EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH Open Access

YouTube QoE evaluation tool for Android wireless terminals

Gerardo Gómez[1*], Lorenzo Hortigüela[2], Quiliano Pérez[2], Javier Lorca[2], Raquel García[2] and Mari Carmen Aguayo-Torres[1]

Abstract

In this paper, we present an Android application which is able to evaluate and analyze the perceived quality of experience (QoE) for YouTube service in wireless terminals. To achieve this goal, the application carries out measurements of objective quality of service (QoS) parameters, which are then mapped onto subjective QoE (in terms of mean opinion score, MOS) by means of a utility function. Our application also informs the user about potential causes that lead to a low MOS as well as provides some hints to improve it. After each YouTube session, the users may optionally qualify the session through an online opinion survey. This information has been used in a pilot experience to correlate the theoretical QoE model with real user feedback. Results from such an experience have shown that the theoretical model (taken from the literature) provides slightly more pessimistic results compared to user feedback. Users seem to be more indulgent with wireless connections, increasing the MOS from the opinion survey in about 20% compared to the theoretical model, which was obtained from wired scenarios.

Keywords: Quality of experience; Mean opinion score; YouTube; Android

1. Introduction

Real-time entertainment services (comprised mostly of streaming video and audio) are becoming one of the dominant web-based services in telecommunications networks. In particular, YouTube service is currently the largest single source of real-time entertainment traffic and the third most visited Internet site (preceded by Google and Facebook). It has emerged to account for more Internet traffic than any other service. Mobile networks have the highest proportion of real-time entertainment traffic. Nowadays, YouTube leads the way, accounting for 20% to 25% of total traffic in mobile networks. Additionally, 27.8% of all YouTube traffic (first half 2012) has been consumed on a smartphone or tablet [1].

The combination of increasing device capabilities, high-resolution content, and longer video duration (largely due to live content) means that YouTube's growth will continue for the foreseeable future. Driven by higher bitrates and enhanced capabilities of mobile devices, the trend is also going towards high-definition (HD) video, which considerably enhances quality demand. That is the reason mobile network operators are following this trend, as it will be hugely influential on network requirements and subscriber quality of experience. In the context of video streaming services, quality of service (QoS) measurements become non-sufficient for evaluating the overall quality because they do not take into account user satisfaction. As a consequence, many network operators are starting to study the evaluation of quality of experience (QoE), which can be considered as the overall performance of a system from a user's perspective.

The QoE has been usually evaluated through subjective tests carried out on the users in order to assess their degree of satisfaction with a mean opinion score (MOS) indicator [2]. This type of approach is obviously quite expensive, as well as annoying to the user. That is why in recent years, new methods have been used to estimate the QoE based on certain performance indicators associated with services. The evaluation methodology used by most network operators to obtain statistical QoE is based on field testing. These tests often use mobile handsets as a modem, with laptop computers that perform the tests and collect statistics. However, this process is expensive in terms of resources and staff, and it does not use the entire protocol stack implemented in the terminal. These drawbacks are solved by integrating QoE analyzers in the

* Correspondence: ggomez@ic.uma.es
[1]Department of Communications Engineering, University of Malaga, 29071 Malaga, Spain
Full list of author information is available at the end of the article

mobile terminal itself so that quality measurements are specific to each terminal. Using this approach, additional measurements can be collected (along the protocol stack) to allow for enhanced analysis of the performance of each service. Furthermore, if mobile terminals are able to report the measurements to a server, the QoE assessment process is simplified significantly.

Recently, a number of works have focused on developing subjective QoE evaluation frameworks for mobile users. For instance, an implementation of a QoE measurement framework on Android platform is presented in [3,4], although results are limited to a laboratory environment. The works in [5-7] present a framework for measuring the QoE for distorted videos in terms of peak signal-to-noise ratio (PSNR) or a modified metric called cPSNR, respectively. A non-intrusive video quality estimator based on parametric models that considers the group of pictures (GoP) length, type of frames, packet loss rate, etc. is presented in [8]. A QoE framework for multimedia services (named as QoM) for run time quality evaluation of video streaming services is presented in [9]; this approach is based on the influence of QoE factors and various network and application-level QoS parameters, although no evaluation of the proposed framework in a context of a real wireless network has been performed. In [10], the problem of YouTube QoE monitoring from an access provider's perspective is investigated, showing that it is possible to detect application-level stalling events by using network-level passive probing only. The work in [11] describes a tool that monitors YouTube application comfort, making it possible to estimate the time when the YouTube player is stalling.

Other works are focused on specific YouTube models to compute the QoE. In [12,13], different QoE YouTube models that take into account the key influence factors in the quality perception (such as stalling events caused by network bottlenecks) are presented. They quantify the impact of initial delays on the user-perceived QoE by means of subjective laboratory and crowdsourcing studies. Other works are devoted to estimate the MOS for video services [3,14,15]; among them, the analysis presented in [15] provides a utility function for HTTP video streaming as a function of three application performance metrics: initial buffering time, mean rebuffering time, and rebuffering frequency.

However, none of the previous works has performed a deep validation of existing models through real tests over different radio technologies. In this work, we describe an Android application that carries out measurements of objective QoS indicators associated to YouTube service; these performance indicators are then mapped onto subjective QoE (in terms of MOS). Our application also informs the user about possible causes that lead to a low MOS, and provides some hints to improve it. After each YouTube session, the users may optionally qualify the session through an opinion survey. This information has been used in a pilot experience to correlate the theoretical QoE model with real user feedback.

The remainder of this paper is structured as follows. A description of the YouTube QoE evaluation method is given in Section 2, specifying its main performance indicators. In Section 3, we describe our Android application for YouTube QoE evaluation. The results from a YouTube evaluation pilot experience are analyzed in Section 4. Finally, some concluding remarks are given in Section 5.

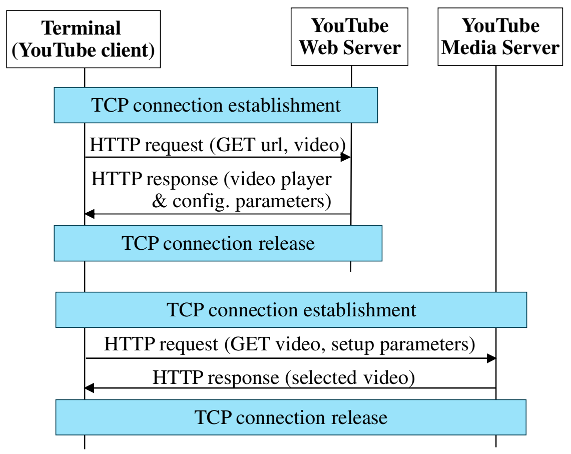

Terminal (YouTube client)
YouTube Web Server
YouTube Media Server
TCP connection establishment
HTTP request (GET url, video)
HTTP response (video player & config. parameters)
TCP connection release
TCP connection establishment
HTTP request (GET video, setup parameters)
HTTP response (selected video)
TCP connection release

Figure 1 Signaling flow chart of a YouTube session (via web browser).

2. YouTube QoE evaluation method

YouTube service employs progressive download technique, which enables the playback of the video before the content downloaded is completely finished [16]. Old YouTube delivery service for mobile terminals (through the mobile YouTube link http://m.youtube.com) was based on conventional video streaming architecture, i.e., Real Time Streaming Protocol (RTSP) and Real Time Protocol (RTP), the latter being transported over User Datagram Protocol (UDP). However, the current delivery service (both for smartphones and PCs) uses progressive video

Table 1 Three valued levels of application performance metrics

L_{ti}	T_{init}	L_{rf}	f_{rebuf}	L_{tr}	T_{rebuf}
1	0 to 1 s	1	0 to 0.02	1	0 to 5 s
2	1 to 5 s	2	0.02 to 0.15	2	5 to 10 s
3	>5 s	3	> 0.15	3	>10 s

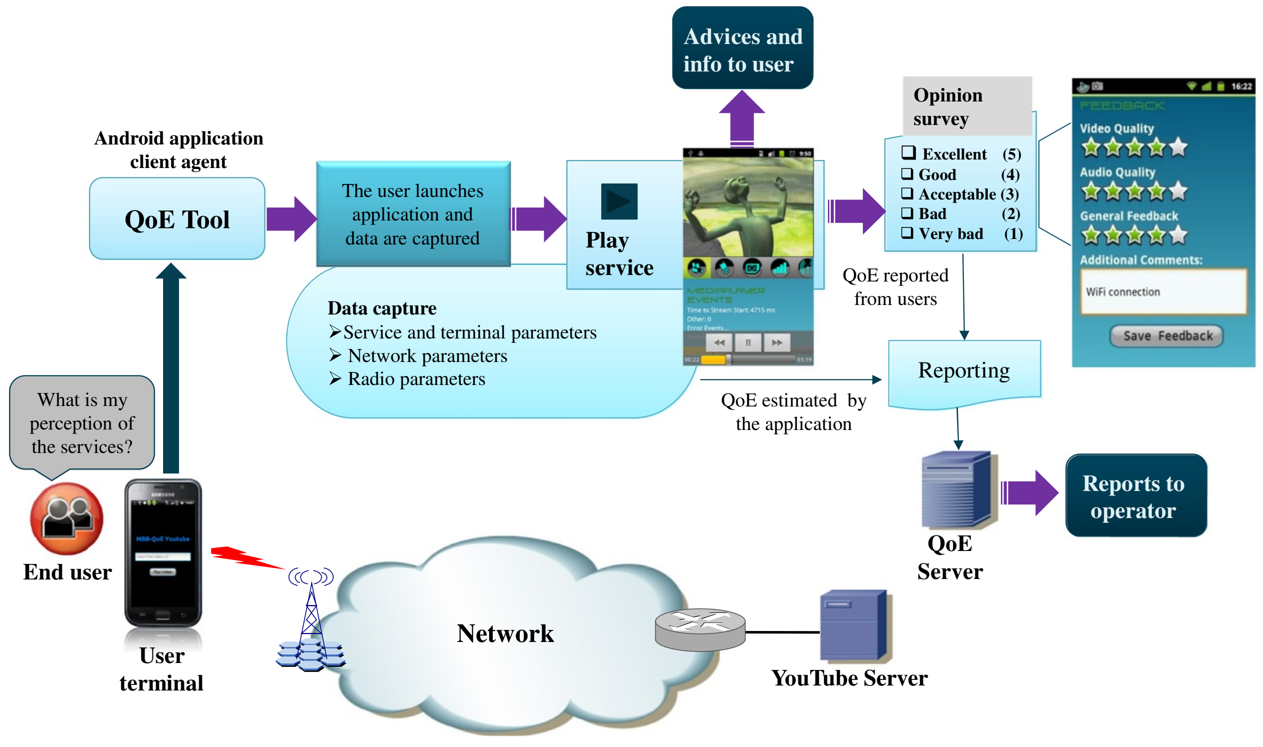
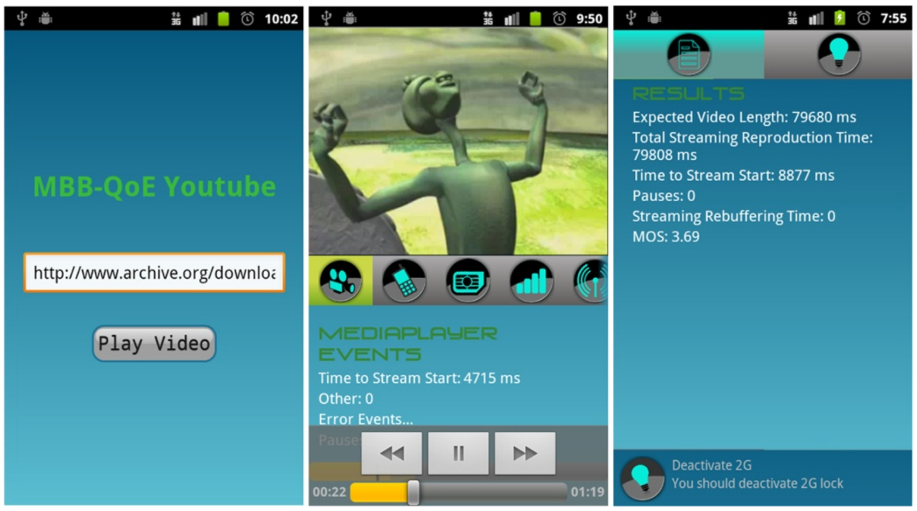


Figure 2 YouTube QoE framework.

download via HyperText Transfer Protocol (HTTP) over Transmission Control Protocol (TCP).

Nowadays, TCP is the preferred transport protocol for YouTube and other video servers since the majority of video content delivery over the Internet is not live and most users' bandwidth is usually greater than the video coding rate [17,18]. The HTTP/TCP architecture also solves the problem of access blocking carried out by many firewalls for unknown UDP ports. Additionally, the continuous improvements in latency reduction and throughput maximization achieved in new cellular technologies have allowed the use of TCP for minimizing the impact of errors without reducing severely the effective throughput.



Time to Stream Start: 4715 ms
Other: 0
Error Events...

Expected Video Length: 79680 ms
Total Streaming Reproduction Time: 79808 ms
Time to Stream Start: 8877 ms
Pauses: 0
Streaming Rebuffering Time: 0
MOS: 3.69

Figure 3 Snapshots of our Android QoE tool (intrusive mode).

The video clip download process is started by the end user when a request (with a link to the desired video clip) is sent to the YouTube web server (see Figure 1). When the client web browser receives the YouTube web page, the embedded player initiates the required signaling with the media server indicating the video to be played out along with some setup parameters [17]. Then, the server starts progressively sending the video data over an HTTP response. The video data is then stored in a play-out buffer at the client side before being displayed. Once the download has been started, there is no further client-to-server signaling (unless the user interacts with the player).

The video data transfer from the media server to the client consists of two phases: initial burst of data and throttling algorithm [17]. In the initial phase, the media server sends an initial burst of data (whose size is determined by one of the setup parameters) at the maximum available bandwidth. Then, the server starts the throttling algorithm, where the data are sent at a constant rate (normally at the video clip encoding rate multiplied by the throttle factor, also denoted in the setup parameters). In a network congestion episode, the data that are not able to be delivered at this constant rate are buffered in the server and released as soon as the congestion is alleviated. When this occurs, data are

Table 2 List of parameters that are reported (from the terminal) to the QoE server

Parameter type	Parameter	Description
Device ID	IMEI	International Mobile Equipment Identity (15-digit format)
Session information	Reproduction Mode	It indicates the application used for video reproduction:
		1 - Embedded video player (based on media player)
		2 - YouTube native application
		3 - Web browser
	Reproduction Time	Total reproduction time (in ms) including rebuffering and user-originated pauses
	Date	Date of the YouTube video session (AAAA-MM-DD)
	Hour	Hour of the YouTube video session (HH:MM:SS)
Application Performance Metrics	InitialBuffering Time (T_{init})	Total time (in ms) since the user starts the session until the video is ready to be played
	Rebuffering Frequency (f_{rebuf})	Frequency of interruption events (not forced by the user) during the playback
	Mean Rebuffering Time (T_{rebuf})	Average duration of a rebuffering event (in ms)
Location of the measurement	Latitude	Expressed in sexagesimal degrees (−90, 90)
	Longitude	Expressed in sexagesimal degrees (−180, 180)
	Altitude	Expressed in meters above sea level
	Accuracy	Precision in meters of the location measurements
	Time	Moment at which the location measurement was done (AAAA-MM-DD_HH:MM:SS format)
	Provider	Method to perform location measurements: GPS or network-assisted
Network information	Connection Type	Type of network data connection active for the session. Possible values: 0(WIFI), 1(GPRS), 2(EDGE), 3(UMTS), 4(CMDA), 5(EVDO_0), 6(EVDO_A), 7(1XRTT), 8(HSDPA), 9(HSUPA), 10(HSPA), 11(IDEN), 12(EVDO_B), 13(LTE), 14(EHRPD), 15(HSPAP)
	LAC	Location area code where the user is located
	Cell ID	Identifier of the cell providing service to the terminal
	RSSI	Received signal strength indication (dBm) measured by the terminal (for either WiFi or cellular connections)
Subjective quality (feedback from users)	Video Quality Feedback	Subjective opinion regarding video quality (scale: 1 to 5)
	Audio Quality Feedback	Subjective opinion regarding audio quality (scale: 1 to 5)
	General Feedback	General feedback from the user (scale: 1 to 5)
	Additional Comments	The user can add any additional comment
Subjective quality (estimated)	Estimated Video Quality	Estimated video quality from QoE model (scale: 1 to 5)

sent at the maximum available bandwidth. Whenever the player's buffer runs out of data, the playback will be paused, leading to a rebuffering event.

Like quality of Internet services in general, Internet video streaming quality is mainly depending on throughput. However, quality requirements in terms of throughput are more demanding than those for other popular Internet applications as file download, web browsing, and messaging. The main differences are that throughput has to meet rather precise requirements and that these requirements are stream-specific, i.e., if data are not transmitted according to the playing rate (corrected by the influence of initial buffering), a rebuffering will likely occur and user QoE will drop down rapidly. It is therefore essential not only to measure the download throughput, but also to check against the bitrate the individual stream is encoded with.

There exist many quality metrics to characterize the video quality. Some of them are based on comparing the received (and degraded) video with the original video (usually called 'reference'). Examples of this type of quality metrics are as follows: mean square error (MSE) [19], peak signal-to-noise ratio (PSNR) [19], video structural similarity (VSSIM) [20], perceptual evaluation of video quality (PEVQ) [21], and video quality metric (VQM) [22]. This type of metrics is useful for obtaining objective metrics in controlled experiments, but these metrics are not applicable for online (real-time) procedures as the full reference is not available. Furthermore, they are suited to measure the image quality degradation, e.g., due to packet losses or compression algorithms. Since using TCP, packet losses are recovered, this type of metrics is less useful for YouTube.

That is why other works are oriented to provide a model for estimating the video quality without a reference. For

Table 3 Examples of causes of low QoE and advices to users (QoE advices module)

Cause	Evidence		Advice
Low throughput	*High traffic load*		
	IF many applications synchronizing	→	Temporarily stop data synchronization
	ELSE IF many apps running	→	Offer some apps/services to be switched off
	ELSE	→	Switch to other technology (WiFi, mobile)
	Low network traffic and connected to a cellular network		
	IF GSM/3G lock on 2G	→	Activate 3G
	ELSE IF low RSSI and WiFi available	→	Switch to a WiFi connection
	ELSE IF low RSSI and WiFi switched on and WiFi not available	→	Switch off WiFi to avoid interference
	ELSE IF low RSSI and Bluetooth switched on	→	Switch off Bluetooth to avoid interference
	Low network traffic and connected to WiFi		
	IF WiFi tethering is activated	→	Switch off WiFi tethering
	ELSE IF Bluetooth switched on	→	Switch off Bluetooth
	ELSE	→	Switch to a cellular network connection
Low memory	*Low memory status flag is TRUE*		
	IF many apps/services running	→	Offer some apps/services to be switched off
	ELSE IF 'hungry' app detected	→	Offer to switch off 'hungry' app
	ELSE	→	Check for system updates
High CPU load	CPU load is high during a period		
	IF many apps/services running	→	Offer some apps/services to be switched off
Low CPU frequency forced	*CPU freq low*		
	IF low battery level		
	OR high battery temperature	→	Wait until battery gets in better conditions
	ELSE IF aggressive power save profile selected	→	Select a performance oriented profile
	ELSE	→	Check for system updates
Video requirements exceeds terminal capabilities	*YouTube API video source and device HW information*		
	IF device capability < video req.	→	Try to select less demanding video files, switch off high quality (HQ) option
Low video quality in origin	*YouTube API video source information*		
	IF low resolution/coding rate	→	Select another file of higher quality

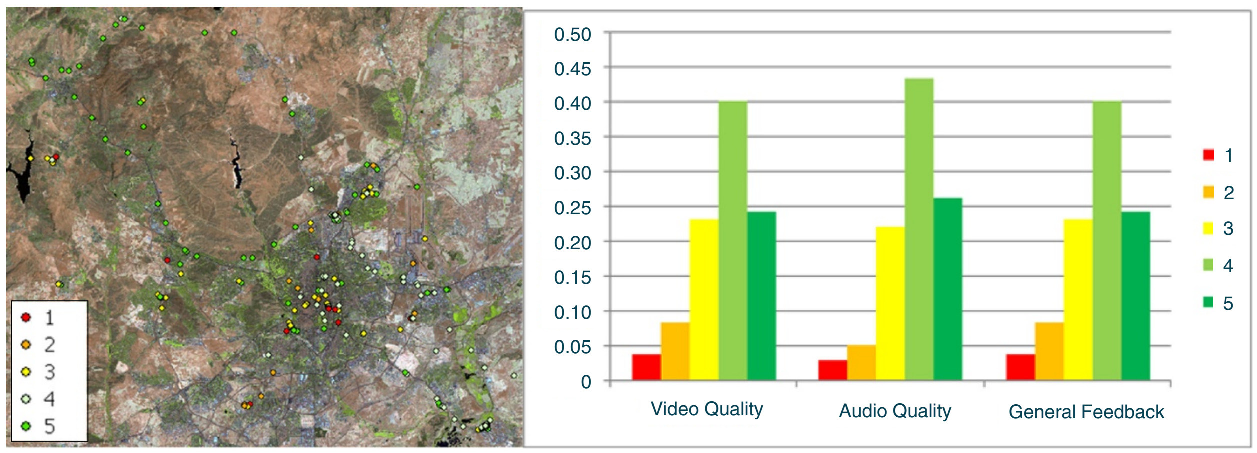


Figure 4 Geographical distribution of the users (left) and quality feedback distribution (right).

instance, the work described in [23] presents a regression model to estimate the visual perceptual quality in terms of MOS for MPEG-4 videos over wireless networks. However, this algorithm requires an image reconstruction process to evaluate the differences between the original and the resulting images (after network transmission), which makes it not adequate for online quality estimations. In [24], the impact of delay and delay variation on user's perceived quality for video streaming is analyzed. However, it does not consider other objective metrics such as resolution, frame rate, or packet losses, which are also important for obtaining an accurate QoE estimation. In [25], a no-reference subjective metric to evaluate the video quality is presented, which considers the frame rate or the picture resolution, although their computation is complex to be used real-time.

Our implementation is based on the work presented in [15], which studied how the network QoS affects the QoE of HTTP video streaming. In this work, they propose a generic procedure to estimate the end user's perceived quality following three steps:

1) Estimate (through modeling) or measure network QoS (e.g., throughput, round trip time, loss rate, etc.).
2) Convert network QoS metrics onto application QoS (application performance metrics) by means of protocols' modeling.
3) Map application QoS onto end user's QoE (in terms of MOS).

It should be noticed that the first step might not be needed if the mobile terminal is equipped with a customized YouTube client that directly monitors and reports the application performance metrics. Otherwise, the mobile terminal shall be able to convert the network QoS onto application QoS by specific protocol modeling. For instance, there are different TCP performance models to estimate TCP throughput from network QoS [26,27]. Afterwards, application performance metrics can be estimated at the receiver from performance indicators at lower layers (e.g., TCP throughput) as well as other parameters like the video coding rate, video length, buffer size at the receiver, or the minimum buffer threshold that triggers a rebuffering event (see [15] for further details).

The third step is performed by applying a utility function for HTTP video streaming as a function of three application performance metrics:

- Initial buffering time (T_{init}): time elapsed until a certain buffer occupancy threshold has been reached so the playback can start
- Rebuffering frequency (f_{rebuf}): frequency of interruption events during the playback
- Mean rebuffering time (T_{rebuf}): average duration of a rebuffering event

In [15], each application performance metric (T_{init}, f_{rebuf}, and T_{rebuf}) is divided into three levels (low, medium, high) which are based on the 25th, 50th, and 75th percentiles of the actual metric. The quantified performance metrics L_{ti}, L_{fr}, and L_{tr} take the numerical values 1, 2, and 3 to represent the 'low', 'medium', and 'high' levels, respectively. The mapping from T_{init}, f_{rebuf}, and T_{rebuf} to L_{ti}, L_{jr}, and L_{tr} is performed according to Table 1 [15].

Table 4 Summary for reproduction time (in minutes)

Connection type	Number of sessions	Mean	Median	Standard deviation	Min	Max
UMTS	911	2.91	1.8	3.92	0.02	49.35
WiFi	524	2.19	1.14	2.89	0.0007	26.88
Total	1,435	2.65	1.6	3.6	0.0007	49.35

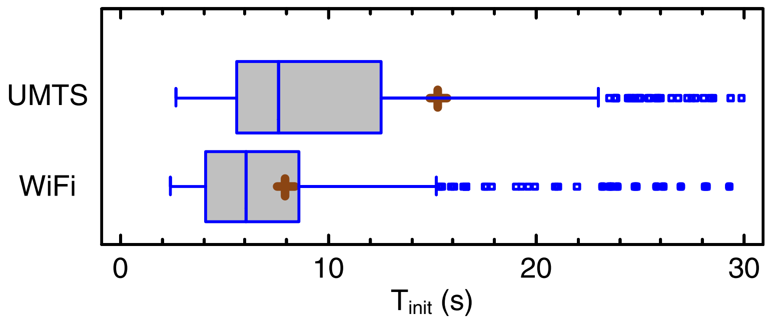


Figure 5 Box-and-whiskers plots for the initial buffering time (in seconds) per technology.

The quantified performance metrics L_{ti}, L_{fr}, and L_{tr} were used in [15] to obtain the MOS estimation from linear regression:

$$\mathrm{MOS}_{\mathrm{QoSmodel}} = 4.23 - 0.0672 \cdot L_{ti} - 0.742 \cdot L_{fr} - 0.106 \cdot L_{tr}. \quad (1)$$

From the previous equation, it can be seen that the maximum predicted MOS is 4.23. Coefficients for all service levels (L_{ti}, L_{fr}, and L_{tr}) are negative as any increase in them would lead to worse service quality. Moreover, the rebuffering frequency metric (f_{rebuf}, quantized as L_{fr}) has the highest impact on the end user's QoE (regression coefficient −0.742), compared to the initial buffering time (regression coefficient −0.06) and the rebuffering duration (regression coefficient −0.106). In this respect, it is reasonable to think that the perceived quality does not only depend on the pause intensity (percentage of time in the pause state), since a higher number of pauses (with lower pause durations) seems more annoying to the user.

3. Android application for YouTube QoE evaluation

The model for estimating YouTube QoE has been implemented as an Android application. Our QoE tool is able to run in two different modes:

1) *Intrusive mode*: Our application includes an embedded video player, thus providing access to the content being consumed through the YouTube Application Programming Interface (API).
2) *Transparent mode*: Our application runs in the background, so monitoring functionalities are associated to YouTube sessions established either through the native YouTube application or through the web browser.

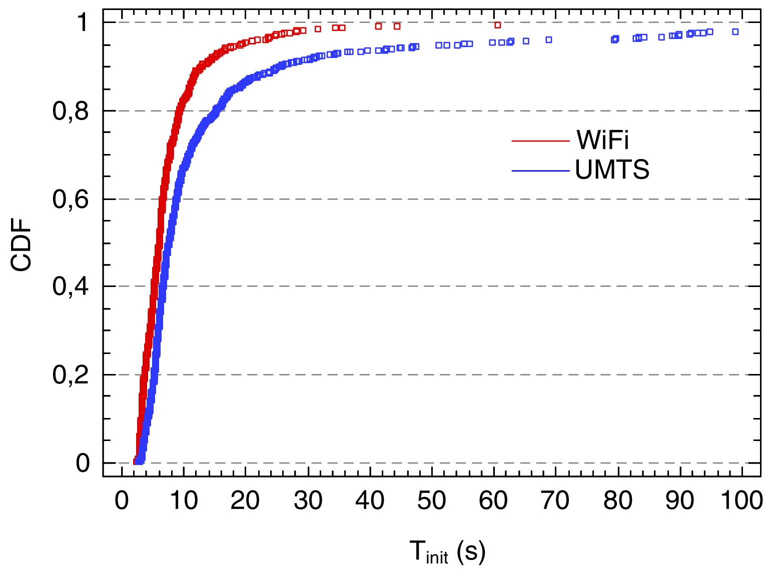


Figure 6 Estimated CDF of the initial buffering time (T_{init}).

Our Android application includes the following modules:

- *Monitoring*: This module is responsible for monitoring network QoS parameters as well as other configuration parameters as required to estimate the application performance monitoring (listed in the previous section). It makes use of the Android Networking and YouTube Data API to get a number of parameters associated to the session, as detailed next.
- *QoE estimation*: This is in charge of (automatically) computing the QoE of a YouTube session (in terms of MOS) from QoS parameters, according to Equation 1.
- *QoE advices*: This informs the user about possible causes that lead to a low MOS and provides some hints to improve it.
- *QoE user feedback*: This allows users to qualify the session through an opinion survey. This information is used to correlate the QoE model with real user feedback.
- *QoE reporting*: This module is responsible for reporting all the performance indicators to a QoE server for post-processing purposes.

A general overview of our YouTube QoE framework is depicted in Figure 2. In addition to the MOS value automatically estimated by the application, users are requested to qualify the session (video, audio, and general feedback) manually in the same MOS scale (from 1 to 5). We have used both types of QoE evaluations to validate the theoretical model proposed in [15], as well as to propose a modified function according to the results of our pilot experience.

Figure 3 shows some snapshots of our Android QoE tool in its *intrusive mode* version, which includes the media player. Once the YouTube session is over, estimated quality results are shown to the user and reported to the QoE server.

In order to estimate the QoE, the monitoring module must collect a set of performance indicators to be subsequently mapped onto QoE. In addition to the three application performance metrics (T_{init}, f_{rebuf}, T_{rebuf}) required to compute the MOS, the monitoring module gathers other relevant information related to the mobile terminal, session information (date, type of player, etc.), location of the measurements, or network information. All this information is reported, together with the estimated QoE and subjective quality specified by the users, to the QoE server. The complete list of parameters that are reported (from the terminal) to the QoE server is given in Table 2.

Our application uses principally standard libraries from Android SDK. Regarding the collection of network statistics, Android terminals allow to get radio-related parameters and performance indicators like the received signal strength as well as traffic-related statistics via standard libraries, like android.telephony or android.net. Time tagging (in milliseconds) associated to performance indicators is performed via standard time class, whereas location information is obtained from Android Location Manager (selecting network providers and GPS, if available).

However, other relevant information is not accessible through the standard Android API, which is more oriented towards applications of general interest. For instance, YouTube application performance metrics (T_{init}, f_{rebuf}, T_{rebuf}) and some parameters related to the YouTube session (like video duration) cannot be directly obtained from Android API. We have discarded the use of Android packet sniffing capabilities (in the form of .pcap files) as they require the terminal to run with root privileges.

When our QoE tool runs in *intrusive mode* (i.e., player embedded in the application), the measurement of the three application performance metrics is straightforward. For this purpose, we use YouTube API, which provides the capability not only to embed a YouTube player in your own application or to control a standalone YouTube player, but also to get certain information about the ongoing YouTube session. Concretely, application performance metrics can be obtained with this API as described next.

Table 5 Summary for initial buffering time (T_{init}) in seconds

Connection type	Mean	Median	Standard deviation	Min	Max
UMTS	15.24	7.604	31.8225	2.69	582.733
WiFi	7.94	6.014	9.69786	2.389	154.245
Total	12.5758	6.977	26.2543	2.389	582.733

Table 6 Summary for rebuffering frequency (f_{rebuf})

Connection type	Mean	Median	Standard deviation	Min	Max
UMTS	1.88e − 3	0	5.55e − 3	0	46e − 3
WiFi	1.05e − 3	0	6.73e − 3	0	0.11
Total	1.57e − 3	0	6e − 3	0	0.11

Table 7 Summary for number of pauses (N_{pauses})

Connection type	Mean	Median	Standard deviation	Min	Max
UMTS	0.54	0	2.43	0	31
WiFi	0.21	0	1.71	0	36
Total	0.42	0	2.20	0	36

Table 8 Summary for mean rebuffering time (T_{rebuf}) in seconds

Connection type	Mean	Median	Standard deviation	Min	Max
UMTS	19.1	0	143.2	0	2.594
WiFi	2.46	0	28.75	0	632
Total	13.04	0	115.73	0	2.594

The YouTube player has internally several states: *unstarted*, *buffering*, *playing*, *paused*, and *ended*. The media player is on unstarted state when it is first loaded, but as soon as it starts downloading the video file, the player changes its state to buffering. When the buffer has enough data, the playback starts, thus changing to playing state. The paused state is given if the video play-out is paused by the user. The player may go back to the buffering state if the buffer runs out of data during the playback due to congestion problems (called rebuffering event); otherwise, the playback will continue until achieving the ended state. Therefore, since the YouTube API provides information about the player state at each moment, it is easy to compute: 1) the initial buffering time (T_{init}) as the time elapsed in the buffering state for the first time, 2) rebuffering frequency (f_{rebuf}) from the number of times that the player enters the buffering state during playback, and 3) mean rebuffering time (T_{rebuf}) from the time elapsed in the buffering state.

However, when our QoE tool runs in *transparent mode*, the computation of these metrics is not so easy as we do not have access to the transitions among states, so they have to be estimated from network-level metrics, as detailed in [15]. In particular, the following basic information is required: average TCP throughput, average playing rate, and player buffer size. The model to estimate application QoS metrics from network QoS is valid under certain assumptions: 1) The network bandwidth, round trip time (RTT), and packet loss rate are assumed to be constant during the video download, and 2) the client does not interact with the video during the playback, such as pausing and forward/backward. However, these assumptions may not be very realistic due to the fact that, as throughput and playing rate may vary along the time, player's buffer utilization depends on the instantaneous throughput and play-out rate rather than their average values. Therefore, this approach is expected to provide slightly optimistic results.

Regarding the QoE advices module, its role is to analyze possible causes that provide a low QoE and subsequently to provide particular advices to the user when certain conditions are given. As an example, Table 3 shows potential causes of low QoE, their associated evidences, and advices.

The application needs permission from the user to read the phone state, to access location information, to access external storage, and to connect to the Internet. Information is stored in the device until it is reported to the server. In the device, the SQLite3 database, included in the operating system, is used for storing all retrieved information.

The proposed Android application is useful for both network operators and YouTube users. There are many use cases that would benefit the rollout of a YouTube

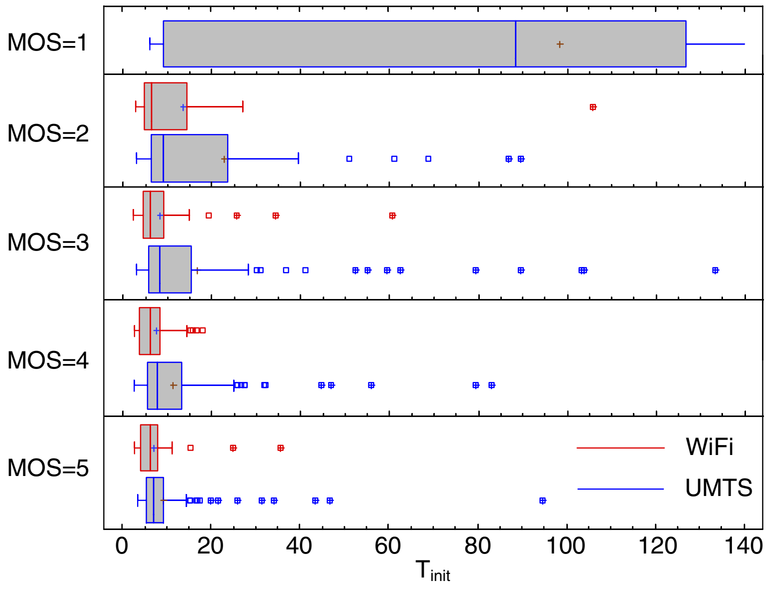


Figure 7 Box-and-whiskers plot for T_{init} per reported MOS.

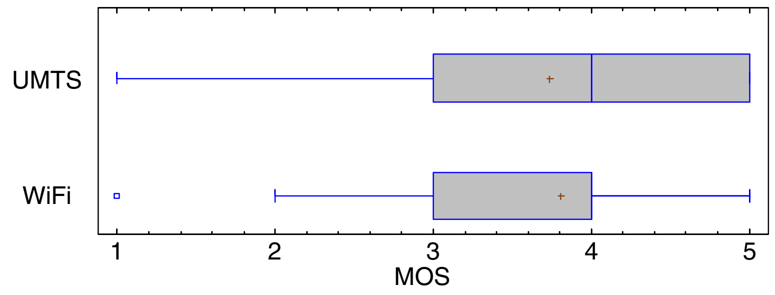


Figure 8 Box-and-whiskers plot for MOS per used technology.

QoE monitoring solution as proposed in this paper, such as:

1. The knowledge of the instantaneous YouTube QoE per user may help the operator to perform actions like, e.g.: a) to modify the subscriber priority when a poor performance in a specific location or particular subscriber is detected; b) to set dynamically different bandwidth limits depending on a number of factors like usage patterns, subscriber, location, time of day, and so forth; c) to send notifications to subscribers like advices to improve their QoE.
2. Network capacity planning: The proposed QoE solution could help on the identification of network bottlenecks and re-dimension the network to ensure the targeted QoE.
3. Traffic forecasting process based on historical data traffic stored in its database.
4. Handset and service performance benchmarking: With the growing number of mobile handsets and multimedia content launched onto the market, it is becoming increasingly important for operators to benchmark each individual terminal and measure its performance. This process enables the identification of problematic handsets and the analysis of the cause for the faults. By identifying problematic handsets, operators can quickly make the required adjustments to their network, thus improving the customer experience
5. Network monitoring and reporting: The proposed QoE solution use passive methods to infer automatically from passive measurements the user perception on the network. The goal is to automatically derive user perception from specific indicators being accessed purely from monitoring (eliminating the need for customer surveys) both from the network and terminal sides.
6. Customer care: The ability of linking perceived (subjective) experience with measured (objective) QoE indicators may lead to significant benefits in terms of achieving a better insight onto customer perceived quality in a much more wide approach than the current one based on sampling of specific customers. QoE monitoring solutions are linked onto customer care centers by means of simplified interfaces and overall status for real-time access to customer-specific information, enhancing the response to customer quality and thus satisfaction. Customer care teams can rapidly diagnose problems and identify whether the root cause is linked to a badly performing network, mobile terminal, or application.

4. YouTube QoE pilot experiment

A set of 17 users (engineers from Telefónica company) were selected to participate in a pilot experiment, which consisted in periodically testing our YouTube QoE tool (installed on different Android smartphones) for 1 month. Every YouTube native session was transparently monitored and evaluated in two ways: 1) automatically by the application (from the QoE model previously described) and 2) by the users through an online opinion survey. A total number of 1,435 YouTube sessions were evaluated during the pilot. The data collected from each user device was sent to a server for post-processing purposes.

The pilot experience was carried out in Madrid (Spain), covering both rural and urban environments (as shown in Figure 4 on the left). Different colors represent the associated subjective quality (from the opinion survey) for a set of YouTube sessions. Such a survey (related to the video quality, audio quality, and overall quality) was requested to be filled after each YouTube session. Figure 4 on the right shows the probability density

Table 9 Percentage of reported MOS per technology

Connection type	MOS =1	MOS =2	MOS =3	MOS =4	MOS =5	Average	Standard deviation
UMTS	3.86	8.13	26.06	34.55	27.44	3.74	1.1
WiFi	3.4	7.46	24.41	46.78	21.02	3.81	0.87
Total	2.69	8.18	25.64	38.46	25.03	3.74	1

Table 10 Average, max, and standard deviation values (in seconds) for T_{init} as per technology and MOS

T_{init} (s)	UMTS				WiFi				Average	Standard deviation
	Ave	Min	Max	Std	Ave	Min	Max	Std		
MOS =1	98.44	6.05	439.2	99.91	-	-	-	-	87.94	96.53
MOS =2	22.9	3.15	147	29.86	13.54	2.87	106	21.59	20.36	27.55
MOS =3	16.77	3.21	167	24.84	8.27	2.39	60.68	8.04	14.67	21.83
MOS =4	11.31	2.69	83.2	11.21	7.67	2.66	154.2	12.99	9.87	12.39
MOS =5	9.30	3.33	94.7	9.85	7.00	2.49	35.50	5.11	8.61	8.57

distribution of the feedback associated to video, audio, and general quality.

According to the statistics collected at the QoE server, the majority of videos consumed by the users are short: nearly 90% of the videos are shorter than 5 min, and average duration is 160 s (see Table 4). Regarding the video characteristics, users had free access to the YouTube repository, so a wide variety of videos with different average bitrates (from 75 kbps to 3 Mbps depending on the resolution and codec) have been downloaded.

Next, statistics related to the application performance metrics (mainly referred to as T_{init}, f_{rebuf}, and T_{rebuf}, which are required to evaluate MOS) are analyzed in detail. Later, their effect on the experienced quality will be described.

First, a box-and-whiskers plot of the *initial buffering time* (T_{init}) per technology is shown in Figure 5. This non-parametric representation depicts quartiles as a box with median drawn as a vertical line inside the box; that is, 50% of values for T_{init} are included in the interval inside the box. Moreover, lines extending from the boxes (*whiskers*) indicate variability outside the upper and lower quartiles. Outliers lying further 1.5 times the inter-quartile range are plotted as individual points, and those further three times that range (extreme points) are besides filled up. Average value is shown as a red cross. A box-and-whiskers plot can be seen as a kind of summary of the cumulative distribution function (CDF) (whose estimation for T_{init} is shown in Figure 6) and a graphical representation of numerical measures, some of which are presented in Table 5.

Results from Figures 5 and 6 show that T_{init} values for WiFi connections are lower than those for UMTS. For 3G sessions, the estimated coefficient of variation (CV) is higher than 2, i.e., the standard deviation of T_{init} for UMTS connections is more than twice its average value. For WiFi, this dispersion measure is reduced to about 1.2. This comes from the fact that T_{init} samples are concentrated around the median for WiFi sessions, whereas UMTS presents a higher range. Such a heavy tail results in a higher average located in the last quartile. In any case, 50% of the videos have experienced an initial buffering time shorter than 7 s. In most connections, no rebuffering is necessary; thus, the median for the rebuffering frequency (f_{rebuf}) is 0 (see Table 6). However, in this case, f_{rebuf} is higher for WiFi than for UMTS; the reason is that, although the number of pauses is smaller for WiFi (Table 7), videos were shorter (see Table 4), thus boosting the frequency of interruption events even if the mean rebuffering time (T_{rebuf}) is lower (Table 8).

Now, we describe the effect of performance indicators in the reported MOS. Figure 7 shows the initial buffering time (T_{init}) box-and-whiskers plot per MOS. As shown in the results, lower T_{init} values are associated with higher MOS. Although a higher feedback quality could be expected for WiFi than for UMTS, it can be observed that users do not assign a significantly lower MOS for UMTS than for WiFi (see Figure 8 and Table 9) even if, from an objective point of view, their performance is better (a summary of the three studied performance indicators can be found in Tables 10, 11 and 12). That is, although WiFi connections achieve

Table 11 Average, max, and standard deviation values for f_{rebuf} as per technology and MOS

f_{rebuf}	UMTS			WiFi			Average	Standard deviation
	Ave	Max	Std	Ave	Max	Std		
MOS =1	13e – 3	0.03	8.7e – 3	-	-	-	0.013	9e – 3
MOS =2	3.6e – 3	0.025	6.8e – 3	1.4e – 3	0.016	4.13e – 3	2.8e – 3	6e – 3
MOS =3	4.3e – 3	0.046	8.6e – 3	2.1e – 3	0.1	13e – 3	3.5e – 3	10.4e – 3
MOS =4	1.2e – 3	0.022	3.8e – 3	6e – 5	5.5e – 3	5.4e – 4	6.6e – 4	2.9e – 3
MOS =5	0	0	0	7e – 5	4e – 3	5.4e – 4	2e – 5	3.1e – 4

All min values are 0.

Table 12 Average, max, and standard deviation values (in seconds) for mean rebuffering time as per technology and MOS

T_{rebuf}	UMTS			WiFi			Average	Standard deviation
	Ave	Max	Std	Ave	Max	Std		
MOS =1	520.88	2,593.56	768.38	-	-	-	494.84	756.9
MOS =2	28.26	372.42	71.18	4.28	53.03	12.73	19.75	58.56
MOS =3	11.25	164.127	32.97	0.99	59.21	7.02	7.56	27.13
MOS =4	2.71	98.056	11.68	0.11	9.07	0.92	1.54	8.78
MOS =5	0	0	0	0.23	14.07	1.79	0.07	1.00

All min values are 0.

better QoS figures, MOS values are very similar to those of 3G. The reason for that might be that subjective users' expectations could be influenced by the type of connection being used. Hence, users might penalize the QoE of WiFi connections due to a higher expected quality. We expect similar or even more demanding user attitude from 4G users (not extensively available at the time of the survey).

Next, the appropriateness of the theoretical model in Equation 1 is analyzed by evaluating the correlation between the theoretical MOS and the MOS reported by users, resulting in a correlation factor of 0.97, i.e., the coefficient of determination R^2 for linear regression through origin is 93.93%. Figure 9 on the left shows the difference between MOS results provided by the theoretical model (see Equation 1) and MOS reported from the users' opinion survey. It can be observed that the QoE model provides an estimation which falls within ±0.5 of the reported score in 23% of sessions. In general, the model in [23] provides more pessimistic results than the opinion of users as the estimated MOS is lower than that reported for about 68% of sessions. A simple modification results from taking a linear regression between the modeled MOS and MOS as reported by users, yielding to:

$$\text{MOS}_{\text{mod}} = 1.1935 \cdot \text{MOS}_{\text{QoSmodel}} \tag{2}$$

Note that this measurement indicates that MOS is about 20% higher than that given in Equation 1. The reason could be that users could be more indulgent with wireless connections than for wired scenarios under which the original model was obtained.

Due to regression properties, the average value for the difference between MOS as obtained by Equation 2 and that reported by users (that is, the residuals) is 0, although no symmetry around 0 exists (see Figure 9 on the right). Differences between subgroups per technology are not significant (estimated slope of 1.1995 for WiFi connections and 1.2089 for UMTS). It was explored whether a multivariant regression could improve

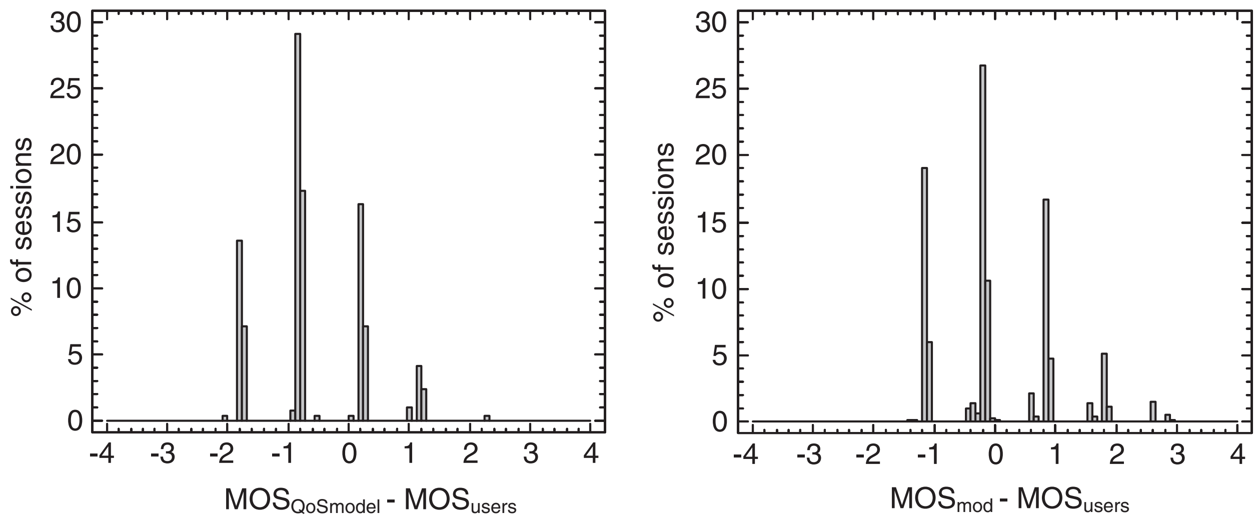


Figure 9 Histogram for the difference between original QoS model (left) or modified QoS model (right) and as reported by users.

those results. Only linear regression was analyzed as a modification of numerical quantities as those proposed in [28] cannot be easily included in the multivariant procedure. The adjusted R^2 including all available parameters results in 90.5%, only a bit lower (90.46%) if the total rebuffering time is taken out from regression. As this value is lower than that obtained with Equation 1, the heuristical measurement quantization proposed in [23] increased by 20% seems to be able to predict well the users' expectations.

5. Conclusions

This work has presented a QoE evaluation tool for Android terminals which is able to estimate the QoE (in terms of MOS) for YouTube service based on theoretical models. In particular, this tool makes it possible to map network QoS onto the QoE of YouTube sessions. Additionally, a QoE advices module analyzes possible causes that lead to low QoE and subsequently provides particular advices to the user under certain conditions.

Our application has been tested on a pilot experience over 17 Android terminals for 1 month. According to the statistics, most of the responses from the users' survey match up with theoretical estimations; however, the QoE model provides slightly more pessimistic results than the opinion of the wireless users, probably as the model was initially generated under wired scenarios. In that sense, we propose a modified utility function from taking a linear regression between the theoretical MOS and the MOS reported by users.

In our opinion, it is critical that application developers provide access to the main key performance indicators (KPIs) associated to their services in order to ease the evaluation and analysis of the QoE.

Competing interests
The authors declare that they have no competing interests.

Authors' information
GG received his B.Sc. and Ph.D. degrees in Telecommunications Engineering from the University of Málaga (Spain) in 1999 and 2009, respectively. From 2000 to 2005, he worked at Nokia Networks and Optimi Corporation (recently acquired by Ericsson), leading the area of QoS for 2G and 3G cellular networks. Since 2005, he is an associate professor at the University of Málaga. His research interests include the field of mobile communications, especially QoS/QoE evaluation for multimedia services and radio resource management strategies for LTE and LTE-Advanced.
LH is a telecommunications engineer from E.T.S.I.T - UPM, specialized in computer science. He has worked mainly in defense aerospace industry as a military avionics engineer and in wireless telecommunication industry as systems engineer. He also worked for public administration as a consultant in computer science and also as a contract reviewer. Since 1996, he has worked for Telefónica I + D, always involved in mobile network R&D. Recently, he worked at the area of PDI - Enabling Platforms (SLA) as a computer technology specialist. Currently, he is working at the area of PTI - Capacity & Traffic Analysis & Solutions, developing tools based on knowledge extraction, machine learning, and analytic prediction techniques.
QP is a physicist for the Complutense University of Madrid from 1986. In 1987, he started working in Telefónica I + D, in aspects related with reliability predictions, failure analysis, and components electronic testing, used in telecommunication systems. In 2011, he was working in QoE issues for mobile broadband networks, analyzing and developing tools and client applications to control and manage the QoS and QoE of mobile services. Currently, he is working in GCTO (Global Chief Technology Office) of Telefónica, analyzing and defining the requirements needed for residential cellular gateways.
JL received his B.Sc. in Telecommunications in 1998 in Universidad Politécnica de Madrid. In 1999, he worked for Teldat on ciphering techniques, and since 2000, he is working in Telefónica I + D on several areas related to mobile communications and physical layer performance, involving terminal specifications and testing, link-level and system-level simulations of 3G/3.5G systems, LTE digital signal processing, and LTE-Advanced. He is currently working on quality of experience in wireless networks.
RG received her B.Sc. degree in Telecommunications Engineering in 1998 from Madrid Polytechnics University. Between 1997 and 1998, she staged at the Digital Communication Systems Department at the Technical University Hamburg-Harburg (Germany), where she carried out her master thesis about multipath fading channel modeling. In July 1998, she signed for Telefónica I + D, becoming part of the Radio Communication Systems Department. Her career has oriented towards mobile communications, especially on radio planning and optimization and QoE in LTE.
MCAT received M.S. and Ph.D. degrees in Telecommunications Engineering from the University of Malaga, Spain, in 1994 and 2001, respectively. Currently, she is working at the Department of Communications Engineering, at the same university. Her main research interests include adaptive modulation and coding for fading channels, multi-user OFDM, SC-FDMA, cross-layer design, and probabilistic QoS guarantees for wireless communications.

Acknowledgements
This work has been partially supported by the Spanish Government and FEDER (TEC2010-18451 and TEC2013-44442).

Author details
[1]Department of Communications Engineering, University of Malaga, 29071 Malaga, Spain. [2]Telefónica I + D, 28006 Madrid, Spain.

Received: 21 January 2014 Accepted: 30 September 2014
Published: 10 October 2014

References

1. Sandvine Corporation, *Global Internet Phenomena Report, 1H, 2012*, 2013. Available at: http://www.sandvine.com/news/global_broadband_trends.asp
2. A Díaz, P Merino, FJ Rivas, Customer-centric Measurements on Mobile Phones, in *Proceedings on 12th IEEE International Symposium on Consumer Electronics*, Vilamoura, Portugal, 14–16 April 2008, pp. 14–16. doi:10.1109/ISCE.2008.4559470
3. I Ketykó, K De Moor, T De Pessemier, AJ Verdejo, K Vanhecke, W Joseph, L Martens, L De Marez, QoE measurement of mobile YouTube video streaming, in *MoViD'10 Proceedings of the 3rd Workshop on Mobile Video Delivery*, Firenze, Italy, 25–29 October 2010, pp. 27–32. doi:10.1145/1878022.1878030
4. K De Moor, I Ketykó, W Joseph, T Deryckere, L De Marez, L Martens, Proposed framework for evaluating quality of experience in a mobile, testbed-oriented living lab setting. Mobile. Netw. Appl. **15**(3), 378–391 (2010). doi:10.1007/s11036-010-0223-0
5. A Alvarez, S Cabrero, XG Paneda, R Garcia, D Melendi, R Orea, A flexible QoE framework for video streaming services, in *IEEE GLOBECOM Workshops*, Houston, Texas, USA, 5–9 December 2011, pp. 1226–1230. doi:10.1109/GLOCOMW.2011.6162377
6. M Ghareeb, C Viho, Hybrid QoE Assessment is Well-Suited for Multiple Description Coding Video Streaming in Overlay Networks, in *2010 Eighth Annual Communication Networks and Services Research Conference (CNSR)*, Montreal, QC, Canada, 11–14 May 2010, pp. 327–333. doi:10.1109/CNSR.2010.15
7. J Klaue, B Rathke, A Wolisz, EvalVid - a framework for video transmission and quality evaluation, in *Proceedings of 13th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, Urbana, IL, USA, 2–5 September 2003, pp. 255–272. doi:10.1007/978-3-540-45232-4_16

8. E Aguiar, A Riker, E Cerqueira, A Abelém, M Mu, T Braun, M Curado, S Zeadally, A real-time video quality estimator for emerging wireless multimedia systems. Wireless Networks **20**(8), 1–18 (2014). doi:10.1007/s11276-014-0709-y
9. K Laghari, TT Pham, H Nguyen, N Crespi, QoM: A New Quality of Experience Framework for Multimedia Services, In *Proc. IEEE Symp. Computers and Communications (ISCC)*, Cappadocia, Turkey, 1–4 July 2012, pp. 851–856. doi:10.1109/ISCC.2012.6249408
10. R Schatz, T Hoßfeld, P Casas, Passive YouTube QoE monitoring for ISPs, in *Proc. of Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, Palermo, Italy, 4–6 July 2012, pp. 358–364. doi:10.1109/IMIS.2012.12
11. B Staehle, M Hirth, R Pries, F Wamser, D Staehle, YoMo: a YouTube Application. Comfort Monitoring Tool, in *Proc. of EuroITV Workshop QoE for Multimedia Content Sharing (QoEMCS)*, Tampere, Finland, 9 June 2010
12. T Hossfeld, M Seufert, M Hirth, T Zinner, P Tran-Gia, R Schatz, Quantification of YouTube QoE via crowdsourcing, in *IEEE International Symposium on Multimedia (ISM)*, Dana Point, California, USA, 5–7 December 2011, pp. 494–499. doi:10.1109/ISM.2011.87
13. T Hossfeld, R Schatz, S Egger, M Fiedler, K Masuch, C Lorentzen, Initial Delay vs. Interruptions: Between the Devil and the Deep Blue Sea, in *Proc. of 4th International Workshop on Quality of Mulitmedia Experience (QoMEX)*, Yarra Valley, Victoria, Australia, 5–7 July 2012, pp. 1–6. doi:10.1109/QoMEX.2012.6263849
14. T Porter, X Peng, An objective approach to measuring video playback quality in lossy networks using TCP. IEEE Commun. Lett. **15**(1), 76 (2011). doi:10.1109/LCOMM.2010.110310.101642
15. RKP Mok, EWW Chan, RKC Chang, Measuring the Quality of Experience of HTTP Video Streaming, in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IFIP/IEEE IM)*, Dublin, Ireland, 23–27 May 2011, pp. 485–492. doi:10.1109/INM.2011.5990550
16. P Gill, M Arlitt, Z Li, A Mahanti, YouTube traffic characterization: a view from the edge, in *Proc. of the 7th ACM SIGCOMM Conference on Internet Measurement*, San Diego, California, USA, 24–26 October 2007, pp. 15–28. doi:10.1145/1298306.1298310
17. P Ameigeiras, JJ Ramos-Munoz, J Navarro-Ortiz, JM Lopez-Soler, Analysis and modeling of YouTube traffic. Trans. Emerg. Telecommunications Technol. **23**(4), 360–377 (2012). doi:10.1002/ett.2546
18. ULC Sandvine Incorporated, *White paper: "Measuring Over-the-Top Video Quality"*, 2013. Available from: https://www.sandvine.com/downloads/general/whitepapers/measuring-over-the-top-video-quality.pdf
19. RC Gonzalez, P Wintz, *Digital Image Processing*, 2nd edn. (Addison-Wesley, Essex, England, 1987)
20. Z Wang, L Lu, AC Bovik, Video quality assessment based on structural distortion measurement. J. Signal Process: Image Commun. **19**(2), 121–132 (2004). doi:10.1016/S0923-5965(03)00076-6
21. ITU-T, *Recommendation J.247. Objective perceptual multimedia video quality measurement in the presence of a full reference*, 2008. Available at: http://www.itu.int/rec/T-REC-J.247-200808-I/en
22. ITU-T, *Recommendation J.144. Objective perceptual video quality measurement techniques for digital cable television in the presence of a full reference*, 2004. Available at: http://www.itu.int/rec/T-REC-J.144-200403-I/en
23. A Khan, L Sun, E Ifeachor, Content-based video quality prediction for MPEG4 video streaming over wireless networks. J. Multimedia **4**(4), 228–239 (2009). doi:10.4304/jmm.4.4.228-239
24. VD Bhamidipati, S Kilari, *Effect of Delay/Delay Variable on QoE in Video Streaming*. M. Thesis (School of Computing at Blekinge Institute of Technology, Sweden, 2010)
25. L Janowski, P Romaniak, QoE as a function of frame rate and resolution changes, in *Proc. of the 3rd International Workshop on Future Multimedia Networking (FMN)*, Krakow, Poland, 17–18 Jun 2010, pp. 34–45. doi:10.1007/978-3-642-13789-1_4
26. J Padhye, V Firoiu, D Twosley, J Kurose, Modeling TCP Reno performance: a simple model and its empirical validation. IEEE/ACM Trans. Networking **8**(2), 133–145 (2000). doi:10.1109/90.842137
27. D Zheng, GY Lazarou, R Hu, A Stochastic Model for Short-lived TCP Flows, in *Proceedings of the IEEE International Conference on Communications*, Anchorage, Alaska, USA, 11–15 May 2003, pp. 76–81. doi:10.1109/ICC.2003.1204146
28. M Fiedler, T Hossfeld, P Tran-Gia, A generic quantitative relationship between quality of experience and quality of service. IEEE Netw. **24**(2), 36–41 (2010). doi:10.1109/MNET.2010.5430142

doi:10.1186/1687-1499-2014-164
Cite this article as: Gómez *et al.*: **YouTube QoE evaluation tool for Android wireless terminals.** *EURASIP Journal on Wireless Communications and Networking* 2014 **2014**:164.