-Clocked Rate Adaptation for Multimedia (SCReAM) [40], developed by Ericsson. Both algorithms are currently being standardized in the IETF Media Congestion Avoidance Techniques (RMCAT) working group.

4 Key Performance Indicators to estimate QoE in WebRTC

A central question when evaluating QoE is the issue of what factors have a significant effect on the quality of the experience of the end user. The already cited Qualinet White Paper [6] highlights three categories of possible Influence Factors (IF) for multimedia QoE: i) System IFs are factors that determine the technically produced quality of an application/service. ii) Context IFs cover a broad range of factors that identify any situational property to describe the user's environment in terms of physical, temporal, or economic characteristics. iii) Human IFs comprise properties related to the human user, such as socio-economic and demographic status, physical and mental constitution, affective state, etc.

These IFs were refined and analysed by Zhao et al. [65] for video quality: system IFs (content, media, network, and device related); context IFs (physical, temporal, economic, social, and technical), and human IFs (physical, emotional, demographic and socio-economic background). Based on that taxonomy, Husić et al. [25] investigated which are the most important IFs according to users' opinions in the context of WebRTC. To that end, they conducted a survey on a group of 140 users of a WebRTC-based video calling service. According to the participants' ratings, the most influential IFs for WebRTC are (from more to less effective): audio quality, video quality (image), and QoS. These IFs were considered to be composite factors, since they depend on several sub-factors.

Based on the classification of the different setups and infrastructure involved in a WebRTC session, presented in Section 3, and based on the relevant IFs for WebRTC proposed by Husić et al., we now propose a group of Key Performance Indicators (KPIs) to estimate the QoE of WebRTC applications. We focus on system factors specific to WebRTC, since the context and human IFs will not be different from those for any other real-time communication service.

The first KPI we have identified is the **call establishment time** (hereinafter called t_{setup}). Following the taxonomy of Husić et al., this KPI can be seen as a type of QoS IF. As previously explained, before a WebRTC peer can start a media session and the media can start flowing, a group of signaling activities has to take place: the SDP negotiation and the gathering of the ICE candidates. Thinking about the time required to accomplish these tasks, one of the most significant aspects is the use of standard ICE or Trickle ICE mechanism. The selection and implementation of one or another option is the

responsibility of the WebRTC application, but in the end, it affects the time that a user waits until the media session is actually established.

Once the session is established, WebRTC peers share their user media over the network. The way in which this conversation is perceived by end users can be defined as **audio and video quality** (hereinafter called Q_a and Q_v, respectively). These KPIs have a direct correspondence with the two factors (audio quality and image quality) that, as mentioned, have been identified by Husić et al. as the ones having the strongest effect on the user's satisfaction in the context of WebRTC based video call services [25]. Moreover, we introduce another KPI related to media quality: **audiovisual quality** (hereinafter called Q_{av}). This indicator can be seen as combining the the other two, as well as including aspects such as audio-visual synchrony. These KPIs can be computed using existing approaches, as was presented in Section 2.

Audio, video, and audiovisual quality are broad terms that encompass all the desired attributes from the user's perspective, such as sharpness, contrast, or high-definition of audio and video, while potential problems such as noise, clipping, ringing, or media freeze are reduced to the minimum. There are many different aspects that contribute to the final media quality of a WebRTC communication. First of all, the type of device used (e.g. a computer with a desktop browser, a mobile device, etc.) shapes the WebRTC media. For instance, the screen resolution or the hardware features (CPU, memory, camera, microphone, etc.) are key during the SDP negotiation to determine the selected codec (VP8, H.264, Opus, G.711, or other) and bitrate. Moreover, the underlying network is an important factor that affects the media quality perceived by end users. First, the access network (3G/4G, WiFi, cable, etc.) determines the available data throughput for the upstream and downstream channels. In addition, the status of the packet path within the core network determines important factors such as packet loss, delay, or jitter. In particular, in WebRTC, a congested network is detected by the RRTCC/GCC algorithms, and as a result, the bitrate is decreased, affecting the perceived media quality.

Since WebRTC is aimed at providing real-time communications, the **end-to-end delay** (hereinafter called d_{e2e}), also known as end-to-end latency, is a significant KPI to be considered. For communications to be real-time imposes a serious restriction on the latency, which must be low so as to allow for a conversational (bidirectional) communication. The first component of d_{e2e} is the network latency. This value aggregates the delays of transmission, propagation, queuing, and processing. After that, delays in the end devices caused by the encoding and decoding processes together with jitter buffers can also affect the value of d_{e2e}. Moreover, in WebRTC, the value of this delay can be increased in several ways. As explained earlier, the use of a TURN server that relays the media (in the case of WebRTC clients behind symmetric NATs) affects the value of d_{e2e}. Even though the use of TURN is the fallback during the session establishment with the ICE protocol, there is a significant percentage of WebRTC peers that are relayed by TURN servers. The documentation of Google's libjingle (an open-source library for building P2P applications) provides a reference point for the performance of STUN/TURN in the real

world. According to that information, 92% of connections can take place directly (STUN), while 8% of the connections requires a relay (TURN). Thus, the value of d_{e2e} experienced by a number of WebRTC peers is increased due to media relay (as shown in Figs 1 and 2). Moreover, in the case of using media server infrastructure to provide advance media capabilities for the WebRTC session (such as transcoding, mixing, archiving, etc.), the media should be routed to the media server. In this case, the value of d_{e2e} increases due to the fact that the media packets are routed to a media server and then to the rest of the peers. Lastly, the load and features of the media server can also affect the final value of d_{e2e}.

5 Systematic review of QoE applied to WebRTC applications

In Section 2 we overviewed methods to measure QoS and QoE. In this section we focus more specifically on methods used in the context of WebRTC applications and perform a systematic literature review (SLR). In addition to the aim of deriving a comprehensive snapshot of existing work, we also aspire to assess the above proposed KPIs by checking if and how the surveyed approaches could be categorized using these KPIs.

The SLR has been carried out following the original guidelines proposed by Kitchenham [43], which include the following steps:

1. Formulation of the research questions (RQs) driving the survey.
2. Definition of the search process (source selection and search keywords).
3. Definition of the exclusion/inclusion criteria and quality assessment.
4. Data collection and extraction.
5. Analysis of the results.

Step 1: Although strictly speaking the focus of this paper is on QoE, as we discussed above, there are several concerns common to measuring QoS and QoE over WebRTC applications, and in much of the literature, the distinction is blurred. Therefore, in searching the literature we considered both QoS and QoE, and we formulated the following RQs:

- RQ1: *Which (QoS/QoE) methods have been employed for the assessment of WebRTC applications?*
- RQ2: *Can these methods be categorized using the KPIs (t_{setup}, d_{e2e}, Q_a, Q_v, and Q_{av}) identified in Section 4?*

Step 2: We carrie out the search in the following repositories: Scopus[6], Microsoft Academic Research[7], ScienceDirect[8], IEEE Xplore Digital Library[9], and ACM Digital Library[10]. The search query was defined, based on the above RQs, as follows:

[6] https://www.scopus.com/
[7] http://academic.research.microsoft.com/
[8] http://www.sciencedirect.com/
[9] http://ieeexplore.ieee.org/
[10] http://dl.acm.org/

Table 2 Exclusion/inclusion criteria and quality assessment

Id	**Exclusion criteria**
E1	Summaries of workshops and tutorials, title pages, editorials, and extended abstracts, since they do not provide sufficient information for the survey
E2	Papers in their early stages or not mature
E3	Books, Masters and PhD theses, as in most similar studies, we only consider papers that have appeared in a peer-reviewed journal. Note that mature results from Masters/PhD theses are generally submitted for publication, and books are typically written extending previously published papers
E4	Double entries. If an extended journal paper is found, it will be chosen over the version in the conference proceedings
E5	Focus of the paper is not on QoE/QoS related with WebRTC
E6	Opinion papers and discussion papers that do not propose a solution
E7	Any paper whose full text is not accessible
E8	Papers not written in English
Id	**Inclusion criteria**
I1	Full versions of journal and conference papers that report on, discuss, or investigate QoS/QoE assessment mechanisms applied to WebRTC
I2	Papers written in English
I3	Papers published since 2014 (WebRTC maturation)
Id	**Quality assessment**
QA1	Is the paper based on research?
QA2	Is there a clear statement of the aim?
QA3	Is there a description of the context in which the research was carried out?
QA4	Are the methods described adequately?
QA5	Is there a clear statement of the findings?
QA6	Did the paper validate its results?

– $\{WebRTC\ and\ QoE\}$ *or* $\{WebRTC\ and\ QoS\}$.

Step 3: The exclusion/inclusion criteria and quality assessment procedure are summarized in Table 2. On the one hand, every inclusion criterion described in this table must be met in order to include a paper in the final selection. On the other hand, each exclusion criterion is sufficient to discard a paper. Finally, we use 6 quality assessment (QA) questions to measure the quality of each paper. Each QA question is answered as: Y (Yes), P (Partial), or N (No), with the following scoring: Y = 1, P = 0.5, N = 0. A paper is included in the final selection only if the sum of the QA scores over the 6 questions is greater than or equal to 4.

Step 4: Following the survey guidelines depicted in the steps above, we found 110 papers[11] (54 from Scopus, 18 from Microsoft Academic Research, 14 from ScienceDirect, 20 from IEEE Xplore, and 4 from ACM). After applying the exclusion/inclusion criteria and quality assessment questions described in Table 2, the number of papers was reduced to 15. Table 3 summarizes the selected primary papers ordered by year of publication.

[11] Our search was last updated in March 2018

Table 3 Selection of QoE WebRTC primary papers

Id	Title	Year	Ref.
S1	A congestion avoidance mechanism for WebRTC interactive video sessions in LTE networks	2014	[42]
S2	Optimization framework for uplink video transmission in HetNets	2014	[48]
S3	VoIP-based calibration of the DQX model	2015	[59]
S4	The impact of mobile device factors on QoE for multi-party video conferencing via WebRTC	2015	[60]
S5	On-demand, dynamic and at-the-edge VNF deployment model application to web real-time communications	2016	[5]
S6	A black box analysis of WebRTC mouth-to-ear delays	2016	[45]
S7	A performance evaluation of WebRTC over LTE	2016	[8]
S8	Video QoE killer and performance statistics in WebRTC-based video communication	2016	[2]
S9	Implementation and analysis of real-time streaming protocols	2017	[56]
S10	Integrating HEC with circuit breakers and multipath RTP to improve RTC media quality	2017	[23]
S11	A comparison of QoS parameters of WebRTC videoconference with conference bridge placed in private and public cloud	2017	[12]
S12	VR video conferencing over named data networks	2017	[64]
S13	WebRTC testing: Challenges and practical solutions	2017	[17]
S14	WebNSM: A novel scalable WebRTC signalling mechanism for many-to-many video conferencing	2017	[15]
S15	QoS analysis for WebRTC videoconference on bandwidth-limited network	2017	[3]

Step 5: In order to analyse the results, first we focus on the different kinds of assessment methods of the selected papers using the classification presented in Section 2. Then, since the number of finally selected papers is not large, we provide a short summary grouped by the different types of assessment methods (QoS, QoE, or both).

As depicted on the left chart of Fig. 3, the results of our survey show that the number of papers about QoE in WebRTC has been increasing from year to year. This suggests that in the research community there is increasing interest in QoE in the specific domain of WebRTC applications. Moreover, as shown in the right chart of Fig. 3, the research groups authoring these papers are quite spread across the world.

In order to answer the RQs, we analyse the QoE/QoS assessment methods that have been employed in the selected papers. Table 4 shows a summary of the results. This table has 4 columns: i) paper identifier (as described in Table 3); ii) QoS parameters (if any) evaluated in the paper; iii) QoE quantitative methods (if any) applied in the paper; iv) which of the KPIs presented in Section 4 is measured by the method applied in the paper. We can check that QoS leads the results, with 53% (8 papers), followed by a combination of QoS and QoE (27%, i.e. 4 papers), and finally papers focused purely on QoE (20%, i.e. 3 papers). Table 4 also contains the specific QoS parameters and

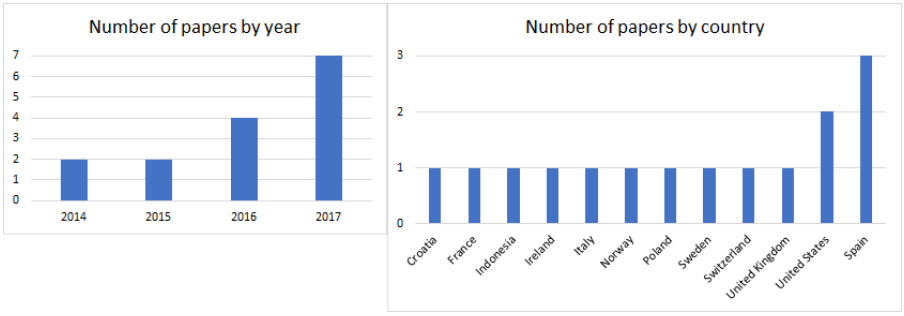

Fig. 3 Number of selected publications, by year and country

Table 4 Categorization of approaches in primary papers

Id	QoS	QoE	KPI
S1	Packet delay, end-to-end delay, bandwidth, throughput, packet loss, jitter, bitrate	✗	d_{e2e}
S2	✗	Objective (VQM)	Q_v
S3	Jitter, latency, packet loss, bandwidth	Subjective (MOS)	Q_v, Q_a, Q_{av}
S4	✗	Subjective (MOS)	Q_v, Q_a, Q_{av}
S5	Call setup time, end-to-end delay, jitter	✗	d_{setup}, d_{e2e}
S6	End-to-end delay, bitrate, jitter	✗	d_{e2e}
S7	Throughput, jitter, packet loss	✗	Q_v, Q_a
S8	Throughput, bandwidth, packet loss, bitrate, picture loss indication, bucket delay	Subjective	Q_v
S9	Call setup time, end-to-end delay	✗	d_{setup}, d_{e2e}
S10	✗	Objective (PESQ, PEVQ)	Q_v, Q_a
S11	Throughput, end-to-end delay, error rate	✗	d_{e2e}
S12	End-to-end delay	✗	d_{e2e}
S13	End-to-end delay	Objective (PESQ, SSIM, PSNR)	Q_v, Q_a, d_{e2e}
S14	Bandwidth	Subjective	Q_v, Q_a
S15	Bandwidth, jitter, bitrate, frame rate, packet rate, packet loss, latency, packet delay	✗	Q_v, d_{e2e}

QoE methods used to evaluate the final user perceived quality in the selected papers. Fig. 4 presents a graphical representation of these values, ordered by number. Regarding QoS, the results show that the preferred parameters to evaluate WebRTC applications are: end-to end delay, jitter, packet loss, bandwidth, throughput, and bitrate. Regarding QoE, subjective MOS is the preferred assessment method, followed by several objective methods (PESQ, PEVQ, VQM, PSNR, and SSIM). However, as a general remark, the number of papers is still quite small, and given that research interest is increasing, it would be worth updating this analysis in a few years.

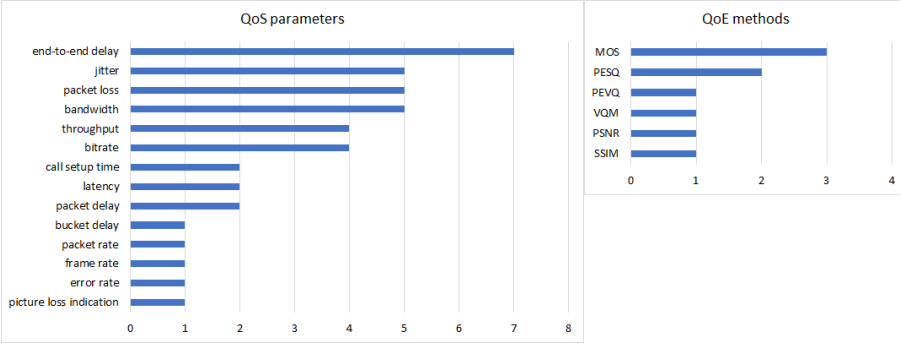

Fig. 4 Number and types of assessment methods (QoS, QoE) in the selected papers

5.1 QoS assessment methods

In the light of these results, we see that **end-to-end delay** is the preferred QoS assessment method in the selected papers, cf. (S1) (S5) (S6) (S9) (S11) (S12), and (S13). This fact can be explained by the real-time nature of WebRTC, which imposes a severs restriction on the total latency between the peers. According to the ITU-T, end-to-end delay in conversational media applications is rated as follows: 0–200 ms is good, 200–300 ms is acceptable, and over 300 ms is poor [30].

Despite its importance, it is significant that the precise interpretation and calculation of end-to-end delay is heterogeneous in the selected papers. In several papers, end-to-end delay is understood as a pure network parameter in a mobile environment. For instance, (S1) calculates the end-to-end delay as $N(D_{pr} + (L/R) + D_p)$, where N - 1 is the number of routers on the media path, D_{pr} is the time needed for the packet transmission, L is the standard packet size, R is the rate, and D_p is the propagation delay. Since that information is usually unknown, other approaches have been proposed to compute the end-to-end delay. Thus, (S9) (S11) and (S12) use network traffic data to calculate this figure as the difference between the time at which a packet is received and the time at which that packet was sent. Lastly, (S5) estimates the end-to-end delay using the value of Round-Trip Time (RTT).

Other papers interpret the end-to-end delay as the total time taken by the media to travel from one WebRTC peer to another from the final user perspective. In this group we find paper (S6), in which the delay is referred to as mouth-to-ear delay and is calculated accurately using an oscilloscope attached to both the input and output WebRTC media. In this category we also find paper (S13), in which the end-to-end delay is calculated using Optical Character Recognition (OCR) of an embedded timer sent through WebRTC.

Jitter is defined as the variation in the latency on a packet flow between two systems of different packets in the same media flow. It is caused by the fact that some packets might take longer to travel than others. Jitter is considered 5 times in the papers selected for our survey: (S1), (S3), (S5), (S6), and (S15),

since it can lead to unintended deviations in audio and video that affect the QoE of WebRTC applications. To avoid Jitter distortions, its recommended maximum value is 75 ms [13].

Packet loss is another important QoS indicator in the selected papers, (S1), (S3), (S7), (S8), and (S15). This fact makes sense since the users' QoE of WebRTC is very sensitive to packet loss. Acceptable values of the packet loss depend on the codecs used. However, in general, packet loss should be less than 3% for audio and less than 1% for video according to ITU-T [29].

Bandwidth is the raw capability of moving data through a communications channel, typically in terms of bits or bytes per second. Bandwidth has been considered in a significant number of papers in our survey (S1), (S3), (S8), (S14), and (S15). In these papers, we can see how the bandwidth of a WebRTC stream can quickly change for diverse causes, including muting and unmuting the voice, activating and deactivating the video, or switching video input (for instance activating desktop sharing). Closely related to bandwidth, we find another important QoS parameter employed in several papers in our survey (S1), (S7), (S8), and (S11), namely **throughput**, which refers to how many bits or bytes are actually sent through the channel. Bandwidth is the theoretical maximum channel capacity whereas throughput is the actual transfer rate used. Lastly, we find another similar parameter, the **bitrate**, in (S1), (S6), (S8), and (S15). Bitrate refers to the bits or bytes per second consumed from source to destination by a given entity, in our case, a WebRTC peer. Codecs play an important role in the bitrate of WebRTC communications. The bitrates in the usual WebRTC codecs, such as Opus and VP8, can vary significantly. According to measurements in the literature, the bitrate of the audio codec Opus for mono is 32 kbit/s and 64 kbit/s for stereo in Google Chrome. Regarding video, the starting video bitrate is, in VP8, 300 kbit/s, the minimum bitrate is 50 kbit/s, and the maximum is about 2 Mbit/s [41].

5.2 QoE assessment methods

As depicted in Table 4, the distribution of QoE assessment methods (i.e. objective and subjective) in the selected papers is quite balanced. In particular, 57% of the papers cover subjective methods, against 43% objective ones.

Regarding **subjective** assessment, we can see that the preferred method is collecting feedback from end users in terms of mean opinion scores (S3), (S4), and (S14). On the other hand, (S8) reports subjective data acquisition following a non-standard approach, evaluating the QoE by expert participants in WebRTC calls, reporting video-related quality issues such as frozen image, slow movement, black/blank screen, screen flash, or video freezes.

Regarding **objective** methods, interestingly, 100% of the objective methods in (S2), (S10), and (S13) follow a full-reference (FR) approach as described in Section 2, namely, VQM, PESQ, PEVQ, SSIM, or PSNR. Therefore, these approaches involve recording the WebRTC stream at the destination in order to compare it with the origin reference, using the above-mentioned algorithms.

6 Discussion

This paper contributes to the understanding of the QoE in WebRTC applications. As shown in the body of the paper, these kinds of applications have specific aspects that can affect the final QoE. In its simpler form, the topology of a WebRTC service follows a pure P2P architecture. Nevertheless, in many cases, some infrastructure is required to support the WebRTC session by means of signaling servers (in order to start, stop, and control the communication), STUN/TURN servers (in order to guarantee NAT traversal between peers), and media servers (in order to provide advance media capabilities, such as bridging, group communication, or archiving). Another significant aspect specific to WebRTC is the congestion control algorithm, implemented at the application layer in the WebRTC stack. These algorithms adapt the WebRTC transmission bitrate when detecting network congestion. All these factors might play an important role in the total amount of end-to-end delay in the WebRTC. As shown in the systematic literature review presented in Section 5, this is the most significant QoS parameter which needs to be minimized in order to guarantee the conversational real-time nature of WebRTC, and therefore the final QoE.

Hence, this article contributes to the modeling of QoE for WebRTC. To that aim, and based on the existing literature, we have proposed a set of KPIs for the modeling of the QoE in WebRTC applications, namely: call establishment time (t_{setup}), end-to-end delay (d_{e2e}), audio quality (Q_a), video quality (Q_v), and audiovisual quality (Q_{av}). As an exercise to validate the proposed KPIs, we consider the primary papers of the systematic survey, checking if and how the proposed approaches can be categorized using our KPIs. The results of this exercise showed that the selected papers use one or more of the KPIs, but none of these papers provides an integrated solution to gather all these KPIs at the same time. This fact suggests that there is room for improvement in the complete QoE assessment of WebRTC applications.

The main limitation of this paper is the lack of concretization of several of the proposed KPIs. On the one hand, d_{e2e} and t_{setup} are well-known QoS parameters with commonly accepted ranges and thresholds in conversational services: d_{e2e} should be less than 300 ms [30] and t_{setup} should be less that 10 s in voice calls [35]. But on the other hand, Q_a, Q_v, and Q_{av} are broad indicators. In the light of the results of the systematic survey, QoE subjective methods based on MOS scores have been preferred. Nevertheless, objective methods have proven to be successful. All in all, an open challenge in this domain is to identify a group of optimal QoE algorithms to assess the audio, video, and audiovisual quality in WebRTC applications.

7 Conclusions

Quality of Experience (QoE) and WebRTC are research topics attracting growing interest. This paper is aimed at facilitating the convergence of these topics

by providing management assets of QoE for WebRTC applications. QoE management involves three differentiated steps: i) understanding and modeling QoE; ii) estimating and monitoring QoE; iii) adapting and controlling QoE. The present research contributes to the first and second steps.

First, in order to understand QoE in WebRTC applications from a system point of view (i.e. excluding context and human factors), a complete analysis of possible WebRTC topologies has been carried out. Despite the fact that WebRTC has been conceived as a P2P technology to share media between browsers, in the real world the situation is more complex. To implement a production-ready WebRTC application, practitioners have to deal with different infrastructures (web and signaling servers, STUN/TURN, and media servers) and heterogeneous mechanisms (ICE, NAT traversal, or congestion control). As explained in the body of this paper, the specific topology of a WebRTC application can affect the QoE for end users.

Second, in order to estimate QoE in WebRTC applications, and based on the existing literature and the specific features of WebRTC, we have proposed 5 Key Performance Indicators (KPIs): call establishment time (t_{setup}), end-to-end delay (d_{e2e}), audio quality (Q_a), video quality (Q_v), and audiovisual quality (Q_{av}).

To conclude, a systematic literature review of the assessment of QoE in WebRTC applications has been conducted. The results of this review have shown that there is a preference for assessing the quality experienced by end users using QoS parameters rather than directly with QoE methods. Specifically, end-to-end delay, jitter, packet loss, bandwidth, throughput, and bitrate are the favorite QoS parameters in the relevant literature. Regarding QoE, on the one hand, the mean opinion score is the preferred subjective method, and on the other hand, all the objective methods found are in the category of full reference methods (PESQ, PEVQ, SSIM, PSNR, and VQM).

We believe the contributions of this article can guide further research for the management of QoE for WebRTC applications. Based on the proposed KPIs, the next steps in this direction could focus on the other steps of the above-mentioned QoE management process, i.e. monitoring, adapting, and controlling QoE in WebRTC applications.

Acknowledgments

This work has been supported by the European Commission under project ElasTest (H2020-ICT-10-2016, GA-731535); by the Regional Government of Madrid (CM) under project Cloud4BigData (S2013/ICE-2894) cofunded by FSE & FEDER; and the Spanish Government under project LERNIM (RTC-2016-4674-7) cofunded by the Ministry of Economy and Competitiveness, FEDER & AEI.

References

1. Alvestrand H, Holmer S (2012) A Google congestion control for real-time communication on the World Wide Web. Tech. rep., IETF
2. Ammar D, De Moor K, Xie M, Fiedler M, Heegaard P (2016) Video QoE killer and performance statistics in WebRTC-based video communication. In: Communications and Electronics (ICCE), 2016 IEEE Sixth International Conference on, IEEE, pp 429–436
3. Bandung Y, Subekti LB, Tanjung D, Chrysostomou C (2017) QoS analysis for WebRTC videoconference on bandwidth-limited network. In: 2017 20th International Symposium on Wireless Personal Multimedia Communications (WPMC), pp 547–553
4. Bevan N (1999) Quality in use: Meeting user needs for quality. Journal of Systems and Software 49(1):89–96
5. Boubendir A, Bertin E, Simoni N (2016) On-demand, dynamic and at-the-edge VNF deployment model application to Web Real-Time Communications. In: Network and Service Management (CNSM), 2016 12th International Conference on, IEEE, pp 318–323
6. Brunnström K, Beker SA, De Moor K, Dooms A, Egger S, Garcia MN, Hossfeld T, Jumisko-Pyykkö S, Keimel C, Larabi MC, et al (2013) Qualinet White Paper on definitions of Quality of Experience
7. Carlucci G, De Cicco L, Holmer S, Mascolo S (2016) Analysis and design of the Google congestion control for web real-time communication (WebRTC). In: Proceedings of the 7th International Conference on Multimedia Systems, ACM, pp 13:1–13:12
8. Carullo G, Tambasco M, Di Mauro M, Longo M (2016) A performance evaluation of WebRTC over LTE. In: Wireless On-demand Network Systems and Services (WONS), 2016 12th Annual Conference on, IEEE, pp 1–6
9. Chandler DM, Hemami SS (2007) VSNR: A wavelet-based visual signal-to-noise ratio for natural images. IEEE Transactions on Image Processing 16(9):2284–2298
10. Chen Y, Wu K, Zhang Q (2015) From QoS to QoE: A tutorial on video quality assessment. IEEE Communications Surveys & Tutorials 17(2):1126–1165
11. Chikkerur S, Sundaram V, Reisslein M, Karam LJ (2011) Objective video quality assessment methods: A classification, review, and performance comparison. IEEE Transactions on Broadcasting 57(2):165–182
12. Chodorek RR, Chodorek A, Rzym G, Wajda K (2017) A comparison of QoS parameters of WebRTC videoconference with conference bridge placed in private and public cloud. In: Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), 2017 IEEE 26th International Conference on, IEEE, pp 86–91
13. Chong HM, Matthews HS (2004) Comparative analysis of traditional telephone and voice-over-internet protocol (voip) systems. In: Electronics and the Environment, 2004. Conference Record. 2004 IEEE International Sym-

posium on, IEEE, pp 106–111
14. Cisco VNI (2017) Forecast and Methodology, 2016–2021. White Paper
15. Edan NM, Al-Sherbaz A, Turner S (2017) WebNSM: A novel scalable WebRTC signalling mechanism for many-to-many video conferencing. In: Collaboration and Internet Computing (CIC), 2017 IEEE 3rd International Conference on, IEEE, pp 27–33
16. Egger S, Schatz R, Scherer S (2010) It takes two to tango—Assessing the impact of delay on conversational interactivity on perceived speech quality. In: 11th Annual Conference of the International Speech Communication Association (ISCA), pp 1321–1324
17. García B, Gortázar F, López-Fernández L, Gallego M (2017) WebRTC testing: Challenges and practical solutions. IEEE Communications Standards Magazine 1(2):36–42
18. García B, López-Fernández L, Gallego M, Gortázar F (2017) Kurento: The Swiss army knife of WebRTC media servers. IEEE Communications Standards Magazine 1(2):44–51
19. Garvin DA (1984) What does "product quality" really mean? MIT Sloan Management Review 26(1):25–43
20. Grigorik I (2013) High Performance Browser Networking: What Every Web Developer Should Know About Networking and Web Performance. O'Reilly Media, Inc.
21. Handley M, Perkins C, Jacobson V (2006) RFC 4566. SDP: Session Description Protocol. Tech. rep., IETF
22. Hekstra AP, Beerends JG, Ledermann D, De Caluwe F, Kohler S, Koenen R, Rihs S, Ehrsam M, Schlauss D (2002) PVQM—A perceptual video quality measure. Signal Processing: Image Communication 17(10):781–798
23. Herrero R (2017) Integrating HEC with circuit breakers and multipath RTP to improve RTC media quality. Telecommunication Systems 64(1):211–221
24. Hoßfeld T, Schatz R, Varela M, Timmerer C (2012) Challenges of QoE management for cloud applications. IEEE Communications Magazine 50(4):28–36
25. Husić JB, Baraković S, Veispahić A (2017) What factors influence the Quality of Experience for WebRTC video calls? In: Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2017 40th International Convention on, IEEE, pp 428–433
26. Huynh-Thu Q, Ghanbari M (2008) Scope of validity of PSNR in image/video quality assessment. Electronics Letters 44(13):800–801
27. ISO (2005) ISO 9000: Quality Management Systems–Fundamentals and Vocabulary. International Organization for Standardization
28. ITU-R (2007) Recommendation BT.1788. Methodology for the subjective assessment of video quality in multimedia applications
29. ITU-T (2001) Recommendation G.1010. End-user multimedia QoS categories
30. ITU-T (2003) Recommendation G.114. Transmission systems and media digital systems and networks

31. ITU-T (2006) Recommendation P.10. Vocabulary for performance and quality of service
32. ITU-T (2008) Recommendation E.800. Definitions of terms related to quality of service
33. ITU-T (2008) Recommendation J.247. Objective perceptual multimedia video quality measurement in the presence of a full reference
34. ITU-T (2011) Recommendation P.863. Perceptual objective listening quality assessment
35. ITU-T (2014) Recommendation E.807 : Definitions, associated measurement methods and guidance targets of user-centric parameters for call handling in cellular mobile voice service
36. ITU-T (2016) Recommendation P.10/G.100. Vocabulary for performance and quality of service (Amendment 5)
37. ITU-T (2016) Recommendation. P.800.2. Mean opinion score interpretation and reporting
38. Ivov E, Rescorla E, Uberti J (2013) Trickle ICE: Incremental provisioning of candidates for the interactive connectivity establishment (ICE) protocol. Tech. rep., IETF
39. Jin J, Nahrstedt K (2004) QoS specification languages for distributed multimedia applications: A survey and taxonomy. IEEE Multimedia 11(3):74–87
40. Johansson I, Sarker Z (2017) Self-Clocked Rate Adaptation for Multimedia. Tech. rep., IETF
41. Khan M (2017) WebRTCPedia! the Encyclopedia! https://www.webrtc-experiment.com/webrtcpedia/, [Online; accessed 11 June 2018]
42. Kilinc C, Andersson K (2014) A congestion avoidance mechanism for WebRTC interactive video sessions in LTE networks. Wireless Personal Communications 77(4):2417–2443
43. Kitchenham B (2004) Procedures for performing systematic reviews. Keele, UK, Keele University 33(2004):1–26
44. Kitchenham B, Pfleeger SL (1996) Software quality: The elusive target [special issues section]. IEEE Software 13(1):12–21
45. Komperda O, Melvin H, Pota P (2016) A black box analysis of WebRTC mouth-to-ear delays. Communications pp 11–16
46. Loreto S, Romano SP (2014) Real-time communication with WebRTC: Peer-to-peer in the browser. O'Reilly Media, Inc.
47. Matthews P, Mahy R, Rosenberg J (2010) RFC 5766. Traversal using relays around NAT (TURN): Relay extensions to session traversal utilities for NAT (STUN). Tech. rep., IETF
48. Muñoz-Gea JP, Aparicio-Pardo R, Wehbe H, Simon G, Nuaymi L (2014) Optimization framework for uplink video transmission in HetNets. In: Proceedings of Workshop on Mobile Video Delivery, ACM
49. Pinson MH, Wolf S (2004) A new standardized method for objectively measuring video quality. IEEE Transactions on broadcasting 50(3):312–322

50. Rix AW, Hollier MP, Hekstra AP, Beerends JG (2002) Perceptual evaluation of speech quality (PESQ) the new ITU standard for end-to-end speech quality assessment Part I—Time-delay compensation. Journal of the Audio Engineering Society 50(10):755–764
51. Rosenberg J (2010) RFC 5245. Interactive connectivity establishment (ICE): A methodology for network address translator (NAT) traversal for offer/answer protocols. Tech. rep., IETF
52. Rosenberg J, Schulzrinne H (2002) RFC 3264. An offer/answer model with SDP. Tech. rep., IETF
53. Rosenberg J, Weinberger J, Huitema C, Mahy R (2003) RFC 3489. STUN, Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). Tech. rep., IETF
54. Rosenberg J, Mahy R, Matthews P, Wing D (2008) RFC 5389. Session traversal utilities for NAT (STUN). Tech. rep., IETF
55. Sale S, Rebbeck T (2014) Operators need to engage with WebRTC and the opportunities it presents. Analysis Mason Report
56. Santos-González I, Rivero-García A, Molina-Gil J, Caballero-Gil P (2017) Implementation and analysis of real-time streaming protocols. Sensors 17(4):1–17
57. Takahashi A, Hands D, Barriac V (2008) Standardization activities in the ITU for a QoE assessment of IPTV. IEEE Communications Magazine 46(2):78–84
58. Timmerer C, Ebrahimi T, Pereira F (2015) Toward a new assessment of quality. Computer 48(3):108–110
59. Tsiaras C, Rösch M, Stiller B (2015) VoIP-based Calibration of the DQX Model. In: IFIP Networking Conference (IFIP Networking), 2015, IEEE, pp 1–9
60. Vucic D, Skorin-Kapov L (2015) The impact of mobile device factors on QoE for multi-party video conferencing via WebRTC. In: Telecommunications (ConTEL), 2015 13th International Conference on, IEEE, pp 1–8
61. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4):600–612
62. Westerlund M, Wenger S (2015) RFC 5117. RTP topologies. Tech. rep., IETF
63. Xiao F (2000) DCT-based video quality evaluation. Final Project for EE392J
64. Zhang L, Amin SO, Westphal C (2017) VR video conferencing over named data networks. In: Proceedings of the Workshop on Virtual Reality and Augmented Reality Network, ACM, pp 7–12
65. Zhao T, Liu Q, Chen CW (2017) QoE in video transmission: A user experience-driven strategy. IEEE Communications Surveys & Tutorials 19(1):285–302
66. Zhu X, Pan R, Ramalho MA, Mena S, Ganzhorn C, Jones PE (2015) NADA: A unified congestion control scheme for real-time media. Tech. rep., IETF