Design and Evaluation of a DASH-compliant Second Screen Video Player for Live Events in Mobile Scenarios

Stefano Petrangeli*, Niels Bouten*, Emanuel Dejonghe*, Jeroen Famaey†, Philip Leroux* and Filip De Turck*

*Department of Information Technology (INTEC), Ghent University- iMinds
Gaston Crommenlaan 8 (Bus 201), 9050 Ghent, Belgium, email: stefano.petrangeli@intec.ugent.be
†Department of Mathematics and Computer Science, University of Antwerp- iMinds
Middelheimlaan 1, 2020 Antwerp, Belgium

Abstract—The huge diffusion of mobile devices is rapidly changing the way multimedia content is consumed. Mobile devices are often used as a *second screen*, providing complementary information on the content shown on the *primary screen*, as different camera angles in case of a sport event. The introduction of multiple camera angles poses many challenges with respect to guaranteeing a high Quality of Experience to the end user, especially when the live aspect, different devices and highly variable network conditions typical of mobile environments come into play. Due to the ability of HTTP Adaptive Streaming (HAS) protocols to dynamically adapt to bandwidth fluctuations, they are especially suited for the delivery of multimedia content in mobile environments. In HAS, each video is temporally segmented and stored in different quality levels. Rate adaptation heuristics, deployed at the video player, allow the most appropriate quality level to be dynamically requested, based on the current network conditions. Recently, a standardized solution has been proposed by the MPEG consortium, called Dynamic Adaptive Streaming over HTTP (DASH). We present in this paper a DASH-compliant iOS video player designed to support research on rate adaptation heuristics for live second screen scenarios in mobile environments. The video player allows to monitor the battery consumption and CPU usage of the mobile device and to provide this information to the heuristic. Live and Video-on-Demand streaming scenarios and real-time multi-video switching are supported as well. Quantitative results based on real 3G traces are reported on how the developed prototype has been used to benchmark two existing heuristics and to analyse the main aspects affecting battery lifetime in mobile video streaming.

I. Introduction

Video streaming is responsible for more than a half of the total global bandwidth consumption on the Internet [1]. Each month over 6 billion hours of video are watched on YouTube, of which 40% is accessed through mobile devices [2]. In the near future, the mobile bandwidth consumption by video streaming is expected to further increase [3].

The diffusion of mobile devices raises new challenges in the field of Internet video streaming. First, mobile devices as tablets and smartphones are often used as *second screens* in conjunction with another device, as a television or a computer desktop, the *primary screen*. In the aforementioned scenario, a user watches an event on the primary screen, while obtaining additional information on his mobile device, as different camera angles. Second, the heterogeneity of devices in terms of processing capabilities, battery lifetime, screen dimensions and the highly varying network conditions typical of a mobile

environment make it more difficult to deliver an acceptable Quality of Experience (QoE) to the end users. This issue is aggravated in case of live events, where the client-side buffer has to be kept as small as possible in order to reduce the playout delay between the user and the live signal. Due to the ability of HTTP Adaptive Streaming (HAS) protocols to dynamically adapt to bandwidth fluctuations, they are especially suited for the delivery of multimedia content in mobile and wireless environments. Microsoft's Smooth Streaming, Apple's HTTP Live Streaming, Adobe's HTTP Dynamic Streaming are examples of commercially available HAS clients. Recently, a standardized solution has been proposed by MPEG, called Dynamic Adaptive Streaming over HTTP (DASH). In a HAS architecture, video content is stored on a server as segments at different quality levels. Each client can request the segment at the most appropriate quality level on the basis of the local perceived bandwidth and buffer status. In this way, video playback dynamically changes according to the available resources, resulting in a smooth video streaming experience. Even though second screen applications and HAS in mobile scenarios have separately received some attention, no research to date has tackled the combined problem yet. In order to overcome this issue, we present a DASH-compliant iOS video player that can support research on rate adaptation algorithms for live second screen scenarios in a mobile environment. The video player allows to monitor the battery consumption and CPU usage of the device and to provide this information to the adaptation heuristic. Live and Video-on-Demand scenarios and real-time multi-video switching are supported as well. The application provides also real-time information on the played video stream, as the quality bitrate, the buffer status and the number of freezes. Automatic testing of the implemented heuristics is also a feature supported by the presented prototype.

The remainder of this paper is structured as follows. Section II reports related work on HAS optimization in live and wireless scenarios. Section III details the design and implementation of the MPEG-DASH iOS video player. In Section IV we present the results of the experiments obtained using the aforementioned application. Section V presents the main conclusions.

II. Related Work

Akshabi et al. present an analysis of the performance and drawbacks of some commercially available HAS heuristics, such as Microsoft Smooth Streaming, Netflix and Adobe
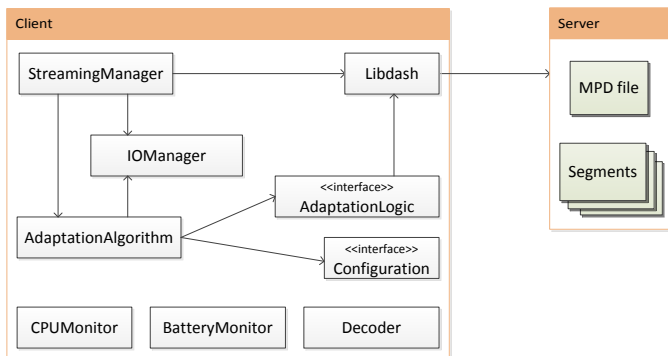
Fig. 1. High-level architecture of the iOS video player application.

Fig. 2. Screenshot of the iOS video player application.

players [4]. It is shown that current heuristics perform quality selection sub-optimally. Particularly, they fail to adapt to rapid bandwidth changes, leading to drops in the client play-out buffer and unnecessary quality switches. Müller et al. [5], as well as De Cicco et al. [6], found that the five most widely used state-of-the-art HAS heuristics perform poorly under wireless network conditions. Recent research has focused on improving this performance, for example through the use of GPS information [7]. Adzic et al. propose to add additional information into the video segments to enhance the quality decision algorithm [8]. The client can decide to switch up or down depending on the effect on bit rate and the intrinsic quality of the next segments to download. In previous research, we propose a self-learning client able to learn the best quality to request based on network conditions and buffer status [9]-[10]. Generally, a large client-side buffer of 30 seconds or more is used to handle unexpected bandwidth fluctuations in wireless environments. However, this is undesirable in live scenarios, where the delay of the user's signal with respect to the live moment, and thus the buffer size, should be minimized. Bouten et al. explored this topic, proposing a rate adaptation algorithm that exploits the advantages of Scalable Video Coding (SVC) to provide a low delay [11]. The use of SVC, in combination with parallel downloads and HTTP pipelining, allows them to reduce the client buffer size down to a few seconds, while guaranteeing satisfactory quality and continuous playback.

As shown above, HAS in mobile environments as well as live scenarios have separately received some attention in the scientific research community. However, no research to date has addressed the combined problem, which we believe is necessary, as the high bandwidth volatility in wireless networks further exacerbates the problems with live delay. Moreover, seamless and real-time multi-video switching has not been studied at all in the context of HAS. In this paper, we present a DASH-compliant video streaming application for iOS devices to support the development of novel HAS rate adaptation heuristics that can minimize live delay under highly dynamic bandwidth conditions, while taking into account mobile device constraints (e.g., screen capabilities and battery lifetime), and coping with multi-video switching.

III. MPEG-DASH Video Player for iOS

In this section we present the iOS video player designed to support research on rate adaptation algorithms in live second screen mobile scenarios. A high level description of the application's architecture can be found in Figure 1.
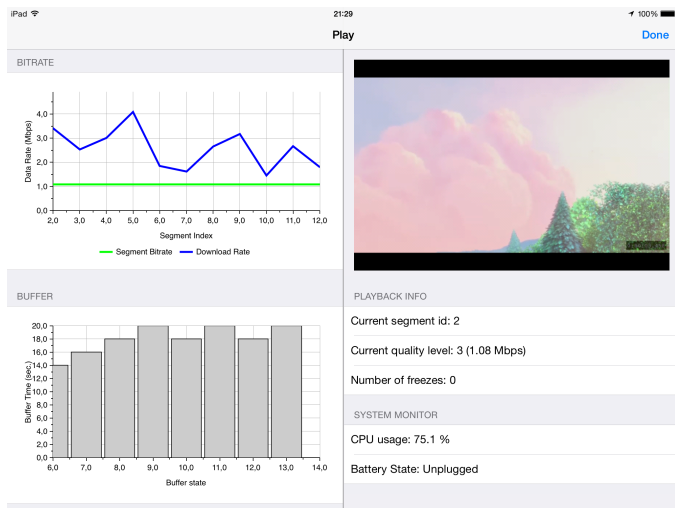
The MPEG-DASH video player is implemented on top of the *Libdash* library, the official ISO/IEC MPEG-DASH reference software for HAS. This library provides an interface to the DASH protocol and is responsible for handling the HTTP requests, parsing the manifest file (also called Media Presentation Description (MPD)) and parsing the segments. The *StreamingManager* module initializes the *AdaptationAlgorithm* and the *IOManager* modules and downloads the MPD of the video with the help of the *Libdash* library.

The *AdaptationAlgorithm* is in charge of deciding when a new segment should be downloaded and of the processing of the data from the segments. It also contains the rate adaption heuristic used to decide at which quality level the next segment has to be requested. This heuristic is implemented via the *AdaptationLogic* interface, which is used to separate the adaptation logic from the rest of the video player application. This way, different rate adaptation heuristics can be easily implemented and tested. The methods specified by the *AdaptationLogic* interface that have to be implemented by the adaptation heuristic are: *setSettings*, to configure the heuristic, *getSegment*, to compute the quality level of the next segment to request and *isBufferReady*, to notify when the buffer is full enough and it is possible to start the play-out of the video. Each *AdaptationLogic* is associated with an implementation of the *Configuration* interface, which contains the parameters to configure the selected heuristic.

The *IOManager* is used to write and read data from the device and is responsible for the buffering of the segments. Due to the closed environment and limitations of iOS SDK, decoding segments directly from memory is not possible. For this reason, the *IOManager* appends the downloaded segments to a movie file stored in the device. The segments are then extracted from this file and decoded by the *Decoder* using the FFMpeg library. The decoded frames are rendered on the screen using OpenGL ES. The *BatteryMonitor* and *CPUMonitor* modules can be used to monitor the battery status and the CPU of the device, respectively.

The main functionalities offered by the designed prototype are:

- Playback of MPEG-DASH video streams;

	Avg MOS score	Avg nr. of freezes	Avg freeze time (sec)	Avg quality	Nr. of switches
MSS	3.0 ± 0.4	0.3 ± 0.6	0.2 ± 0.4	3.2 ± 0.2	61.0 ± 6.1
QoE-RAHAS	2.7 ± 0.4	1.0 ± 1.0	0.6 ± 0.7	3.3 ± 0.2	90.3 ± 22.5

TABLE II.   PERFORMANCE SUMMARY OF QoE-RAHAS AND MSS
WITH SEGMENT DURATION OF 2 SECONDS

	Avg MOS score	Avg nr. of freezes	Avg freeze time (sec)	Avg quality	Nr. of switches
MSS	2.9 ± 0.6	1.0 ± 1.7	0.4 ± 0.6	3.2 ± 0.2	42.0 ± 5.3
QoE-RAHAS	2.8 ± 0.6	0.7 ± 1.1	0.2 ± 0.4	3.3 ± 0.2	64.7 ± 13.5

- Multi-video switching at run time;
- Support for live and Video-On-Demand scenarios;
- Selection and configuration of the desired rate adaptation heuristic at run time;
- Display of relevant statistics in real time (e.g., buffer filling level, available bandwidth and video bitrate);
- Obtain measurements on the battery consumption and CPU status;
- Perform automatic tests to benchmark and optimize the implemented rate adaptation heuristics.

As far as multi-video switching is concerned, in a DASH-complaint MPD it is possible to specify multiple video streams available for the client, the so-called *adaptation sets* (e.g., different camera angles of the same event, different events occurring at the same time etc.). Multi-video switching has been implemented by allowing the user to select at run time the different adaptation sets to play. The play-out video switching time can also be configured by specifying the number of segments from the previous adaptation set to be kept in the buffer, before the actual switch occurs.

A screenshot of the video player is reported in Figure 2. A window on the right top-corner of the screen allows to follow the played video stream. It is also possible to track the number of freezes, the CPU usage and the battery status. Two windows on the left of the screen report real-time statistics on the buffer status, the played quality and the download rate of the segments.

IV.   RESULTS

The prototype presented is Section III has been used to (i) optimize the behavior and benchmark the performance of two rate adaptation heuristics and (ii) analyze the main factors in HAS impacting the battery life duration of mobile devices. In all the experiments, the HAS Server is implemented through a MAMP 2.1.3 server installed on a MacBook Pro equipped with OS X Mavericks 10.9.2. As far as the video player application is concerned, an iPad 3 Wi-Fi has been used. In order to simulate a realistic mobile network, the bandwidth of the WiFi link connecting the iPad to the HAS Server is throttled. The bandwidth profiles are obtained by measurements performed on a real 3G/HSDPA network in Norway [12].

A.   Rate Adaptation Heuristics Comparison

The DASH video stream is a 10 minutes version of the *Big Buck Bunny* movie, which is offered at four quality levels with bitrates 333, 468, 750 and 1083 Kbps, respectively. As far as the segment duration is concerned, two different versions of the video stream are created, with chunks of 1 and 2
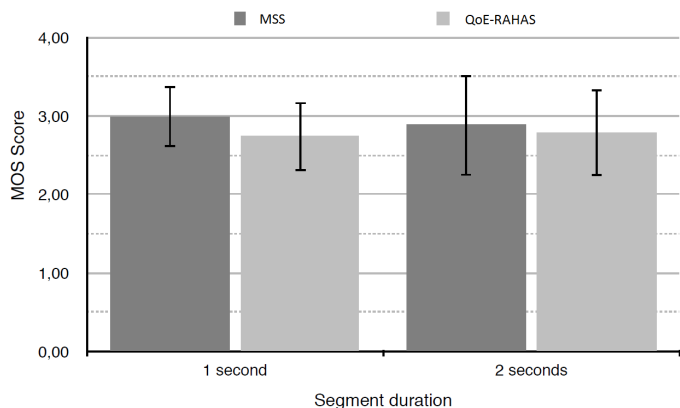

Fig. 3. Comparison between MSS and QoE-RAHAS, from a QoE perspective, for different segment durations.

TABLE III.   IMPACT OF SEGMENT DURATION ON BATTERY LIFETIME

Device	Segment duration 10 sec.		Segment duration 2 sec.	
	Batterylevel (%)	Time (sec)	Batterylevel (%)	Time (sec)
iPad 3	-15	3357	-15	3324
iPad 4	-15	3515	-15	3462

seconds. Two different rate adaptation heuristics have been evaluated: a proprietary HAS client, the Microsoft ISS Smooth Streaming[1] (MSS), and the QoE-RAHAS client [13]. In order to compare the performance of the two clients, a metric in the same range of the Mean Opinion Score (MOS) has been used [9]-[14]. This metric assigns a score between 1 and 5 to a played video stream. A MOS score of 1 indicates a bad QoE, whereas a MOS score of 5 indicates an excellent QoE. A MOS score of 3 is considered a fair QoE. All experiments are conducted with a buffer size of 10 seconds. Since the performance of a rate adaptation heuristic strongly depends on the available bandwidth, three different bandwidth traces are tested. First of all, optimal values for the parameters of the MSS and QoE-RAHAS algorithms are determined. For the QoE-RAHAS heuristic, these experiments have shown that for a stream with 1 second segments, a configuration with *buffer minimum* of 4 seconds, *quality window* of 40 seconds and *buffer percentage* of 100% results in the highest MOS score. For a stream with 2 seconds segments, a *buffer percentage* of 80% should be used. As far as the MSS is concerned, the best configuration is obtained with a *panic threshold* of 25%, a *lower threshold* of 40% and an *upper threshold* of 80%. A comparison of the two heuristics is presented in Figure 3 and Tables I and II. MSS slightly outperforms QoE-RAHAS in terms of MOS score. However QoE-RAHAS presents a higher average quality level during the video stream. The lower MOS score for the QoE-RAHAS algorithm is mainly due to the increased number of quality switches. This is due to the more aggressive behavior of the QoE-RAHAS heuristic, whereas MSS has a more conservative behavior. This results in a higher QoE for MSS.

TABLE IV.   IMPACT OF QUALITY LEVEL ON BATTERY LIFETIME

Device	Bitrate 800 Kbps		Bitrate 3100 Kbps		Bitrate 5800 Kbps	
	Batterylevel (%)	Time (sec)	Batterylevel (%)	Time (sec)	Batterylevel (%)	Time (sec)
iPad 3	-15	3357	-15	3221	-15	3000
iPad 4	-15	3515	-15	3422	-15	3100

B. Battery Consumption Analysis

In a second set of experiments, the impact of the segment duration and video characteristics on the battery consumption of an iPad was investigated. This parameter is especially important in a mobile scenario, where the battery life of the devices is limited. For these experiments, three different DASH streams with a bitrate of 800, 3100 and 5800 Kbps are played for about one hour (i.e., no quality switching occurs). The quality levels at 800 and 3100 Kbps are obtained by encoding the *Big Buck Bunny* video with a resolution of 864x480 with a Quantization Parameter (QP) of 35 and 25, respectively. The quality level at 5800 Kbps is obtained by encoding the same video with a resolution of 1296x720 and a QP of 25. All streams are segmented in chunks of 2 and 10 seconds. For all the experiments, the video client is directly connected to the HAS Server through a WiFi link at 6 Mbps. Since battery draining can differ from device to device, these experiments are conducted on an iPad 3 WiFi and an iPad 4 WiFi. We also ensured that the screen brightness was set to 100% for all the experiments, all the video statistics were disabled and the battery was fully charged at the beginning of each experiment. We measure the amount of time needed to deplete the battery level of 15%. Table III shows the impact of the segment duration on the battery consumption, for a video at 800 Kbps. The experiment shows that the impact of segment duration on the battery consumption is very limited. On the other hand, the impact of the video quality is significant, as shown in Table IV. In this case, three streams at 800 Kbps, 3100 Kbps and 5800 Kbps with segment duration of 10 seconds are played for one hour. The battery consumption for the two streams at 800 and 3100 Kbps is almost the same, while it increases for a stream at 5800 Kbps. Particularly, battery lifetime is reduced between 7% and 11% when streaming the highest quality. This means that streaming a video at a high spatial resolution increases the energy consumption, since the decoding and rendering of the video is computationally more intensive. By keeping into account this information, it is possible to develop a new adaptation heuristic that tries not only to maximize users' QoE but also battery duration.

V. CONCLUSIONS

In this paper, we presented the design and evaluation of a DASH-compliant video player for iOS devices, designed to support research on rate adaptation algorithms in second screen live scenarios in a mobile environment. Quantitative results were shown on how the video player can be used to benchmark two rate adaptation heuristics, the MSS and the QoE-RAHAS [13], and to understand which are the main factors impacting battery duration in HAS. We reported that the spatial resolution of the video content is the main factor impacting battery lifetime. Particularly, in the evaluated scenario, playing the video at the highest resolution decreases battery lifetime between 7% and 11% compared to the lowest resolution. The outcome of the presented analysis will serve as basis for the development of new rate adaptation heuristics, which try not only to maximize users' QoE but also battery lifetime.

REFERENCES

[1] Cisco Systems, *Cisco Visual Networking Index: Forecast and Methodology, 2012–2017*. 2013. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.pdf.

[2] Youtube, *Youtube statistics*. 2014. [Online]. Available: http://www.youtube.com/yt/press/statistics.html.

[3] Cisco Systems, *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013–2018*. 2014. [Online]. Availabel: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.pdf.

[4] S. Akhshabi, S. Narayanaswamy, A. C. Begen and C. Dovrolis, *An experimental evaluation of rate-adaptive video players over HTTP*. Signal Processing: Image Communication, Volume 27, Issue 4, pp. 271-287, April 2012.

[5] C. Müller, S. Lederer and C. Timmerer, *An evaluation of dynamic adaptive streaming over HTTP in vehicular environments*. In Proceedings of the 4th Workshop on Mobile Video (MoVid '12). ACM, New York, NY, USA, pp. 37-42.

[6] L. De Cicco, S. Mascolo and C.T. Abdallah, *An experimental evaluation of Akamai adaptive video streaming over HSDPA networks*. Computer-Aided Control System Design (CACSD), 2011 IEEE International Symposium on, pp.13-18, 28-30 Sept. 2011.

[7] H. Riiser, P. Vigmostad, C. Griwodz and P. Halvorsen, *Bitrate and video quality planning for mobile streaming scenarios using a GPS-based bandwidth lookup service*. In 2011 IEEE International Conference on Multimedia and Expo (ICME), pp 1-6.

[8] V. Adzic, H. Kalva and B. Furht, *Optimized adaptive HTTP streaming for mobile devices*. Proc. SPIE 8135, 2011.

[9] M. Claeys, S. Latré, J. Famaey, T. Wu, W. Van Leekwijck and F. De Turck, *Design and Optimization of a (FA)Q-Learning-based HTTP Adaptive Streaming Client*. Connection Science 26, 01 (2014), pp. 27-45. 2014.

[10] S. Petrangeli, M. Claeys, S. Latré, J. Famaey and F. De Turck, *A multi-agent Q-Learning-based framework for achieving fairness in HTTP Adaptive Streaming*. In Network Operations and Management Symposium (NOMS), pp. 1-9. May 2014.

[11] N. Bouten, S. Latré, J. Famaey, F. De Turck and W. Van Leekwijck, *Minimizing the impact of delay on live SVC-based HTTP adaptive streaming services*. In proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM), 2013.

[12] H. Riiser, T. Endestad, P. Vigmostad, C. Griwodz and P. Halvorsen, *Video Streaming Using a Location-based Bandwidth-lookup Service for Bitrate Planning*. ACM Trans. Multimedia Comput. Commun. Appl. 8, 3, Article 24 (Aug. 2012), 19 pages.

[13] S. Petrangeli, J. Famaey, M. Claeys and F. De Turck, *A QoE-Driven Rate Adaptation Heuristic for Enhanced Adaptive Video Streaming*. Technical Report, 2014. [Online]. Available: http://users.ugent.be/~spetrang/QoE-RAHAS.pdf.

[14] J. De Vriendt, D. De Vleeschauwer and D. Robinson, *Model for estimating QoE of video delivered using HTTP adaptive streaming*. In Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on. pp. 1288-1293.

---

[1] https://slextensions.svn.codeplex.com/svn/trunk/SLExtensions/AdaptiveStreaming