This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

Measurement of Quality of Experience of Video-on-Demand Services: A Survey

Parikshit Juluri[1], Venkatesh Tamarapalli[2], Deep Medhi[1]

***Abstract*—Video on demand streaming (VoD) services have gained popularity over the past few years. An increase in the speed of the access networks has also led to a larger number of users watching videos online. Online video streaming traffic is estimated to further increase from the current value of 57% to 69% by 2017 [1]. In order to retain the existing users and attract new users, service providers attempt to satisfy the user's expectations and provide a satisfactory viewing experience. The first step towards providing a satisfactory service is to be able to quantify the users' perceptions of the current service level. Quality of Experience (QoE) is a quality metric that provides a holistic measure of the users perception of the quality. In this survey, we first present a tutorial overview of the popular video streaming techniques deployed for stored videos, followed by identifying various metrics that could be used to quantify the QoE for video streaming services; finally, we present a comprehensive survey of the literature on various tools and measurement methodologies that have been proposed to measure or predict the QoE of online video streaming services.**

***Index Terms*—Quality of Experience (QoE), video streaming, Video On Demand (VoD), measurement, online video**

I. Introduction

Watching videos online has become a popular form of entertainment in recent years. Online VoD services enable users to watch video content such as user generated videos, movies, TV shows, music videos, and live streams. In the past few years, it has been observed that video streaming has accounted for most of the traffic carried over the Internet and it continues to steadily increase every year. The global video streaming traffic is expected to account for 69% of the consumer traffic on the Internet by 2017, up from 57% in 2012 [1]. YouTube, Netflix, and Hulu are the most popular online video streaming service providers. According to [2], in 2013 Netflix accounted for 32.25% of the total downstream traffic during peak periods in North America, followed by YouTube with 17.1%, and Hulu at 2.41%.

The increasing dominance of video traffic over other Internet traffic could soon make it a prime online service for the average user. In order to meet the satisfaction of the user with these services, it is necessary to be able to measure the performance of these services. Traditionally, the Quality of Service (QoS) metrics have been used to study the performance of online services and networked elements. QoS metrics, which are more suitable to measure the performance and reliability of the network elements, however, do not capture the actual experience of the user. Quality of Service(QoS) as defined by ITU [3] reflects the performance of the network and its components. It measures the network's ability to satisfy the needs of the service and is thus, a network-centric metric. The common QoS metrics used are throughput, bandwidth, packet loss, delay, or jitter. The service received by the user, however, is affected and influenced not just by the network components but also by several other factors. These are end-to-end factors that include the effects of the service infrastructure, terminal, client, and network. On the other hand, Quality of Experience (QoE) [4], [5] is a user-centric metric that captures the overall acceptability of the service and includes the end-to-end factors. It has also been defined as the degree of delight or annoyance of the user of an application or service [6]. QoE measures the performance as subjectively perceived by the user. However, QoE and QoS are not mutually exclusive, rather, QoE is an extension to QoS. The user-perceived quality of video streaming services is influenced by four different categories of influence factors [7].

- **System Level** factors encompass the effect of the factors that work at the technical level. These factors include the factors related to the network (packet-loss, delay), end-devices(system hardware, screen size), and also the application layer(video buffering strategies, browser).
- **Context Level** considers the environmental factors associated with the user such as the user's location, purpose of using the service such as entertainment, education, etc.
- **User Level** considers the psychological factors such as the expectations of the user, browsing history, and hour of the day.
- **Content Level** considers the defining characteristics of the video file such as the encoding rate, format, resolution, playback duration, quality of the video, age of the video etc.

Due to the varied influence factors that affect the QoE, it presents a unique challenge to measure it. In this survey, we present the different tools and methodologies that have been developed to measure the QoE of online video-on-demand (VoD) streaming services for stored videos. Most video services are not provided with any special provisioning for end users and hence, need to compete with other services to be able to maintain a satisfactory service. In contrast, broadcast television and cable television have dedicated infrastructures and do not typically share network resources with other services. The Internet Protocol Television (IPTV) provides television services over a managed IP network and ensures a superior entertainment experience [8], [9]. IPTV services utilize networks that guarantee a Quality of Service, which differentiates them from other online streaming services such

[1]University of Missouri–Kansas City, USA; [2]Indian Institute of Technology–Guwahati, India.

as YouTube, which are provided with no such guarantees; such managed video delivery over IPTV is outside the scope of this survey. Secondly, we emphasize that this survey focuses on stored videos as opposed to live streams. It may be noted that, for stored videos, the size of a video file is known a priori. VoD for stored videos supports VCR-like functionality with the ability to play, pause, rewind, and forward the playback. We limit our survey to users that are connected to wired access networks such as on residential ISPs and accessing VoD services.

In order to set the stage for this survey, we first present a tutorial overview of the popular video delivery techniques and existing services that use them in Section II. We then identify the most important QoE metrics used in each of these techniques in Section III. Section IV presents a comprehensive categorization of the different measurement studies. We conclude with Section V in which we identify a few future research directions to improve and extend the existing QoE measurement techniques.

II. Video Delivery Techniques

Compared to an HTML page, the size of a typical video file is much larger. The average web page size is found to be 1.9MB in October, 2014 [10]. In 2007, based on their measurements, the authors in [11] found that the average YouTube video size was 10MB. Since then, YouTube has increased the maximum video file size to 2GB from 100MB and also started hosting HD videos. The HD videos hosted by YouTube have an average size of 32MB per minute [12]. Using these numbers as reference, a user would need to wait for significantly longer durations if the users' video players were to download the entire file before starting the playback. This clearly would affect the users' QoE. Instead, online video streaming services are set up in such a way that users can start viewing the content once the initial part of the video is downloaded. In order to understand how this is achieved, we start with an overview of the popular video streaming techniques: real-time video streaming, Real-time messaging protocol streaming, progressive download streaming, and HTTP based adaptive bitrate streaming.

A. Real-Time Video Streaming Techniques

Real-time video streaming techniques use Real-Time Streaming Protocol (RTSP) [13] for video playback. RTSP supports VCR-like functionality for play, pause, and skip forward or backward operations. RTSP is a stateful protocol. The RTSP server keeps track of the session as long as the client is connected and it also maintains a continuous feedback channel with the client. This feedback is used to modify the media delivery rate based on parameters such as available bandwidth and end-to-end delay. The server tries to adjust the packet delivery rate to avoid any interruptions in the playback. A single RTSP session can be used to establish and control one or more media streams. The role of RTSP is limited to to establishing and maintaining the session while the actual transfer of the data is managed by the Real-time Transport Protocol (RTP) [14]. A complete RTSP session (Fig. 1) consists of a client establishing a streaming session with the server by sending an `RTSP-SETUP` message. After the setup is completed, the client sends an `RTSP-PLAY` message to begin the playback. Following this, the server starts sending the multimedia streams using RTP packets. RTP uses a supporting protocol, Real-time Control Protocol (RTCP) [14], to provide an out of band control of the flow. The RTCP messages are exchanged either over TCP or UDP and the actual data packets are transferred using RTP running over TCP, or UDP. If the client is behind a firewall, then RTSP and RTP packets can also be sent over an HTTP tunnel [15]. After completion of the video playback, the session is terminated by an `RTSP-TEARDOWN` message and the server state is cleared.

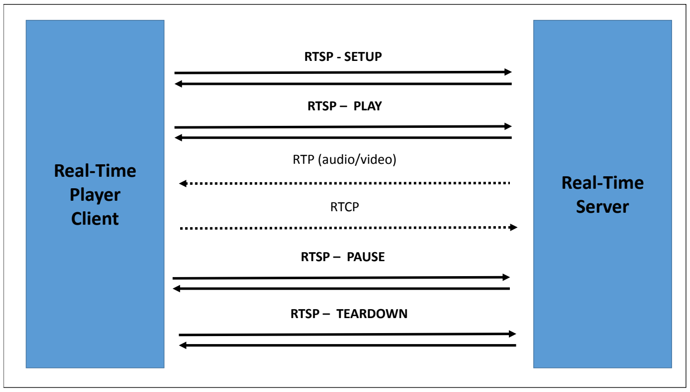

Fig. 1: RTSP Streaming Session

RealNetworks' *RealPlayer*, Apple's *QuickTime* player and Microsoft's *Windows Media Player* are three of the most popular RTSP based video streaming desktop applications used to play real-time video streams. The RTSP video server, which delivers the video streams to the video player can serve both VoD and live content. RTSP is used to support the session control features and RTP/RTCP are used as the transport suite. The players are configured to automatically select the best data-delivery protocol for RTSP/RTP among TCP, UDP, and HTTP to support the highest quality (least interruptions, and best possible video quality) of playback [16]–[18]. In case the client is behind a firewall, the RTSP and RTP/RTCP packets are transferred using HTTP tunneling.

B. Real Time Messaging Protocol Streaming Techniques

Real Time Messaging Protocol (RTMP) is a proprietary multimedia streaming protocol developed by Macromedia. After Adobe acquired Macromedia, it released an incomplete specification of the RTMP protocol in 2009 [19]. The specification released by Adobe provides limited information about the implementation of the protocol and also does not include explanations for certain control messages (e.g., control message ids 31 and 32). RTMP is a stateful protocol that delivers video and audio data between a *Flash Server* and a *Flash Player*. Based on the limited documentation, we can learn that the RTMP runs over TCP and supports parallel streams to transfer video, audio, data, user commands, and the control messages.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

The data delivered to RTMP by the client or the server are treated as *messages*. The RTMP message types include *audio*, *video*, *command*, *data*, and *user control messages*. Fig. 2 depicts a simple RTMP session. After establishing a TCP connection, the RTMP session is initialized by `RTMP-HANDSHAKE` messages. The RTMP client and the server subsequently exchange messages using two different types of commands, *NetConnection* and *NetStream*. The *NetConnection* commands are used to create and manage a full duplex connection between the client and the server. After the connection establishment phase, the *NetStream* commands are used to transfer the audio, video, and control messages between the server and the client. For example, the client initializes the playback by sending a *NetStream-Play* message. The server responds with a control message followed by sending parallel streams of audio and video messages. After the media playback is completed, the RTMP client closes the session by sending a *NetConnection-Close* message.

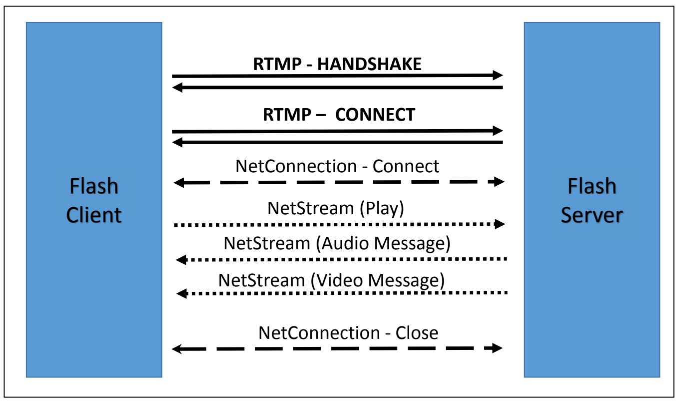

Fig. 2: RTMP Streaming Session

Hulu, a popular video streaming service, uses RTMP over TCP on port 1935. If the TCP port is blocked, it uses RTMP tunneled over HTTP (RTMPT) to transfer the multimedia content [20]. The multimedia content is downloaded by setting up an RTMP connection between an embedded Flash player and a Macromedia Flash Communication Server (Macromedia-FCS). When accessed using mobile devices, Hulu uses adaptive streaming over HTTP technique (discussed in Section II-D).

C. Progressive Download Streaming Techniques

The progressive download streaming techniques refers to the continued download of the video file, while the video player plays out the video content received so far. In *Progressive Download Streaming* techniques, the multimedia file is downloaded like a regular file using HTTP over TCP from a web server hosting the content. Unlike the two techniques discussed earlier, there is no direct feedback loop between the client and the server. The stateless approach of delivery reduces the load on the server to maintain persistent feedback loops with the client and enables the system to scale better. HTTP natively and easily supports mirroring and edge caching, thus enabling large-scale expansion if needed [21]. On the client-side, the entire video file is being downloaded as quickly as possible and stored on the local disk. However, in order to reduce the amount of data delivered to the user and to reduce the network resources utilized, the server-side could implement certain flow control mechanisms. The video player (typically an embedded Flash player) starts playing the video (for the data received so far) while it is still being downloaded.

Using HTTP over TCP for video transfers ensures reliable data delivery, but in the case of low bandwidth availability or excessive packet losses during the transmission, the playback may be interrupted. By using HTTP, the complexity on the server-side is reduced and the clients are not affected by a firewall or a proxy. However, since a progressive download tends to download the video as fast as possible, it leads to a waste in bandwidth if the user quits before completely viewing the video.

YouTube is the most popular service that uses progressive download. An embedded *Flash Player* is used to play the video directly from any suitable web browser [22]. The player downloads the video as a regular file and stores it in a temporary folder. The downloaded video file is usually an interleaved stream of audio and video blocks with tags to indicate the time at which they are supposed to be played. These tags help the Flash Player to skip backward or forward within the video file already downloaded. Since the entire video is stored locally, the backward skip is simply a local operation. In case of a forward skip to a position in the video that has not yet been downloaded, the Flash Player sends a new HTTP request with the byte-range indicating the new position to the server. Although the progressive download technique has been traditionally used for VoD, YouTube has been using it for live content as well.

Since the server does not maintain any state, the player handles the video playback in two stages to ensure smooth playback. During the initial stage, the player waits for a certain period of time known as the *Initial Buffering* period, during which the content that can play the first 30 to 40 seconds of playback is downloaded as fast as possible from the server [23]. Once the *initial buffer* is filled, the video playback starts. By delaying the start of the playback, the player avoids the effect of delay jitter on the playback. The content is continuously downloaded and buffered while it is played by the Flash player. If the difference between the amount of content played and the content downloaded falls below a certain threshold, the video playback is paused resulting in an interruption. The video playback is interrupted until the threshold is crossed again after which the playback resumes.

At the server-side, YouTube employs flow control mechanisms, to improve the initial buffering time and also reduce the amount of unnecessary data transferred in case the user decides to quit before watching the entire video [24]. The YouTube server sends the video as fast as possible for an initial buffering period before settling down to a constant sending rate. The sending rate during this steady phase is maintained at or slightly above the playback speed to ensure smooth playback.

YouTube supports the playback of multiple formats and resolutions of the same video [25]. Each of these are stored

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

as separate files with a different URL. Before initializing the video streaming, the default format and the resolution are determined by the player and based on the hardware platform and screen resolution settings (such as full-screen and wide screen). The selected format is typically fixed for the entire duration of the playback. However, the player supports the ability to automatically change the format/resolution of the playback depending on the network conditions. This shift is initiated by the player at the user-end and results in the new video format file being downloaded.

At the time of writing this paper, YouTube has started supporting most of its videos using MPEG-DASH (discussed in Sec. II-D) in addition to progressive download [26].

D. HTTP Based Adaptive Bitrate Streaming (HABS) Technique

Using HTTP for multimedia streaming has several advantages. First, HTTP has the ability to traverse firewalls/NATs since outgoing connections are always supported. Second, HTTP-based services do not have to make any major modifications to the existing web-servers (unlike RTSP-based services which require the server to maintain session state). Moreover, the existing Internet architecture consisting of Content Distribution Networks (CDNs), proxies, and caches already support HTTP traffic. Although progressive download has the advantages of HTTP-based streaming, it does not have the built-in capability to adapt the streaming rate. The HTTP-based Adaptive Bitrate Streaming (HABS) technique was proposed to support adaptive streaming over HTTP [27]. A HABS server does not maintain any state information during the session. The rate adaptation is handled at the client-end. By offloading the decision making process onto the client-end, the system is able to scale well while still providing dynamic adaptive streaming. It was observed in [28] that the above advantages of HABS resulted in a better QoE for the users when compared to the progressive download techniques.

HABS dynamically adapts the video streaming quality to ensure seamless playback. HABS considers a media file to be a collection of several components: audio, video, and subtitles that are stored separately on the HABS server. These components are delivered to the user independently but combined at the time of the playback [27]. Each of these components is further divided into smaller chunks called *segments* while each *segment* could be encoded into multiple versions. Every *segment* is identified by a unique URL. The different versions of the same media file are called *representations*. Different representations vary in bit rate, resolution, format, language, and other characteristics.

When the user requests a video, the server sends a Media Presentation Description (MPD) file that is a manifest with the list of all the available representations, minimum and maximum bandwidth to be supported, the accessibility features, the digital rights management (DRM) information, the location of each *segment* on the network (URL) and other attributes of the media file. In order to play the video, not all representations need to be supported by the client device.

At the beginning of a session, the video player sends an HTTP request for the first *segment* of the video using its URL. Typically, the *segment* with the lowest resolution is requested first. While the first *segment* is being downloaded, the player monitors factors such as the current observed bandwidth, throughput, delay, client-buffer status, or the bit-rate of each segment. [29] reviews several tools that could be used by the HABS streaming services to estimate the bandwidth. Based on these factors, end-device capabilities, and the user's preferences, the HABS player employs adaptation algorithms([30], [31]) to determine the most suitable representation for the next segment. The player thus decides to either stay with the same representation or shift to a higher or lower representation. The representation is selected to play the best possible quality of the video with the least number of interruptions. Once the decision is made, a new HTTP request is sent for the next *segment*.

The support for HABS started with three different companies developing their own techniques: Adobe's HTTP Dynamic Streaming (HDS) [32], Microsoft's HTTP Smooth Streaming (HSS) [33], and Apple's HTTP Live Streaming (HLS) [34]. Although the basic design and working principles of all the three version are comparable, each one of them has an independent implementation of HABS and also use their respective players. HDS supports an Adobe Flash player, HSS supports a Microsoft Silverlight player, and HLS works with a Quicktime player. HDS and HSS use an MP4 segment format with 2-4 seconds of typical segment duration while HLS uses an MPEG-2 TS segment format with each 10 second segment. In November 2011, MPEG released Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [27], which is the first open standard for adaptive video streaming. MPEG-DASH has been designed to work with an MPEG segment playback that supports both MP4 and MPEG-2 TS segment formats, and the segment duration is flexible.

Netflix is the leading subscription-based service provider for VoD content surpassing HBO in 2013 with a total number of subscribers in the U.S.A alone to be 31.1 million [35]. Currently, Netflix uses the *Microsoft Silverlight* [36] platform with an MPEG-DASH protocol. The first time the user requests a video, the browser downloads the Silverlight player. The server initially sends an MPD file that includes the list of representations available, the URL for each segment, the list of CDNs, and instructions for the playback. The server divides the content into small audio and video segments encoded into multiple representations. The video segments are encoded into bit rates between 100 Kbps and 1750 Kbps (2350 Kbps and 3600 Kbps for HD videos) [37]. The Silverlight player initially selects the lowest representation, and depending on the network conditions, determines the representation of the next segment using a *rate determination algorithm*. The quality of the video is adapted seamlessly at the segment boundaries based on the network conditions.

III. QoE Metrics

The users of video streaming services usually have a different set of expectations as compared to other types of web services. Typically, the VoD streaming sessions last from several minutes to hours. If there is a noticeable delay after a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

video link is clicked, or if there is an interruption during the playback or any perceivable drop in the visual quality of the video, these can affect the perceived quality of the user. The traditional network-centric QoS metrics such as packet loss, packet reordering, and delay are not sufficient to measure the satisfaction of the users. Although the users can perceive the effects of these metrics on the video streaming quality, they cannot use these metrics to quantify their QoE. Hence, it is necessary to have metrics that can be directly perceived by the users who also are able to use them to quantify their QoE.

In traditional video broadcast services QoE metrics are measured by comparing the reference with the outcome. Reference reflects the undistorted content in its original form at the server end. The content received at the other end could be potentially distorted or delayed. QoE metrics have been classified into the following categories based on reference [38].

- **Full reference(FR) metrics**: Both the reference and outcome are available and, thus, enable detailed subjective and objective comparisons of the videos. These metrics are suitable for traditional broadcasting and television systems. Eg. Peak Signal to Noise Ratio (PSNR) [39], Structural Similarity (SSIM) [40], and Video Quality Metric (VQM) [39].
- **No reference(NR) metrics**: Only the outcome is available and the quality is to be estimated with no reference. These types of metrics are more applicable to online services where only the outcome at the user end is available. In video streaming services, it is hard to determine if the discrepancy in the quality is due to the quality of the reference or due to the intermediate elements.
- **Reduced reference(RR) metrics**: The same set of parameters are derived for both the reference and the outcome. These parameters could be at the application layer: bit-rate, frame-rate or at the network layer: packet-loss.

For readers who are interested in the metrics and methodologies relevant to the tractional video delivery, we refer them to the survey in [41]. The current survey paper aims at studying the measurement methods that are used to measure the QoE for online video streaming. Depending on the type of measurement mechanisms employed, the QoE metrics in this section are classified into objective and subjective metrics [5]. Later, in Section IV we present the different studies that have used these metrics to quantify the QOE.

A. Objective Metrics

Objective QoE metrics are metrics that are the QoE metrics that can be quantified with a measurement tool. The following are the common objective QoE metrics that capture the factors that influence the user's QoE.

1) Playback Start Time: The *playback start time* or the *initial delay* is the time duration before a video starts to playout. The *playback start time* typically includes the time taken to download the HTML page (or manifest file), load the video player plugin, and to playback the initial part of the video. In the case of streaming videos, the player starts playing the video only after a part of the file is downloaded (called *Initial Buffering*). By doing so, the player overcomes the effect of the delay jitter incurred during the data transfer on the video playback. The *playback start time* is an important factor that affects the QoE, and hence, certain VoD services (such as YouTube) usually tend to push data at higher rates initially and settle down for a lower rate later [42]. It was observed in [43] that playback start time had a significant influence on user retainment. If the playback start time extends by more than 2 seconds it could result in the viewer abandoning the video completely [43]. Playback start time can be used to quantify the QoE in any type of video streaming.

2) Number of Interruptions: An interruption occurs when the playback of the video is temporarily stalled. A streaming player downloads the initial parts of the multimedia content into a playout buffer before the video has started playing. As long as the rate at which the buffer is being filled is greater than or equal to the rate at which the video is played, the playback is not interrupted. If the download rate falls below the playback rate, the buffer gets depleted and the player waits for the buffer to be partially filled before resuming the playback; this wait time is the interruption of the playback. The interruption is thus a direct consequence of *buffer starvation* at the player. The interruptions are also referred to as *(re)buffering events* and the frequency with which the buffering events occur is called the *buffering frequency*. The re-buffering could also be a consequence of user interaction. When a user skips to a different part of the video or changes the quality of the video during the playback, the player needs to fetch the requested content from the server and then continue the playback from the desired position. In this case, the player again waits for a certain buffering period before resuming the playback.

In case of adaptive streaming, if the player observes any drop in the receiving rate, it may automatically switch to a lower bit rate so that the video playback is not interrupted. However, if the network condition becomes noticeably bad that even the low bit rate content cannot be downloaded in time, then the user would see interruptions. The interruption lasts until the playout buffer is partially filled with the content of the desired bit rate. These interruptions or stall events during the playback lead to a poor user experience [44]. It is found that the number of interruptions have a significant impact on the QoE. Users who experienced more interruptions in the video tend to watch the video for shorter durations [43] and are likely to be dissatisfied in the case of four or more interruptions for videos [45]. Hence, it is an important metric to measure the satisfaction of the users. [38].

3) Duration of Interruptions: Apart from the number of times the playback is interrupted, the duration of the interruption ("buffering duration") is also an important QoE metric. If the interruption duration is one second, the users are less dissatisfied when compared to 3 seconds of interruption while watching YouTube videos [44], [45]. In [46], it was concluded that the viewers prefer a single but long stall event instead of several short stall events. Hence, not just the number of interruptions but the duration of each interruption has a distinct effect on the QoE of the user.

4) Quality of the Video File: The quality of a video stream is based on the encoding rate. The Encoding rate is the average

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

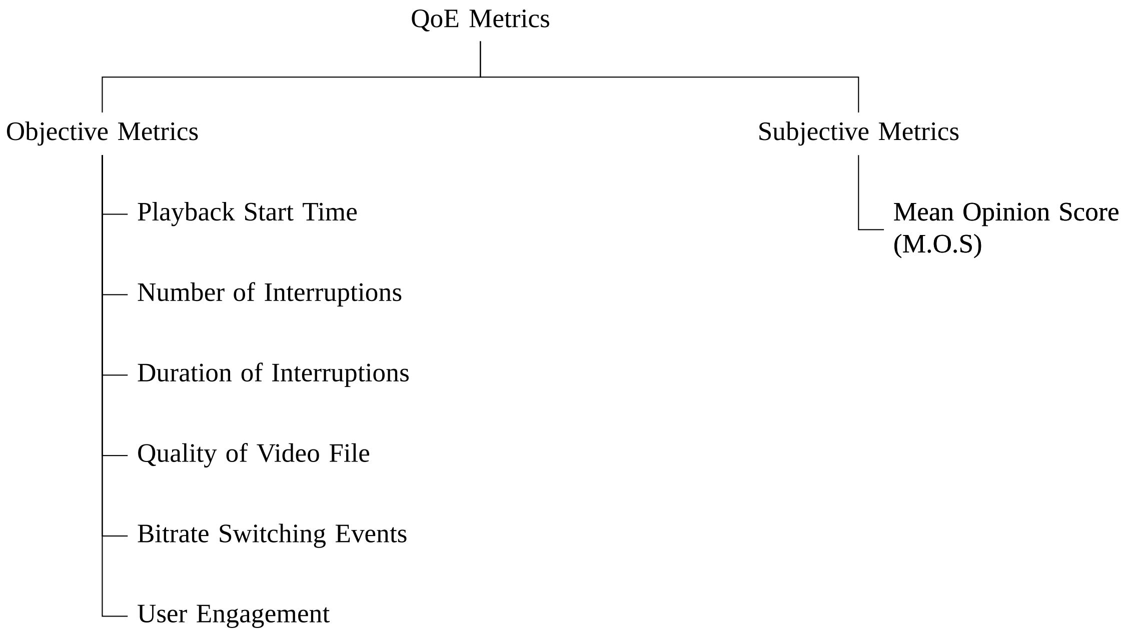

Fig. 3: Classification of QoE Metrics for Online Videos

data required to play one second of the video. The encoding rate of the video does affect the QoE of the users [47]. A higher quality video (HD) would usually require a larger amount of data for each frame and hence, results in a higher encoding rate. Progressive download streams typically stick to the same encoding rate throughout the duration of the playback, irrespective of the change in the network quality. Adaptive streaming techniques, like real-time streaming and HABS, vary the encoding rate depending on the network parameters.

There are other video characteristics that have been used to represent the quality of the video such as the contrast, blurring [48], and blockiness [49], [50], [51], [52]. Blockiness manifests as a block appearing in the video. It is caused by the block-based coding schemes such as H.261, H.263, MPEG-1.

5) Bitrate Switching Events: The Bitrate switching events are related to the HTTP based adaptive streaming technique(HABS)(Sec. II-D). For DASH videos, the player tends to pick a lower initial startup and gradually keeps increasing the quality before settling at a suitable bitrate. The bitrate could later be reduced when the rate of the playback exceeds the buffering rate due to degraded network conditions. By lowering the quality of the video streaming, the player minimizes the interruptions during the playback. On the other hand, when the network conditions improve, the bitrate is increased. However, frequent switching in bit rates can degrade the users QoE [28]. Hence, it is necessary to ensure that the number of bitrate switching events are reduced. The startup bitrate, number of bitrate switching events, and the average bitrate affect the QoE [28], [47].

6) User Engagement: User Engagement reflects the user involvement and interaction with the video. User engagement is measured in terms of the number of views and the play time of the video. However, the play time might not reflect the amount of time the user actually spends watching the video without getting distracted. It is hard to quantify the user's focus which is a subjective metric. The users who are satisfied with the content and the QoE of the streaming session tend to spend more time watching the video [53].

B. Subjective Metrics

Subjective metrics are the QoE metrics based on collecting the data from the users directly based on their experience with the service. A limited set of human subjects are exposed to the video service in a controlled environment and are asked to rate them. Due to the use of actual human subjects, the ratings could be affected by several physical and psychological confounding factors and hence, subject to user-bias. The confounding factors can be broadly classified as (1) user-dependent: user interest, purpose (educational, entertainment) (2) content-based: genre, age of the content and popularity; or (3) device-dependent: quality of the Internet connection, screen size, and the capabilities of the device. Unlike the objective metrics, users are asked to rate video services on a standard measuring scale. Such subjective metrics are susceptible to bias and hence, can vary from one subject to another. The best way to measure the effects of these factors is to collect direct feedback from multiple users using robust sampling methodologies and to use statistical analysis techniques to avoid any bias.

1) Mean Opinion Score(MOS): The *Mean Opinion Score* (MOS) is the most popular subjective metric measurement scale that is often used to quantify these factors. Users watch videos and rate them on a five-point discrete scale: 1-bad, 2-poor, 3-fair, 4-good, and 5-excellent.

The use of the MOS as a subjective metric has become the de facto standard for subjective assessment. It is, however, not easy to automate the MOS measurement since the influence of

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

the human psychological factors and the user bias needs to be considered. In order to predict the MOS, a good understanding about the psychology of the users to predict the MOS ratings is necessary.

C. Other Metrics in Practice

Traditionally, the quality of video playback for television and broadcasting media services and networked media was measured by using metrics such as Peak Signal to Noise Ratio (PSNR) [39], Structural Similarity (SSIM) [40], and Video Quality Metric (VQM) [39]. The PSNR is computed by calculating the Mean Square Error (MSE) in each pixel, represented in dB by comparing both the original and received frames. SSIM extended PSNR to include the effects of luminance, contrast, and structural similarity of the frames. The VQM was an improvement over the PSNR and SSIM as it also considered blurring, global noise, and color distortions. However, these three metrics can be measured only if a copy of the original video is available for comparison with the distorted video. This requirement has made these metrics unsuitable for online video streaming.

For VoD streaming, application layer metrics such as the application layer packet loss and reordering are also used to measure the QoE. These metrics are applicable to services that use UDP as the underlying transport layer protocol. Decodable Frame Rate (Q) [54] is another application layer metric that captures the packet loss rate at the application layer. Q is defined as the fraction of the number of successfully decoded frames in the total number of frames sent by an MPEG source, expressed as:

$$Q = \frac{N_{dec}}{N_{total-I} + N_{total-P} + N_{total-B}}, \tag{1}$$

where N_{dec} is the total number of successfully decoded frames, and $N_{total-I}$, $N_{total-P}$, and $N_{total-B}$ are the total numbers of inter (I), forward predicted (P), bidirectionally predicted (B) frames transmitted, respectively.

Apart from the common QoE metrics specified in Section III-A, HABS services also use the play list, MPD information, the result of the different HTTP transactions (MPD requests, segment requests) to measure the performance of the service [55]. The survey, [56] discusses metrics and measurement metrologies that are related to HTTP based adaptive streaming.

D. QoE Metrics and Video Streaming Techniques

The objective and subjective metrics discussed above are used to measure the QoE in the context of different services. The extent to which the metrics are relevant depends on the type of video streaming service considered. The playback start time, the number and the duration of interruptions, and application layer metrics can be used for the different video streaming techniques discussed in Section II. The quality of the video stream remains constant in progressive download-based service (irrespective of the playback conditions) whereas for and adaptive streaming over HTTP, the quality changes during the playback based on the network conditions. These changes are reflected by the bitrate changing events. Hence, the quality of the video and bitrate switching events are more suitable to measure the QoE of adaptive streaming-based services. The MOS is obtained by a direct user feedback and the measurement methodology is independent of the streaming technology used. However, it is effective in quantifying the effect of confounding factors on the QoE of various services.

IV. QoE Measurement Methodologies

The importance of QoE in representing user satisfaction has long been established. However, the metrics to be used and the techniques to measure the QoE for online video streaming services have not been standardized. Recently, several measurement methodologies have been proposed to measure the QoE for VoD services.

The International Telecommunication Union (ITU) has classified the QoE measurement methods into the following five categories depending on the type of input data used by the models [57], [58]:

1) Media-layer models: These models utilize the audio or video signals to predict the QoE. These models do not require a priori knowledge of the system under testing, such as codec type or packet loss and hence can be applied to the evaluation of unknown systems. The media-layer models quantify the QoE using pixel-based schemes. These systems (by definition) cannot work if the media signals are not available.
2) Parametric packet-layer models: These models predict the QoE based on the packet header information and do not require any access to the media. This model provides a lightweight solution for evaluating the QoE and unlike the media-layer models can be used in a network where the packet header (TCP, RTP, UDP, IP etc.) information is available. Analyzing the information from the packet headers, the metrics such as packet loss, bit-rate, and frame-rate can be measured.
3) Bitstream-layer models: These models do not inspect the payload and hence, do not measure the impact of the audio and video quality on the QoE. On the other hand, it is difficult to implement the media layer models that analyze the payload as they are computationally intensive. As a trade-off, the bitstream-layer models extract features from the encoded payload and the packet-layer information to measure the QoE. These models utilize features of the media without using the pixel information.
4) Parametric planning models: These models take the quality parameters, such as bandwidth and packet loss rate, in the network and at the terminal to monitor the QoE. These models require a priori knowledge of the system being considered.
5) Hybrid Models: These models are the result of combining the above models. They try to exploit as much information as possible to predict the QoE.

The Media-Layer models have been studied extensively and the survey [59] presents many of these models. Since these models use the media signals, they have been mostly used for television and other classical video delivery systems. The

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

models that we consider in this survey can be classified as the parametric packet-layer and bitstream-layer models. Apart from these two models, we also review models that collect direct feedback from users, which are not part of the ITU classification, but enable researchers to collect the subjective QoE metrics. Based on the data-collection methodology and their location in the network, we classify them in the following categories: measuring QoE using client-side instrumentation, measuring QoE from within the network, measuring QoE using direct user feedback, and estimating the QoE using predictive models

A. Measuring QoE Using Client-Side Instrumentation

In order to precisely capture the QoE as experienced by the user many studies have proposed using tools closer to the client-end to measure the objective metrics or application layer metrics. These metrics are collected by using measurement tools that run on the user devices. Depending on how the data is collected by the tool, the user-end measurement studies can be classified further into the following two categories.

Measurements Based on Passive Analysis: Passive analysis tools collect QoE metrics for the videos that are being watched by the users. These tools run in the background and the QoE metrics are obtained in real-time by analyzing the video streams being played. In this case, the videos for which the QoE is measured is purely dependent on the interest of the user and the tool has no control over the selection or the playback duration of the videos.

In [22], the authors developed a client side passive measurement tool, YOMO that collects the QoE metrics for progressive download videos, specifically YouTube videos. YOMO estimated the number of interruptions occurring while watching a YouTube video by tracking the status of the playout buffer that reflects the total time, say β, that the playback can continue in case of an interruption in the download. Here, β is the difference between the time in which the content could be played from the playout buffer, T and the current play time of the video, t. In case β is smaller than a threshold β_0, the playback is interrupted. In case of YouTube, they found that the API waited for a certain duration from the time the first byte of the video is downloaded before the playback is started. However, this duration was observed to be inconstant and it did not have any correlation with the video characteristics. The accuracy with which the duration of the interruption is estimated depended on the accuracy in estimating t, which is also the time since the playback started. The authors proposed two methods to estimate t. In *Method 1*, it was assumed that the video playback began as soon as the first Flash Video (FLV) tag was downloaded. The accuracy of this method was found to be directly related to the bandwidth. In this method, the error was sufficiently low for broadband connections. In *Method 2*; YOMO uses a Firefox plugin that retrieved t from the YouTube player. This method resulted in an estimation independent of the bandwidth. The maximal error in this method was 0.5 seconds whereas it was 20 seconds in the case of slow connections with *Method 1*.

In [60], the application layer metrics are used to estimate the QoE of *Windows Media Player* users. They developed a wrapper for the player to collect the application layer metrics such as the number of packets lost, recovered, and received, the current data rate of the stream, and buffer starvation, if any. The data collected from the player was sent to a centralized location to analyze and report on the statistics. The current work used a distributed architecture that consisted of assessment servers, media clients, data collection points, and report servers. The data collection was scheduled and collected by the central assessment servers. The playback metrics collected from the application were transferred to the data collection points and sent to the assessment servers, where the data was analyzed. The analyzed results were sent to the report servers where the analyzed data was made available to the customers. They demonstrated that the application layer metric, player buffer starvation, could be used as an indicator to predict the playback interruptions. A similar approach is taken in [61] where the metrics such as the number of packets lost and retransmitted at the application layer are measured and compared against similar metrics for a reference stream to predict the user-perceived quality. The authors in [61] concluded that the re-buffering frequency and initial buffering time are the main factors affecting the QoE. In [62], the authors extended their earlier study by also collecting the *initial buffering period*, which is equal to the *Playback Start Time* along with other application layer metrics to predict the MOS ratings for the videos.

In [63], [64], the authors used client-side instrumentation for HTTP based progressive download video streaming. The instrument kept track of the user initiated actions such as pausing, resuming playback, jumping within a video, and screen switching, etc. The QoE metrics such as initial buffering time, mean buffering duration, re-buffering frequency, and bit rate switching were measured for the video streaming sessions. Apart from these objective metrics, the users were also asked to rate the videos using the MOS scale. The objective QoE metrics and the subjective feedback from the users thus collected were used to correlate the objective metrics with the MOS. Similar to the previous studies, they concluded that the re-buffering frequency is the main factor affecting the QoE in terms of the MOS.

In [65], client-side instrumentation was used to collect data from 50 million viewers for 200 million views of both VoD and live streaming services that were served by 91 different content providers. They used the QoE metrics such as *re-buffering frequency, playback start time, average bit rate, and failure to start video*. Using these metrics, three issues that could result in poor video quality were identified: (1) client-side bandwidth variations during the playback, (2) variation in the CDN performance across the time and geographic regions, and (3) heavily loaded ISPs. In order to overcome these issues, they proposed a video control plane that utilized measurement-driven feedback on the performance. The feedback enabled the control plane to dynamically adapt the video parameters such as the CDN that was used and the bit rate to improve the quality.

In [53], client-side instrumentation was used to measure the *initial playback time, buffering ratio (defined as the ratio between buffering time and play time), buffering frequency,*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

and average bit rate for different types of videos: long, short, and live. These videos were accessed from multiple different content providers, that used varied streaming techniques. A quantitative analysis was done on the correlation between the three different factors: QoE metrics, content type, and user engagement. Based on the measurements, it was concluded that the buffering ratio impacted the user-engagement for all content types whereas the bit rate mainly affected the user-engagement for live streaming videos. They also determined that the *initial buffering period* was critical for the user engagement.

In the client-side passive measurement tools presented above, the authors developed wrappers around the players for different streaming services to capture different metrics. They either polled the video player or observed the buffer status of the player to estimate the status of the video playback. The data collection in such tools requires the active involvement of the users.

Measurements Based on Active Analysis: In measurements based on active analysis, the requests for the videos are generated artificially and the QoE metrics for these videos are collected. Active measurements circumvent the need for a user to sit and watch the entire set of videos that need to be evaluated. Active measurement approaches typically use crawlers or bots that crawl through the video streaming websites and collect the QoE metrics for a large number of videos. The advantage of using such tools is that they can be easily used to measure the QoE for a large number of videos and do not require any user participation, thus eliminating any subjective bias.

In [66], the authors presented a tool *Pytomo* that actively crawls the YouTube website and downloads the videos simultaneously collecting the network latency metrics, the QoE metrics, and the CDN information. The QoE metrics, such as the initial buffering duration, number of the interruptions, and encoding rate were measured. In [67], the QoE metrics were collected from users across France and other European countries. Based on these measurements, the QoE metrics and YouTube CDN distribution policies across the locations were compared. In a separate study presented in [68], the authors collected the QoE metrics for several YouTube videos viewed by residential subscribers in the Kansas City metropolitan area. The QoE of users served by three different ISPs was measured. It was observed that even though the users were located in the same geographic area, the QoE and server selection policy varied based on the ISP, and more importantly, the server cluster from which the users were served.

In [42], the latency of three video services, YouTube, Dailymotion, and Metacafe, was measured and compared using active measurements. Automated scripts were run on several PlanetLab nodes to fetch 1 MB of each video in 50 KB increments. They computed the mean service delay for a video by averaging the service delays of three consecutive 1 MB chunks to eliminate the effect of the size of the files. This service delay, referred to as the *incremental service delay*, was independent of the client-side application as it did not account for the flow control or the buffering scheme used. By comparing the incremental service delay for the three services, it was observed that YouTube delivered 1 MB of video content nearly 6 times slower than Dailymotion and Metacafe. However, the incremental service delay is not a direct quantifier of the QoE.

Similar to the studies based on passive analysis, the active measurements are also done as close to the user as possible. These tools also capture the effect of the performance of the user device and the last hop connectivity on the QoE. Since active analysis uses crawlers, it requires minimal involvement of the users. But access to the user devices is still required to run the crawler and the data needs to be gathered manually. There is a need to develop active measurement tools that can send the collected data to centralized servers that will enable us to coordinate the data collection across several machines and analyze the data in real-time. The existing studies only look at collecting the objective QoE metrics. The possibility of using the objective metrics collected to predict the MOS score with active techniques could be an interesting direction for future research.

B. *Measuring QoE From Within the Network*

While the measurements at the user-end estimate the user satisfaction, they cannot be implemented easily by the network providers. A network provider can collect metrics within the network in a faster and easier manner. There has been extensive research on measuring the QoS metrics such as throughput, loss rate, delay, jitter, and packet reordering and it was demonstrated that these network-level parameters affect the perceived quality [69], [70], [71]. If the network-level metrics could be used to predict the QoE of the user, it would help the network providers to extend the existing measurement tools to evaluate the QoE. The following studies propose models and tools to predict the QoE based on QoS metrics collected at a vantage point in the network.

The authors of [72] used Deep Packet Inspection (DPI) in the network to develop a preliminary model to estimate the QoE in the case of progressive download services. From the TCP headers of the *pcap* traces collected, the timestamps of the TCP acknowledgments at the client were monitored. By tracking the TCP segments and the corresponding acknowledgments (ACKs), they were able to estimate the playout buffer level at the client and predict the QoE metrics such as initial buffering time, number of interruptions, and total duration of the interruptions. However, this requires the complete packet trace and is an offline analysis approach.

In [25], the authors used Tstat [73], a DPI tool that implements a traffic classifier from the flow-level statistics. YouTube traffic was analyzed from vantage points within the ISPs and university campuses across Europe and the U.S.A. It was observed that all YouTube video requests contained `HTTP videoplayback` tags in their HTTP headers. By observing the time difference between the `HTTP videoplayback` request and the reception of the video data in the payload, the start up latency of the video was predicted (the initial buffering period at the user was ignored). The user behavior during the playback was observed from the user-initiated bit rate switching, screen mode (full-screen) switching, and the portion of the video actually watched to correlate with the system performance. It was observed that although progressive download maintains the quality of streaming with aggressive download, the amount of unused data was significantly high as the users typically watched only a part of the video. The behavior of the users across different devices (such as mobiles and PCs) was also studied and it was observed that the device, location, and infrastructure had no effect on the playback quality of the YouTube videos.

An on-line QoE estimation algorithm for progressive download videos was presented in [74]. The packet-level metrics were obtained from the TCP headers and the meta data information of the video. The network layer information was collected using the Access Network TCP Monitoring Algorithm (ANTMA) [75] to predict the number of interruptions and the duration of the interruptions in real-time. However, the algorithm is designed only for an MP4 video format and makes two assumptions to work in real-time. First, the node should see all the TCP packets in both directions. Second, no re-ordering of packets takes place between the monitoring node and the TCP receiver. The authors were able to predict the QoE metrics such as the number of interruptions, and the playback start time in real-time by restricting the tool to run in the node in the access network of the user.

The QoE of services using HABS was estimated by using the session logs collected from a node within the network and a server in [76]. The session logs were generated using a packet capture method. These logs consisted of the complete MPD file, and three timestamps for each video segment downloaded: interception time of the `HTTP-GET` request from the client (including the URL), the interception time of the first packet with the video payload (this gives the segment size), and the `OK` time stamp that indicates the end of the last segment. The session log data collected was used to reconstruct the HABS session that reflected the adaptation of the bit rate and the evolution of the buffer-filling (in seconds) over the duration of the session. Based on the reconstructed sessions, the authors were able to estimate the QoE metrics such as the number of interruptions, initial buffering duration, average duration of each interruption, number of bit rate switching events, and average re-buffering time due to user interaction.

Three in-network methods to measure the QoE of YouTube videos were presented in [77] by estimating the number of interruptions during the playback. The first method, M1 measures the total stalling time T as the difference between the total video download time Y and the video duration D, i.e., $T = \max\{Y - D, 0\}$. The second method, M2, is based on the stalling frequency $F(x)$ approximately defined by (2) from [45] where x is the normalized video demand defined as the ratio of video bit rate (V) to the bottleneck capacity (B) and α, β, and γ are constants.

$$F(x) = \alpha e^{\beta x} + \gamma \tag{2}$$

Both, M1 and M2 require access to the video metadata and are not accurate since they estimate the total stalling time T and/or the number of stalling events N, assuming that the distribution of the stalling length L is known.

The third and the most preferable method, M3, estimates the video buffer status at the client-side from the network level measurements. The size of each video frame and the video frame rate are extracted from the metadata in the FLV container. The total video data downloaded, τ_i is calculated in terms of the video playback duration, based on the arrival times of TCP ACKs at time *t* and the video frame rate.

The M3 approach is developed according to the way the YouTube player buffers and displays the video. The YouTube player uses two different playing and stalling thresholds to control the playback from the buffered data. The Boolean variable ψ_i is used to indicate if the video is experiencing a stalling(ψ = 1) or not (ψ = 0) after the i^{th} TCP ACK. The first threshold Θ_0 defines the minimum duration of the buffered video to be exceeded before a stalled playback is resumed; the second threshold Θ_1 specifies the minimum duration of the buffered video to continue the playback once it started initially. So if we assume the playout buffer size to be β_{i-1} at t_{i-1}, then it means that when β_{i-1} exceeds Θ_0, then the video starts playing; on the other hand, if β_{i-1} falls bellow Θ_1, then the video stalls. Hence, stalling occurs if the following condition is true: $\psi_{i-1} \wedge (\beta_{i-1} < \Theta_0) \vee \neg\psi_{i-1} \wedge (\beta_{i-1} < \Theta_1)$. Further, ρ_i in (4) and σ_i in (5) indicate the video playtime and stalling time experienced by the user, respectively computed on receiving the i^{th} TCP ACK. The total duration of the buffered video (β_i) in (6) corresponds to the difference between the duration of the downloaded video τ_i and the duration of the playback ρ_i.

$$\psi_i = \psi_{i-1} \wedge (\beta_{i-1} < \Theta_0) \vee \neg\psi_{i-1} \wedge (\beta_{i-1} < \Theta_1) \tag{3}$$

$$\rho_i = \rho_{i-1} + \begin{cases} 0, & \text{if } \psi_i \\ t_i - t_{i-1}, & \text{if } \neg\psi_i \end{cases} \tag{4}$$

$$\sigma_i = \sigma_{i-1} + \begin{cases} t_i - t_{i-1}, & \text{if } \psi_i \\ 0, & \text{if } \neg\psi_i \end{cases} \tag{5}$$

$$\beta_i = \tau_i - \rho_i \tag{6}$$

Depending on the state of the video: stalling ($\psi = 1$) or playing, the stalling time (σ) or playtime (ρ) are updated, respectively. For YouTube, which waits for an initial buffering duration(Θ_0) before starting the playback, the variables in the above equation are initialized as follows:

$$\sigma_0 = 0, \rho_0 = 0, \psi_0 = 1 \tag{7}$$

Although M3 adds to the system complexity it was observed to have better accuracy compared to M1 and M2 and hence, it is the preferred method. Using M3, they were able to measure 50% of the number of interruptions to the exact value and for an additional 30% of the videos with a difference of one interruption. However, they found that the relative error for the estimation was below 20% for 90% of the videos. For YouTube videos, they were able to predict the MOS correctly for 80% of the videos.

The use of the in-network QoE estimation techniques eliminates the need to modify the existing video players or deploy additional software on the client or the server. These tools can be deployed easily by network providers who have greater control over the network compared to the user devices and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

are also platform independent. However, due to the limited processing capabilities of the nodes in the network and large packet processing times involved in Deep Packet Inspection (DPI), these techniques are best suited for off-line processing. Packet loss and caching between the network monitoring node and the client can pose significant challenges to the estimation of the playout buffer state using these tools.

C. Measuring QoE Using Direct User Feedback

A popular method to evaluate subjective QoE metrics is to directly ask the user to provide the feedback after viewing a video. The users are asked to watch a set of videos and rate each one of them using the MOS scale. These types of studies can measure the effect of the objective metrics as well as the confounding factors on the QoE. The users who are asked to rate the videos could be volunteers such as friends, colleagues, and acquaintances or paid personnel (crowd-sourcing). Such studies are performaed in a controlled environment that enables the researchers to have complete control over the stream settings and the user interface. These types of studies can be easily extended to any type of video streaming services as they depend only on the user's feedback.

For accurate and consistent subjective QoE assessment it the viewing conditions are crucial. The standard in [78] specifies the conditions that are needed to be considered to conduct experiments that describe the viewing distance, room lighting conditions, selection of the subjects, video content selection, and assessment and data analysis methods. The tests are conducted using a set of participants who rate the quality of the videos on the the discrete MOS scale.

In [64], the authors created a platform that played HTTP streaming videos by varying the objective QoE metrics and collected the MOS rating from different subjects. They studied the ratings submitted by 10 different subjects who were non-experts in the estimation of the video quality. The authors simulated an environment to introduce re-buffering events by pausing and playing the videos. It was observed that the number of interruptions is the main factor affecting the MOS given by the users.

In [79], [80], the authors created a customized RTSP-based video player *RealTracer* that played Real video clips. The servers hosting the videos were spread out geographically. *RealTracer* collected several objective metrics: encoded bandwidth, measured bandwidth, transport protocol, encoded frame rate, measured frame rate, playout jitter, frames dropped, and CPU utilization. The authors solicited 60 users across 12 different countries through personal contacts and online forms to watch several videos. For each of the videos, the users were requested to provide a score between 0 and 10 based on the quality of streaming. The authors were able to correlate the user ratings with the metrics collected and observed that there was hardly any difference in the perceived quality when the video was streamed from different servers in the same country. However, there was a noticeable difference in the perceived quality when the videos were streamed from different countries.

Contacting a sufficiently large number of users to perform subjective measurements involving direct feedback from the users is a non trivial task. It is more difficult to involve users from different geographic locations. A crowd-sourcing model where the tasks are outsourced to several anonymous users, could be very useful in this context. These users could be located across the globe and could be remunerated to watch a certain set of videos under varying conditions and provide the feedback in the form of MOS. Examples of crowd-sourcing websites are Mechanical Turk [81] and Microworkers [82].

However, it is important to understand that the users in a crowd-sourcing environment might be trying to maximize their remuneration. The researchers cannot directly monitor the crowd-sourced tasks and hence, the data collected from such studies are not entirely reliable. In order to gather the data that is more reliable, the tasks assigned need to be interleaved with gold standard data tests, consistency tests, usage monitoring, and content questions as suggested in [45]. The verification tests are a set of questions whose answers are known a priori, to ensure that the response by the users is correct. Consistency tests involve asking the same set of questions with a slightly different wording to ensure that the responses are consistent. Usage monitoring is done to determine the duration for which the video player was in focus, thus ensuring that the users are not browsing other websites during the test. Another method for ensuring reliability is to provide a set of questions related to the content of the videos. A reliable crowd-sourcing data collection model would use the above tests to process the data collected. If the user data does not satisfy any of the tests, then it cannot be trusted and should be discarded.

In [44], a subjective study was performed using two different experiments. The first experiment collected MOS from volunteers in a laboratory and the second experiment collected MOS by crowd-sourcing. The data from both the experiments were used to quantify the impact of initial delay and interruptions on the QoE for different scenarios. This study compared the sensitivity to initial delay and the interruptions while watching YouTube videos. Apart from YouTube videos, they also studied the effect of the authentication time for the social networks and Wireless 3G Internet connection setup time on the QoE. Based on the MOS data collected, the authors found that the users in both the experiments preferred some delay before playback started instead of an interruption. Even a short duration of an interruption significantly reduced the satisfaction of the users. It was also observed that the results from crowd-sourcing were similar to the results from the laboratory tests and the length of the video clip did not have any influence on the MOS.

In [83] video characteristics in terms of duration and video bit-rate in addition to the stallings are used to map the bottleneck bandwidth to the subjective QoE. They also present a UDP based YouTube video streaming model. Using subjective tests they are able to compare the QoE of UDP and TCP based YouTube streaming.

In [28] the authors developed a framework to obtain the subjective ratings from the volunteers to measure the QoE for an adaptive video streaming service and compare it with fixed bitrate video streaming (progressive download) technique. They used 141 volunteers to rate artificially spliced variable bitrate samples that emulated a DASH session. The

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

volunteers were asked to rate the overall video experience using a MOS rating and also to rate their satisfaction with objective metrics such as video-definition (video quality), fluency (interruptions), response speed (initial delay). Based on their studies, they found that for the HABS streams the number of interruptions affects the QoE more than the video quality, the startup bitrate and slow bit-rate ramp up affects the QoE, and the users are sensitive to frequent bit-rate switching. They also concluded that the QoE of HABS videos was better than the progressive download videos.

[84] presented two approaches to subjective evaluation of Multiple HABS streaming solutions including Microsoft Smooth Streaming, Apple HTTP Live Streaming. In order to subjectively test the HABS in typical conditions they developed a lab network which represented an end-to-end network. In the first approach, which acted as the baseline, they compared the three different adaptive streaming techniques with the Microsoft Mediaroom Internet Protocol Television solution. This comparison was done by collecting the MOS ratings for the different streams with unlimited bandwidth and no network impairments. Based on this study they were able to conclude that all three HABS solutions performed better than IPTV. In the second approach, they introduced impairments to multiple clips and collected the user MOS rating for them. The impairments represent typical network errors related to: bandwidth, latency, random packet loss, burst packet loss on both downstream and upstream. In the second test, they found that the adaptive streaming services experienced a smaller drop in the MOS rating than IPTV when network errors were introduced. For example, when the bandwidth was reduced from 6.2 Mbps to 2 Mbps, the change in the MOS for HLS was 0.66, whereas for IPTV, it was 2.89 points. Hence, they were able to conclude that HABS streaming services perform better than IPTV services when the network faces any impairments.

The MOS feedback collected from the users either in a controlled lab or crowd-sourcing environment can be used to understand the effects of various metrics on the QoE. However, it is difficult and time consuming for the users to watch and provide feedback on every video. This process of collecting data is also expensive and not easily repeatable.

D. *Estimating the QoE Using Predictive Models*

Measurement studies that obtain feedback from the users on the perceived quality are found to be good at assessing the QoE of the users. However, they are time consuming, expensive, subject to bias, affected by confounding factors, and do not scale well. If we are able to replicate the rating by users in an automated method we should be able to overcome some of the limitations of the direct-user feedback mechanism. This can be achieved by developing predictive models that estimate the subjective QoE metrics in an automatic, quantitative, and repeatable manner.

In this section, we review the work that has led to the development of models to predict the MOS score from the objective metrics collected. The design of such models is challenging because the QoE metrics are interdependent and the relationship between these metrics is found to be non-linear. The first challenge, in order to develop MOS prediction models based on the objective metrics, is to be able to define a relationship between them. The predictive models rely on studying the objective and confounding factors for the videos. Depending on the amount of information available about the source videos and the video received at the user end the models can be classified as Full-reference (FR), Reduced-reference (RR), and No-reference (NR) models [85]. For full-reference (FR) models, the complete original and the distorted videos are available for comparison. These models are difficult to be implemented in online systems and are more commonly used in traditional broadcasting services. In case of the reduced-reference (RR) models, the distorted video and some extracted structural features of the original video are available. The last model, which is the no-reference (NR) model has only access to the distorted videos at the user end. Since these models do not need any comparison, they can be used in real-time and are most suitable for online streaming.

In order to predict the MOS from the QoE and/or QoS metrics collected, it is important to identify the relationship between the subjective MOS score and the objective QoE or QoS metrics. The problem to find this relationship between the different QoE and QoS metrics and to predict the MOS from their relationship has been studied for various web applications that have later been extended to video streaming applications.

A generic expression that captures the exponential relation between the QoE and QoS parameters was proposed in [38] and is called the IQX hypothesis. Here, the QoE for streaming services is considered in terms of the MOS and is expressed as a function of loss and reordering ratio (caused by jitter). According to the IQX hypothesis, the change of the QoE depends on the current level of the QoE given the same amount of change in the QoS value but with a different sign, as shown in (8). If the QoE is already very high, then even a small disturbance can effect the QoE; however, if the QoE was already low to begin with, then a further disturbance will not be perceived.

$$\frac{\partial QoE}{\partial QoS} \sim -(QoE - \gamma). \tag{8}$$

It was demonstrated that the proposed relationship provided a better approximation when compared to the original logarithmic approximations presented in [86]. A similar pattern was seen in [43] where users watching the videos with better conditions would have less patience with the initial startup delay and would abandon sooner.

The authors in [86] presented a simple model that defined the relationships between one of the QoS parameters: bandwidth, response times and the QoE. In [87], this model was extended to capture the effect of multiple QoS parameters such as throughput and delay on the QoE. A new discrete scale called the Opinion Score (OS) that ranges from 0 to 5 was introduced to eliminate the constant γ from (8). Using the Opinion Score values and applying the Multiple Linear Regression Model(MLR) from [88], a linear relationship between the QoE and multiple QoS parameters was defined as:

$$\log(\mathrm{QoE}) = a_0 + a_1\mathrm{QoS}_1 + a_2\mathrm{QoS}_2 + ... + a_n\mathrm{QoS}_n \tag{9}$$

where constants a_i were estimated by the least squares

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

method. On applying an exponential transformation on (9), the QoE/QoS exponential correlation is modeled as

$$\text{QoE} = e^{a_0} + e^{a_1 \text{QoS}_1 + a_2 \text{QoS}_2 + \ldots + a_n \text{QoS}_n} \tag{10}$$

The models in (8) and (10) were defined for web-pages and files. However, based on the following studies, it was found that the aforementioned models were applicable for video streaming services. In [89], the authors presented a similar non-linear model (11) using a *psychometric model* to estimate the MOS from the QoS parameters for HTTP-DASH streaming systems using any N parameters.

$$\text{QoE} = \sum_{i=0}^{N-1} a_i \text{QoS}_i^{k_i} \tag{11}$$

where a_i are the constants and k_i are the exponents for N metrics.

The *Perceived QoE* model [90] defines a non-linear relationship between the MOS and objective QoE/QoS metrics. This no-reference model extracts parameters such as the bit rate b, frame rate f, packet loss rate l, video jerkiness j, and quantization parameter q, to approximately map the MOS. These parameters are combined by the following non-linear formula (12):

$$\text{MOS} = B \times b^{bb} + F \times f^{ff} + L \times l^{ll} + J \times j^{jj} + .. + Q \times q^{qq} ... \tag{12}$$

The above formula uses specific weights (capital letters) and exponents that can be estimated according to the service and the device being studied. This composite metric helps capture the effect of multiple parameters on the MOS rating.

A preliminary model with limited scope that predicts the QoE in terms of MOS for progressive download videos has been presented in the ITU-T P.1201 standard [91]. This model considers sequences between 30 to 60 seconds and considers the interruptions, and playback start time.

A cross-layer monitoring architecture was proposed in [54] to monitor the packets at the node and the network level to measure the QoS metrics (packet loss ratio) and to build a physical and network level view. A mapping tool estimates the Decodable Frame Rate (Q) at the user for the streaming service based on the QoS measurements. The tool then predicts the degradation in the QoE in terms of the MOS from the Decodable Frame Rate (Q).

Pseudo Subjective Quality Assessment (PSQA) is a hybrid approach that combines subjective and objective evaluation [92]. It is used to predict the subjective ratings of videos using automation. In this approach, an initial set of reference videos are evaluated subjectively by users. These reference videos are samples of distorted videos whose objective metrics are known. The result of the initial evaluation is used to train a self-learning tool to predict the subjective rating in real time.

In [92] a framework for measuring the QoE of MPEG videos was presented. The framework consisted of a streaming server, monitoring nodes (called probes), a data collector server, a PSQA learning tool and a web-based reporting application called *Webstat*. The streaming server was used to deliver video content (MPEG) and audio content (MP3) using several protocols, HTTP, RTP, and UDP. The probes collected frame level metrics such as the loss rate (LR) of video frames, and the mean size of loss bursts (MLBS)), defined to be the average length of a consecutive sequence of frames lost but not part of a longer such sequence. The probes transferred the collected metrics to the data collector server using the Simple Network Transfer Protocol (SNMP). The perceptual QoE was calculated by the PSQA tool, and the computed QoE metrics were presented using the *webstat* application. Before the framework was used for real-time measurements, the PSQA tool was passed through a pre-processing stage during which it learned the effect of LR on the QoE. During this stage, a fixed set of videos with different LR and MLBS values were evaluated by a group of five human experts. The five experts provided the MOS values for each of the initial sequences. These MOS values were used as input to train a Random Neural Network (RNN) on the two variables, LR and MLBS, and mapped them into a perceived quality on a $[0, 1]$ range. Once the pre-processing was complete, the framework was ready to predict the MOS for the videos. During a real-time evaluation of the QoE, the probes collected the frame level information of the video streams and transferred the LR and MLBS metrics to the data collector server. Using the trained RNN model, the PSQA tool used the LR and MLBS metrics as input to generate the MOS in real-time.

The authors in [61], used the PSQA tool to predict the perceived quality of the videos streamed using a *Windows Media Player* that uses UDP to transfer the data. Objective metrics, such as *Application Layer Packets Lost* and *Application Layer Packets Retransmitted* were used to predict the MOS rating. Before applying this method in real-time, the MOS ratings of certain sets of videos (called the reference streams) with known objective metrics were evaluated by a group of participants. These volunteers were asked to rate the reference video streams twice, once without loss and once with loss. The MOS and the objective metrics for each stream were fed to a *Dynamic Time Warping* (DTW) predictor during its training stage. On completion of the training stage, the DTW predictor was ready to rate the streams in real-time. During real-time measurements, the application layer metrics were obtained by polling the *Windows Media Player* periodically. It was found that the DTW predictor matched the patterns of the metrics of the active streams with the reference streams to predict the MOS in real-time.

In [93], [94] the authors extended the PSQA to predict the QoE by using some parts of the videos rather than the complete videos. These parts were used in both training as well as in real-time prediction. With a set of preliminary experiments, it was demonstrated that the MOS could be predicted with 70-80% accuracy.

A_PSQA proposed in [95] is a reduced-reference model based on PSQA measures. In this model, several distorted videos were evaluated subjectively and the data was used to train an RNN to capture the relationship between the network parameters and the perceived quality. This model used the packet loss rate but can be extended to other network metrics such as the packet loss rate, mean loss burst size, end-to-end delay, and jitter. It was observed that the results of the A_PSQA model correlate very well with the subjective scores

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

when compared to the full-reference models and no-reference models.

In [96], the authors used a no-reference PSQA model based on RNN to estimate the QoE for Adaptive HTTP over TCP video streaming. The parameters considered were the interruptions, average interruption duration, and the maximum interruption duration. Another QoE metric unique to Adaptive HTTP streaming, *Quantization Parameter* (QP), which controls the degree of video compression was also considered. The MOS collected from the users viewing an initial set of distorted reference videos was used to train the PSQA-RNN tool that was later used to predict the MOS for other videos.

In [97], the authors proposed a predictive model for the user-engagement as a function of the QoE metrics. The user-engagement is defined as the amount of time spent by the user in watching the video before quitting. In order to capture the effect of the QoE metrics they use a machine learning algorithm. The QoE metrics, such as the average bit rate, join time (playback start time), rate of buffering events, buffering ratio (ratio between buffering and play time), and frequency of buffering were collected using client-side instrumentation for both VoD and live sessions. These metrics were used to develop a machine learning model for which the relationship between the metrics was determined by using *Information Gain Analysis*. Later, they used a compacted decision tree to predict the impact of different quality metrics on the user-engagement. The also concluded that the main confounding factors are the type of video, device, and connectivity.

In [98], the authors presented an online learning QoE management tool that uses machine learning to understand the effects of the objective metrics and application layer metrics on the MOS ratings. The effect of objective metrics such as the video bit rate, audio bit rate, and frame rate on the perceived subjective QoE was studied. In the current model, continuous feedback from the users (MOS) was collected and served as a datapoint for the online learning algorithm. As the number of feedback instances increased so did the accuracy of the model. By using online learning algorithms they avoided the need for complex and expensive a priori subjective tests that use reference streams. They also demonstrated that the learning algorithms can retain the accuracy in estimation of the QoE without the use of a priori subjective studies.

In [99], the authors developed a framework that uses prediction models to realize a QoE-aware QoS management strategy. The framework would predict the MOS rating from the QoS parameters by using a statistical prediction model based on Discriminant Analysis. The QoS parameters considered were the encoding bit rate and the frame rates of the videos. A set of subjective measurements, where several users are presented with reference videos with degrading levels of QoS, were conducted. During these measurements the feedback from the subjects was recorded in the form of a binary response (“acceptable” or “unacceptable”). The main aim of these tests was to determine at what level of the QoS metrics and the perceived quality becomes unacceptable. The data collected from these measurements was then used as the input for the model that correlates the QoS parameters with the QoE perception. Using this model, the degree of influence of those QoS metrics had on the QoE was determined. The prediction model developed was used to realize a management strategy to control the QoS parameters that could guarantee a satisfactory QoE level.

A content-adaptive packet layer model to assess the QoE in terms of the frame rate quality for video services using RTP/UDP was presented in [100]. The bit-rate, packet loss rate, temporal complexity (the acuteness of temporal changes of a video sequence), and the frame type information of the transported video were determined from the packet headers. Using all these factors, a model to predict the MOS was developed.

All the studies done so far attempted to build models to capture the relationship between the QoS metrics, the objective QoE metrics, and the MOS. Due to the non-linear relationship between these metrics, it is not easy to construct a simple model. Considering the work done in the literature to date, it can be concluded that the relation between the QoS metrics and the QoE score follows a non-linear exponential relationship. The use of supervised learning techniques and neural network-based models could be used to predict the QoE in real time. The effect of confounding factors on the subjective QoE metrics also needs deeper investigation and the inter-dependence of the metrics needs to be studied further.

E. Comparison of the Studies

In Table I, we present a comprehensive view of the QoE measurement studies in the literature, highlighting varied approaches taken to measure the QoE metrics. These approaches vary in tools used for the data collection, the location of the collection, the video delivery techniques used, and the QoE metrics used in the studies. We observed that most of the studies collected data passively closer to users to capture the perceived quality. This approach has the combined advantage of capturing the metrics at close proximity to the users with minimum user involvement.

QoE measurement studies at the user-end are classified based on how the videos are analyzed. These measurements could either be based on passive or active analysis. Passive measurements are done on the user device with tools executing in the background without the involvement of the user. Various measurement tools have been developed to capture the metrics using passive analysis for streaming services based on RTSP ([80], [60]), and progressive download (YOMO [22]). The Pytomo [66] tool collects the user-end metrics using active analysis for progressive download videos. Most of the tools developed to date work are designed for a particular streaming service or video player. In order to measure the QoE of a wider range of services there is a need to develop an integrated tool that can measure the metrics for different services. There is also a lack of client-side measurement tools for *adaptive streaming over HTTP services*. The increasing popularity of adaptive streaming over HTTP makes this an important direction for future research.

The measurements that collect feedback from the users directly use either a controlled laboratory environment or a crowd-sourcing technique where the researchers control the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

TABLE I: QoE Measurement Studies: A Comprehensive View

Ref.	Point of Collection		Data Collection Method					VDT	QoE Metrics					
	User	Network	Passive	Active	DUF	PS	CL		IBT	I	RF	AL	MOS	Other
[22]	✓		✓					PD			✓			
[63]	✓		✓		✓			PD	✓	✓	✓		✓	User-Viewing Activities
[62]	✓		✓					RTSP	✓	✓	✓	✓		
[61] [93] [94]	✓		✓			✓		RTSP				✓	✓	
[96]	✓		✓			✓		HABS		✓	✓			
[53]	✓		✓					Multiple	✓	✓	✓			AverageBitrate, Rendering Quality
[65]	✓		✓					Multiple	✓		✓			Average Bitrate, FailureRate, & Failure to Start video
[60]	✓		✓	✓				RTSP		✓				
[66] [67] [68]	✓			✓				PD	✓	✓	✓			
[42]	✓			✓				PD				✓		
[79] [80]	✓		✓		✓			RTSP				✓	✓	Bandwidth, Jitter, Framerate, & CPU Utilization
[64]	✓				✓		✓	PD	✓	✓	✓		✓	
[45]	✓				✓			PD		✓	✓		✓	
[44]	✓				✓			PD	✓	✓			✓	
[92]	✓		✓			✓		Multiple					✓	LR, MLBS
[25]		✓	✓					PD	✓					Bitrate Ratio
[72]		✓	✓					PD	✓	✓	✓			
[74]		✓	✓					PD	✓	✓	✓			
[77]		✓	✓					PD	✓	✓	✓		✓	
[76]		✓	✓					HABS	✓	✓	✓			Bitrate switching, Rebuffering (User initiated)
[28]	✓				✓			HABS	✓	✓				Startup and average bitrate, bitrate switching events

VDT = Video Delivery Technique, PD = Progressive Download, DUF = Direct User Feedback, CL = Controlled Lab, PM = Passive Measurements, AM = Active Measurements, PS = Pseudo Subjective Quality Assessment (PSQA),HABS = HTTP Based Adaptive Bitrate Streaming, IBT = Initial Buffering Time, I = Duration of Interruption, RF = Rebuffering Frequency, AL = Application Layer Metrics (Packet Loss)

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

playback. However, collecting such feedback for a large set of videos can be difficult and expensive. To be able to extend these studies to a wider range of subjects, a feedback module that is integrated with the video delivery service (using an embedded feedback button) can be helpful. Such a feedback module would enable the actual viewers (instead of volunteers) to provide feedback when they face any deterioration in the service during the course of regular viewing sessions.

While the client-side instrumentation tools have been successful in capturing the QoE of the users, they require access to the user devices. This requirement could be a bottleneck for wider applicability of this approach due to privacy and security concerns from the users. For the network and service providers who might want to monitor the QoE on a regular basis, it is more convenient and scalable to perform these measurements from within the network. Although, there have been a few studies that have looked at collecting the QoS metrics from the network to predict the QoE ([25] [72]), these studies are limited to progressive download services. The work in [76] used session reconstruction techniques to reconstruct the HABS streams in order to predict the player buffer status at the user. The development of tools to measure the QoE from within the network for other video streaming services based on RTSP/RTMP is an important research direction. The existing in-network tools also do not handle the effects of caching between the probe and the user. If the tools could work with the presence of caches in the network, they would be able to estimate the QoE more effectively.

Several QoE prediction models have been presented in IV-D. These models help us understand that the relationship between the MOS and the objective metrics follows a non-linear exponential relationship. In order to predict the MOS more accurately the use of supervised machine learning techniques could be further investigated. The client-side tools that work on the PSQA, measure the QoE for Windows Media Streaming Services using the application layer metrics([61], [93], [94]). These tools could be be extended to measure the QoE for Quicktime and RealPlayer clients as well. If these application layer metrics could be collected using APIs embedded in the web-browsers (similar to [22]) then the PSQA techniques could be extended for all services that use embedded players (YouTube, Netflix, Dailymotion).

Apart from the client-side and in-network measurements, there is a third possible location where the QoE metrics could be collected from the video streaming servers. Although several video service providers already published the relative performance of their services across various ISPs, there are not many results on the QoE of individual users. Netflix has been publishing the relative speeds of the ISPs measured at the servers [101], but not any information on the QoE metrics. The video streams can be monitored directly at the servers to measure the QoE of the users. The development of tools to measure and monitor the streams on the server to predict the subjective QoE metrics is an interesting direction of research.

V. Conclusion and Future Direction

Online video streaming services are being touted as the next largest video entertainment services that could replace television and cable services. There has been significant work done in the television and broadcast industry to measure the QoE of videos; however, these studies are based on full-reference and reduced reference metrics. Also, the networks used by the traditional video delivery systems did not have to compete with other services as is the case with the online video streaming services. Hence, the online video streaming services provide a unique challenge to measure the QoE. Although online video services have been available for many years now, the metrics and the methods used for the quantification of the quality of the online video streaming services have not yet been standardized. This survey paper focusses on capturing various QoE metrics that have been proposed and the different measurement methodologies used to measure the QoE metrics. There is still a significant amount of work that is to be done in order to be able to measure and predict the QoE for online video streaming services. In the following paragraphs, we try to point out some of the future directions that could be taken by researchers in order to improve the measurement of QoE for online video streaming services.

The relationship between the QoE metrics, the confounding factors, and the QoS metrics has been found to be complex and non-linear. It has not yet been fully understood; so far, it is still considered to be an open problem. There needs to be more work done to understand how the different confounding parameters affect the QoE metrics. By understanding the relationship between the factors, we can get closer to developing a definite model that can capture all the QoE metrics to precisely predict the MOS.

Client side instrumentations collect the QoE metrics at the client machine. These tools help capture the effects of performance of the service provider, the network and also the connectivity of the last hop link. The various tools designed [22], [60], [65], [66] are limited to a single work with a single video service. These tools could be extended to not just collect the video buffer metrics, but also other confounding factors that affect the QoE of the users. These tools could also be extended to different popular video services. These client side instrumentations could also be integrated with a centralized server architecture to be able to automate the data collection. For such an architecture, the privacy and data collection methods that do not affect the streams and scalability could be an important challenge.

The use of in-network QoS measurements to predict the QoE needs additional effort as it is an important challenge from the network providers' perspective. Since the network providers already have tools in place to measure the QoS metrics, by increasing the accuracy and speed of the QoE prediction algorithms, these tools could be extended to measure the QoE. New machine learning and neural network based prediction algorithms could be applied to online video streaming QoE prediction.

The measurement of user QoE from metrics collected from the server-end could be another important challenge. So far there is limited published work in this field, but this could enable the service providers to understand and better serve the video content. The service providers are in a good position to directly collect metrics such as quality of the video, user-engagement, bit-rate switching events for all the video sessions. For example, YouTube provides a feature for its channel publishers where they can track the average view duration for videos. However, there needs to be more work done to develop a mechanism that can predict the other QoE metrics such as: playback start time, number of interruptions etc. as observed by the user based on the measurements at the server-end. These server-end based models can help measure the QoE on a larger scale and limit the user involvement.

Most of the tools look at measuring the metric at either the user-end, or network-end. Each of these locations gives a different perspective of the QoE. If a system can be designed with a hybrid approach that could collect metrics simultaneously from both the network and user-end, such approaches would help to correlate the QoS metrics on the QoE and generate a better MOS prediction tool. Also, such hybrid studies will allow the study of the impact of the variations in the performance of the network on the users' QoE. Other approaches such as the multi-dimensional ARCU model [102] which categorizes the confounding metrics into application, resource, context, and user spaces could also be used to understand the multi-dimensional relationship between the confounding factors and the QoE.

The QoE metrics can be used to optimize the video delivery techniques and the rate adaptation algorithms for a better user experience. Initially, the video delivery techniques were designed to improve the QoS metrics. More recently the video delivery techniques and the rate adaptation algorithms have been developed to improve the QoE metrics. However, since the different QoE metrics are interdependent, the optimization of these metrics presents a unique challenge while it is a worthwhile issue to address in the future. For example, in the case of progressive download videos, the tradeoff is between the playback start time and interruptions. Increasing the playback start time ensures that any jitter in the network during the playback does not cause interruptions. On the other hand, from [43], we see that when the playback start time increases beyond 2 seconds, the user might give up watching the video. In HTTP based videos, the challenge is to determine the tradeoff between the best playback time, initial bitrate, and the selection of the bitrate for the next segment to be downloaded. There are studies that have used QoS metrics, such as bandwidth and throughput to determine the most optimal bit-rate to be used for the next segment to be downloaded [30], [31]. Improving the rate adaptation schemes in order to improve the QoE for the users for various streaming is a popular and important challenge.

Finally, this survey does not consider QoE for users accessing from hand-held mobile devices served from cellular access networks or QoE of video streaming in wireless networks. While many of the QoE metrics discussed here are applicable to such environments, research on this area is still in a nascent stage [103]–[105]. This will be a worthwhile area to conduct research in the future.

Acknowledgement

The authors would like to thank the anonymous reviewers for their detailed, valuable and constructive comments and suggestions, as well as Nancy Lorenz for proof reading.

References

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 20122017," Retrieved 4.3.2014. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf

[2] Sandvine, "Global Internet Phenomena Report: 1H 2013," Retrieved 4.3.2014. [Online]. Available: https://www.sandvine.com/downloads/general/global-internet-phenomena/2013/sandvine-global-internet-phenomena-report-1h-2013.pdf

[3] ITU-T Recommendation P.10/G.100-Amendment 3, "Vocabulary for performance and quality of service," Recommendations of the ITU,Telecommunications Sector, 12 2012.

[4] ITU-T Recommendation P.10/G.100-Amendment 2, "Vocabulary for performance and quality of service," Recommendations of the ITU,Telecommunications Sector, 12 2012.

[5] R. Schatz, T. Hoßfeld, L. Janowski, and S. Egger, "From packets to people: Quality of experience as a new measurement challenge," in *Data Traffic Monitoring and Analysis*, ser. Lecture Notes in Computer Science, E. Biersack, C. Callegari, and M. Matijasevic, Eds. Springer Berlin Heidelberg, 2013, vol. 7754, pp. 219–263. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36784-7_10

[6] P. L. Callet, S. Mller, and A. Perkis, "Qualinet white paper on definitions of quality of experience," *European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003)*, 2013.

[7] T. Hoßfeld, R. Schatz, E. Biersack, and L. Plissonneau, "Internet video delivery in YouTube: From traffic measurements to quality of experience," 2013.

[8] N. Staelens, S. Moens, W. Van den Broeck, I. Marien, B. Vermeulen, P. Lambert, R. Van de Walle, and P. Demeester, "Assessing quality of experience of IPTV and video on demand services in real-life environments," *Broadcasting, IEEE Transactions on*, vol. 56, no. 4, pp. 458–466, 2010.

[9] Alliance for Telecommunications Industry Solutions (ATIS), "ATIS IPTV Exploratory Group Report and Recommendation to the TOPS Council," July 2006.

[10] "HTTP Archive." [Online]. Available: http://httparchive.org/trends.php

[11] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "YouTube traffic characterization: A view from the edge," in *In: Proc. of IMC*, 2007.

[12] D. K. Krishnappa, D. Bhat, and M. Zink, "DASHing YouTube: An analysis of using DASH in YouTube video service," *38th Annual IEEE Conference on Local Computer Networks*, vol. 0, pp. 407–415, 2013.

[13] H. Schulzrinne, A. Rao, and R. Lanphier, *RFC2326 - Real Time Streaming Protocol (RTSP)*, Std., April 1998.

[14] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," 7 2003, rFC 3550.

[15] Real Networks, "Real player 8 plus user manual." [Online]. Available: http://docs.real.com/docs/playerplus/plus_manual.8_pdfs/rp8manualfull.pdf

[16] ——, "Configuring RealPlayer version 7 to work from behind a firewall," Retrieved 4.3.2014. [Online]. Available: http://service.real.com/firewall/configRP7.html

[17] Apple, "Quicktime 7.3," Retrieved 4.3.2014. [Online]. Available: http://www.apple.com/quicktime/

[18] Microsoft, "Which protocols does windows media player use for streaming?" Retrieved 10.13.2014. [Online]. Available: http://windows.microsoft.com/en-us/windows/which-streaming-protocols-media-player

[19] Adobe, "Adobe's Real Time Messaging Protocol," Retrieved 4.3.2014. [Online]. Available: http://www.adobe.com/devnet/rtmp.html

[20] V. Adhikari, Y. Guo, F. Hao, V. Hilt, and Z.-L. Zhang, "A tale of three CDNs: An active measurement study of Hulu and its CDNs," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, March 2012, pp. 7–12.

[21] Cisco, "HTTP versus RTMP: Which way to go and why?" Retrieved 4.3.2014. [Online]. Available: http://www.cisco.com/c/en/us/products/collateral/video/as5100-series-media-processor/white_paper_c11-675935.html

[22] B. Staehle, M. Hirth, F. Wamser, R. Pries, and D. Staehle, "YoMo: A YouTube application comfort monitoring tool," Institut für Informatik, Technical Report 467, March 2010.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

[23] M. Ghobadi, Y. Cheng, A. Jain, and M. Mathis, "Trickle: Rate limiting YouTube video streaming," in *Proceedings of the 2012 USENIX Conference on Annual Technical Conference*, ser. USENIX ATC'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 17–17.

[24] S. Alcock and R. Nelson, "Application flow control in YouTube video streams," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 2, pp. 24–30, Apr. 2011. [Online]. Available: http://doi.acm.org/10.1145/1971162.1971166

[25] A. Finamore, M. Mellia, M. M. Munafò, R. Torres, and S. G. Rao, "YouTube everywhere: Impact of device and infrastructure synergies on user experience," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11, 2011, pp. 345–360.

[26] R. Leider, "MPEG DASH supersession - a session sponsored by DASH," IBC, Tech. Rep., 2014. [Online]. Available: http://www.ibc.org/

[27] I. Sodagar, "The MPEG-DASH standard for multimedia streaming over the internet," *MultiMedia, IEEE*, vol. 18, no. 4, pp. 62–67, April 2011.

[28] L. Yitong, S. Yun, M. Yinian, L. Jing, L. Qi, and Y. Dacheng, "A study on quality of experience for adaptive streaming service," in *Communications Workshops (ICC), 2013 IEEE International Conference on*, June 2013, pp. 682–686.

[29] T. Arsan, "Review of bandwidth estimation tools and application to bandwidth adaptive video streaming," in *High Capacity Optical Networks and Enabling Technologies (HONET), 2012 9th International Conference on*, Dec 2012, pp. 152–156.

[30] K. Miller, E. Quacchio, G. Gennari, and A. Wolisz, "Adaptation algorithm for adaptive streaming over HTTP," in *Packet Video Workshop (PV), 2012 19th International*, May 2012, pp. 173–178.

[31] C. Liu, I. Bouazizi, and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proceedings of the Second Annual ACM Conference on Multimedia Systems*, ser. MMSys '11. New York, NY, USA: ACM, 2011, pp. 169–174. [Online]. Available: http://doi.acm.org/10.1145/1943552.1943575

[32] Adobe, "Adobe HTTP Dynamic Streaming," Retrieved 4.3.2014. [Online]. Available: http://www.adobe.com/products/httpdynamicstreaming/

[33] Microsoft, "IIS smooth streaming transport protocol," Retrieved 4.3.2014. [Online]. Available: http://go.microsoft.com/?linkid=9682896

[34] R. Pantos and E. W. May, "HTTP Live Streaming, informational internet draft," http://tools.ietf.org/html/draft-pantos-http-live-streaming-06, march 2011.

[35] "Netflix quarterly earnings," http://ir.netflix.com/results.cfm, [Online; accessed 17-April-2013].

[36] "Microsoft Silverlight," Retrieved 4.3.2014. [Online]. Available: http://www.microsoft.com/silverlight/

[37] V. Adhikari, Y. Guo, F. Hao, M. Varvello, V. Hilt, M. Steiner, and Z.-L. Zhang, "Unreeling Netflix: Understanding and improving multi-CDN movie delivery," in *Proc. IEEE INFOCOM*, 2012, pp. 1620 –1628.

[38] M. Fiedler, T. Hoßfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *Network, IEEE*, vol. 24, no. 2, pp. 36–41, 2010.

[39] R. Serral-Gracià, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, and X. Masip-Bruin, "An overview of quality of experience measurement challenges for video applications in IP networks," in *Proc. International Conference on Wired/Wireless Internet Communications (WWIC)*, 2010, pp. 252–263.

[40] Z. Wang, L. Lu, and A. Bovik, "Video quality assessment using structural distortion measurement," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 3, 2002, pp. III–65–III–68 vol.3.

[41] Y. Chen, K. Wu, and Q. Zhang, "From QoS to QoE: A survey and tutorial on state of art, evolution and future directions of video quality analysis," *Communications Surveys Tutorials, IEEE*, 2014.

[42] M. Saxena, U. Sharan, and S. Fahmy, "Analyzing video services in Web 2.0: A global perspective," in *Proc. International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, 2008, pp. 39–44.

[43] S. S. Krishnan and R. K. Sitaraman, "Video stream quality impacts viewer behavior: Inferring causality using quasi-experimental designs," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, ser. IMC '12. New York, NY, USA: ACM, 2012, pp. 211–224. [Online]. Available: http://doi.acm.org/10.1145/2398776.2398799

[44] T. Hoßfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial delay vs. interruptions: Between the devil and the deep blue sea," in *Quality of Multimedia Experience (QoMEX), 2012 Fourth International Workshop on*, July 2012, pp. 1–6.

[45] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz, "Quantification of YouTube QoE via crowdsourcing," in *Proc. IEEE International Symposium on Multimedia (ISM)*, 2011, pp. 494 –499.

[46] Y. Qi and M. Dai, "The effect of frame freezing and frame skipping on video quality," in *Proc. International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, pp. 423–426.

[47] B. Lewcio, B. Belmudez, T. Enghardt, and S. Moller, "On the way to high-quality video calls in future mobile networks," in *Proc. Quality of Multimedia Experience (QoMEX)*, 2011, pp. 43–48.

[48] M. Montenovo, A. Perot, M. Carli, P. Cicchetti, and A. Neri, "Objective quality evaluation of video services," *Proceedings of the Third International Workshop on Video Processing and Quality Metrics for Consumer Electronics.*, 2006.

[49] S. Winkler, A. Sharma, and D. McNally, "Perceptual video quality and blockiness metrics for multimedia streaming applications," in *in Proceedings of the International Symposium on Wireless Personal Multimedia Communications*, 2001, pp. 547–552.

[50] W. Lin and C.-C. J. Kuo, "Perceptual visual quality metrics: A survey," *Journal of Visual Communication and Image Representation*, vol. 22, no. 4, pp. 297 – 312, 2011.

[51] U. Engelke and H.-J. Zepernick, "Perceptual-based quality metrics for image and video services: A survey," in *Next Generation Internet Networks, 3rd EuroNGI Conference on*, May 2007, pp. 190–197.

[52] J. You, U. Reiter, M. M. Hannuksela, M. Gabbouj, and A. Perkis, "Perceptual-based quality assessment for audio-visual services: A survey," *Image Commun.*, vol. 25, no. 7, pp. 482–501, Aug. 2010.

[53] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang, "Understanding the impact of video quality on user engagement," in *Proc. ACM SIGCOMM*, vol. 41, 2011, pp. 362–373.

[54] M. Sidib, H. Koumaras, I. Kofler, A. Mehaoua, A. Kourtis, and C. Timmerer, "A novel monitoring architecture for media services adaptation based on network QoS to perceived QoS mapping," *Signal, Image and Video Processing*, vol. 2, no. 4, pp. 307–320, 2008.

[55] O. Oyman and S. Singh, "Quality of experience for http adaptive streaming services," *Communications Magazine, IEEE*, vol. 50, no. 4, pp. 20–27, April 2012.

[56] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *Communications Surveys Tutorials, IEEE*, 2014.

[57] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *Communications Magazine, IEEE*, vol. 46, no. 2, pp. 78–84, February 2008.

[58] F. Yang and S. Wan, "Bitstream-based quality assessment for networked video: a review," *Communications Magazine, IEEE*, vol. 50, no. 11, pp. 203–209, November 2012.

[59] S. Chikkerur, V. Sundaram, M. Reisslein, and L. Karam, "Objective video quality assessment methods: A classification, review, and performance comparison," *IEEE Transactions on Broadcasting*, vol. 57, no. 2, pp. 165–182, June 2011.

[60] A. C. Dalal and E. Perry, "A new architecture for measuring and assessing streaming media quality," in *Proc. of PAM*, 2003.

[61] A. Csizmar Dalal, A., D. Musicant, J. Olson, B. McMenamy, S. Benzaid, B. Kazez, and E. Bolan, "Predicting user-perceived quality ratings from streaming media data," in *Proc. IEEE International Conference on Communications (ICC)*, 2007, pp. 65 –72.

[62] A. Dalal and K. Purrington, "Discerning user-perceived media stream quality through application-layer measurements," in *Proc. International Conference on Multimedia Services Access Networks (MSAN)*, 2005, pp. 44 – 48.

[63] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang, "Inferring the QoE of HTTP video streaming from user-viewing activities," in *Proc. ACM SIGCOMM workshop on Measurements Up the Stack (W-MUST)*, 2011, pp. 31–36.

[64] R. Mok, E. Chan, and R. Chang, "Measuring the quality of experience of HTTP video streaming," in *Proc. IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2011, pp. 485 –492.

[65] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang, "A case for a coordinated internet video control plane," in *Proc. ACM SIGCOMM*, 2012, pp. 359–370.

[66] P. Juluri, L. Plissonneau, and D. Medhi, "Pytomo: A tool for analyzing playback quality of YouTube videos," in *Proc. International Teletraffic Congress (ITC)*, 2011, pp. 304–305, (poster paper).

[67] L. Plissonneau, E. Biersack, and P. Juluri, "Analyzing the impact of YouTube delivery policies on user experience," in *Proc. International Teletraffic Congress (ITC)*, 2012, pp. 1–8.

[68] P. Juluri, L. Plissonneau, Y. Zeng, and D. Medhi, "Viewing YouTube from a Metropolitan Area: What Do Users Accessing from Residential ISPs Experience?" in *Proc. IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2013.

[69] S. Jumisko-Pyykk, V. Kumar, M. V, M. Liinasuo, and M. Hannuksela, "Acceptance of audiovisual quality in erroneous television sequences over a DVB-H channel," *Proc. International Workshop in Video Processing and Quality Metrics for Consumer Electronics*, 2006.

[70] M. Claypool and J. Tanner, "The effects of jitter on the perceptual quality of video," in *Proc. ACM Multimedia Conference*, 1999, pp. 115–118.

[71] O. Teyeb, T. B. Sørensen, P. Mogensen, and J. Wigard, "Subjective evaluation of packet service performance in UMTS and heterogeneous networks," in *Proc. ACM international workshop on Quality of service & security for wireless and mobile networks (Q2SWinet)*, 2006, pp. 95–102.

[72] S. Rugel, T. Bauschert, T. M. Knoll, and M. Eckert, "A network-based method for measurement of Internet video streaming quality," in *ITG-FG 5.2.3 - Next Generation Networks*, 2011.

[73] Tstat, "Tstat: TCP statistic and analysis tool," Retrieved: 4.3.2014. [Online]. Available: http://tstat.tlc.polito.it/index.shtml

[74] S. Latr, N. Staelens, P. Simoens, B. D. Vleeschauwer, W. V. de Meerssche, F. D. Turck, B. Dhoedt, P. Demeester, S. V. den Berghe, R. Huysegems, and N. Verzijp, "On-line Estimation of the QoE of Progressive Download Based Services in Multimedia Access Networks." in *Proc. International Conference on Internet Computing*, 2008.

[75] B. D. Vleeschauwer, W. V. D. Meerssche, P. Simoens, F. D. Turck, B. Dhoedt, P. Demeester, K. Struyve, T. V. Caenegem, E. Gilon, H. Dequeker, and E. Six, "Enabling autonomic access network QoE management through TCP connection monitoring." in *Proc. IEEE Workshop on Autonomic Communications and Network Management of (ACNM)*, 2007, pp. 56–63.

[76] R. Huysegems, B. De Vleeschauwer, K. De Schepper, C. Hawinkel, T. Wu, K. Laevens, and W. Van Leekwijck, "Session reconstruction for HTTP adaptive streaming: Laying the foundation for network-based qoe monitoring," in *Proc. IEEE International Workshop on Quality of Service (IWQoS)*, 2012, pp. 1–9.

[77] R. Schatz, T. Hoßfeld, and P. Casas, "Passive YouTube QoE Monitoring for ISPs," in *Proc. International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS),*, 2012, pp. 358–364.

[78] ITU-T Recommendation P.910, "Subjective video quality assessment methdods for multimedia applications," Recommendations of the ITU,Telecommunications Sector.

[79] Y. Wang, M. Claypool, and Z. Zuo, "An empirical study of RealVideo performance across the Internet," in *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement (IMW)*, 2001, pp. 295–309.

[80] Y. Wang and M. Claypool, "RealTracer–tools for measuring the performance of RealVideo on the Internet," *Multimedia Tools Appl.*, vol. 27, no. 3, pp. 411–430, Dec. 2005.

[81] Amazon, "Amazon Mechanical Turk," Retrieved: 4.3.2014. [Online]. Available: https://www.mturk.com/mturk/welcome

[82] Microworkers, "Microworkers," Retrieved 4.3.2014. [Online]. Available: {http://microworkers.com/}

[83] T. Hoßfeld, R. Schatz, and U. Krieger, "Qoe of youtube video streaming for current internet transport protocols," in *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, ser. Lecture Notes in Computer Science, K. Fischbach and U. Krieger, Eds. Springer International Publishing, 2014, vol. 8376, pp. 136–150. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-05359-2_10

[84] D. C. Robinson, Y. Jutras, and V. Craciun, "Subjective video quality assessment of HTTP adaptive streaming technologies," *Bell Labs Technical Journal*, vol. 16, no. 4, pp. 5–23, 2012. [Online]. Available: http://dx.doi.org/10.1002/bltj.20531

[85] F. Yang, S. Wan, Q. Xie, and H. R. Wu, "No-reference quality assessment for networked video via primary analysis of bit stream," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 11, pp. 1544–1554, Nov 2010.

[86] S. Khirman and P. Henriksen, "Relationship between Quality-of-service and Quality-of-Experience for public Internet service," in *Proc. 3rd Passive Active Measurement Workshop*, 2002.

[87] J. Shaikh, M. Fiedler, and D. Collange, "Quality of experience from user and network perspectives," *annals of telecommunications - annales des tlcommunications*, vol. 65, no. 1-2, pp. 47–57, 2010.

[88] S. Aroussi, T. Bouabana-Tebibel, and A. Mellouk, "Empirical QoE/QoS correlation model based on multiple parameters for VoD flows," in *Proc. IEEE Global Communications Conference (GLOBECOM)*, 2012, pp. 1963–1968.

[89] C. Alberti, D. Renzi, C. Timmerer, C. Mueller, S. Lederer, S. Battista, and M. Mattavelli, "Automated QoE evaluation of dynamic adaptive streaming over HTTP," in *Proc. Quality of Multimedia Experience (QoMEX)*, 2013, pp. 58–63.

[90] B. Shao, D. Renzi, P. Amon, G. Xilouris, N. Zotos, S. Battista, A. Kourtis, and M. Mattavelli, "An adaptive system for real-time scalable video streaming with end-to-end QoS control," in *Proc. 11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2010, pp. 1–4.

[91] ITU-T P.1201 (2012) Amd. 2, "Amendment 2: New appendix iii - use of p.1201 for non-adaptive, progressive download type media streaming," Recommendations of the ITU, Telecommunications Sector, 12 2013.

[92] D. Vera, P. Rodrguez-Bocca, and G. Rubino, "QoE monitoring platform for video delivery networks," in *IP Operations and Management*, ser. Lecture Notes in Computer Science, D. Medhi, J. Nogueira, T. Pfeifer, and S. Wu, Eds. Springer Berlin Heidelberg, 2007, vol. 4786, pp. 131–142.

[93] A. Csizmar Dalal, E. Kawaler, and S. Tucker, "Towards Real-Time stream quality prediction: Predicting video stream quality from partial stream information," in *Quality of Service in Heterogeneous Networks*, 2009, vol. 22, pp. 20–33.

[94] H. French, J. Lin, T. Phan, and A. Dalal, "Real time video QoE analysis of RTMP streams," in *Proc. 30th International Performance Computing and Communications Conference (IPCCC)*, 2011, pp. 1 –2.

[95] W. Cherif, A. Ksentini, D. Negru, and M. Sidibe, "A-PSQA: Efficient real-time video streaming QoE tool in a future media internet context," in *IEEE International Conference on Multimedia and Expo (ICME)*, 2011, pp. 1–6.

[96] K. Singh, Y. Hadjadj-Aoul, and G. Rubino, "Quality of experience estimation for adaptive HTTP/TCP video streaming using H.264/AVC," in *Proc. Consumer Communications and Networking Conference (CCNC)*, 2012, pp. 127–131.

[97] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "Developing a predictive model of quality of experience for internet video," in *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 339–350.

[98] A. L. Vlado Menkovski, Georgios Exarchakos, "Online learning for quality of experience management," in *Proc. 19th Machine Learning conference of Belgium and The Netherlands*, 2010.

[99] F. Agboma and A. Liotta, "QoE-aware QoS management," in *Proc. 6th International Conference on Advances in Mobile Computing and Multimedia (MoMM)*, 2008, pp. 111–116.

[100] F. Yang, J. Song, S. Wan, and H. R. Wu, "Content-adaptive packet-layer model for quality assessment of networked video services," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 6, no. 6, pp. 672–683, Oct 2012.

[101] Netflix, "The ISP Speed Index From Netflix," Retrieved: 4.3.2014. [Online]. Available: http://ispspeedindex.netflix.com/

[102] L. Skorin-Kapov and M. Varela, "A multi-dimensional view of qoe: the arcu model," in *MIPRO, 2012 Proceedings of the 35th International Convention*, May 2012, pp. 662–666.

[103] A. Moorthy, L. K. Choi, A. Bovik, and G. de Veciana, "Video quality assessment on mobile devices: Subjective, behavioral and objective studies," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 6, no. 6, pp. 652–671, Oct 2012.

[104] S. Xiang, L. Cai, and J. Pan, "Adaptive scalable video streaming in wireless networks," in *Proceedings of the 3rd Multimedia Systems Conference*, ser. MMSys '12. New York, NY, USA: ACM, 2012, pp. 167–172.

[105] A. Bokani, M. Hassan, and S. Kanhere, "Http-based adaptive streaming for mobile clients using markov decision process," in *Packet Video Workshop (PV), 2013 20th International*, Dec 2013, pp. 1–8.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/COMST.2015.2401424, IEEE Communications Surveys & Tutorials

Parikshit Juluri received his B.Tech in Electrical and Electronics Engineering from Jawaharlal Nehru Technological University, Hyderabad, India in 2007, MS degree in Electrical Engineering from University of Missouri-Kansas City, USA in 2008. He is currently working towards the Ph.D degree in Networking and Telecommunications at the University of Missouri-Kansas City, USA. His interests include QoE of video streaming applications, performance, measurement and network management.

Venkatesh Tamarapalli received his Ph.D. degree in computer science and engineering from the Indian Institute of Technology Madras in 2009. Since then, he has been working as an Assistant Professor at the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati. His research interests include performance analysis of high-speed networks, sensor networks, and multimedia streaming.

Deep Medhi is a Curators' Professor in the Department of Computer Science & Electrical Engineering at the University of Missouri-Kansas City, USA, and a honorary professor in the Department of Computer Science & Engineering at the Indian Institute of Technology–Guwahati, India. He received B.Sc. in Mathematics from Cotton College, Gauhati University, India, M.Sc. in Mathematics from the University of Delhi, India, and his Ph.D. in Computer Sciences from the University of Wisconsin-Madison, USA. Prior to joining UMKC in 1989, he was a member of the technical staff at AT&T Bell Laboratories. He served as an invited visiting professor at the Technical University of Denmark, a visiting research fellow at Lund Institute of Technology, Sweden, and as an International Research Visitor at the State University of Campinas, Brazil. He has also been a visitor at Princeton University, and Université Pierre et Marie Curie (UPMC), Paris, France. As a Fulbright Senior Specialist, he was a visitor at Bilkent University, Turkey, and Kurukshetra University, India. He is the Editor-in-Chief of Springer's *Journal of Network and Systems Management*, and is on the editorial board of *IEEE/ACM Transactions on Networking*, *IEEE Transactions on Network and Service Management*, and *IEEE Communications Surveys & Tutorials*. He is co-author of the books, *Routing, Flow, and Capacity Design in Communication and Computer Networks* (2004) and *Network Routing: Algorithms, Protocols, and Architectures* (2007), both published by Morgan Kaufmann Publishers, an imprint of Elsevier Science. His research interests are multi-layer networking, network virtualization, data center optimization, and network routing, design, and survivability. His research has been funded by NSF, DARPA, and industries.