Towards a Framework for Comparing Application-Network Interaction Mechanisms

Susanna Schwarzmann*, Thomas Zinner* and Ognjen Dobrijevic†
*Institute of Computer Science, University of Würzburg, Am Hubland, 97074 Würzburg, Germany
Email: {susanna.schwarzmann,zinner}@informatik.uni-wuerzburg.de
†University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, 10000 Zagreb, Croatia
Email: ognjen.dobrijevic@fer.hr

***Abstract*—Multimedia services, such as HTTP adaptive streaming (HAS) and audio conferencing, hold a dominant share in Internet traffic and are critical to end-user quality of experience (QoE). Since a distributed, application-level adaptation to the available network resources may not result in an optimal and fair QoE among heterogeneous clients, there is the need for a coordinated application-network interaction (App-Net), which includes exchange of monitoring and control information between the involved entities. Numerous App-Net approaches have been proposed, but typically for very specific use cases. A systematic comparison of such solutions, which would identify their pros and cons, as well as potential application scenarios, is missing. Our paper addresses this gap by providing an overview of QoE-centric App-Net mechanisms and deducing an abstract interaction model. In order to evaluate a trade-off between QoE gains and the messaging overhead, we use the latter model to compare two App-Net mechanisms for HAS against a baseline HAS implementation.**

***Keywords*-QoE-Centric Management, Application-Network Interaction, Application Control, Application Monitoring, Network Control, Network Monitoring**

I. Introduction

Multimedia content remains dominantly consumed over the Internet, with, for example, the traffic share of video services expected to rise from 64% in 2014 up to 80% by 2019 [1]. However, the Internet's architecture has not been designed for stringent network resource demands imposed by multimedia services and for differentiation among them. Since over-provisioning network resources is often economically unfavorable, other proposals to overcome these issues comprise techniques to adapt configurations of either multimedia applications, the underlying network, or both. In this context, network-level mechanisms include bandwidth reservation (e.g. [2], [3]), traffic prioritization (e.g. [4], [5]), and flow re-routing (e.g. [6], [7]).

In general, applications are capable of dynamically adapting their behavior, with solutions ranging from conventional TCP-based and UDP-based implementations to technologies such as Dynamic Adaptive Streaming over HTTP (DASH) [8]. But, as most application-level approaches perform local client or server optimizations, this may not lead to a uniform assignment of the available network resources among competing traffic flows. Even if such a resource share is reached, an unfair end-user quality distribution among heterogeneous clients may still occur [2]. This reflects well in an example of a smartphone and a desktop computer being allocated a similar network bandwidth for video streaming, despite the former requiring less bandwidth for smooth video playout.

In order to provide a fair distribution of user-perceived quality, or quality of experience (QoE) [9], an explicit exchange of monitoring and control information between entities representing applications and the network is called for, thus forming *application-network interaction* (App-Net). Several solutions discuss a coordinated approach to controlling applications and network (e.g. [3], [4]), where control actions are orchestrated by a logically centralized entity that is receiving the monitoring information. In this paper, we will refer to such an entity as *policy manager* (PM). With numerous App-Net approaches and use cases, the contribution of this paper is two-fold:

(1) an *abstract QoE-centric interaction model*, which is deduced from our overview of App-Net mechanisms and would later on enable us to produce a systematic comparison of different solutions; and
(2) an *evaluation comparison* of two App-Net mechanisms for improving QoE in a DASH-based video service against a baseline DASH implementation, which considers achieved video quality, duration of video freezes, and the messaging overhead.

The rest of the paper is organized as follows. Section 2 gives an overview of QoE-centric App-Net mechanisms and derives a generic interaction model, while Section 3 presents the App-Net evaluation methodology, as well as our testbed for conducting experimental measurements. Evaluation results are described and analyzed in Section 4, followed by the conclusion and a future work outlook.

II. Background and Related Work

In recent years a lot of research effort has been devoted to maximizing QoE for multimedia services, such as video streaming and audio conferencing. Given video's prevailing Internet traffic share, most researchers focus on its delivery via HTTP adaptive streaming (HAS) and one of HAS dominant standards, DASH [8]. DASH splits up media content into short segments, each in different representations with respect to, e.g., encoding bitrate and resolution. All necessary meta-information on media is stored in a media presentation description (MPD) file. In a typical scenario, a DASH client first retrieves the MPD file, and then decides which media segments to download and play out, by considering their representations, state of the client buffer and network throughput.

978-0-9883045-1-2/16 $31.00

As one of the key prerequisites for enhancing QoE is the supported optimization of application/network control, our overview uses this classification of App-Net solutions:

- *Application-level optimization* (ALO) – an application entity is informed on application and/or network performance, so as to decide to adjust application parameters or to invoke network-level mechanisms;
- *Network-level optimization* (NLO) – a network entity learns of application and/or network performance, in order to execute network-level mechanisms or to instruct applications on how to adapt;
- *PM optimization* (PMO) – applications and network performance indicators are delivered to a PM, which orchestrates control actions for applications and the network based on its "global" system view.

A. App-Net Approaches for Improving QoE

An ALO-based framework for over-the-top (OTT) services, such as Skype conferencing, is proposed in [10]. Congestion indication and other mobile network information is conveyed to Skype application, which utilizes it to adjust media transmission parameters (such as intra-flow packet prioritization). QoE-aware DASH [11] introduces a proxy-assisted measurement of network throughput so as to provide more precise values and ALO instructions to DASH clients when selecting media representations.

Several NLO-centric mechanisms that target YouTube are discussed in [5], [6], [12]. All of them employ an estimator of the client buffer state, which is based on deep packet inspection in the network. A software-defined networking (SDN) approach that dynamically re-routes traffic based on buffer state is proposed in [6], while [12] considers a flexible aggregation of resources from one or more access networks if a mobile client buffer is drained out (so as to ensure a smooth media reproduction). A dynamic prioritization of YouTube flows in home network gateways over other traffic is presented in [5].

Nam *et al.* adopt the SDN concept to increase QoE for HAS services [7]. The central part of their PMO-based solution is an SDN application, which calculates a new route when network congestion takes place. Monitoring is performed periodically by clients, which report application-level metrics such as buffer status, and an SDN controller, which reports on network performance (e.g. jitter). A similar SDN-centric proposal that includes NLO of QoE-aware network paths is presented in [13], with the latter design addressing different service categories (e.g., audio conversation and video streaming).

The QoE Fairness Framework [2] is a PMO approach for a fair QoE distribution among heterogeneous DASH clients. It realizes App-Net by monitoring network bandwidth, collecting user device features and MPDs, and reporting them to a PM. The PM then calculates appropriate media bitrate for each client and directs the clients which media representation to choose. Another PMO proposal for DASH services, which reduces video freezes (or stallings), is presented in [4]. There, network devices are regularly polled for traffic statistics, while clients report on application metrics such as buffer state. If a client buffer runs empty, a PM prioritizes specific media segments in the network and instructs the associated client to request the matching media quality.

Cofano *et al.* [3] describe several NLO strategies that target QoE fairness among different clients. The bandwidth reservation strategy uses application-level information so as to assign media flows to dedicated queues on network interfaces, while a hybrid strategy combines bandwidth reservation with media bitrate calculation (similarly to [2]). Server and network assisted DASH (SAND) [14] is an ongoing standardization to specify a DASH messaging framework, with the goal to enhance multi-client QoE. SAND seeks to exploit monitoring information provided by media content servers and network devices, which encompasses MPDs and measured network throughput. The key messaging interface involves interaction between DASH clients/content servers and network devices.

More generic App-Net approaches are investigated in [15]–[17]. Ferguson *et al.* propose the participatory networking paradigm [15], which enables end-users, user devices and applications to interact with the network. For example, end-users can invoke application re-configuration based on the network state or delay execution of, e.g., a conference call until sufficient network resources are available. OpenADN, or Open Application Delivery Networking [16], [17], introduces an abstraction layer between applications and the network, in order to facilitate the mapping of policies across various multimedia services to network behavior. OpenADN utilizes the SDN principles, allowing, e.g., a network provider to customize message routing by including different middleboxes in the path.

B. Cartography of App-Net Approaches

The approaches presented previously employ several distinct functions to realize App-Net, but all exhibit a common control loop. First, relevant monitoring information on applications, network, or both is collected. Then, control actions are computed and enforced, at network-level, application-level, or both. Disregarding implementation-specific details of these App-Net functions, we group them into the following categories: application or network monitoring, and application or network control. The latter categories are used to identify fundamental building blocks and derive an abstract App-Net model (Figure 1).

The model depicts four functional blocks, namely *Application Monitoring* (AM), *Application Control* (AC), *Network Monitoring* (NM), and *Network Control* (NC). AM supervises the state of running applications, while NM captures performance parameters that indicate the network state. Control functions of the model are shared between AC and NC – the former features adaptation actions implemented in applications, while the latter does the same for the network level. In order to jointly optimize behavior of applications and network, and achieve a fair QoE distribution, monitoring information about applications and network may be provided to a logically centralized

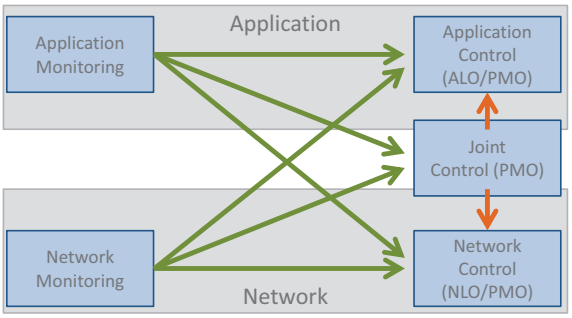

Figure 1. Key building blocks for application-network interplay

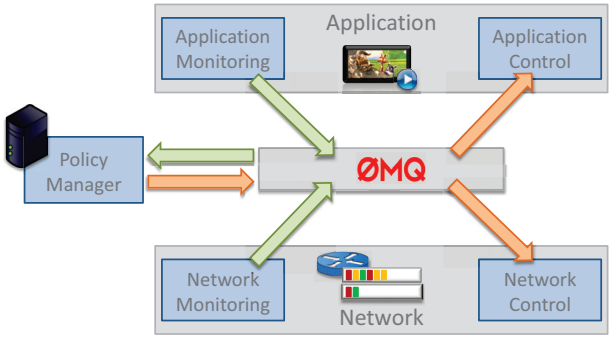

Figure 2. ØMQ-based messaging service for App-Net

PM. Utilizing this information, the PM can decide on adaptation actions and instruct AC/NC to enforce them, thus realizing a PMO (joint control) approach.

The actual realization of the control loop varies between the App-Net approaches, which differ in monitoring accuracy, the degree of control support, and the implementation domain of the specific monitoring/control function. Additionally, the App-Net mechanisms manifest their dissimilarity in the frequency of control actions and technical implementation specifics. Thus, we classify the presented mechanisms with respect to the identified App-Net functional blocks (Table I), also comparing monitoring accuracy and the frequency of control actions. *Initial* refers to monitoring/control being performed only during service establishment, *triggered* assumes that monitoring or control is initiated by an irregular event, while *periodic* regards the actions in regular time intervals.

Table I
CLASSIFICATION OF THE PRESENTED APP-NET MECHANISMS

Source	Type	NM	NC	AM	AC
[2]	PMO	Periodic	None	Initial	Periodic
[3]	NLO	Initial	Periodic	Initial	Periodic
[4]	PMO	Periodic	Triggered	Periodic	Triggered
[5]	NLO	Periodic	Triggered	Periodic	None
[6]	NLO	Periodic	Periodic	Periodic	None
[7]	PMO	Periodic	Triggered	Periodic	None
[10]	ALO	Periodic	Triggered	Periodic	Periodic
[11]	ALO	Periodic	Periodic	Periodic	Periodic
[12]	NLO	Periodic	Periodic	Periodic	None
[13]	NLO	Periodic	Triggered	Initial	None
[15]	ALO	Triggered	Triggered	Initial	Periodic
[16]	NLO	Periodic	Periodic	Initial	None
[17]	NLO	Periodic	Periodic	Initial	None

III. EVALUATION METHODOLOGY

In this section, we first present a messaging service that is used to realize the information exchange as underlined in the abstract App-Net model. Afterwards, we highlight two PMO-based App-Net mechanisms, which are evaluated in Section IV. Finally, the evaluation testbed and the chosen performance metrics are described.

A. Messaging Service for the App-Net Realization

The interaction between application and network entities calls for the exchange of monitoring and control information (Figure 2). In order to enable this exchange, we develop a light-weight publish-subscribe messaging system that is based on ØMQ[1]. The central point of this system is a messaging broker (ØMQ-Broker), which receives all messages and forwards them to the targeted entities. Figure 2 illustrates the associated information flow. Application and network monitoring data is first delivered to the PM, which then forwards its control decisions to the corresponding application and network control entities via the ØMQ-Broker.

B. Investigated App-Net Mechanisms

For our evaluation we choose two PMO-based App-Net solutions that enhance QoE for DASH-based video streaming. The first approach is QoE Fairness Framework (QoE-FF) [2], which employs a centralized controller (PM) to govern the video quality selection among heterogeneous DASH clients and to produce a fair QoE distribution. When starting a new video stream, a DASH client communicates its screen resolution, the requested video bitrate and the associated bandwidth requirements to the PM. While monitoring network load and active DASH clients, the PM evaluates QoE for each client, computes the fair-share video quality levels, and enforces them at each client.

The second approach is based on the work by Petrangeli *et al.* [4]. Their App-Net mechanism avoids buffer under-runs in DASH clients by prioritizing download of video segments over other traffic. We will refer to the latter approach as *Stalling Prevention Mechanism (SPM)*. Network monitoring is realized by having a centralized controller (PM) regularly poll throughput of pre-configured queues in an SDN switch. To enable application monitoring, DASH clients are customized to add the video buffer status to HTTP requests, which are forwarded by the switch to the PM. The PM is then able to extract the application-level parameters and calculate whether a video segment will arrive before the corresponding client buffer runs out or not. In the case of a late arrival, the given segment is prioritized, while the associated DASH client is forced to request the lowest video quality so as to allow a fast refill of the buffer.

[1] http://zeromq.org

In order to assess different data-plane effects on video quality, we implement two versions of the SPM approach. In the first implementation, which is referred to as SPM1, video packets are forwarded in the data plane without any intervention. This may result in video packets being delivered to a DASH client before control actions are decided by the PM. The second implementation, denoted as SPM2, delays video packets in the data plane until possible control actions are enforced.

C. Evaluation Metrics

QoE influence factors for DASH-based video streaming are initial buffering delay, video stalling, video playback quality, and the number of quality level switches [8]. Stallings during video playback have a high impact on user's QoE, while initial buffering delay has a minor impact [18]. In the absence of stallings, video reproduction quality on the highest level dominates the user experience, as presented in [19]. In that case, the number of quality level switches can be neglected. Accordingly, we rely on the overall stalling duration and the average video quality to compare the gains between the implemented App-Net mechanisms. To quantify the video quality, we use the structural similarity (SSIM) metric [20].

The investigated App-Net mechanisms differ in their complexity, which encompasses the number of involved functional entities, extent of the software implementation, and the number of exchanged messages. As a first step, we evaluate this complexity by considering the number of exchanged messages sent via the brokering service.

D. Implemented Testbed and Evaluation Scenario

Our testbed consists of four common personal computers. First one acts as a DASH server, second one as a network emulator, third one is running a PM and the ØMQ-Broker, while fourth one hosts six instances of the TAPAS DASH client [21]. TAPAS clients use DASH heuristic *Conventional* [22] (baseline implementation). The DASH server stores the Big Buck Bunny[2] clip in three different resolutions (1080p, 720p, 360p), assuming different screen resolutions (i.e. client characteristics), and encodes them in several bitrates. We divide video clips into segments of four seconds long and create the matching MPD file in the *m3u8* format by using the following *ffmpeg* command:

```
ffmpeg -i bbb1080.yuv -codec:v libx264 -codec:a libfaac
    -map 0 -segment_time 4 -segment_list $newdir/out.
    m3u8 -force_key_frames "expr:gte(t,n_forced*4)" -f
    stream_segment -b:v "$(($br * 1000))" $newdir/
    output_%03d.ts
```

Obtained video bitrates per resolution are shown in Figure 3, as well as the corresponding SSIM values.

The evaluation scenario involves six TAPAS clients running in parallel, two for each resolution. An evaluation run lasts for six minutes, whereas the clients issue a video request randomly within the first 60 seconds of the run. Four bandwidth limitation values are considered,

[2]https://peach.blender.org/

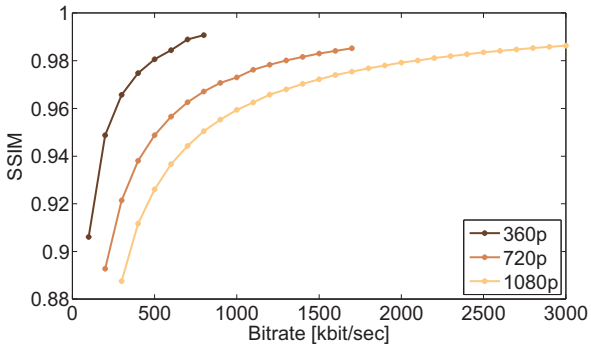

Figure 3. Obtained SSIM values per resolution and encoding bitrate

namely 1440 kbit/sec, 2400 kbit/sec, 4800 kbit/sec, and 7000 kbit/sec, and for each of them we repeat evaluation runs five times. For SPM1 and SPM2 we consider four configurations, which combine different values of network monitoring frequency (time intervals of two and four seconds) and of safety margin for the computed segment arrival time (set as 5% and 10% of the calculated value). For QoE-FF, we consider 100%, 95% and 90% of the network bandwidth as available for the quality distribution algorithm.

IV. Evaluation Results

In this section, we discuss evaluation results in terms of messaging overhead and key QoE indicators for the investigated App-Net approaches. We focus on the "steady-state" phase between two and six minutes, i.e. the impact of DASH clients joining the system is neglected. The mechanism/configuration pairs and the corresponding label used in some of the result figures are shown in Table II. Confidence intervals for the evaluations are small and thus omitted.

A. Quantification of the Message Exchange

The messaging overhead, including the message types, is illustrated in Figure 4. The y-axis shows the number of messages averaged over all runs and bandwidth configurations. For the same configuration, both SPM implementations yield a similar number of messages. While the monitoring frequency has a significant impact on the overall number of exchanged messages, the impact of the arrival time safety margin is negligible. For QoE-FF, the number of exchanged messages is considerably smaller as compared to the SPM implementations, since it features no

Table II
Configurations for SPM and QoE-FF

	Monitoring frequency / Download time margin			
	2 s / 0.05	2 s / 0.1	6 s / 0.05	6 s / 0.1
SPM 1	[1,1]	[1,2]	[1,3]	[1,4]
SPM 2	[2,1]	[2,2]	[2,3]	[2,4]
	Used bandwidth share			
	100 %	95 %	90 %	
QoE-FF	[3,1]	[3,2]	[3,3]	

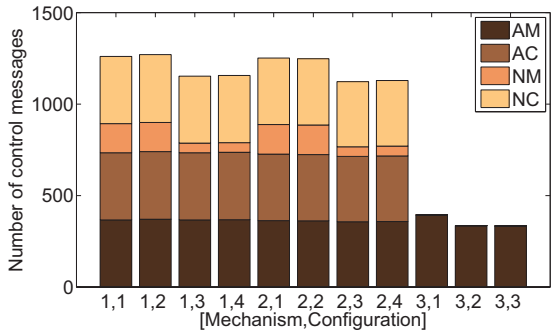

Figure 4. Number and types of exchanged messages for different configurations

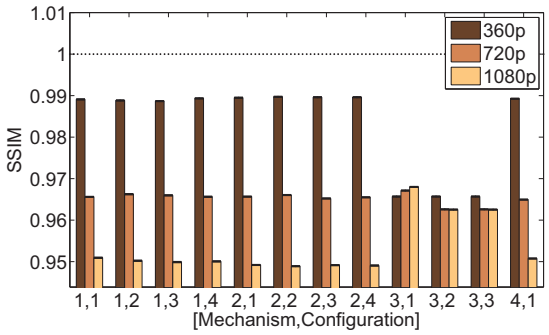

Figure 6. SSIM values as the fairness indicator

network control messages. Application control for QoE-FF is conducted when clients join or leave the system, i.e. no messages are exchanged in the steady state phase. However, application monitoring messages indicating a client is active are exchanged regularly.

B. Video Quality vs. Stalling

Figure 5 highlights the trade-off between average video quality in terms of SSIM and the associated stalling duration for different bandwidth configurations. In case of the 1440 kbit/sec bandwidth, none of the investigated App-Net approaches is capable of streaming without stalling. Depending on the mechanisms, either less stalling occurs, or a lower video quality is achieved. The different configuration parameters of each approach have a minor impact on this behavior.

The SPM 2 approach reduces the overall stalling time compared to the baseline implementation, while SPM 1 and QoE-FF provide a better video quality, but also a longer stalling time. For network bandwidth configurations higher or equal to 2400 kbit/sec the baseline implementation and both SPM approaches perform similar. No stalling times occur, and the average video quality is almost equal. The QoE-FF approach, however, significantly differs from the other approaches. If the mechanism is configured to use 100 % of the available bandwidth share the average video quality per user is higher, but also stalling occurs.

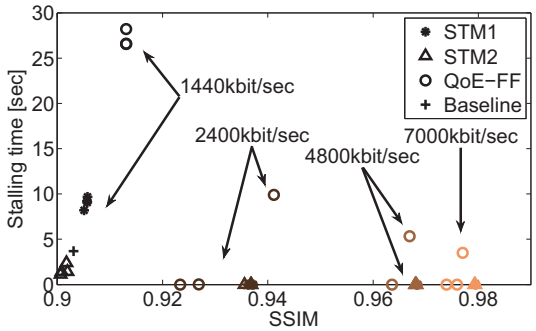

Figure 5. Comparison of SSIM values and stalling duration

This is due to the fact that the QoE-FF configuration does not take network and transport protocol overhead into account. This results in an overestimation of the available bandwidth and thus a too optimistic quality allocation to the clients.For the other configurations, 95 % and 90 % of the available bandwidth share, no stalling occurs, but the average video quality is lower as compared to the other mechanisms. A similar behavior can be observed for higher network bandwidth configurations, 4800 kbit/sec and 7200 kbit/sec.

C. Video Quality Fairness Among the Clients

Next, we highlight the impact of each approach on a fair QoE distribution among the involved clients. Figure 6 shows the measured SSIM values for different resolutions and configurations at the bandwidth limit of 4800 kbit/sec. Baseline DASH (mechanism 4) and all configurations for both SPM approaches show similar results in terms of video quality per client. Devices with the 360p resolution have higher SSIM values than devices with the 720p resolution, while 1080p devices suffer the most. This is due to the TCP fair share of network bandwidth among the clients, which does not regard different client resolutions and video quality per client. In case of QoE-FF, video quality between the heterogeneous clients is distributed in a fair manner. However, this quality fairness may come at the cost of overall video quality being decreased, as it was shown in the previous subsection.

V. Conclusion

This paper is a first step towards a systematic comparison of different QoE-centric App-Net interaction mechanisms. Firstly, we give an overview of such approaches and use them to derive an abstract interaction model based on the identified functional blocks. Secondly, we evaluate two of the App-Net approaches for adaptive video streaming.

The investigated approaches differ with respect to the number of exchanged messages and their influence on stalling and average video quality. The stalling prevention mechanism requires a constant message exchange and is thus very costly. It reduces stalling if little network resources are available, i.e., if all clients can be barely

supplied with the lowest quality. Its impact on the user-centric metrics is negligible if enough network resources for higher video qualities are available. The QoE fairness framework provides a fair video quality among heterogeneous clients for the price of a lower average video quality. As long as enough network resources are available no stalling times occur. If the lowest video quality can be barely supplied large stalling times are observed indicating that the approach should not be used in such a scenario.

Future work will provide a broader investigation of the App-Net approaches and an extended complexity evaluation beyond the messaging overhead. In addition, the adaptation potential of combining specific approaches will be addressed.

ACKNOWLEDGEMENT

This work has been supported/partially supported by the ICT COST Action IC1304 - Autonomous Control for a Reliable Internet of Services (ACROSS), Nov 2013 - Nov 2017, funded by European Union.

REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2014-2019 White Paper," Cisco Systems, Inc., San Jose, USA, Tech. Rep., 2015.

[2] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race, "Towards Network-wide QoE Fairness using Openflow-assisted Adaptive Video Streaming," in *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking (FhMN)*, Hong Kong, China, 2013, pp. 15–20.

[3] G. Cofano, L. D. Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia, and S. Mascolo, "Design and Experimental Evaluation of Network-assisted Strategies for HTTP Adaptive Video Streaming," in *Proceedings of the 7th ACM Multimedia Systems Conference (MMSys 2016)*, Klagenfurt, Austria, May 2016.

[4] S. Petrangeli, T. Wauters, R. Huysegems, T. Bostoen, and F. De Turck, "Network-based Dynamic Prioritization of HTTP Adaptive Streams to Avoid Video Freezes," in *Proceedings of the 2015 IFIP/IEEE Int'l Symposium on Integrated Network Management (IM)*, Ottawa, Canada, 2015, pp. 1242–1248.

[5] F. Wamser, L. Iffländer, T. Zinner, and P. Tran-Gia, *Mobile Networks and Management: 6th Int'l Conference, MONAMI 2014*. Würzburg, Germany: Springer International Publishing, 2015, ch. Implementing Application-Aware Resource Allocation on a Home Gateway for the Example of YouTube, pp. 301–312.

[6] M. Jarschel, F. Wamser, T. Höhn, T. Zinner, and P. Tran-Gia, "SDN-based Application-Aware Networking on the Example of YouTube Video Streaming," in *Proceedings of the 2nd European Workshop on Software Defined Networks (EWSDN 2013)*, Berlin, Germany, 2013, pp. 87–92.

[7] H. Nam, K. H. Kim, J. Y. Kim, and H. Schulzrinne, "Towards QoE-aware Video Streaming using SDN," in *Proceedings of the 2014 IEEE Global Communications Conference (GLOBECOM)*, Austin, TX, USA, 2014, pp. 1317–1322.

[8] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2015.

[9] "Qualinet White Paper on Definitions of Quality of Experience (2012)," 2013, European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003), Patrick Le Callet, Sebastian Möller and Andrew Perkis, eds., Lausanne, Switzerland, version 1.2.

[10] J. Zhu, R. Vannithamby, C. Rodbro, M. Chen, and S. Vang Andersen, "Improving QoE for Skype Video Call in Mobile Broadband Network," in *Proceedings of the 2012 IEEE Global Communications Conference (GLOBECOM)*, Anaheim, CA, USA, 2012, pp. 1938–1943.

[11] R. K. Mok, X. Luo, E. W. Chan, and R. K. Chang, "QDASH: A QoE-aware DASH System," in *Proceedings of the 3rd Multimedia Systems Conference (MMSys 2012)*, Chapel Hill, NC, USA, 2012, pp. 11–22.

[12] F. Wamser, T. Zinner, P. Tran-Gia, and J. Zhu, "Dynamic Bandwidth Allocation for Multiple Network Connections: Improving User QoE and Network Usage of YouTube in Mobile Broadband," in *Proceedings of the 2014 ACM SIGCOMM Capacity Sharing Workshop (CSWS)*, Chicago, IL, USA, 2014, pp. 57–62.

[13] O. Dobrijevic, M. Santl, and M. Matijasevic, "Ant Colony Optimization for QoE-centric Flow Routing in Software-defined Networks," in *Proceedings of the 11th Int'l Conference on Network and Service Management (CNSM 2015)*, Barcelona, Spain, 2015, pp. 274–278.

[14] ISO/IEC, "ISO/IEC DIS 23009-5 Information technology – Dynamic adaptive streaming over HTTP (DASH) – Part 5: Server and network assisted DASH (SAND)," International Organization for Standardization, Geneva, Switzerland, Under development, 2015.

[15] A. D. Ferguson, A. Guha, C. Liang, R. Fonseca, and S. Krishnamurthi, "Participatory Networking: An API for Application Control of SDNs," *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4, pp. 327–338, 2013.

[16] S. Paul and R. Jain, "OpenADN: Mobile Apps on Global Clouds Using OpenFlow and Software Defined Networking," in *Proceedings of the 2012 IEEE Global Communications Conference (GLOBECOM) Workshops*, Anaheim, CA, USA, 2012, pp. 719–723.

[17] S. Paul, R. Jain, J. Pan, J. Iyer, and D. Oran, "OpenADN: A Case for Open Application Delivery Networking," in *Proceedings of the 22nd Int'l Conference on Computer Communication and Networks (ICCCN 2013)*, Nassau, Bahamas, 2013, pp. 1–7.

[18] 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 *Proceedings of the 4th Int'l Workshop on Quality of Multimedia Experience (QoMEX 2012)*, Yarra Valley, Australia, 2012, pp. 1–6.

[19] T. Hoßfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing Effect Sizes of Influence Factors Towards a QoE model for HTTP Adaptive Streaming," in *Proceedings of the 6th Int'l Workshop on Quality of Multimedia Experience (QoMEX 2014)*, Singapore, 2014, pp. 111–116.

[20] Z. Wang, L. Lu, and A. C. Bovik, "Video Quality Assessment Based on Structural Distortion Measurement," *Signal Processing: Image Communication*, vol. 19, no. 2, pp. 121–132, 2004.

[21] L. De Cicco, V. Caldaralo, V. Palmisano, and S. Mascolo, "TAPAS: a Tool for rApid Prototyping of Adaptive Streaming algorithms," in *Proceedings of the 2014 Workshop on Design, Quality and Deployment of Adaptive Video Streaming (VideoNext)*, Sydney, Australia, 2014, pp. 1–6.

[22] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran, "Probe and Adapt: Rate Adaptation for HTTP Video Streaming at Scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 4, pp. 719–733, 2014.