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

Multi-access Edge Computing video analytics of ITU-T P.1203 Quality of Experience for streaming monitoring in dense client cells

Roberto Viola[1] · **Mikel Zorrilla**[2] · **Pablo Angueira**[3] · **Jon Montalbán**[4]

Received: date / Accepted: date

Abstract 5G promises unseen network rates and capacity. Furthermore, 5G ambitions agile networking for specific service traffic catalysing the application and network symbiosis. Nowadays, the video streaming services consume lots of networking assets and produce high dynamics caused by players mobility meaning a challenging traffic for network management. The Quality of Experience (QoE) metric defined by ITU-T P.1203 formulates the playback issues related to widely employed Dynamic Adaptive Streaming over HTTP (DASH) technologies based on a set of parameters measured at the video player. Monitoring the individual QoE is essential to dynamically provide the best experience to each user in a cell, while video players compete to enhance their individual QoE and cause high network performance dynamics. The edge systems have a perfect position to bring live coordination to dense and dynamic environments, but they are not aware of QoE experienced by each video player. This work proposes a mechanism to assess QoE scores from network dynamics at the cell and manifests of DASH streams without an explicit out of band messaging from video players to edge systems. Hence, this paper implements an edge proxy, independent from video servers and players, to monitor and estimate QoE providing the required information to later decide streaming qualities in a coordinated manner in a dense client cell. Its lightweight computation design provides real-time and distributed processing of local sessions. To check its validity, a WiFi setup has been exercised where the accuracy of the system at the edge is checked by assessing the ITU-T P.1203 QoE of individual players.

Keywords 5G · MEC · MOS · MPEG-DASH · QoE

R. Viola
E-mail: rviola@vicomtech.org
M. Zorrilla
E-mail: mzorrilla@vicomtech.org
P. Angueira
E-mail: pablo.angueira@ehu.eus
J. Montalbán
E-mail: jon.montalban@ehu.eus
[1]Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), Mikeletegi 57, 20009 San Sebastián, Spain. PhD Candidate at UPV/EHU.
[2]Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), Mikeletegi 57, 20009 San Sebastián, Spain.
[3]University of the Basque Country (UPV/EHU), Department of Communications Engineering, 48013 Bilbao, Spain.
[4]University of the Basque Country (UPV/EHU), Department of Electronic Technology, 20018 San Sebastián, Spain.

1 Introduction

Popularity of video streaming platforms to consume live sports events and over-the-top (OTT) services like Netflix, when combined with mobility and city contexts, result in a complex traffic congestion scenario to be managed. Specifically, when video players share the same path, they try to individually enhance their Quality of Experience (QoE) considering instant available bandwidth and display setup. As the cell gets more subscribers and the Base Station (BS) serves more traffic sessions, dynamics turn higher and available bandwidth estimation at client side gets coarser, fuelling video player fluctuations on requested bandwidth which affect the QoE [15]. Therefore, when congestion at the radio link comes into place, a dense client cell means a pool of clients competing for the available network assets. This competition leads to player instability, unfairness between players, and bandwidth under-utilization [1].

Maximizing customer satisfaction through QoE-based networking is a crucial challenge for video streaming services. As most of the traffic flooding the networks comes from video streaming services, Mobile Network Operators (MNOs) need to configure the network according to cost-effective policies to cope with video demand while providing best QoE with the available resources. To this end, MNOs need to monitor the QoE without access to video streaming servers, Content Delivery Network (CDN) systems or video player applications.

In the upcoming horizon 5G aims to improve current Key Performance Indicators (KPIs), such as traffic density, volume and latency. It will flatten barriers to allow network operators and/or 3rd parties to provide dynamic networking performance to respond to specific traffic demands [6]. To this end, network management frameworks analyse network assets and traffic features to adapt network setup to concurrent traffic needs. This means a revolution on networks with a digital transformation which leaves behind the concept one-fits-all and moves to a specialized and mutable network which dynamically changes its configuration over a common bare-metal infrastructure.

In the line of catalysing synergies of network and traffic from services, edge computing is gaining relevance due to benefits of local processing. Enabled by the transition of networking assets to cloud infrastructures, the provision of hosting at the edge means new revenues for MNOs. Furthermore, the location of some parts or entire systems of a service closed to the clients enables advanced possibilities considering the user environment. The application of analytics at the edge to optimise the video streaming traffic is a use case compiled in the ETSI working documents as a key representative of Multi-access Edge Computing (MEC) application [5].

The proposed 5G MEC-enabled proxy estimates the end users' QoE according to ITU-T P.1203 [28] and derived from monitored QoS metrics in the radio cell to allow a decision by other edge services operating streaming qualities in a coordinated manner in a dense client cell. Thus, the proposed system catalyses the use of Video Streaming Analytics at the network edge to realise the network/application symbiosis [5].

The major contributions of this paper can be summarized as follows:

- A novel mechanism to estimate ITU-T P.1203 [28] QoE values from network dynamics at the cell and parameters parsed from manifests of DASH streams without an explicit out of band messaging from video players to edge system.
- A lightweight implementation of an edge computing proxy, independent from video servers and players, to monitor and assess in real-time ITU-T P.1203 QoE values for each local session.
- The analysis of the accuracy of the proposed system to assess the ITU-T P.1203 QoE of individual players in a WiFi-based experimentation setup and a dense client cell. To this end, two scenarios with different levels of concurrency and congestion are performed.

The rest of this paper is organized as follows. First, the related work is in Section 2. The MEC system model and architecture and the streaming analytics service is in Section 3. Furthermore, the system performance analysis, defined metrics, the evaluation setup and the results are described in Section 4. Finally, the Section 5 concludes the paper.

2 Related Work

QoE of streamed video contents is intrinsically related to customers' satisfaction. Mean Opinion Score (MOS) compiles the results of subjective evaluations over a variety of representative spectators/audience surveyed to score perceived quality, ranging 1 to 5 values [27]. However, this surveys process results in high expensive costs since it cannot be automated, as it is difficult to be repeated as updates on operational parameters are applied and it is complex to process the results in real-time.

Therefore, research community investigates models to map subjective distortion to objective QoS and application metrics which can be automatically monitored and computed in real-time producing actionable data [14]. These subjective quality models change as the streaming technologies evolves. Nowadays, HTTP Adaptive Streaming (HAS) technologies are widely used for live and on-demand video streaming services. Then, QoE models shift from pixel-level distance evaluation of received frames from the original served, such as Peak Signal to Noise Ratio (PSNR) based on Mean Squared Error (MSE) or Structural Similarity Index (SSIM), to content-agnostic models with sophisticated parametric equations comprising throughput, frequency of bitrate changes from available representations and buffering duration. These models better reflect quality level from a video streaming perspective, they are lightweight and can be processed in real-time. Per-title encoding techniques [3] are widely employed by major video services, adapting the employed bitrate to the image complexity to save bandwidth costs, so representation bitrate has a direct translation to image fidelity and pixel-level distortion is not required anymore.

Objective QoE models [18] are widely employed at media player side to enable decision making on representation/bitrate selection to accommodate the demand to the available network resources and maximise the local QoE. Some of the QoE models at player side are heavily dependent on buffer metrics [12]. At the server side, buffer information is not available and QoE models usually employ HTTP delivery information [19]. Some server-side models exploit also lower-level information, such as TCP connections [20], or combine both TCP and HTTP information [21]. The ITU-T P.1203, whose last version was published in October 2017, is a standardized model for QoE evaluation of adaptive audio and video streams including video and audio quality and buffering issues, where quality is closely related to representation bitrate of each played media segment, number of representation switches, number of stalling events and their duration.

Dense client cells are tough environments where the autonomous decisions from video players may produce unfair, biased, unsteady and inefficient use of the radio link. There, intrinsic ON-OFF periods of HAS technologies may cause oscillations on the experienced network throughput and, accordingly, on the selected bitrate[1]. Client-side algorithms, selecting representation to be downloaded and played, have been proposed to detect and compensate bitrate oscillations [23]. However, the lack of coordination across the video players may turn available network resources into stochastic and bitrate selections into erratic as the density of sessions or the audience dynamics mutate quicker. There, a server-assisted solution to compensate oscillations allows improving video player bitrate selection [2], it does not need modifications on client-side algorithms to work, but it does not solve uncoordinated behaviour across the players. Here, a more coordinated approach is needed to apply common policies to all the local subscribers. Evolved from legacy Real-Time Transport Control Protocol (RTCP), which periodically produces QoS statistics

to allow the server to change operational ranges, server-side QoE estimation mechanisms [16] have been also employed to manage video delivery and provision platform resources.

However, once different representations are available at the server or CDN to match display preferences and changeable network conditions due to mobility, server-side approach is less scalable and slower to react than an edge system.

5G brings several technologies to leap density KPIs. At the Radio Access Network (RAN), massive multiple-input multiple-output (MIMO) technology, millimeter wave communications, and small-, micro-, pico-cell concepts have appeared to improve the spectrum efficiency raising communication throughput while saving energy [10]. Concerning the network backhaul and core, Software-defined Network (SDN) and Network Function Virtualization (NFV) paradigms adopted in the ETSI stack are the pillars to realise agile and smart networking for delivering traffic from different services to different users and under specific level of performance [25]. However, in the case of dense client cells, it is the MEC architecture which will make the difference to monitor QoE, apply policies to video delivery and coordinate video players' decisions transparently by managing content manifests [8].

Different systems have exploited the exceptional position of MEC architectures to host QoS/QoE monitoring services. Some of them to apply video transcoding operations [4], to perform in network caching [31] or to negotiate the wireless interface to use [33] to satisfy a target QoE. However, they add new systems processing provided data at the network edge losing transparency for media servers and players meaning messaging overheads [4]. From the scalability perspective, proposed systems tend to perform heavy processing such as Random Neural Network (RNN) to assess the QoE at the viewer [4] or perform statistical mapping of MOS from QoS metrics [33]. More focused in 5G MEC architectures, lightweight monitoring and processing systems embodied in a Virtual Network Function (VNF) which infers Dynamic Adaptive Streaming over HTTP (MPEG-DASH) clients' QoE, independent from the media server and players, means a viable service to process real-time QoE metrics in a dense client environment [9]. Here, objective metrics which influence QoE, such as initial playout delay and the rebuffering number and duration metrics, are estimated from requests timestamps, as the representation bitrate switches can be directly obtained by the requested URL from the media manifest. This approach still relies on a LTE network for testing, meaning that the MEC service is not actually running at network edge, but at the LTE Core (Evolved Packet Core) location. Our approach is similar, expanding QoE evaluation to include subjective results estimation through the QoE model defined by ITU-T P.1203 [28] and assessing actual bitrate of each requested segment, as the nominal bitrate of the manifest is coarse affecting inference accuracy. We add a novel mechanism to estimate ITU-T P.1203 QoE by analysing network dynamics and parsing manifest parameters of DASH streams without an explicit out of band messaging from video players to edge system. Moreover, we deploy our MEC service on a WiFi access point, according to MEC architecture defined by ETSI [7].

3 Edge Analytics Service

3.1 Architecture

The analysis of QoE is crucial for decision making systems enforcing or boosting video streaming services. Video players apply algorithms to adapt bitrate demand to the changeable network performance to autonomously enforce its QoE. However, in dense client cells a more coordinated approach across the sessions sharing the radio link is needed to avoid individual competition for available network assets [1], thus fostering steady and smooth playback. Edge systems have an exceptional position to enforce

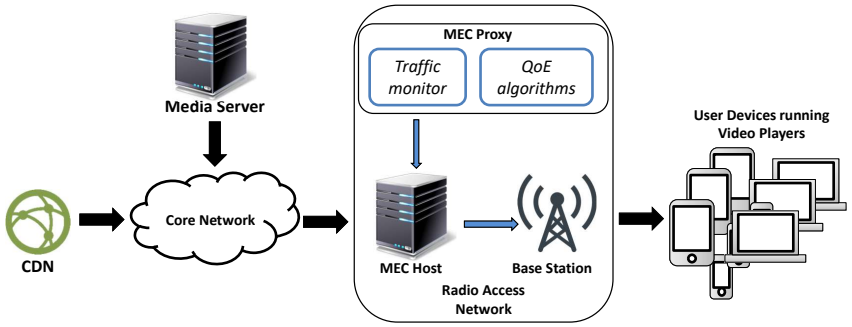


Fig. 1: Architecture of MEC-based QoE estimation approach.

or enhance QoE of video streaming services applying common policies to all the users in a cell while bringing scalable, transparent and zero latency processing of local metrics [5].

This section describes the proposed MEC proxy according to ETSI standardized MEC architecture [5]. The Proxy performs the estimation of the standard ITU-T P.1203 [28] parameters as depicted in Figure 1. To keep video players and server architectures, no additional communications are introduced inside the delivery chain to capture and report players' metrics. Thus, communication overheads are avoided. This means that the MEC systems do not have direct access to all the factors which comes into play in the equation of the QoE which are measured at the media player. This manner, it is necessary to estimate the QoE by inferring the values of involved factors.

To achieve it, the MEC Proxy is deployed at the MEC Host and monitors all the traffic exchanged at RAN between the video players, media server and CDN [5]. During this operation, it can assess specific metrics related to the streaming session from two different activities:

- download of video manifest from media server, which includes different metadata for each representation such as resolutions, nominal bitrates, or language for the different media streams;
- player's representation selection, when matching the HTTP segment requests with representations available in the manifest.

When a streaming session is started, the MEC Proxy downloads the video manifest and serves it to the player. The MEC Proxy can locally analyse the manifest to list the available representations and their features since the manifest includes information for all the available video, audio and subtitles tracks. Later, when the session is already playing, the video player must retrieve segments from the media server or CDN through HTTP requests. In this case, the MEC Proxy can act in two different ways:

1. MEC Proxy can parse the information contained inside the HTTP URL and header as well as obtain request/response timestamps;
2. MEC Proxy can analyse all the HTTP packets payload, meaning an active analysis of the media stream.

Our solution is on top of first option to reduce the processing demands of the Proxy at the edge and favour a more scalable solution. In any case, it is security/privacy sensitive since standard encryption mechanisms of HAS standards just encrypts the payload of media segments to avoid unauthorized playback, so parsing the media manifest and process the HTTP requests endpoints are allowed.

The general communication is presented in Figure 2. First, the HTTP Proxy needs to identify each streaming session to link captured timestamps and QoE factors from each request and response to a

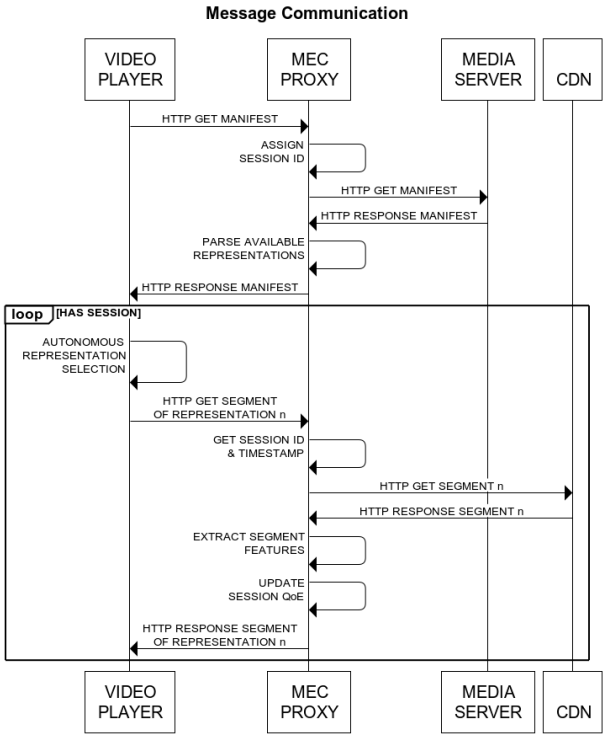


Fig. 2: Message communication.

specific item to infer its QoE metrics. To achieve it, the MEC Proxy generates a random ID when the first HTTP request from an IP requesting a manifest is received. Therefore, the MEC Proxy maintains a list with all the active streaming sessions. A session is considered expired when no HTTP requests are received during the duration of two segments. Then, inactive sessions are removed from the list.

When the manifest is received by the video player it decides which representation better fits with the display features, the user preferences, the service subscription, and the network performance. This decision is revised by the video player for each segment to request. Accordingly, it starts to request a specific representation bitrate. The MEC Proxy stores the timestamps to later process request pace to detect any buffering issue when comparing nominal segment duration and time elapsed between requests. Then the MEC Proxy performs the CDN request and analyse some high-level features from the provided segment to accurately estimate the bitrate which may differ from the nominal value declared at the manifest. From that information, the MEC Proxy is can infer an estimated ITU-T P.1203 QoE. Last, the segment is delivered to the video player.

The list of sessions and the estimated QoE, updated for each segment request, is available for other systems at the edge or in the cloud to facilitate the enforcement or enhancement of video streaming in a dense client cell concerning QoE.

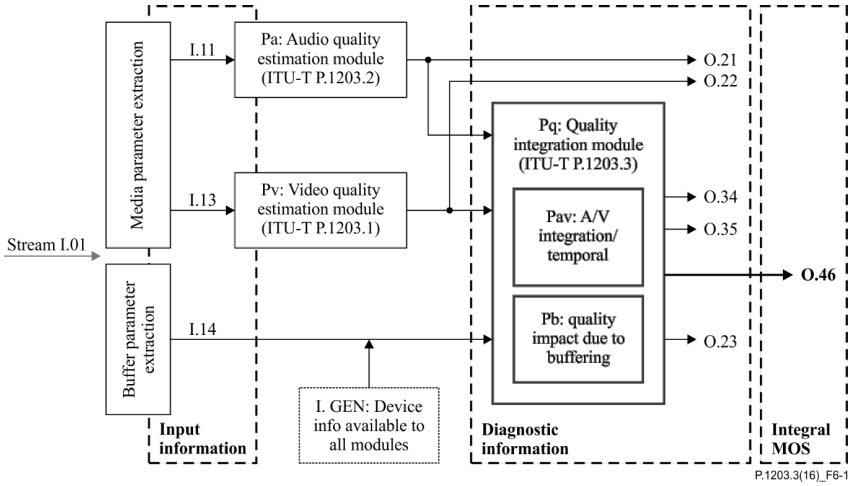


Fig. 3: Building blocks of the ITU-T P.1203 model. Source: [28], Figure 1.

3.2 QoE Estimation Algorithm

ITU-T P.1203 [28] describes a model for monitoring media session quality while delivering content through HAS technologies. The building block of the ITU-T P.1203 model is presented in Figure 3.

The ITU-T P.1203 model receives media stream information and playback device features to generates the inputs (*I.11, I.13, I.14, I.GEN*) for the internal modules (*Pa, Pv, Pq, Pav, Pb*). The model generates the following input signals:

- I.GEN: Playback display resolution and device type.
- I.11: Information on played audio segments, including audio codec and representation features.
- I.13: Information on played video segments, including video codec and representation features.
- I.14: Stalling event information, including stalling start time and its duration.

The inputs may be extracted or estimated in different ways since the ITU-T P.1203 does not provide information on *Buffer parameter extraction* and *Media parameter extraction* modules. The internal modules process the inputs signals to achieve several output QoE scores:

- O.21 and O.22: *Pa* and *Pv* modules provide one score per sampling interval for audio and video, respectively.
- O.34, O.35 and O.23: *Pav* and *Pb* modules provides cumulative scores for audio-visual and buffering, respectively.
- O.46: *Pq* module integrate audio-visual and buffering scores to provide the overall score.

All the outputs have 1-5 quality scale, where "1" means "bad" quality and "5" means "excellent" quality, according to MOS specifications [27].

The ITU-T P.1203 also establishes 4 modes of operation (mode 0 to 3) [28]. Mode 0 employs only content metadata. All the other modes work only with unencrypted content to acquire information from the media stream. Modes 2 and 3 also require decoding it. Consequently, if we employ mode 1-3 at the MEC Proxy, it may cause security issues. For this reason, we employ mode 0 for our MEC Proxy. Moreover, mode 0 is also the less intensive in terms of processing.

A software implementation of ITU-T P.1203 standard internal modules is provided in [13]. The software implements the internal modules [30] according to the ITU-T P.1203 and provides customized *Media parameter extraction* and *Buffer parameter extraction* modules (the ITU-T P.1203 does not specifies these modules). These customized modules are useful to generate compliant inputs signals and evaluate the internal modules, but they are limited for working with locally stored video files. So, they could not be used while streaming a content. All the modules provided by this software implementation are also capable to analyse the media content through any of the four available modes [26].

To feed the ITU-T P.1203 while streaming a content, we have designed and implemented our custom solutions to generate the inputs. *I.GEN* can be easily known by analysing the header of the HTTP requests since it contains a User-Agent field [22] that allows the recognition of the HTTP client type (mobile or desktop device, browser or application, etc.), while keeping anonymous the identity of the user. To extract the remain input signals, we design both *Media parameter extraction* and *Buffer parameter extraction* modules which execute Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Media parameter extraction

function MEDIAPARAMETER(Manifest, segment_n, $\{\text{segment}\}_{n-1}$) ▷ for each downloaded segment
Input: Manifest ▷ media manifest
Input: segment_n ▷ current downloaded segment
Input: $\{\text{segment}\}_{n-1}$ ▷ session information including last segment
Output: $\{\text{segment}\}_n$ ▷ session information updated with current segment
d_n, res_n, fps_n, $\text{codec}_n \leftarrow \text{parseManifest}(\text{Manifest}, \text{segment}_n)$ ▷ segment duration, resolution, framerate and codec
$\text{size}_n \leftarrow \text{getBitSize}(\text{segment}_n)$ ▷ current segment size
$\text{bitrate}_n = \frac{size_n}{d_n}$ ▷ current segment bitrate
$\{\text{segment}_n\} = \{\text{d}_n, \text{res}_n, \text{fps}_n, \text{codec}_n, \text{bitrate}_n\}$ ▷ current segment information
$\{\text{segment}\}_n \leftarrow \{\{\text{segment}\}_{n-1}, \{\text{segment}_n\}\}$ ▷ updated session information
end function

The Algorithm 1 is executed for each segment downloaded by the MEC Proxy. It receives the media manifest, the most recent downloaded segment, and the accumulated metadata for the past downloaded segments of a specific session. In the case of employing MPEG-DASH video streaming technology, the manifest would be a Media Presentation Description (MPD). First, specific metadata is captured by matching the manifest and the segment URL, identifying the specific selected representation by the video player for each segment time slot. Second, a more accurate metric in terms of the actual bitrate is captured from the segment size and its nominal duration. Usually, the duration of the segments is fixed as the Group of Pictures (GOP) size is fixed at the encoders to start the segment with a keyframe, so the nominal duration declared in the manifest is accurate. Once all the metadata for the current segment ($\{\text{segment}_n\}$) are collected, i.e., bitrate, duration, resolution, framerate and employed codec, this information is stored for all the session long ($\{\text{segment}\}_n$), updating the series for the last downloaded segment ($\{\text{segment}\}_{n-1}$).

The Algorithm 2 is executed at the MEC Proxy once each segment download is completed by the video player. It estimates stall occurrence and duration without access to video player buffer or playback issues. The MEC Proxy is remotely inferring the playback issues from the timestamps on video player requests. As the video players download a new segment once another has been played, the inter-arrival on video player requests should follow segment duration. Any identified drift likely means a buffering issue. To this end, this function, also executed for each downloaded segment, gets the duration of the segment, the download timestamps of the video player from the segment provided by the MEC Proxy,

Algorithm 2 Buffer parameter extraction

function BUFFERPARAMETER(n, d, t_0, t_n, $\{stall\}_{n-1}$) ▷ for each downloaded segment
Input: n ▷ segment index
Input: d ▷ segment duration
Input: t_0 ▷ first segment download time
Input: t_n ▷ current segment download time
Input: $\{stall\}_{n-1} = \{\{start_0^{stall}, d_0^{stall}\},...,\{start_k^{stall}, d_k^{stall}\}\}$ ▷ session stalls including last segment
Output: $\{stall\}_n$ ▷ session stalls updated with current segment
  $k \leftarrow \{stall\}_{n-1}$ ▷ number of estimated stalls along the session
  $D^{stall} = \sum_{i=0}^{k} d_i^{stall}$ ▷ total duration of estimated stalls along the session
  $t_{playback} = (t_n - t_0) - D^{stall}$ ▷ playback time
  $d_{downloaded} = (n-1)*d$ ▷ total duration of downloaded segments
  $d^{stall} = t_{playback} - d_{downloaded}$ ▷ candidate stall duration
  if $(d^{stall} > 0)$ **then** ▷ check if stalling in the recent segment
    $d_{k+1}^{stall} = d^{stall}$ ▷ record stall duration
    $start_{k+1}^{stall} = t_n - d^{stall}$ ▷ stall start time
    $\{stall_{k+1}\} = \{start_{k+1}^{stall}, d_{k+1}^{stall}\}$ ▷ estimated parameters of new stall (start time and duration)
    $\{stall\}_n = \{\{stall\}_{n-1}, \{stall_{k+1}\}\}$ ▷ update stall series
    $k{+}{+}$ ▷ new stall when playing the current segment
  else
    $\{stall\}_n = \{stall\}_{n-1}$ ▷ unchanged stall information
  end if
end function

and the records of past estimations of stalls for previous segment time slots in the session. First, the total duration of estimated stalls along the session is calculated. Then, the current playback time is measured from the elapsed time from the player's start-up to the last downloaded segment, including the total estimated stall for all the session. The MEC Proxy assumes the first downloaded segment time as the start-up time. It is a reasonable choice since the player starts decoding and displaying the content only when the first segment is completely downloaded. With this assumption, an error could be introduced on the start-up time selection since the proxy cannot measure decoding delay of the first frame. In any case, decoding operation is done in real time, meaning that the maximum error is in the order of tens of milliseconds (1/framerate). Downloading time duration and player's internal buffer size are in the same order of magnitude of segment duration (seconds), which is 100 times higher than the decoding delay. Thus, the error due to decoding delay has a negligible impact on the overall measurement of the start-up time. To calculate the duration of content available at the video player, the algorithm only includes the previously downloaded segments $(n-1)$ excluding the most recent one (n). As this function is evaluated just once the segment has been downloaded by the video player, the most recent segment has not been decoded yet. If the current playback time ($t_{playback}$) is lower than the duration of content available at the video player ($d_{downloaded}$), the video player has buffered content to be played, so stall should not happen. If the playback time is ahead the duration of available content, the stall duration is estimated (d^{stall}), and the start time of this stall is assumed shifting current segment download time (t_n). Last, estimated timestamp and duration of detected stall is stored.

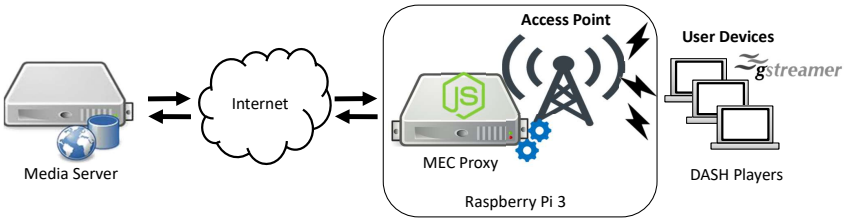


Fig. 4: Testbed.

4 Results

4.1 Experimental setup

To demonstrate the effectiveness of the proposed approach to assess the ITU-T P.1203 [28] QoE scores, we deployed the testbed shown in Figure 4. The experimental setup comprises the following:

- Media server: we use a mirror server located at Polytechnic University of Turin and belonging to D-DASH dataset and infrastructure [17] which provides a Dynamic Adaptive Streaming over HTTP (DASH) standard content.
- MEC proxy and access point: this is a unique node deployed through a Node.js [24] proxy application running on a Raspberry Pi 3 single-board computer [29]. Raspberry Pi 3 includes both computing capabilities to execute edge processing and a WiFi internal module to deploy a local wireless network such to provide connection to WiFi clients. Thus, this design guarantees ultra-low latency, proximity and high bandwidth [7]. This node has Internet access to download the MPD and the corresponding media segments stored at the Media Server which are served to the clients on demand.
- UE node: client node featuring WiFi interface and running several DASH players, based on GStreamer multimedia framework [11], which download the video stream. Players run headless (without GUI) since we are not interested in displaying the content and allowing the client to save computing capabilities.

Node.js application at MEC node employs *http-server* module for acquiring HTTP request and response time and *xml2js* module for parsing the MPD to feed Algorithms 1 and 2. During the experiments, Algorithms 1 and 2 are also implemented inside Node.js application and run in real time to generate the inputs of ITU-T P.1203 model. However, ITU-T P.1203 QoE evaluation is performed offline, after that all the metrics are collected. Thus, it simplifies the process of metrics collection and QoE evaluation, while the QoE results remain valid as the input metrics for the ITU-T P.1203 model do not change.

The dataset at the Media Server includes the Red Bull Playstreet video sequence, which is owned by Red Bull Media House and licensed for scientific purposes. This sequence is encoded for 17 video representations through advanced video coding (H.264/AVC) and 4 dual channel audio representations through advanced audio coding (AAC). Both audio and video are segmented with different segment lengths of 2, 4, 6, 10, and 15 seconds, and multiplexed in ISO MPEG4 files (ISO/IEC 14496-12 - MPEG-4 Part 12). For our experiments, we employed 6 seconds segments as such duration favour accuracy of the proposed solution when delivering segments with enough size in bytes to get a more solid assessment of the network performance while downloading, while the serving latency is still valid for streaming of live events.

The representations range from a resolution of 320 x 240 and 30 fps at 100 kbps to a resolution of 1920 x 1080 and 30 fps at 6000 kbps. As the client-side bitrate adaptation mechanism does not target a specific resolution, then each client struggles to achieve the highest representation bitrate.

We use the outcomes of [32] to model players behaviour. The authors provide an extensive analysis on user behaviour while accessing streaming services. The player inter-arrival time fits a modified version of the Poisson distribution. This means that the players are starting and stopping their sessions (joining and leaving the cell/hotspot) along the experiment according to a Poisson distributed inter-arrival time. Moreover, the duration of streaming sessions of the players is variable and follows the declared sections of 5 (37.44%), 10 (52.55%), or 25 min (75.25%). As a result, the effective number of players employed during the experiments is variable since the model is aleatory. Modelling players inter-arrival time and their streaming session duration according to a real distribution allows to emulate a real media traffic scenario.

To test with different network loads, we also consider two different scenarios where we change the limit of the maximum number of concurrent players:

- Scenario 1: 10 players at a time. Here, no more than 10 players at a time are connected to the WiFi access node and downloading the content.
- Scenario 2: 20 players at a time. The number of players concurrently consuming the video streaming is increased to 20.

We perform a test over each scenario for one hour. It results in 66 players taking part in the first scenario and 144 players participating in the second scenario.

4.2 Evaluation metrics

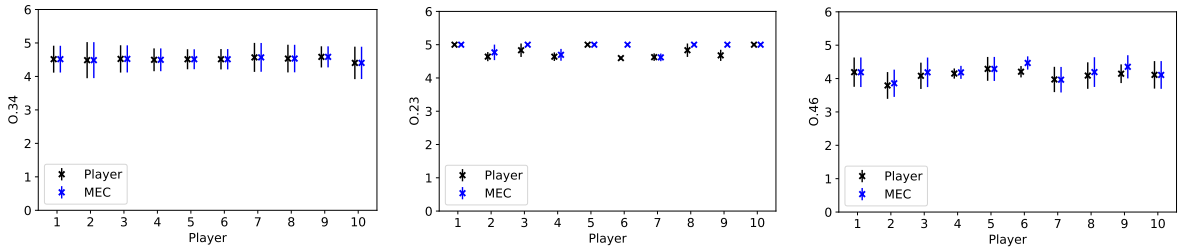
To evaluate the proposed solution, we employ the outputs of the ITU-T P.1203, which provide subjective evaluation of the streaming sessions. Since we are focused on evaluating QoE of video representation, we target the following outputs from the ones analysed by ITU-T P.1203:

- O.34: it provides a video quality score per output sampling interval. The default interval of the ITU-T P.1203 implementation is 1s, so we have 6 new values per every downloaded segment.
- O.23: it provides overall score considering stalling events. We update this value every time the player downloads a new segment.
- O.46: it provides overall score overall quality score, considering video and stalling events. We update this value every time the player downloads a new segment.

To check the accuracy of the values obtained by the MEC Proxy, we compare the outcomes at the MEC with the outputs at the player side. At the player it is not necessary to design algorithms to extract, estimate or infer the target signals of the ITU-T P.1203 model since all of them are directly available at the video player. The player establishes the video representation, then it knows the features of the downloaded segment after selecting one. Moreover, to know if a stall is experienced, it is enough to check when the internal playback buffer goes empty.

4.3 QoE estimation at the Edge results

As described in Section 4.1, the player inter-arrival time and session duration are modelled as described at [32]. While testing the proposed solution for one hour, it resulted in 66 video players for Scenario 1



(a) (b) (c)

Fig. 5: Average and standard deviation of ITU-T P.1203 QoE scores for 10 players from Scenario 2: O.34 (a), O.23 (b) and O.46 (c).

Table 1: Scenario 1: number (N_{stall}) and total duration (T_{stall}) of stalls.

	$\mathbf{N_{stall}}$	$\mathbf{T_{stall}}$
Player	661	289s
MEC	160	263s

and 144 for Scenario 2. In terms of computational capabilities consumption, Raspberry Pi 3, which runs the MEC Proxy (Algorithms 1 and 2) and the access point, experiences 13% CPU and 122MB RAM usage during Scenario 1 and 17% CPU and 156MB RAM usage during Scenario 2.

Figure 5 shows average value and standard deviation along the streaming session of the considered ITU-T P.1203 outputs. It shows only the results for 10 players randomly chosen among the executed ones of the Scenario 2 since it is the most demanding scenario, where more stalls and quality changes are experienced. Scenario 1 has similar results, but as expected the average values are higher due to the lower competition for the available network resources. The results for O.34 (Figure 5a) obtained at the MEC are like the ones obtained at the player. It is reasonable since most of the video information to provide to the ITU-T P.1203 model comes from the MPD, which is available at the player, as well as at the MEC Proxy. On the contrary, the results for O.23 (Figure 5b) shows some differences since the Algorithm 2 may not detect all the stalling events at the player side from the MEC Proxy. The sampling time to check for stalls at the MEC Proxy is equal to the segment duration. Then, a stall experienced by the player, whose duration is short, and it is recovered before the next check at the MEC, may not be detected correctly. Consequently, estimated values at the MEC are higher that the captured ones at the player. Tables 1 and 2 also provides more in-depth details of the stalls. It is clear from the tables that the MEC cannot detect all the stalls. Anyway, the total duration of stalls tends to actual one experienced by the player. It means that MEC Proxy cannot detect micro-stalls and tends to concentrate several of them into a macro-stall. Again, this is due to the sampling time to check for stalls. If two micro-stalls are experienced during the same segment, the MEC Proxy will consider that they are one longer stall since it checks only at the end of each segment download. Finally, O.46 scores (Figure 5c) are directly influenced by the results obtained by the other two scores (O.34 and O.23). It has a significant standard deviation due to video scores (O.34), but lower average value due to the impact of buffering score (O.23).

Tables 3 and 4 resume the scores of all the players executed under the Scenarios 1 and 2, respectively. The Tables shows the results by averaging the values obtained by the players and evaluating the standard deviation for each considered score. The Tables confirms that O.34 has the same values when estimated

Table 2: Scenario 2: number (N_{stall}) and total duration (T_{stall}) of stalls.

	$\mathbf{N_{stall}}$	$\mathbf{T_{stall}}$
Player	1623	757s
MEC	334	765s

Table 3: Scenario 1: average and standard deviation of ITU-T P.1203 QoE scores.

	$\mathbf{O.34_{avg}}$	$\mathbf{O.34_{std}}$	$\mathbf{O.23_{avg}}$	$\mathbf{O.23_{std}}$	$\mathbf{O.46_{avg}}$	$\mathbf{O.46_{std}}$
Player	4.71	0.28	4.81	0.20	4.38	0.28
MEC	4.70	0.28	4.84	0.19	4.39	0.28

Table 4: Scenario 2: average and standard deviation of ITU-T P.1203 QoE scores.

	$\mathbf{O.34_{avg}}$	$\mathbf{O.34_{std}}$	$\mathbf{O.23_{avg}}$	$\mathbf{O.23_{std}}$	$\mathbf{O.46_{avg}}$	$\mathbf{O.46_{std}}$
Player	4.53	0.31	4.76	0.19	4.22	0.27
MEC	4.53	0.31	4.85	0.20	4.28	0.28

Table 5: Scenario 1: MAE and RMSE for ITU-T P.1203 QoE scores.

	O.34	**O.23**	**O.46**
MAE	0.21	0.11	0.18
RMSE	0.39	0.21	0.36

at the MEC and captured at the player, while O.23 scores assessed at the MEC tends to be higher than the real values issued at the player. Again, O.46 scores, which are influenced by the other two scores, shows a standard deviation coming from O.34 and lower average values inherited from O.23.

As expected, Scenario 1 presents higher values for any of the considered metrics since the network resources are the same as Scenario 2, but the number of players sharing them is lower. The resulting standard deviations are similar for both scenarios.

Tables 5 and 6 compare the results obtained at the MEC and at the player side by providing the Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) between the different scores. The results show that the scores obtained at the MEC tends to be like the ones obtained at the player, achieving more accurate results in the Scenario 1, as expected.

Note that O.34 has higher MAE and RMSE than O.23 and O.46. It could be considered a contradictory to the fact that the values obtained at MEC Proxy compared to the values captured at the player for O.34 are more accurate than O.23 and O.46. Anyway, it is important to remember that these metrics have different sampling rates. O.34 provides a value every second, while O.23 and O.46 every segment (6s). Then, O.34 has higher values for MAE and RMSE since they are evaluated over a number of scores which is 6 times bigger than O.23 and O.46. Finally, if we compare the two scenarios, the results tend to be similar since the MAE and RMSE are relative values, they are not conditioned by the average values of the metrics. It means that the MEC Proxy performs similarly in both scenarios.

In summary, the proposed MEC Proxy can monitor network dynamics and estimate the individual QoE of each player according to Recommendation ITU-T P.1203. The results show that video related scores (O.34) estimated follow the actual ones captured at the player, while buffering scores (O.23) tend to miss some small stalls unperceived from segment to segment. Anyway, the overall scores (O.46) are

Table 6: Scenario 2: MAE and RMSE for ITU-T P.1203 QoE scores.

	O.34	**O.23**	**O.46**
MAE	0.14	0.14	0.19
RMSE	0.43	0.24	0.36

still valid showing a small difference of the estimated values at the MEC from the actual values captured at the player.

5 Conclusion

The trend for the following years is an increasing consumption of media content due to the popularity of video streaming platforms. To cope with this increasing demand, MNOs need to manage the network according to cost-effective policies, requiring monitoring solutions which provide actionable data to management systems to provide the best QoE with the available network resources.

The proposed 5G MEC-enabled proxy aims to estimate the ITU-T P.1203 QoE metrics to enable edge services operating coordinated decisions on the streaming qualities. The MEC Proxy assesses QoS metrics at the radio cell and parses manifests of requested DASH streams to estimate the parameters employed to evaluate ITU-T P.1203 QoE scores. Consequently, there is no need for explicit out-of-band messaging from video players to send playback statistics to the MEC.

The solution has been implemented in a real testbed where WiFi was employed as access network between MEC and players and tested in two scenarios with different demands and traffic loads over the network. The results demonstrate the capability of the MEC Proxy to estimate ITU-T P.1203 QoE scores close to actual ones.

Acknowledgements This work was fully supported by the 5G-TEST project (Gipuzkoa's research and innovation programme) and Open-VERSO project (Red Cervera program, Spanish government's Centre for the Development of Industrial Technology).

References

1. Akhshabi S, Anantakrishnan L, Begen AC, Dovrolis C (2012) What happens when HTTP adaptive streaming players compete for bandwidth?. Proceedings of the 22nd international workshop on Network and Operating System Support for Digital Audio and Video: 9-14. doi: 10.1145/2229087.2229092
2. Akhshabi S, Anantakrishnan L, Dovrolis C, Begen AC (2013) Server-based traffic shaping for stabilizing oscillating adaptive streaming players. Proceeding of the 23rd ACM workshop on network and operating systems support for digital audio and video: 19-24. doi: 10.1145/2460782.2460786
3. De Cock J, Li Z, Manohara M, Aaron A (2016) Complexity-based consistent-quality encoding in the cloud. IEEE International Conference on Image Processing (ICIP):1484-1488. doi: 10.1109/ICIP.2016.7532605.
4. Dutta S, Taleb T, Frangoudis PA, Ksentini A (2016) On-the-Fly QoE-Aware Transcoding in the Mobile Edge. IEEE Global Communications Conference (GLOBECOM):1-6. doi: 10.1109/GLOCOM.2016.7842074
5. ETSI (2018) ETSI GS MEC 002 version 2.1.1 - Multi-access Edge Computing (MEC): Phase 2: Use Cases and Requirements. `https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/02.01.01_60/gs_MEC002v020101p.pdf` Accessed 13 December 2020.
6. ETSI (2018) ETSI TS 122 261 version 15.6.0 - 5G; Service requirements for next generation new services and markets. `https://www.etsi.org/deliver/etsi_ts/122200_122299/122261/15.06.00_60/ts_122261v150600p.pdf` Accessed 13 December 2020.

7. ETSI (2020) ETSI GS MEC 028 version 2.1.1 - Multi-access Edge Computing (MEC); WLAN Information API. https://www.etsi.org/deliver/etsi_gs/MEC/001_099/028/02.01.01_60/gs_MEC028v020101p.pdf Accessed 13 December 2020.
8. Fajardo J, Taboada I, Liberal I (2015) Improving content delivery efficiency through multi-layer mobile edge adaptation. IEEE Network 29(6):40-46. doi: 10.1109/MNET.2015.7340423.
9. Ge C, Wang N (2018) Real-time QoE estimation of DASH-based mobile video applications through edge computing. IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS):766-771. doi: 10.1109/INFCOMW.2018.8406935.
10. Ge X, Tu S, Mao G, Wang C, Han T (2016) 5G Ultra-Dense Cellular Networks. IEEE Wireless Communications 23(1):72-79. doi: 10.1109/MWC.2016.7422408.
11. GStreamer: open source multimedia framework. https://gstreamer.freedesktop.org/ Accessed 13 December 2020.
12. Huang W, Zhou Y, Xie X, Wu D, Chen M, Ngai E (2018) Buffer State is Enough: Simplifying the Design of QoE-Aware HTTP Adaptive Video Streaming. IEEE Transactions on Broadcasting 64(2):590-601. doi: 10.1109/TBC.2018.2789580
13. ITU-T Rec. P.1203 Standalone Implementation. https://github.com/itu-p1203/itu-p1203 Accessed 13 December 2020.
14. Juluri P, Tamarapalli V, Medhi D (2016) Measurement of Quality of Experience of Video-on-Demand Services: A Survey. IEEE Communications Surveys & Tutorials 18(1):401-418. doi: 10.1109/COMST.2015.2401424.
15. Khan K, Goodridge W (2018) What happens when adaptive video streaming players compete in time-varying bandwidth conditions?. International Journal of Advanced Networking and Applications: 3704-3712. doi: 10.35444/IJANA.2018.10015
16. Koskimies L, Taleb T, Bagaa M (2017) QoE estimation-based server benchmarking for virtual video delivery platform. IEEE International Conference on Communications (ICC):1-6. doi: 10.1109/ICC.2017.7996445
17. Lederer S, Mueller C, Timmerer C, Concolato C, Le Feuvre J, Fliegel K (2013) Distributed DASH dataset, Proceedings of the 4th ACM Multimedia Systems Conference:131-135. doi: 10.1145/2483977.2483994
18. Liotou E, Tsolkas D, Passas N (2016) A roadmap on QoE metrics and models. International Conference on Telecommunications (ICT):1-5. doi: 10.1109/ICT.2016.7500363
19. Mangla T, Halepovic E, Ammar M, Zegura E (2017) MIMIC: Using passive network measurements to estimate HTTP-based adaptive video QoE metrics. 2017 Network Traffic Measurement and Analysis Conference: 1-6. doi: 10.23919/TMA.2017.8002920
20. Mangla T, Halepovic E, Ammar M, Zegura E (2018) emimic: Estimating http-based video qoe metrics from encrypted network traffic. 2018 Network Traffic Measurement and Analysis Conference: 1-8. doi: 10.23919/TMA.2018.8506519
21. Mazhar MH, Shafiq Z (2018) Real-time video quality of experience monitoring for https and quic. IEEE INFOCOM 2018-IEEE Conference on Computer Communications: 1331-1339. doi: 10.1109/INFOCOM.2018.8486321
22. Mozilla MDN web docs, User-Agent. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent Accessed 13 December 2020.
23. Mueller C, Lederer S, Grandl R, Timmerer C (2015) Oscillation compensating dynamic adaptive streaming over http. 2015 IEEE International Conference on Multimedia and Expo (ICME): 1-6. doi: 10.1109/ICME.2015.7177435
24. Node.js: asynchronous event driven JavaScript runtime. https://nodejs.org/en/ Accessed 13 December 2020.
25. Quadri C, Gaito S, Bruschi R, Davoli F, Rossi GP (2018) A MEC Approach to Improve QoE of Video Delivery Service in Urban Spaces. IEEE International Conference on Smart Computing (SMARTCOMP):25-32. doi: 10.1109/SMARTCOMP.2018.00095
26. Raake A, Garcia MN, Robitza W, List P, Göring S, Feiten B (2017) A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1, 2017 Ninth International Conference on Quality of Multimedia Experience (QoMEX):1-6. doi: 10.1109/QoMEX.2017.7965631
27. Recommendation ITU-T P.800.1 (2016) Mean opinion score (MOS) terminology.
28. Recommendation ITU-T P.1203 (2017) Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport.
29. Richardson M, Wallace S (2012) Getting started with raspberry PI. O'Reilly Media, Inc.
30. Robitza W, Göring S, Raake A, Lindegren D, Heikkilä G, Gustafsson J, List P, Feiten B, Wüstenhagen U, Garcia MN, Yamagishi K, Broom S (2018) HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 – Open Databases and Software, 9th ACM Multimedia Systems Conference:466-471. doi: 10.1145/3204949.3208124
31. Younis A, Tran TX, Pompili D (2019) On-Demand Video-Streaming Quality of Experience Maximization in Mobile Edge Computing. IEEE International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM):1-9. doi: 10.1109/WoWMoM.2019.8793052
32. Yu H, Zheng D, Zhao BJ, Zheng W (2006) Understanding user behavior in large-scale video-on-demand systems, ACM SIGOPS Operating Systems Review 40(4):333-344. doi: 10.1145/1218063.1217968
33. Zhang X, Wang J (2018) Joint heterogeneous statistical-QoS/QoE provisionings for edge-computing based WiFi offloading over 5G mobile wireless networks. Annual Conference on Information Sciences and Systems (CISS):1-6. doi: 10.1109/CISS.2018.8362265