Quality of Experience of Web-based Adaptive HTTP Streaming Clients in Real-World Environments using Crowdsourcing

Benjamin Rainer
Institute of Information Technology
Alpen-Adria-Universität Klagenfurt
9020 Klagenfurt, Austria
benjamin.rainer@itec.aau.at

Christian Timmerer
Institute of Information Technology
Alpen-Adria-Universität Klagenfurt
9020 Klagenfurt, Austria
christian.timmerer@itec.aau.at

ABSTRACT

Multimedia streaming over HTTP has gained momentum with the approval of the MPEG-DASH standard and many research papers evaluated various aspects thereof but mainly within controlled environments. However, the actual behaviour of a DASH client within real-world environments has not yet been evaluated. The aim of this paper is to compare the QoE performance of existing DASH-based Web clients within real-world environments using crowdsourcing. Therefore, we select Google's YouTube player and two open source implementations of the MPEG-DASH standard, namely the *DASH-JS* from Alpen-Adria-Universitaet Klagenfurt and the *dash.js* which is the official reference client of the DASH Industry Forum. Based on a predefined content configuration, which is comparable among the clients, we run a crowdsourcing campaign to determine the QoE of each implementation in order to determine the current state-of-the-art for MPEG-DASH systems within real-world environments. The gathered data and its analysis will be presented in the paper. It provides insights with respect to the QoE performance of current Web-based adaptive HTTP streaming systems.

Categories and Subject Descriptors

C2.4 [**Computer-Communication Networks**]: Distributed Systems; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems

General Terms

Algorithms, Design, Measurement, Experimentation

Keywords

Dynamic Adaptive Streaming over HTTP; Crowdsourcing; Subjective Quality Assessment; Quality of Experience; QoE; DASH; MPEG

VideoNext'14, December 2, 2014, Sydney, Australia.

http://dx.doi.org/10.1145/2676652.2676656.

1. INTRODUCTION

In recent years MPEG's Dynamic Adaptive Streaming over HTTP (MPEG-DASH; or just DASH) has gained momentum both within industry and research [15]. Various implementations of the the standard have become available, partially due to the specification of the Media Source Extensions (MSE) [2] which are currently (or being soon) supported by major Web browsers (i.e., Internet Explorer, Mozilla Firefox, Google Chrome, Safari). The MSE provide an easy access to the multimedia decoding chain of the Web browser via the HTML5 video element and, thus, provides support for an Javascript-based implementation of current multimedia communication technologies. This encourages the implementation of adaptive HTTP streaming technologies like DASH because one can solely focus on implementing the mandatory components required by DASH thanks to the MSE. A typical implementation supporting DASH comprises the following components:

- **MPD parser**: receives and parses the XML-based media presentation description (MPD) from a source (e.g., HTTP server).
- **Segment handler**: requests the segments via a standard HTTP client according to the available representations defined within MPD and based on the information determined by the adaptation logic.
- **Adaptation logic**: this is the core of every DASH client and decides which media representation shall be used for a given content based on the client's context such as network parameters, terminal characteristics, and user preferences in order to maximize the Quality of Experience (QoE).

Figure 1 provides an overview of a typical DASH-based system architecture. The parts in red are specified within the DASH standard, i.e., the MPD and segment formats. The other parts are left open in order to encourage researchers and industry to come up with novel algorithms and implementations. All segments are identified using HTTP URLs described within the MPD. The MSE provide support for segment handling (including parsing, demuxing – if needed –, decoding, and rendering) and the Web browser environment provides the HTTP client. Thus, a Web-based implementation of DASH can focus on the MPD parsing and adaptation logic.

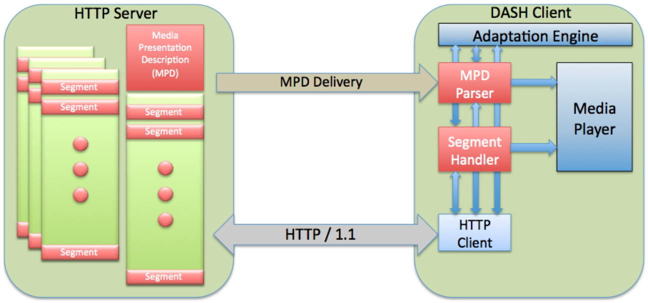

Figure 1: MPEG-DASH-based System Architecture.

Current research efforts in this domain focused on the adaptation logic to increase media throughput at the client in order to maximize the QoE under the given context conditions. The evaluation of the proposed adaptation algorithms is mainly done based on predefined context conditions (i.e., predefined bandwidth trajectories or captured network traces) within a laboratory environment following a given experimental setup. In this work, however, we investigate the QoE of Web-based adaptive streaming clients within real-world environments which helps to better understand the dynamics of real-world deployments. Therefore, we selected three DASH-enabled Web clients, namely: *DASH-JS*, *dash.js*, and *YouTube*. DASH-JS is one of the first open source implementations developed by the Alpen-Adria-Univerisität Klagenfurt and proposes a simple rate-based adaptation logic [13]. dash.js is the official reference implementation of the DASH Industry Forum (DASH-IF) and is also available as open source[1]. Finally, the official YouTube client adopted DASH playout recently (in HTML5 only, i.e., right mouse click over the video playback, "Stats for nerds", "DASH" flag set to "yes") [17].

In order to compare the QoE delivered by these three Web clients we conduct a subjective quality assessment using crowdsourcing. Therefore, we used a common content configuration (i.e., based on YouTube) and made the content available both on YouTube and our universities' server located in Klagenfurt, Austria. We employ the crowdsourcing platform Microworker where we hired so-called microworkers from countries within Europe (east/west), North America (USA, Canada), and India. The microworkers access the content over the open Internet using one of the supported Web browsers. The results provide insights which of the DASH-enabled Web clients provide the best QoE in a real-world environment. In addition to the explicit feedback from the microworkers, we measure other Quality of Service (QoS) parameters such as the number of stalls, startup time/delay, and the client's media throughput in terms of the provided representation bitrate.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of related work. In Section 3 we describe the methodology for conducting the subjective quality assessment using crowdsourcing. The obtained results of the statistical analysis are reported in Section 4 and discussed in Section 5. Finally, Section 6 concludes the paper and outlines future work.

[1]https://github.com/dash-industry-forum/dash.js, last access: August 2014.

2. RELATED WORK

A comparison of YouTube using progressive HTTP download and with DASH enabled is described in [9]. Authors found that approximately 43% of the videos are not watched in their entirety (according to a recorded trace). The bandwidth and cost savings correlate with the segment size of the DASH compliant stream. With longer segments (10s) the bandwidth and cost savings decrease compared to shorter segments (2s). The QoE impact of stalling caused by buffer outages for YouTube videos is investigated in [7]. The results and their analysis show that the QoE degrades exponentially with an increase in stalling events.

In [3] different rate-based adaptation strategies are objectively compared. The results state that the adaptation strategy that uses the average bitrates of the segments compared to the average bandwidth available is the most robust against buffer underflows. Furthermore, a quality based adaptation strategy is introduced which takes into account PSNR values of the segments. A similar evaluation of DASH-like system is done in [11] which uses bandwidth traces captured in wireless/3G networks and used within a laboratory setup to determine the media throughput (among others) of different formats and client implementations.

Using crowdsourcing for QoE evaluations is an active research topic for which best practices [6] and a trusted framework [18] are available. In this paper, we have adopted these principles for our needs as documented in the next section.

To the best of our knowledge no research paper provides a QoE evaluation using crowdsourcing of Web-based DASH clients in a real-world environment.

3. METHODOLOGY

In this section we discuss the participants, the selected stimulus, and the evaluation methodology for assessing the QoE of the selected DASH-enabled Web clients.

3.1 Platform & Participants, Stimulus, and Evaluation Method

For the subjective quality assessment using crowdsourcing we have selected the crowdsourcing platform Microworker [10]. This platform allows creating so-called campaigns which microworkers will carry out. For validating whether a microworker has successfully participated in the campaign, microworkers are asked to hand in a proof, i.e., a unique identification number which is provided at the end of the subjective quality assessment. We set a compensation of 0.2 to 0.33 US$ depending on the region for which the campaign is available [5]. We made our campaign available to east-/west Europe (west: Austria, France, Germany, Spain, Italy, Netherlands, Portugal, and Sweden; east: Bulgaria, Macedonia, Greece, Lithuania, Romania, Poland, and Turkey), North America (USA, Canada), and India.

The stimulus is an excerpt of the open source trailer Tears of Steal [16] comprising a video sequence without audio (duration: 160 seconds; resolution: 720p). As we do not evaluate encoding parameters or content-specific options but the streaming experience of different DASH-enabled Web clients, we believe one video sequence is sufficient. We compare the following DASH-enabled Web clients: DASH-JS, dash.js and YouTube. The DASH-enabled Web clients are used as they are provided (i.e., no modifications to the adaptation logic or any other component). For DASH-JS we used

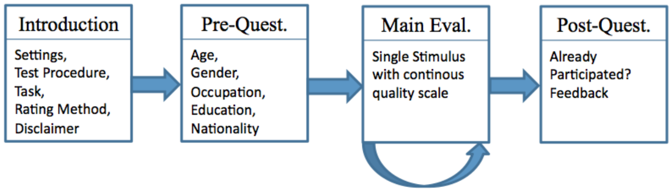

Figure 2: Overview of the Methodology.

the version provided[2] and for dash.js we used version 1.1.2[3]. In order to provide the same content configuration for all clients we adopted the configuration of YouTube with the resolutions and corresponding bitrates as follows: 144p (250 kbit/s), 240p (380 kbit/s), 360p (740 kbit/s), 480p (1308 kbit/s), and 720p (2300 kbit/s). The segment size was 2 seconds for all configurations. For DASH-JS and dash.js we hosted the content on our universities' server which has a symmetric bandwidth of approximately 10Gbit/s. For the YouTube player, the content was hosted by YouTube.

Figure 2 illustrates the methodology we have used for conducting the subjective quality assessment. First, we provide a detailed *introduction* explaining participants the structure of the actual task and how the QoE shall be assessed. Second, a *pre-questionnaire* is presented which asks for demographic data (e.g., age, gender, country of residence). Third, the *main evaluation* is presented. The stimulus is presented using a Web page with a gray background as recommended in [1]. The stimulus is only presented once to each participant and only one of the three Web clients is selected for the playback. The selection of the Web player is uniformly distributed ($p = \frac{1}{3}$) among the participants. The size of the Web client was fixed to a resolution of 1280×720. After the stimulus presentation the participants have the possibility to rate the QoE using a slider with a continuous scale from 0 to 100. The slider is initially set to 50 (middle position) and the time for rating the QoE is limited to eight seconds [1]. In the end a *post-questionnaire* is presented which asks the participant whether she/he has already participated in a similar study. Furthermore, the participants have the possibility to provide feedback using a free text (e.g., what they liked and what not).

For conducting subjective quality assessments using crowdsourcing we use an existing Web platform [14] which is extensible and measures further parameters such as the length of the stimulus presentations, the time used for rating a particular stimulus presentation, and fingerprints the participant's browser [12]. These mechanisms are used to identify participants that try to cheat, repeat the subjective quality assessment, or are not paying attention as further discussed in the following.

3.2 Screening Techniques

We investigate the implicit and explicit data obtained by the subjective quality assessment using crowdsourcing with respect to participants that try to cheat or do not pay attention to the task. Therefore, we screen participants as follows:

Browser Fingerprinting. We use browser fingerprinting in order to avoid that a single participant repeats the subjective quality assessment. If a fingerprint was recognized more than once by the subjective evaluation platform, the participant was excluded from the result set.

Stimulus Presentation Time. We screened those participants regarding the stimulus presentation time. A lower presentation time than the total duration of the presented video sequence indicates that the participants tried to shorten its presentation and, therefore, introduced a bias. Furthermore, if the presentation duration of the stimulus is greater than the time needed for downloading the segments plus the total duration, we also removed the participant from the result set as this indicates that participants paused the playback.

QoE Ratings and Pre-Questionnaire. For screening participants according to the QoE scores we employed the Median Absolute Deviation (MAD) estimator. Furthermore, we crosschecked the data gathered by the pre-questionnaire with respect to the country of residence.

4. RESULTS

In total 288 microworkers participated in the subjective quality assessment from which 33 participants were screened according to the mechanisms mentioned in the previous section (20 due to fingerprinting, six due to presentation time, and seven due to ratings and pre-questionnaire). From the remaining 255 participants were 175 male and 80 female, the majority (80%) is aged between 18 and 37. 17 participants stated that they have already participated in a similar subjective quality assessment.

Figure 3 shows an overview of the results along four dimensions: average *representation bitrate* (i.e., media throughput at the client), average *startup time* (or startup delay), average number of*stalls*, and the *QoE* in terms of Mean Opinion Score (MOS). DASH-JS maintains the lowest number of stalls (0.5 stalls on average) and the average representation bitrate is about 1,330 kbit/s. However, DASH-JS has the highest average startup time. The reason for this high startup time is that DASH-JS estimates the initial bandwidth when downloading the MPD and, thus, may select a higher bitrate in the beginning than the other clients. dash.js is outperformed by the other two DASH-enabled Web clients in three of the four dimensions. In particular, dash.js provides the lowest average representation bitrate, the highest number of stalls, and the lowest QoE. YouTube outperforms all other clients in three cases, specifically in the representation bitrate, startup time, and QoE. Furthermore, Figure 3 shows some tendencies for possible correlations. For instance, it indicates that there might be a correlation between the number of stalls and the QoE and that the representation bitrate impacts the QoE.

Figure 4 illustrates the QoE of the countries for which we conducted the subjective quality assessment clustered into "Europe East", "Europe West", "India", and "USA & Canada". We conducted for each pair of clients F-Tests in order to test whether the variances are equal. In the case that the F-Test rejected our hypotheses we used the student's t-Test proposed by Welch for unequal sample variances otherwise we used the student's t-Test for equal sample variances. A student's t-Test revealed a statistical signif-

[2]https://github.com/dazedsheep/dash-js/tree/master/dash-js, last access: August 2014.
[3]https://github.com/Dash-Industry-Forum/dash.js/releases/tag/v1.1.2, last access: August 2014.

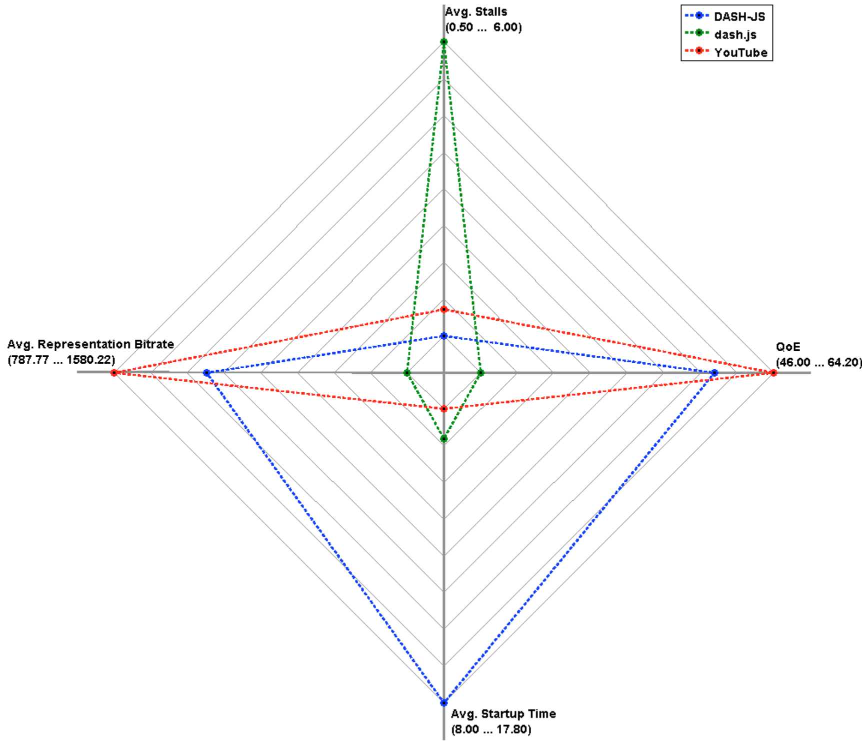

Figure 3: Overview of the results for the evaluated DASH-enabled Web clients.

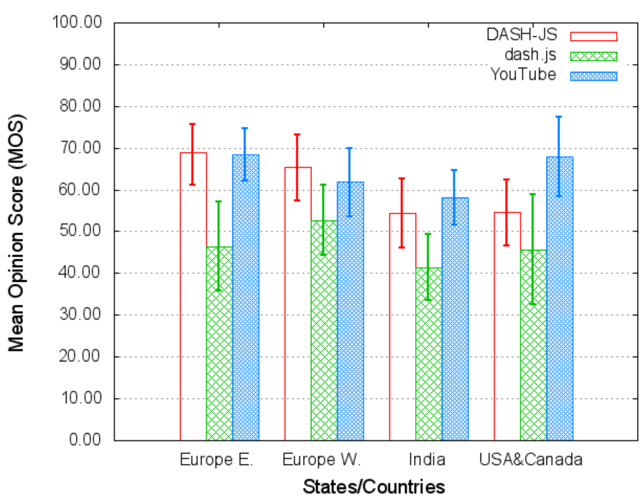

Figure 4: MOS grouped by the regions and presented for each Web client.

icant difference for the MOS between dash.js and DASH-JS ($p = 0.0021$, $t = 3.35$, $\alpha = 0.05$) and YouTube and dash.js ($p = 0.0005$, $t = 3.72$, $\alpha = 0.05$) for both "Europe West" and "Europe East". For "India", a student's t-Test showed a significant difference for the MOS between dash.js and DASH-JS ($p = 0.03$, $t = 2.24$, $\alpha = 0.05$). Furthermore, there exists a significant difference between YouTube and dash.js ($p = 0.002$, $t = 3.22$, $\alpha = 0.05$) but no significant difference between YouTube and DASH-JS. For the region "USA & Canada" we have identified a significant difference between YouTube and DASH-JS ($p = 0.042, t = 2.08$, $\alpha = 0.05$) and between YouTube and dash.js ($p = 0.063, t = 1.92$, $\alpha = 0.1$).

In our subjective assessment both influence factors stalling and different representation bitrates were present. Thus, we cannot distinguish which of the influence factors had the highest impact on the resulting QoE because if a user experiences stalls they may be accompanied by a lower representation bitrate (and vice versa). There is only a low linear correlation between the representation bitrate and QoE (i.e., $\rho = 0.38, p = 2.6 \cdot 10^{-11}, t = 6.95$). An interesting finding is that there is no linear correlation between the startup time and the QoE (e.g., $\rho = 0.18, p = 0.002, t = 3.07$) and, thus, the startup time has no significant impact on the QoE (as already indicated in Figure 3).

Figure 5 depicts the average representation bitrate selected by each client's adaptation logic grouped by region. Interestingly, the results of DASH-JS and YouTube within Europe are similar whereas YouTube outperforms the other clients in the other two regions, specifically within USA & Canada. One explanation of this behaviour is that the content for DASH-JS and dash.js is hosted solely at our universities' premises whereas for YouTube the content is hosted directly at YouTube and, thus, the existence of a content delivery network and additional server infrastructure may contribute to a better performance. However, it seems that India does not benefit from the YouTube infrastructure in the same way as USA&Canada. Finally, as seen in the results, dash.js adopts a rather conservative approach for all regions.

The results clearly state that the main influence factors for the QoE of DASH-enabled Web clients are the number of

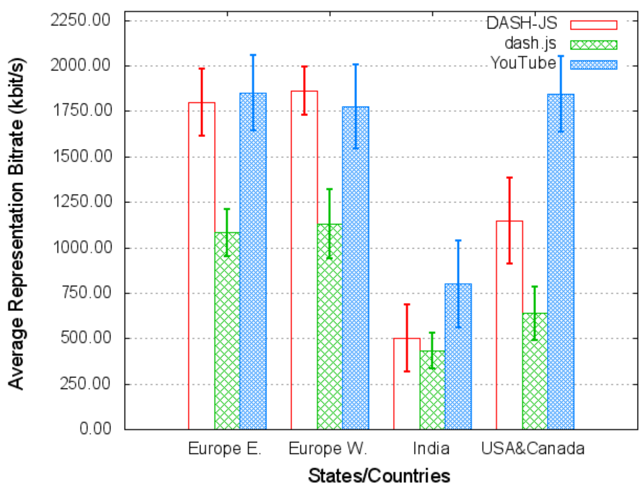

Figure 5: Average representation bitrate selected by each client grouped by region.

stalls and the selected/provided representation bitrate. As long as the startup time does not exceed a certain maximum it has no significant impact on the QoE (cf. Figure 3) [8]. A result that is worth investigating is the impact of representation switches on the QoE. Our study revealed that dash.js switches the representation 36 times on average. If we take into account the duration of the presented video sequence which was 160 seconds, this leads to an average representation switch every 4.4 seconds or nearly at every second segment boundary. DASH-JS has 10 representation switches on average and YouTube switches only once on average. However, there was only very little evidence that this high frequency of representation switches have a significant impact on the QoE.

5. DISCUSSION

The conducted subjective quality assessment using crowdsouring covers three DASH-enabled Web clients. We used them as they are provided without any modifications and optimizations with respect to the QoE. If we take possible extensions of the adaptation logic into account, dash.js provides the easiest way of changing the adaptation behavior. It provides the possibility of adding "rules" which define the behavior of the adaptation mechanism. DASH-JS demands rewriting the adaptation logic or an extension of the existing one. YouTube does not provide the possibility to allow any modifications to its adaptation logic.

Since we could not find a correlation between the startup time and the resulting QoE does not hold in general because the participants were hired through microworkers. Furthermore, participants that dropped out because of high startup times have not been considered by the web-based assessment platform.

The screening of the participants by using finger printing the browser has been investigated in [4]. It can reveal those participants that try to participate more than once in the study. Screening participants that had lower playback presentation times does filter those participants that have shortened the playback of the stimulus presentation and, therefore, it is feasible to exclude these participants.

DASH-JS provides a simple throughput-based adaptation logic, yet results are comparable with commercial deployments like YouTube. The YouTube client achieves a much higher average representation bitrate (throughput) in USA & Canada compared to other clients. However, the QoE difference between YouTube and others in that region is not as high as the average representation bitrate which indicates that also other factors impact the QoE. In general, the average representation bitrate in India is much lower than for other regions although the QoE is still in an acceptable range. This could be explained by regional aspects and context conditions, e.g., lower available bandwidth or that in some regions crowdsourcing is not done as honest as microworkers do not want to risk not receiving the compensation. The impact of a dedicated delivery infrastructure (i.e., CDN, servers, proxies, caches) becomes apparent for results in USA&Canada which provides the same average representation bitrate as in Europe and also yields similar QoE results. DASH-JS does not benefit from such an infrastructure and has a significant lower bitrate and QoE than YouTube.

Finally, dash.js is the most sophisticated DASH-enabled Web client among the three because it fully supports the DASH standard including the live profile, although DASH-JS has been recently extended with live support[4]. With respect to the effort of embedding the clients into a Web page all three can be embedded without any problems and with a minor expenditure of time.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have investigated the QoE of DASH-enabled Web clients within real-world environments using crowdsourcing. Therefore, we selected an already deployed client (YouTube) and research prototypes (DASH-JS and dash.js) using the same content configurations and conducted a crowdsourcing campaign to assess the QoE when consuming video content over the Internet. The results indicate that the delivered representation bitrate and the number of stalls are the main influence factors on the QoE. Interestingly, this confirms similar results achieved in previous evaluations but conducted within a controlled environment without real-world context conditions. Therefore, the findings documented in this paper provide evidence about QoE aspects of DASH-enabled Web clients within real-world environments. Additionally, it shows the feasibility of using crowdsourcing for subjective quality assessments within this application domain.

Future work comprises a more comprehensive evaluation of various adaptation logics (both objective and subjective) and the impact of dedicated delivery infrastructures aiming to improve DASH-based services.

7. ACKNOWLEDGEMENTS

This work was supported in part by the EC in the context of the SocialSensor (FP7-ICT-287975) and QUALINET (COST IC 1003) projects and partly performed in the Lakeside Labs research cluster at AAU.

8. REFERENCES

[1] Rec. ITU-R BT.500-11. Technical report.

[4] http://streaming.polito.it/2013/03/18/dash-js-with-live-streaming/, last access: August 2014

[2] A. Colwell, A. Bateman, and M. Watson. Media Source Extensions. W3C Candidate Recommendation, July 2014.

[3] F. De Simone and F. Dufaux. Comparison of DASH Adaptation Strategies based on Bitrate and Quality Signalling. In *Multimedia Signal Processing (MMSP), 2013 IEEE 15th International Workshop on*, pages 087–092, Sept 2013.

[4] P. Eckersley. How unique is your web browser? In M. Atallah and N. Hopper, editors, *Privacy Enhancing Technologies*, volume 6205 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin Heidelberg, 2010.

[5] M. Hirth, T. Hossfeld, and P. Tran-Gia. Anatomy of a Crowdsourcing Platform - Using the Example of Microworkers.com. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2011 Fifth International Conference on*, pages 322–329, June 2011.

[6] T. Hossfeld, C. Keimel, M. Hirth, B. Gardlo, J. Habigt, K. Diepold, and P. Tran-Gia. Best Practices for QoE Crowdtesting: QoE Assessment With Crowdsourcing. *Multimedia, IEEE Transactions on*, 16(2):541–558, Feb 2014.

[7] T. Hossfeld, M. Seufert, M. Hirth, T. Zinner, P. Tran-Gia, and R. Schatz. Quantification of YouTube QoE via Crowdsourcing. In *IEEE International Symposium on Multimedia (ISM) 2011*, pages 494–499, 2011.

[8] S. S. Krishnan and R. K. Sitaraman. Video Stream Quality Impacts Viewer Behavior: Inferring Causality Using Quasi-experimental Designs. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 211–224. ACM, 2012.

[9] D. Krishnappa, D. Bhat, and M. Zink. DASHing YouTube: An Analysis of using DASH in YouTube Video Service. In *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pages 407–415, Oct 2013.

[10] Microworkers. http://www.microworkers.com.

[11] C. Mueller, S. Lederer, and C. Timmerer. An Evaluation of Dynamic Adaptive Streaming over HTTP in Vehicular Environments. In *Proceedings of the Fourth Annual ACM SIGMM Workshop on Mobile Video (MoVid12)*, pages 37–42. ACM, feb 2012.

[12] N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna. Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 541–555, May 2013.

[13] B. Rainer, S. Lederer, C. Mueller, and C. Timmerer. A Seamless Web Integration of Adaptive HTTP streaming. In B. Pesquet-Popescu and C. Burileanu, editors, *Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 1519–1523. European Signal Processing (EURASIP) Society, aug 2012.

[14] B. Rainer, M. Waltl, and C. Timmerer. A Web based Subjective Evaluation Platform. In *Proceedings of the 5th International Workshop on Quality of Multimedia Experience (QoMEX'13)*, pages 24–25. IEEE, jul 2013.

[15] I. Sodagar. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE MultiMedia*, 18(4):62–67, 2011.

[16] Tears of Steel. http://tearsofsteel.org/.

[17] N. Weil. The State of MPEG-DASH Deployment. Streaming Media Europe, April 2014.

[18] C.-C. Wu, K.-T. Chen, Y.-C. Chang, and C.-L. Lei. Crowdsourcing Multimedia QoE Evaluation: A Trusted Framework. *IEEE Transactions on Multimedia*, 15(5):1121–1137, Aug 2013.