Concept and Implementation of Video QoE Measurements in a Mobile Broadband Testbed

Anika Schwind*, Michael Seufert*, Özgü Alay†, Pedro Casas‡, Phuoc Tran-Gia*, Florian Wamser*

*University of Würzburg, Institute of Computer Science †Simula Research Laboratory ‡AIT Austrian Institute of Technology
Würzburg, Germany — Fornebu, Norway — Vienna, Austria
{anika.schwind | seufert | trangia | florian.wamser}@informatik.uni-wuerzburg.de
ozgu@simula.no
pedro.casas@ait.ac.at

***Abstract*—The MONROE testbed enables the objective performance assessment of Mobile Broadband (MBB) networks from the end-user perspective, using highly distributed measurements from fixed and mobile nodes. To quantify the performance of MBB networks for popular Internet services from a user-centric perspective, dedicated tools are needed. In this paper we extend the MONROE testbed to the Quality of Experience (QoE) domain, presenting the design and implementation of a QoE-capable measurement tool for YouTube video streaming. The measurement concept is based on emulating a virtual end-user device requesting video streams, which are then monitored at the network and application layers, on the basis of QoE-relevant features. The initial measurements conducted in the MONROE testbed and reported in this paper demonstrate the applicability of the implemented measurement concept.**

***Index Terms*—Distributed Active Measurements; Cellular Networks; QoE; YouTube Video Streaming; YoMo.**

I. Introduction

The evolution of technology, especially of mobile phones, has changed the way we use the Internet. Smartphones are today the most typical device to access the Internet, and the trend is growing. According to Cisco's global mobile data traffic forecast [1], smartphones will be responsible for more than three-quarters of the mobile data traffic generated by 2019. Mobile video traffic takes the main share in this striking growth, already accounting for more than 60% of the total mobile data traffic in 2016. In the light of these trends, mobile broadband (MBB) network operators are becoming more and more interested in understanding how to dimension their networks and how to manage their customers' traffic to capture as many of these new mobile Internet users as possible.

Quality of Experience (QoE) provides a plausible solution to this network management problem. Closely linked to the subjective perception of the end-user, QoE enables a broader understanding of the factors that influence the performance of systems from the end-user perspective, complementing traditional technology-centric concepts such as QoS. The need for QoE-aware approaches for improved network operation has boosted the research interest in scaling QoE out of the traditional lab setup. However, dealing with QoE in operational environments is challenging and involves proper monitoring and understanding of the interplays between end-user satisfaction, application behavior, and network performance.

The MONROE[1] project has conceived the first European transnational open platform for independent, multi-homed, large-scale monitoring and performance assessment of MBB networks. Being one of the goals of the MONROE project to measure and analyze the QoE of MBB networks, this paper presents the design, implementation and initial experiments conducted with a QoE-capable measurement tool for video streaming in MBB networks. The measurement approach targets YouTube as video streaming platform, given its overwhelming popularity. In a nutshell, this QoE monitoring tool implements an hybrid-measurements monitoring approach, firstly by emulating a virtual end-user device requesting video streams, and then by passively monitoring the resulting video traffic at both the network and application layers. The QoE-relevant monitored KPIs (e.g., throughput, stalling, playback delay, video quality level, etc.) can be used for multiple purposes, for example to analyze and benchmark the QoE of different MBB networks, to optimize the operation of the streaming protocols, to construct QoE prediction models which can be further integrated into large-scale monitoring systems, and many more. By enabling pervasive measurement of YouTube KPIs at all the layers of the stack (i.e., network and application), the proposed system benefits everyone: operators can get a fine-grained picture of the YouTube QoE-based network status, empowering effective management and operation; end-users and regulators can objectively compare the user-centric performance offered by different operators, improving competition in the market and even verifying adherence to SLAs; researchers can get access to large-scale QoS/QoE measurements directly taken in the operational field, opening the door to many interesting problems; application developers gain powerful insights for handling performance issues of their video-streaming applications.

To demonstrate the applicability and monitoring capabilities of the proposed approach, we additionally report the analysis of the first initial measurements collected in the MONROE

[1]The MONROE project is funded by the European Union's H2020 research and innovation programme under grant agreement No.644399. For more information, visit https://www.monroe-project.eu/

testbed with the proposed system. Measurements were conducted during three and a half weeks of February 2017, running the YouTube QoE monitoring tool on six different testbed nodes, distributed in Sweden, Italy and Norway.

The remainder of the paper is organized as follows: Sec. II presents an overview of the related work, focusing on the specific case of QoE-aware network monitoring. Sec. III overviews the MONROE testbed and the MONROE measurement setup. Sec. IV presents the design and implementation of the proposed QoE monitoring tool for video streaming. Sec. V describes the experiments and collected measurements, reporting and discussing the obtained results. Finally, Sec. VI concludes this work.

II. Related Work

There is a long literature in the study of QoE for video streaming services such as YouTube, and it is well accepted today that stalling, initial playback delays and quality switches are the most relevant Key Performance Indicators (KPIs) for adaptive streaming QoE. A comprehensive survey on adaptive video streaming QoE is provided in [2].

Regarding QoE-related measurements for YouTube, previous work has already proposed different tools to collect QoE-relevant KPIs at end-devices, including YoMo [3], YouSlow [4], and YoMoApp [5], [6] among others. These tools are capable to passively collect QoE-relevant features for YouTube video streaming in desktop PCs and smartphones mainly at the application layer side. There are also multiple studies on the analysis of QoE for cellular networks and smartphones using end-device measurements and crowdsourced, end-user QoE measurements [7], [8]. Dedicated testbeds for network measurements mainly focus on fixed core networks, such as PlanetLab [9], G-Lab [10], and GENI [11].

In [12] authors present a system for on-line monitoring of YouTube QoE in cellular networks using in-network measurements only. In [13], authors introduce QoE Doctor, a tool to measure and analyze mobile app QoE, based on active measurements at the network and application layers. Additional papers in a similar direction tackle the problem of modeling QoE and user abandonment for video [14], [15], using both end device and network measurements. Papers such as [16], [17] take a step further and develop different approaches to predict the QoE of mobile users using passive in-network and in-device measurements, applying machine learning techniques to obtain mappings between QoS and QoE.

III. MONROE Testbed

A. Testbed Description

The MONROE platform is the first open access hardware-based platform for independent, multihomed, large-scale experimentation in commercial MBB heterogeneous environments [18]. Figure 1 provides an overview of the main building blocks. The platform comprises a set of 250 nodes[2], both

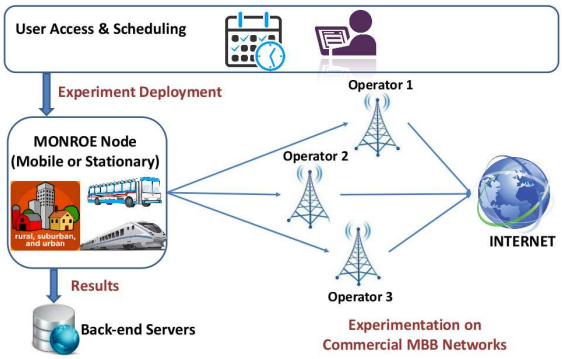


Fig. 1: Key components of the MONROE platform. MONROE users access resources and deploy their experiments via the User Access and Scheduling function. Measurement results are periodically synchronized to external repositories.

mobile (e.g., operating in delivery trucks and on board public transport vehicles, such as trains or busses) and stationary (e.g., volunteers hosting nodes in their homes). Nodes are multihomed to three different MBB operators using commercial subscriptions in five countries in Europe (Spain, Italy, Sweden, Norway and the UK). All software components used in the platform are open source and available online[3].

Each MONROE Node integrates two small programmable computers (a PC Engines APU2 board[4], interfacing with three 3G/4G MC7455 miniPCI express (USB 3.0) modems[5] using Long-Term Evolution (LTE) CAT6 and a WiFi modem. LTE CAT6 can connect to two bands simultaneously.

The node software is based on Debian GNU/Linux *"stretch"*[6]. Linux provides accessibility of the source code, flexibility and community maintenance to ensure interoperability with other systems and flexibility in the hardware required to support research and implementation of transport protocols.

Each Node runs: (i) *the management software* ensures that the node remains operational (e.g. MBB modems correctly configured and connected, routing enabled) and enable remote update of all the other software components, (ii) *the maintenance software* that monitors operational status and reduces the need for manual maintenance and (iii) *the experimentation enablers*, facilitating experiment deployment (via the scheduler client) and feeding rich context information to the experiments.

The management software provides: (i) a Device Listener to detect, configure and connect USB network interfaces, (ii) a routing daemon that use DHCP to acquire an IP address and set up routing tables and (iii) a network monitor to monitor interface state, check connectivity and configures default routes. The Metadata Multicast collects and multicasts metadata from the MBB modems and the node, such as node status, connection technology, GPS location or node sensor

[2]At the time of writing, 50 nodes have been deployed, with deployment scheduled to be completed by June 2017

[3]https://github.com/monroe-project/
[4]http://www.pcengines.ch/apu.htm
[5]https://techship.com/products/sierra-wireless-mc7455-lte-cat6/
[6]https://www.debian.org/releases/stretch/

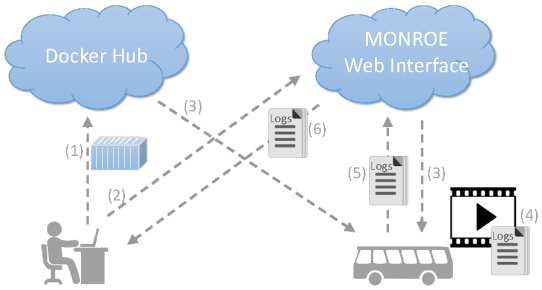


Fig. 2: Procedure for scheduling an experiment in the MONROE testbed in six steps.

data. A messaging API relays real-time metadata in JSON format to experiments through ZeroMQ[7].

Experiments running on the platform use the Docker containers (i.e. light-weight virtualized environment) to provide agile reconfiguration. This also controls access to external actions in a node.

User Access to the platform resources is through a web portal, which allows an authenticated user to use the OpenMBB scheduler to deploy experiments. This enables exclusive access to nodes (i.e. no two experiments run on the node at the same time). The results from each experiment are periodically transferred from the nodes to a repository at a Back-end server, while the OpenMBB scheduler also sets data quotas to ensure fairness among users.

B. Measurement Setup

Figure 2 illustrates how an experiment can be scheduled in the MONROE testbed by showing the procedure in six steps labeled by the numbers (1) up to (6). First, a Docker container which includes the experiment has to be designed and then pushed to DockerHub[8], a cloud-based registry service (1). To start an experiment, the MONROE web interface has to be used (2). Here, the DockerHub repository of the experiment's container, the node on which the experiment should run and other parameters have to be selected to schedule the measurement. Next, the MONROE scheduler informs the node that an experiment is scheduled and the node downloads the image of the Docker container from DockerHub (3). Then, at the defined starting time, the node runs the experiment and generates the resulting log files from the monitoring (4). After that, all log files and other corresponding files were uploaded to the MONROE database (5) which offers the log files for download for the project members (6).

IV. Measurement Container Setup

The aim of this work is to design and implement video QoE measurements in a MBB testbed. Therefore, a measurement tool for YouTube video streaming was implemented and initial experiments were conducted in the MONROE testbed.

[7]http://zeromq.org/
[8]https://hub.docker.com/

A. Container Design

To perform experiments in the MONROE testbed, a Docker container has to be designed. MONROE provides a base image that contains a set of packages and libraries. The experimenter builds their containers using the base image, to minimize the size of the container. This is crucial considering the limited data quotas of MBB networks.

Required Packages: In the following, a list of the most important packages which were required for our measurement container is explained. To measure network traffic information during the whole experiment, the network protocol analyzer *tshark*[9] is used. Tshark allows to capture packet data from a live network like, for example, the TCP segment length or the source address. For video streaming, the open-source browser *Mozilla Firefox*[10] is chosen. To simulate a YouTube video request in a Firefox browser within the Docker container, we are using the web browser automation tool *Selenium*[11]. The YouTube video is monitored with our Firefox add-on YoMo, which will be explained in more detail later. Moreover, to run the application headless in the MONROE platform, which does not provide any physical screen, the *X virtual frame buffer (Xvfb)*[12] is chosen. The X virtual frame buffer is used to create a fake display port of the Firefox server, so headless Firefox can be run via the X server. Moreover, the hidden screen can be sized over a specific option. Except for tshark, all required packages were included in the MONROE base image.

Container Execution: When starting the container, a bash script is automatically called, which takes the following actions before gaining measurement data. First, the Firefox browser has to be configured. Since the browser is started headless and there is no one who can interact with it, it is important that no pop-up message appears. Thus, for example, the pop-up, in which the browser checks whether it is set as default or not, has to be deactivated. Furthermore, it has to be allowed that XPI (Cross-Platform Install) data can be installed without signatures, since the YouTube video monitoring add-on YoMo has to be installed this way. Additionally, the cache has to be disabled to not distort the measurement results. After setting up the browser, tshark is started to measure the network traffic information during the video playback. Afterwards, a python script, which handles the video playback with selenium, can be started and the monitoring begins.

B. Monitored Data

While a YouTube video is played on a MONROE node within the Docker container, several performance indicators are monitored. Therefore, not only the application layer, but also the network and the metadata layer have to be observed. Parameters on application layer are monitored via the Firefox browser add-on YoMo. The network traffic information is measured with the help of tshark, an open source packet analyzer.

[9]https://www.wireshark.org/docs/man-pages/tshark.html
[10]https://www.mozilla.org/en-US/
[11]http://www.seleniumhq.org/
[12]https://packages.debian.org/en/wheezy/xvfb

Metadata are automatically stored for every MONROE node and can be retrieved from the MONROE back-end.

Application Layer: To monitor the key performance indicators (KPIs) on application layer, the browser plugin YoMo was used. YoMo is a JavaScript based Firefox add-on, which monitors the media events and the media properties of the HTML5 player on YouTube. It allows, for example, to determine exactly the buffered playtime and the streamed video quality (i.e., resolution).

The YoMo add-on listens to player events and stores them in the video event log file. These events include the player state like stalling, seeking, paused, playing, and ended, as well as changes in the duration of the video, the current playback quality and volume. Additionally, also the YouTube ID and the title were stored. To handle the events, which are throw by the HTML5 video player, a JavaScript event listener is added. As soon as an event is detected, the listener stores it, together with the current timestamp. In order to detect changes in the video quality, duration, or volume, as well as to detect if a new video is played (change of the YouTube ID and the title), a JavaScript function to check these parameters in specified time steps is used. Whenever a parameter has changed, the new value is logged with the corresponding timestamp.

Not only player events, but also information about the buffer level is measured by the YoMo add-on. Every second a snapshot of the current video playback time, the buffered video playback time, and the available playback time is taken and stored in a log file.

To monitor the network activity, a JavaScript HTTP activity observer was used. Therefore, the *nsIHttpActivityObserver*[13] interface has to be implemented. Here, request information and the request headers were watched and logged in the response header log file. Request information are the information from the browser, whereas on the contrary, response headers are sent from the web server. Request headers tell the server about the client, for example, what it can accept or what exactly it wants. This includes information, for instance, the mime type, itag, and the requested range. The server returns headers detailing the object type, size, validity and so on.

Network Layer: To monitor the network traffic during the experiment, tshark was used. The captured packet data include, for example, a timestamp, information about the frame length, and the source and destination address. Thus, for example, the throughput during the video playback can be calculated.

Metadata Layer: Each MONROE node generates metadata, like information about the node's CPU or location, which are stored in a database. A daily dump of the metadata of all nodes is generated by the MONROE project and is offered via the MONROE web interface.

V. Study Description and Results

A. Measurement Description

To analyze the Quality of Experience in mobile networks, measurements were started in the MONROE testbed. An experiment starts by checking the Internet access and selecting the network interface. MONROE nodes offer three different interfaces, which are fixed to a network operator in the given country. Since not all interfaces might be available on every node, first, a list of the active interfaces has to be generated. Therefore, each interface is set successively as default route and a website is pinged to check whether the Internet access is available and on this interface. Finally, one of the available network interfaces is selected for the measurement and the monitoring of the network traffic on the selected interface is started.

After that, the main part of the experiment is executed. A Firefox browser is started and a specified baseline video is requested. This video is the same in all measurements to compare the streaming performance of different measurement runs without the impact of the video content. The selected baseline video is a movie trailer with duration 2:33 minutes, which was published on March 8, 2014[14]. It was chosen because it is available in many quality levels (up to 2160p) and is also very popular, having over six million views.

In addition to the baseline video, a random second video is streamed in every measurement to compare the streaming for different video contents. Due to the vast amount of possible video contents, a pre-selection had to be conducted to obtain a significant number of measurement results per video content. This means, a core set of five videos with different numbers of views, durations, categories, release dates, and available video qualities was defined, which have a probability of 12% each to be selected as the second video.

With a probability of 40 %, a video is selected randomly from an extended video set. This set should cover a broad variety of YouTube videos, such that the insights from the measurement results can be checked for generality. Therefore, a list of 1 000 YouTube video was obtained from the website *Random YouTube Videos*[15]. To keep the measurement node occupation short, from this list, only 736 videos with a duration lower than 10 minutes were considered.

After the streaming of the two videos, the browser is closed, the network capturing is terminated, and all monitored data is written to log files. Finally, the generated log files from the YoMo add-on and the network traffic monitoring are uploaded from the measurement node to the MONROE database. Via the MONROE websites, the log files can be accessed and downloaded. Additionally, also the metadata files are gathered from the MONROE database.

B. Study Description and General Streaming Results

The first long term measurement study was started on February 2, 2017 at 12:17 am GMT+1, and ran until February 27, 2017 at 12:15 am GMT+1. During this time, 141 experiments were performed. As the testbed is still in the development phase, several measurements could not be finished successful. Thus, only the results of 89 successful measurements could be collected for evaluation.

[13] https://developer.mozilla.org/en-US/docs/Mozilla/Tech/XPCOM/Reference/Interface/nsIHttpActivityObserver

[14] https://www.youtube.com/watch?v=QS7lN7giXXc

[15] http://randomyoutube.net/

These experiments were scheduled on six different nodes of the MONROE testbed, two in Karlstad (Sweden), two in Torino (Italy), and two in Oslo (Norway). All in all, 165 video streaming sessions, which comprised 30 different YouTube IDs during all measurement runs, were monitored. The measurements of 13 video streaming sessions could not be used for further evaluations because they were aborted for unknown reasons. The average video duration was about 192.99 s while the maximal video duration was 569.34 s and the minimal was 19.49 s. On average, the mean throughput of all videos was 370.93 kbps. The lowest throughput was 124.39 kbps, while the highest throughput was 812.72 kbps.

With respect to the QoE of adaptive video streaming sessions, especially, initial delay, stalling, and quality adaptation have to be considered. The average initial delay of all video playbacks was 6.43 s, while the minimal initial delay was 1.32 s and the maximum was 18.09 s. It can be seen that long initial delays can occur during mobile video streaming, which negatively affect the QoE [19]. The same conclusion can be drawn for stalling, i.e., playback interruptions due to video buffer depletion, which are the worst QoE degradations of video streaming. [20] found that users tolerate at most one stalling event of at most 3 s duration. Although, during the study, 77.58 % of the streaming sessions showed no stalling, the mean total stalling time of the other sessions was 4.21 s and the maximum total stalling time was 25.64 s. This indicates that stalling is a severe problem in this mobile network, which results in unsatisfied end users with poor QoE.

Taking a look at the played out quality layers, the following resolutions were observed with the respective ordinal rank of the resolution in brackets: 720p (8), 480p (7), 360p (6), 270p (5), 240p (4), 180p (3), 144p (2), and 108p (1). The overall mean of the weighted time on quality layers, i.e., the time average of the ordinal quality rank, was 4.56, which means that the quality was on average between 270p and 240p. The average start and end quality was 360p and the average number of quality switches was 3. This indicates that often the throughput in the mobile network fluctuated and, as the time average was below 360p, was not high enough to support 360p throughout the whole streaming session.

C. Streaming Results for Baseline Video

In the following, only the measurement data of the baseline video are evaluated to obtain statistically significant results without video content dependencies. In total, the playback of the baseline video was measured 88 times at different daytimes and at different nodes.

Figure 3a shows the cumulative distribution function of the mean throughput during each video playback. The mean throughput of all videos was 394.31 kbps. Only 20 % of the sessions had a considerably lower throughput below 359.99 kbps, and only 10 % had a higher throughput above 453.42 kbps. The lowest throughput was 177.96 kbps, while the highest throughput was 505.15 kbps.

The cumulative distribution function of the initial delay can be seen in Figure 3b. The average initial delay of all video playbacks of the baseline video was 5.06 s, while the minimal initial delay was 2.79 s and the maximal was 13.55 s. As initial delay is generally considered a minor QoE degradation, from the QoE results on initial delay [19], this distribution suggests that all of the sessions had an acceptable initial delay.

In Figure 3c, the distribution of the total stalling time of the baseline video is depicted. 84.09 % of the videos could be streamed without stalling. The average stalling time of the videos with stalling (15.91 %) was 0.83 s, while the maximum observed stalling time was 6.16 s.

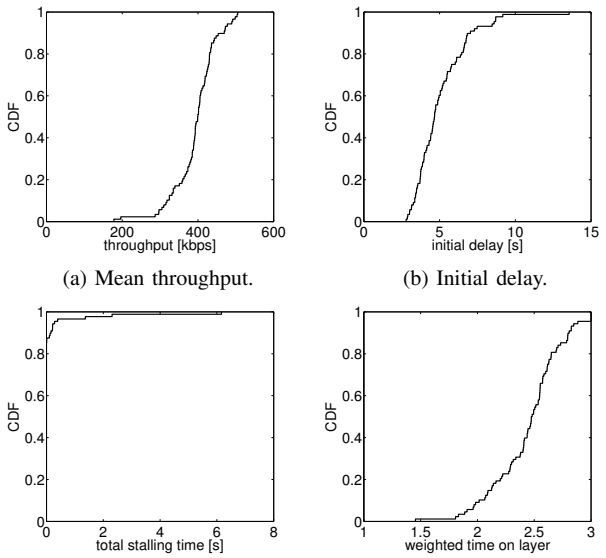
Figure 3d shows the distribution of the weighted time on (ordinal) quality layers of all measurements of the baseline video in the MONROE testbed. Note that only three quality layers were played back, namely 144p (1), 240p (2), and 360p (3). The overall mean of the weighted time on quality layers was 2.45, which indicates an average quality between 240p and 360p. The distribution of each quality layer is depicted in Figure 4a. It can be seen that 360p was played out most of the time (53.86 %), followed by 240p (37.63 %). Layer 1 (144p) was played out only 8.51 % of the time.

Every playback started on 360p, while the average number of quality switches was 2.53 with a median of 3 switches per video playback. 92.05 % of the videos also ended on 360p. The minimal number of quality switches was 0 (4.55 %) and the maximal number 4 (1.14 %). The reason that the quality of 360p cannot be supported all the time is that the average bit rate of the 360p quality layer is 765.87 kbps, which is larger than the received average and maximum throughput. It also follows that the playback of videos without quality switches faced frequent stalling events.

Figure 4b investigates the distribution of quality switch amplitudes. The notation $a > b$ indicates a quality switch from layer a to layer b. With 61.4 %, the probability for switching up the quality layer was higher than for switching down. This behavior is caused by the fact that down-switches with a large amplitude are possible, while up-switches only have a small amplitude. It can be seen that YouTube mostly switched up only one layer at a time, as the probability of a switch $1 > 3$ was below 1 %. When reducing the quality, the large switch amplitude $3 > 1$ occurred more frequently (26.9 %) than $3 > 2$ (10.8 %). If the resolution was already at 240p, a down-switch to 144p rarely occurred with a probability below 1 %. This confirms findings from [21] that mobile YouTube shows a very conservative adaptation behavior.

VI. Conclusion and Outlook

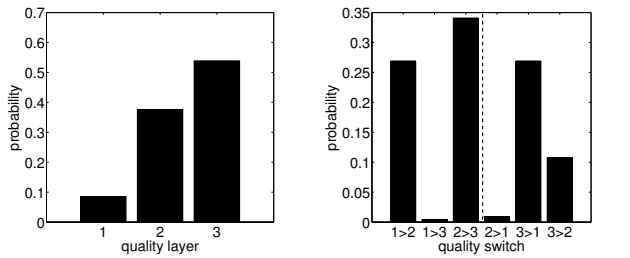
This paper presented a measurement tool for large scale measurements of QoE in MBB networks, which was implemented as Docker container in the MONROE testbed. On application layer, the playback events and the media properties of the HTML5 player, are monitored. This allows to quantify all relevant QoE factors of HTTP adaptive video streaming, i.e., stalling, initial delay, and adaptation. Moreover, the complete network traffic is captured including all HTTP request and response information. This monitored information



(a) Mean throughput. (b) Initial delay.

(c) Total stalling time. (d) Weighted time on quality layers.

Fig. 3: Throughput and QoE-related KPIs of the measurements of all baseline videos.



(a) Distribution of time on layer. (b) Distribution of quality switch amplitudes.

Fig. 4: Adaptation-related QoE influence factors.

can be enriched with metadata about the measurement node provided by the MONROE platform.

A first long term study was designed and conducted in the MONROE testbed. Over the course of three and a half weeks, 165 streaming sessions of various video contents could be measured. This provides valuable insights into the streaming behavior of mobile YouTube and the resulting QoE for end users in mobile networks. A thorough analysis was conducted for 88 measurements of a specific baseline video to exclude the impact of the video content. The results confirmed findings of [21] that mobile YouTube currently shows a conservative adaptation behavior. The resulting QoE for end users can still be poor, which showed that MBB networks still have to be improved, especially considering ever growing streaming popularity and quality demands.

As future work, the large-scale measurement campaigns in MONROE have to be continued to gather more results in both stationary and mobile scenarios. The analyses of such measurements are crucial to advance our understanding of the QoE of mobile video and the underlying application- and network-layer factors that impact it. Mobile operators should rely on such QoE benchmarks of their network and the resulting insights to further improve their networks and traffic management policies, e.g., in terms of QoE-aware traffic management.

ACKNOWLEDGMENT

This work was partly funded in the framework of the EU ICT projects INPUT (H2020-2014-ICT-644672) and MONROE (H2020-2014-ICT-644399, through open call project Mobi-QoE). The authors alone are responsible for the content.

REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016-2021 White Paper," 2017. [Online].
[2] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hobfeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, 2015.
[3] B. Staehle, M. Hirth, R. Pries, F. Wamser, D. Staehle, "YoMo: A YouTube Application Comfort Monitoring Tool," in QoEMCS, 2010.
[4] H. Nam, K. Kim, H. Schulzrinne, D. Calin, "YouSlow: A Performance Analysis Tool for Adaptive Bitrate Video Streaming," in SIGCOMM, 2014.
[5] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz, "Understanding YouTube QoE in Cellular Networks with YoMoApp – a QoE Monitoring Tool for YouTube Mobile," in ACM MOBICOM, 2015.
[6] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz, "YoMoApp: a Tool for Analyzing QoE of YouTube HTTP Adaptive Streaming in Mobile Networks," in EuCNC, 2015.
[7] P. Casas, R. Schatz, F. Wamser, M. Seufert, and R. Irmer, "Exploring QoE in Cellular Networks: How Much Bandwidth do you Need for Popular Smartphone Apps?" in ACM ATC, 2015.
[8] P. Casas, M. Seufert, F. Wamser, B. Gardlo, A. Sackl, and R. Schatz, "Next to You: Monitoring Quality of Experience in Cellular Networks From the End-Devices," *IEEE Transactions on Network and Service Management*, vol. 13, no. 2, 2016.
[9] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: an overlay Testbed for Broad-Coverage Services," in ACM SIGCOMM, 2003.
[10] D. Schwerdel, D. Günther, R. Henjes, B. Reuther, and P. Müller, "German-lab experimental facility," in FIS, 2010.
[11] M. Berman, J.S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "GENI: A Federated Testbed for Innovative Network Experiments," *Computer Networks*, vol. 61, 2014.
[12] P. Casas, M. Seufert, and R. Schatz, "YOUQMON: A System for On-line Monitoring of YouTube QoE in Operational 3G Networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, 2013.
[13] Q. A. Chen, H. Luo, S. Rosen, Z. M. Mao, K. Iyer, J. Hui, K. Sontineni, and K. Lau, "QoE Doctor: Diagnosing Mobile App QoE with Automated UI Control and Cross-layer Analysis," in ACM IMC, 2014.
[14] H. Nam, H. Schulzrinne, "YouSlow: What Influences User Abandonment Behavior for Internet Video?," Tech. report 2017. [Online].
[15] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang, "Understanding the Impact of Network Dynamics on Mobile Video User Engagement," in ACM SIGMETRICS, 2014.
[16] V. Aggarwal, E. Halepovic, J. Pang, S. Venkataraman, and H. Yan, "Prometheus: Toward Quality-of-Experience Estimation for Mobile Apps from Passive Network Measurements," in ACM HotMobile, 2014.
[17] P. Casas, A. D'Alconzo, F. Wamser, M. Seufert, B. Gardlo, A. Schwind, P. Tran-Gia, R. Schatz, "Predicting QoE in Cellular Networks using Machine Learning and in-Smartphone Measurements,", in QoMEX, 2017
[18] O. Alay, A. Lutu, R. Garcia, M. Peon-Quiros, V. Mancuso, T. Hirsch, T. Dely, J. Werme, K. Evensen, A. Hansen, S. Alfredsson, J. Karlsson, A. Brunstrom, A. S. Khatouni, M. Mellia, M. A. Marsan, R. Monno, and H. Lonsethagen, "Measuring and Assessing Mobile Broadband Networks with MONROE," in WoWMoM, 2016.
[19] T. Hoßfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial Delay vs. Interruptions: Between the Devil and the Deep Blue Sea," in QoMEX, 2012.
[20] T. Hoßfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz, "Quantification of YouTube QoE via Crowdsourcing," in ISM, 2011.
[21] M. Seufert, P. Casas,F. Wamser, N. Wehner, R. Schatz, and P. Tran-Gia, "Application-layer Monitoring of QoE Parameters for Mobile YouTube Video Streaming in the Field," in ICCE, 2016.