Can context monitoring improve QoE? A case study of video flash crowds in the Internet of Services

Tobias Hoßfeld*, Lea Skorin-Kapov†, Yoram Haddad‡, Peter Pocta§, Vasilios A. Siris¶, Andrej Zgank‖, Hugh Melvin**

* University of Duisburg-Essen, Modeling of Adaptive Systems, Essen, Germany, Email: tobias.hossfeld@uni-due.de
† University of Zagreb, Faculty of Electrical Engineering and Computing, Zagreb, Croatia, Email: lea.skorin-kapov@fer.hr
‡ Jerusalem College of Technology, Jerusalem, Israel, Email: haddad@jct.ac.il
§ Dept. of Telecommunications and Multimedia, FEE, University of Zilina, Zilina, Slovakia, Email: pocta@fel.uniza.sk
¶ Department of Informatics, Athens University of Economics and Business, Greece, Email: vsiris@aueb.gr
‖ Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, Slovenia, Email: andrej.zgank@um.si
** Discipline of Information Technology, National Univ. of Ireland, Galway, Ireland, Email: hugh.melvin@nuigalway.ie

***Abstract*—Over the last decade or so, significant research has focused on defining Quality of Experience (QoE) of Multimedia Systems and identifying the key factors that collectively determine it. Some consensus thus exists as to the role of System Factors, Human Factors and Context Factors. In this paper, the notion of context is broadened to include information gleaned from simultaneous out-of-band channels, such as social network trend analytics, that can be used if interpreted in a timely manner, to help further optimise QoE. A case study involving simulation of HTTP adaptive streaming (HAS) and load balancing in a content distribution network (CDN) in a flash crowd scenario is presented with encouraging results.**

I. Introduction

The variety and consumption of multimedia services delivered across the Internet are continuously increasing. In addition, a paradigm shift is being witnessed in Internet service delivery, whereby we see a transition towards the Internet of Services (IoS), envisioning everything on the Internet as a service [1]. Such a transition will potentially lead to new services being realized as largescale service chains combining and integrating the functionality of other services offered by third parties (e.g., infrastructure, software, or platform providers). Key aspects and challenges to address will be the reliability and quality of service delivery, relying inherently on monitoring and quality estimation/prediction mechanisms.

In light of high market competition, one key differentiator between providers will be centered around end-user Quality of Experience (QoE). In order to successfully manage QoE, it is necessary to identify and understand the multiple factors affecting user QoE. Resulting QoE models dictate the parameters to be monitored, with the ultimate goal being effective QoE optimization strategies [2]. The majority of QoE-based management approaches to-date may be primarily related to either network management (based on monitoring and exerting control on access and core network level) or application management (adaptation of quality and performance on end-user and application host/cloud level) [3].

Going beyond an ordinary QoE management, additional information may be exploited to optimize the services on a system level, e.g. resource allocation and utilization of system resources, resilience of services, but also the user perceived quality. While (in-session) QoE management mainly targets the optimization of current service delivery and currently running applications, the exploitation of context information by network operators may lead to more sophisticated traffic management, reduction of the traffic load on the inter-domain links, and a reduction of the operating costs for the ISPs. Context monitoring in the broadest sense aims to obtain maximum information about the current system situation.

The paper is organized as follows. Section II provides a brief overview of QoE monitoring and potential extensions towards context monitoring. A case study involving cloud-based HTTP adaptive streaming (HAS) in a flash crowd scenario is presented in Section III, including simulations and numerical results. Discussions on technical solution path and concluding remarks are given in Section IV.

II. Involving context in QoE monitoring

QoE monitoring can be performed at the end-user or terminal level, the network level, or a combination of the two, e.g. see [4]. Network monitoring focuses on measuring QoS parameters (e.g., packet loss, delay), whereas QoE monitoring focuses on understanding how various parameters collectively impact the end-user quality, and includes QoS–QoE mapping. [5]. Application layer monitoring may also be important, e.g., for YouTube, estimating the buffer status can help recognize or predict stalling events [6]. To estimate application-specific parameters inside the network, mechanisms such as Deep Packet Inspection (DPI) are required.

We suggest that further improvement in QoE can be achieved by monitoring additional context data, potentially gathered from a broad range of sources. Context monitoring (Figure 1) can, in its broadest sense, significantly enrich QoE monitoring by providing better current and future information about a system's properties and state, which cannot be obtained using mechanisms such as DPI. Exploitation of context information can lead to more sophisticated system and

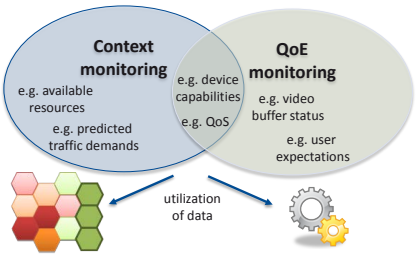

Fig. 1. QoE and context monitoring.

traffic management, reduced traffic load on inter-domain links, and eventually reduced operating costs for ISPs. Moreover, predictions of future traffic demands may be used to improve resource allocation. Such predictions can be based on monitoring and modeling network parameters [4] or other important resources, which reflect the current or future user behavior. Media and social networks can play an important role. For example, monitoring social networks' text streams with natural language processing and machine learning algorithms can identify short-term trends in popular content or possible 'high-profile' future events [7].

Context monitoring requires models and metrics that capture the system state at the network and service layer, application/service demands, and the capabilities of end-user devices. Challenges for context monitoring include (1) identification of the relevant context information that influences QoE, (2) improved QoE estimation and control that exploits context information, and (3) specification of a cross-layer QoE monitoring and management architecture that can incorporate context information from various sources. SDN can help address the above challenges [8], as discussed in Section IV.

III. Example Use Case: Video Flash Crowd

To assess the potential of context monitoring, a simulated use case is implemented to provide qualitative data. In the simulated scenario, a flash crowd of users watching the same video is examined. This sudden (but often temporary) increase of popularity is a typical phenomenon in video platforms with user-generated contents like YouTube. Video cascades emerge due to the announcement of a 'new' or 'popular' content on the video platform website, but also through the announcement via other channels. Those flash crowds may also be temporally limited because of event-related content, spatially limited because of regional interests or socially limited due to social interest groups [9].

The underlying content delivery network (CDN) consists of multiple server edge clusters to serve the video requests. According to some CDN load balancing strategy, the user requests are delegated either to CDN 1 or CDN 2 in our simulation. The information about the actual load in the edge cluster is updated infrequently, in the order of minutes. Thus, in a flash crowd scenario, many users may arrive within a short timeframe and may be delegated to a single particular CDN.

The delivery of the actual video to the customer uses HTTP adaptive streaming (HAS). The video contents are downloaded in segments, each with specific quality levels. At the client side, QoS parameters may be monitored and based on this information, the quality level of the next video segment is then determined in such a way, that video interruptions are minimised whilst optimising quality. In contrast to the CDN load balancing, this intelligence is client-driven.

A. Simulation Model

The simulation model includes the following components: video player, HAS algorithm, video contents, flash crowd arrival of users, CDN load balancing strategy. The concrete values of the parameters are not relevant, since the focus is on qualitative statements. The video player starts with the video playout, as soon as there are more than $x = 6$ video seconds in the video buffer. If the buffer depletes, the video stalls until the video buffer reaches the threshold x [6]. For the sake of simplicity, we consider a video with a constant segment size of 2 s. The video is available in two different quality layers. In total, the higher quality layer leads to a video size 2.72 times larger than the lower layer. The flash crowd consist of $N = 30$ users which arrive according to a Poisson process with $\lambda = 1/2\text{s}$, i.e. with high probability $P(T \leq 90\,\text{s}) = 99.27\,\%$ the time T until all N users arrive is below 90 s, as T follows an Erlang-N distribution with parameter λ.

Two different CDN load balancing strategies are implemented. First, the CDN directs the first K users to CDN 1, subsequent users are assigned to CDN 2. Second, the CDN implements context monitoring and obtains information about the flash crowd scenario from a third party. In that case, the users are assigned to the CDN with the lowest number of users which ensures a fairer load balancing.

Three different HTTP adaptive streaming algorithms are implemented in the simulation for each of the two CDN strategies. The first HAS strategy only considers the actual buffer and the throughput of the last segment, in order to estimate if the next segment can be downloaded in high quality, such that the video does not stall. The 2nd HAS strategy receives context information from the network which evaluates the current load in the CDN. In particular, this information details the overall capacity of the CDN and the current amount of requested video volume per CDN. Based on this information, it decides whether the next segment can be downloaded in high or low quality. Finally, to compare the results, a 3rd non-adaptive streaming algorithm is implemented which always requests segments in its highest representation.

B. QoE Model

According to [6], the main influence factors on HTTP video streaming are stalling events which significantly decrease the QoE. In particular, an exponential decrease of QoE is observed depending on the number X of stalling events. The quality switching due to HAS additionally affects QoE. The results in [10] clearly demonstrate the high impact of the switching amplitude between two played back representations. Further,

the time t on highest video quality layer is a major QoE influence factor, while the actual number of quality switches can be neglected. As a result, a simplified model is proposed which yields an exponential relationship according to the IQX hypothesis depending on the time t on highest layer.

Since stalling events additionally degrade the QoE, the following QoE model is considered. The time on the highest layer defines the upper bound of QoE. Any additional stalling event decreases the QoE according to the IQX hypothesis [5]. This rationale leads to the following function to quantify QoE depending on the number X of stalling events and the time t on the highest layer: $Q(X,t) = Q_1(t) \cdot Q_2(X)$ with $Q_1(t) = 0.003 \cdot e^{0.064t} + 3$ and $Q_2(X) = ((3.5 \cdot e^{-0.3X} + 1.5) - 1)/4$. The parameters are taken from [6], [10] derived from subjective tests. Q_2 is normalized to be between 0 and 1. Then, $Q(X,t)$ returns the QoE value on a MOS absolute category rating scale from 1 to 5 based on the rationale above.

C. Numerical Results

In the simulation, CDN 1 can serve on average 35 and 13 users in low and high quality, respectively, such that no stalling occurs. CDN 2 has lower capacity - 26 and 10 users, respectively. As maximum capacity from both CDNs for high quality streams is 22, HAS is required to avoid stalling, as $N = 30$ users arrive over a short time. The 1st CDN strategy sets the threshold parameter, $K = 13$, which should allows all users in CDN 1 to watch the video in highest quality.

Figure 2 shows the simulation results for CDN strategy 1 and HAS strategy 1, i.e. without using any context information. For each user the number X of stalling events, the average video quality expressed as time t on high layer, and the resulting QoE value $Q(X,t)$ is plotted on the y-axis, while the x-axis depicts the arrival time of the user.

The CDN load balancing strategy (with $K = 13$) does not know about the flash crowd and reacts too slow. Hence, some users experience stalling even though enough capacity is available to serve all. Stalling events are rather limited for users served by CDN 1 (not shown), as the network requirements are reduced by switching to lower video quality. However, users in CDN 2 suffer more from stalling. The HAS algorithm is necessary to adapt to the current network situation and to reduce the number of stalling events. If the high video quality layer is requested (labelled above in Table I 'No HAS', i.e. strategy 3), most of the users suffer from stalling resulting in bad QoE. In that case, the average number of stalling events per user is 4.40 instead of a value of 3.77 for strategy 1 labelled 'client' above i.e. default HAS (Figure 2) that only considers the throughput of the last segment and the actual video buffer status. If additional context information about the current load in the CDN is taken into account, labelled 'context' above, the number of stalling events can be reduced to 0.07 by lowering the video quality (strategy 2).

TABLE I
IMPACT OF CDN LOAD BALANCING STRATEGIES AND HAS ADAPTATION MECHANISMS ON AVERAGE QOE WITH AND W/O CONTEXT INFORMATION.

HAS quality selection		CDN load balancing strategy			
type	information used	context	$K = 8$	$K = 13$	$K = 18$
no HAS	none, high quality	3.67	2.30	2.66	2.55
client	throughput, buffer	3.62	2.07	2.36	2.08
context	load in CDN	3.68	3.50	3.45	2.82

Table I summarizes all results. It quantifies the impact of the CDN load balancing strategies (1-Default with different K values or 2- Context) and the HAS mechanisms (3 strategies) on the average QoE. The simulation is repeated 15 times and the mean values are given. It can be seen that the utilization of context information by the CDN strategy and/or the HAS mechanism improves QoE. It is difficult to select a proper value K for the CDN threshold, as the results strongly depend on the actual flash crowd. Therefore, a proper information exchange mechanism is required to make the information available across layers.

Next, we consider how early the video flash crowd has to be determined by the CDN load balancing. The earlier it is recognized, the better the system performance should be. As an upper bound, the context CDN load balancing has global knowledge and immediately balances between both CDNs.

The results in Figure 3 show that context monitoring can

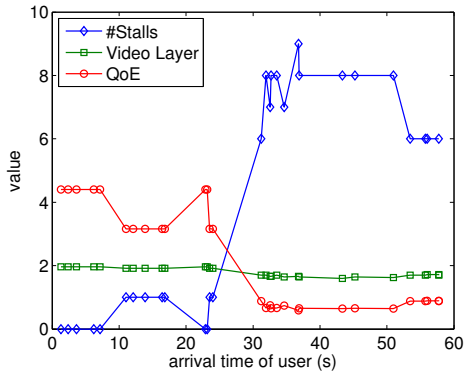

Fig. 2. No context information is used by the CDN load balancing strategy and HAS quality adaptation mechanism.

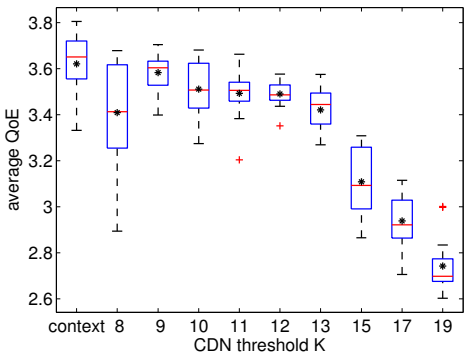

Fig. 3. Boxplot of the average QoE for all users depending on CDN load balancing threshold in comparison with optimal CDN load balancing strategy.

significantly increase QoE. However, the results are very sensitive due to the dynamics and interactions of the HAS control loop and the CDN load balancing. For example, a threshold of $K = 8$ leads to worse results than $K = 13$, whereas better results might be expected with earlier load balancing, i.e. for smaller values of K. Thus, in practice, realistic tests and input models are required to quantify the results and to derive reasonable thresholds.

IV. Discussions and Conclusions

The described results serve to demonstrate the potential through exploitation of relevant context data. In the use case, the contextual information regarding the formation of a flash crowd is collected by a third party . Other studies also addressed related improvements, such as the approach proposed in [11] suggesting a video control plane that can dynamically adapt both the CDN allocation and video bitrate based on global context knowledge of network state, distribution of active clients, and CDN performance variability.

While we have addressed a video flash crowd scenario for context monitoring, other scenarios addressed in related work draw similar conclusions. E.g., significant work has addressed the challenges arising from multiple concurrent clients accessing HAS video in a given access network, thereby competing for bandwidth across a shared bottleneck link [12]. Problems arise due to individual clients making adaptation decisions based on local observations, hence clients' adaptation behaviors interact with each other and result in quality oscillations. Various solutions proposed in the literature to tackle this issue involve centralized network-based solutions deployed using software defined networking (SDN) [13], enhanced client-side adaptation to improve fairness among flows [14], and server-based traffic shaping [15]. What is clear in all cases, however, is that context data in the network , could likely be utilized in driving/controlling quality adaptation decisions (on a domain wide level), consequently reducing oscillations and improving QoE.

The flash crowd scenario illustrates the benefits of employing SDN as a technological solution. With traditional IP networks, decisions are made based on local knowledge, so even if a server that did data mining on social networks could forecast a high download rate, it will be challenging to perform load balancing. This would require changing the routing policy at the BGP protocol but maybe also have some impact on the IGP which is confined to its Autonomous System. However, with SDN, this very complex and time consuming task is simplified. The forwarding decision is taken higher up by the controller, so an update to the routing decision at the controller implements the load balancing. From a technical perspective, further insight is needed regarding the information exchange between SDN controllers that are responsible for different domains, and between SDN controllers and other entities that can provide context information.

As regards future work, one aspect is to specify relevant context monitoring and exploitation scenarios for numerous additional use cases, e.g., online gaming and large scale video conferencing, which present more significant challenges in terms of high interactivity and stringent realtime requirements for effective management. Finally, underlying IoS business models will play a key role in putting an effective QoE management scheme based on enhanced monitoring into practice.

Acknowledgment

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] R. Moreno-Vozmediano, R. S. Montero, and I. M. Llorente, "Key challenges in cloud computing: Enabling the future internet of services," *Internet Computing, IEEE*, vol. 17, no. 4, pp. 18–25, 2013.

[2] T. Hoßfeld, R. Schatz, M. Varela, and C. Timmerer, "Challenges of QoE Management for Cloud Applications," *IEEE Communications Magazine*, vol. 50, no. 4, 2012.

[3] R. Schatz, M. Fiedler, and L. Skorin-Kapov, "Qoe-based network and application management," in *Quality of Experience: Advanced Concepts, Applications and Methods, S. Möller and A. Raake (eds)*. Springer, 2014, pp. 411–426.

[4] R. Schatz, T. Hoßfeld, L. Janowski, and S. Egger, "From Packets to People: Quality of Experience as New Measurement Challenge," in *Data Traffic Monitoring and Analysis: From measurement, classification and anomaly detection to Quality of experience*, E. Biersack, C. Callegari, and M. Matijasevic, Eds. Springers Computer Communications and Networks series, Volume 7754, 2013.

[5] M. Fiedler, T. Hoßfeld, and P. Tran-Gia, "A Generic Quantitative Relationship between Quality of Experience and Quality of Service," *IEEE Network*, vol. 24, Jun. 2010.

[6] T. Hoßfeld, R. Schatz, E. Biersack, and L. Plissonneau, "Internet Video Delivery in YouTube: From Traffic Measurements to Quality of Experience," in *Data Traffic Monitoring and Analysis: From measurement, classification and anomaly detection to Quality of experience*, E. Biersack, C. Callegari, and M. Matijasevic, Eds. Springers Computer Communications and Networks series, Volume 7754, 2013, pp. 264–301.

[7] A. Hrriyetoglu, F. Kunneman, and A. van den Bosch, "Estimating the time between twitter messages and future events." in *DIR*, ser. CEUR Workshop Proceedings, C. Eickhoff and A. P. de Vries, Eds., vol. 986. CEUR-WS.org, 2013, pp. 20–23.

[8] M. Jarschel, T. Zinner, T. Hoßfeld, P. Tran-Gia, and W. Kellerer, "Interfaces, attributes, and use cases: A compass for sdn," *Communications Magazine, IEEE*, vol. 52, no. 6, pp. 210–217, 2014.

[9] X. Cheng, H. Li, and J. Liu, "Video sharing propagation in social networks: Measurement, modeling, and analysis," in *IEEE INFOCOM 2013*. IEEE, 2013, pp. 45–49.

[10] T. Hoßfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing Effect Sizes of Influence Factors Towards a QoE Model for HTTP Adaptive Streaming," in *6th Int. Workshop on Quality of Multimedia Experience (QoMEX)*, Singapore, Sep. 2014.

[11] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang, "A case for a coordinated internet video control plane," in *ACM SIGCOMM 2012*, 2012, pp. 359–370.

[12] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hossfeld, and P. Tran-Gia, "A survey on quality of experience of http adaptive streaming," *Communications Surveys Tutorials, IEEE*, vol. PP, no. 99, pp. 1–1, 2014.

[13] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race, "Towards network-wide qoe fairness using openflow-assisted adaptive video streaming," in *ACM SIGCOMM 2013 workshop on Future human-centric multimedia networking*. ACM, 2013, pp. 15–20.

[14] J. Jiang *et al.*, "Improving Fairness, Efficiency, and Stability in HTTP-Based Adaptive Video Streaming With FESTIVE," *IEEE/ACM Transactions on Networking*, vol. 22, no. 1, p. 326/340, February 2014.

[15] S. Akhshabi, L. Anantakrishnan, C. Dovrolis, and A. C. Begen, "Server-based traffic shaping for stabilizing oscillating adaptive streaming players," in *23rd ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*. ACM, 2013, pp. 19–24.