Mobile DASH Services using the FLAME Future Media Platform

Fotis Foukalas
Athanasios Tziouvaras*
fotis@cogninn.com
thanasis@cogninn.com
Cognitive Innovations
Athens, Greece

Carolina Fernández
August Betzler*
carolina.fernandez@i2cat.net
august.betzler@i2cat.net
Fundació i2CAT
Barcelona, Spain

ABSTRACT

In this work we present MoDASH, a mobile DASH solution for Future Multimedia applications such as Virtual Reality (VR). We focus both on managing efficiently the available network resources as well as delivering a high Quality of Experience (QoE) to the end-users; which is achieved respectively by employing data and computational offloading techniques as well as QoE-aware resource management signaling. Our solution relies on a Network Function Virtualization (NFV) and trigger-based platform that can be used in small cell networks. This is demonstrated in the FLAME platform, where the supported monitoring of the DASH streaming and triggers allows to measure the real-time performance of the solution and dynamically adapt it to the changing conditions.

CCS CONCEPTS

• **Networks → Cloud computing**; **Location based services**; **Network management**; **Network monitoring**.

KEYWORDS

DASH; NFV; QoE; data offloading; computation offloading; resource management

ACM Reference Format:
Fotis Foukalas, Athanasios Tziouvaras, Carolina Fernández, and August Betzler. 2020. Mobile DASH Services using the FLAME Future Media Platform. In *The Adjunct Proceedings of ACM TVX/IMX 2020, Barcelona (Spain), June 2020. Copyright is held by the author/owner(s).*. ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

MoDASH targets some of the core challenges of the Future Media Internet (FMI). In particular, we aim to support future mobile DASH-enabled VR applications. The low latency, reliable distributed networking and computing requirements are addressed by our solution. The FLAME platform provides the tools to deploy MoDASH on top of a city-wide infrastructure that allows for a high degree of prototyping and ease of use [5]. As such, FLAME allows to deploy a solution to the edge of the network that is responsible for orchestrating the QoE for the end-users connected over the infrastructure's Wireless Access Network [9]. The QoE-based orchestration introduced by MoDASH provides many relevant functionalities for 5G, such as reporting of QoE parameters, SDN-enabled deployment [6] and estimation of metrics that are relevant to enhance the VR user experience. Overall, MoDASH targets QoE-based orchestration for provisioning resources under mobility and interactivity use cases, which are relevant in future Multimedia Services such as VR over Wireless Access Networks [1]. Such orchestration is deployed on the edge via the network function virtualization (NFV) and monitoring functionalities facilitated by FLAME. Future Media Service providers can support dynamic service request and routing capabilities, and MoDASH offers the potential to reduce the overall costs while ensuring fast availability of services towards end users.

The trial in the FLAME infrastructure at Barcelona demonstrated the management and orchestration capabilities of the Media Service Functions through the server at the edge (street cabinet) and the cluster of servers at the (*OMEGA*) Data Center (DC) locations. This validated the techniques used by MoDASH:

- **Data offloading**: transfers and caches chunks of multimedia files at the edge, where dedicated storage for media content can be deployed on demand.
- **Computation offloading**: migrates the transcoding service, which is critical for the DASH functionality to the edge.
- **QoE-aware resource management**: in case of an interactive video, video content transcoding to a large set of videos with different bitrates and resolutions is provided.

The rest of this paper details the MoDASH solution in Sec. 2, the novel signaling solutions being used in Sec. 3 and the use cases shown in the demonstration in Sec. 4. Sec. 5 concludes the paper.

2 CONCEPT

Fig.1 depicts the MoDASH Service Function Chain (SFC) as composed of six distinct Foundation Media Services (FMS) that are provided by FLAME [5], plus one service which allows the user to run the experiment. The deployed SFC consists of the following services:

- **fms-master-streaming**: handles the user-generated DASH requests and provides the streaming service delivery. This service segments the video content into smaller segments that are periodically delivered to the user.
- **fms-proxy-streaming**: as the content is stored at the edge (by *fms-proxy-storage*), this service caches the incoming video segments to the edge, before forwarding them to the users for consumption.
- **fms-transcoding**: transcodes the stored content on the edge to produce different video qualities.
- **fms-master-storage**: stores the content at the DC and creates a database which refers to such content.
- **fms-proxy-storage**: like *fms-master-storage*, it stores the content at the edge. It creates a database which refers to such content and is used to find the corresponding video ID, so that the video can be transcoded and forwarded to the *fms-proxy-streaming* service. The latter can utilize the

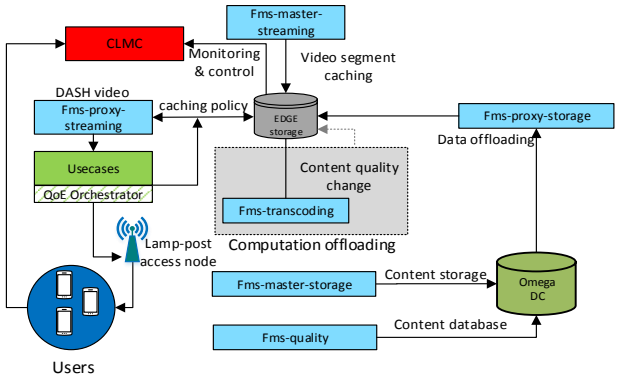

Figure 1: MoDASH SFC

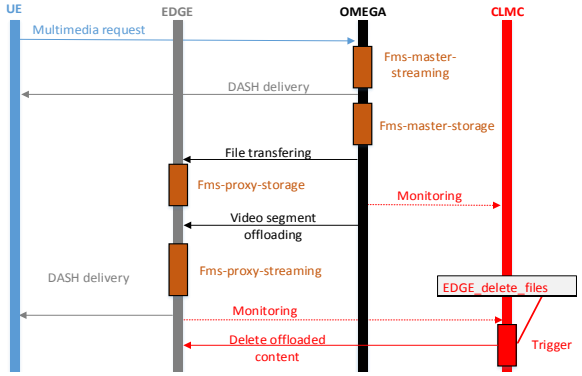

Figure 2: Data offloading for mobile DASH services

existing video qualities to adapt the streaming quality based on the network capacity.

- **fms-quality**: obtains content-related information, such as file size and resolution of transcoded videos. This information is complemented with that provided by the FLAME Cross-Layer Management and Control (CLMC) component, which enables the definition of the states of the SFC in runtime based on environment conditions that are triggered, allowing MoDASH to adapt to the traffic conditions. The CLMC measurements are also used to monitor the execution of the experiment and for post-mortem evaluation.
- **use_cases**: presents a portal to the user where each of the use cases described in the following sections can be tested.

The *fms-master-storage* and *fms-proxy-storage* services are the cornerstone of the data offloading signaling procedure used in the MoDASH experiment, thus being essential a proper communication between them. Both the data offloading procedure and the DASH delivery are essential to the experiment.

3 NOVEL SIGNALING TECHNIQUES

The introduction of signaling solutions enables the dynamic adaptation of the environment. MoDASH uses the triggering system of the FLAME platform to first raise alarms based on the monitoring of specific network conditions and then to react based on these, thus tailoring the usage of resources (disk space, CPU, RAM, number of connections) to the needs of the experiment. This, along with the definition of the state of any FMS upon each trigger, makes it possible to minimize the usage of resources at every time. For instance, a service can be on stand-by, consuming a restricted subset of resources, and be ready for activation with less response time. Compared to a typical deployment in the cloud, this is a convenient feature in both performance and economic terms.

In order for MoDASH to provide the expected QoE we deploy a QoE-based orchestration framework for mobile DASH services that is capable of both providing functionalities that enhance the QoE and managing the infrastructure resources. The QoE-based orchestrator is composed of a data offloading signaling procedure, a computation offloading signal processing module and a QoE-aware resource manager. Below we describe such methodologies and we highlight how they differ from existing approaches.

Fig. 2 depicts the data offloading signaling process we employ. In this scenario the User Equipment nodes (UEs) periodically create multimedia requests that are redirected to the DC, acting as the main cluster of the infrastructure. First, the *fms-master-storage* service is invoked, which allows the user to retrieve any content stored on the DC. Afterwards, the data offloading mechanism takes place, so segments of the media files are transferred to the edge through the *fms-proxy-storage* service. The edge, in return, utilizes the *fms-proxy-streaming* service to cache the video segments received. The *fms-master-streaming* service is also used as a master streaming service and thus, collaborates with the *fms-proxy-streaming* service to operate properly. Both the *fms-proxy-streaming* and the *fms-master-streaming* services achieve the DASH content delivery to the end user. The CLMC component is used to monitor the storage capacity in the edge. If an event is triggered, the alert system is notified to take action to contain the problem. Specifically, Fig. 2 depicts how the red highlighted "EDGE_delete_files" trigger detects low disk storage and the edge server is notified to free space according to the employed caching policy. In this way the storage resources at the edge are efficiently managed as we make sure to have enough space available to store new content requested by the users. To ensure higher QoE levels we commence the DASH delivery from the DC before the data offloading sequence completes and then we switch to edge side streaming when the data offloading procedure finishes.

A previous study in data offloading methodologies in [7] suggests that migration of large data chunks to the edge of the network raises the QoE perceived by the users by 22% to 26% compared to a baseline solution which does not employ data offloading. As such, we employ the data offloading signaling procedure on top of the FLAME platform. Our solution goes beyond the standard data offloading techniques as we also deploy a resource management technique to monitor the remaining edge resources and to react based on such measurements.

Fig. 3 depicts the computation offloading signaling procedure. The main task of this procedure is to redistribute the computation workload across the available servers of the network in order to manage the available resources, as shown in [3]. In this scenario, the *fms-transcoding*, *fms-proxy-storage* and *fms-proxy-streaming* services are deployed at the edge while the *fms-master-storage* and

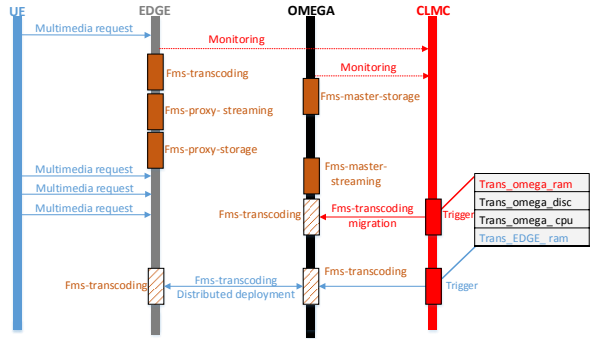

Figure 3: Computation offloading for mobile DASH services

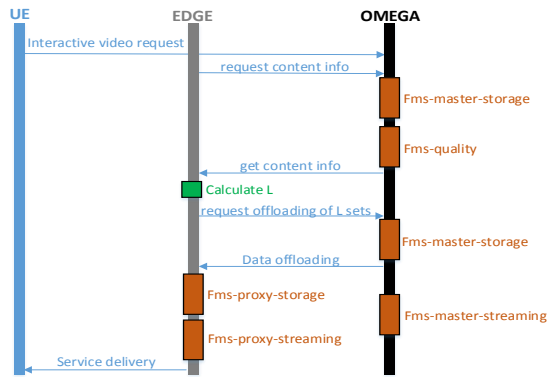

Figure 4: QoE-aware resource management for mobile DASH services

fms-master-streaming are deployed at the DC. Starting with a single UE requesting video content and periodically adding more UEs, the overall demand keeps increasing. This procedure is closely monitored by the CLMC component and when the workload of the *fms-transcoding* service reaches a high threshold; a trigger is fired. In figure 3, due to continuous user requests, the RAM usage of the *fms-transcoding* service has reached the defined threshold and the red trigger ("Trans_omega_ram") is fired. In order to avoid RAM congestion, which in return may reduce the performance of the service, the *fms-transcoding* service is migrated to the DC server, consequently relieving the workload of the edge. When the number of requests from the UE drops, a significant portion of the RAM on the edge is released. The CLMC detects the low RAM utilization and the blue-colored trigger ("Trans_EDGE_ram") is fired. This time the *fms-transcoding* service is deployed back at the edge in a distributed fashion, utilizing both the edge and the DC resources.

Previous work in computation offloading [8] suggests that by deploying a proper computation offloading algorithm, the execution time of a network service can be significantly reduced while also the power consumption of the UE will drop, thus allowing the device to operate for longer periods of time[10]. We also employ such an approach and we combine it with a QoE-oriented methodology to further increase the QoE of the end-user. As such, and as suggested by [9], we opt to transfer the transcoding service to the DC server (which resembles the cloud in our experiment) when we observe a high resource utilization at the edge. Due to the dynamic nature of our deployment mechanism, our approach differs from the standard paradigm in that we do not only migrate services to the cloud (DC here), but we also deploy such services in a distributed way when certain criteria are met for improved resiliency and more efficient usage of the SDN-enabled FLAME architecture.

Fig. 4 depicts the QoE-aware resource management signaling procedure. The QoE-aware resource management focuses on maximizing the efficiency of the data offloading signaling procedure for DASH streaming applications, considering both the QoE and the available resources at the edge. Specifically, the UEs periodically generate interactive video requests, which are then headed to the DC alongside with a "content-info" request that originates from the edge. After that, the *fms-master-storage* and the *fms-quality* services respond to the edge and forward any data related to the corresponding content information. Such data includes the characteristics of the available transcoded video sets. Next, an L value is selected such that it minimizes the following formula:

$$L = argmin_j(C_t + \mu \sum_{k=0}^{k=j} CS_k) \quad (1)$$

Where C_t is the transcoding cost of the j-segment measured in bitrate, μ is a constant coefficient measuring the cost of synthesizing the j-segment of the interactive video and CS_k is the caching cost of the k-segment measured in bytes. After L is computed, the *fms-proxy-storage* service is notified to offload the L lowest quality transcoded video sets to the edge by invoking the data offloading procedure. Finally the *fms-master-streaming* and *fms-proxy-streaming* services deliver the content to the end-user. Although equation 1 can be employed for regular use case data offloading, we opt for using such formula on interactivity use case only due to the increased storage requirements of interactive video applications compared to regular video streaming applications.

The QoE-aware resource management employs a signaling procedure that is applied to interactive video applications. An interactive video is transcoded to a large set of videos with different bitrates and resolutions [4]. When the viewer requests a video service, a transcoded copy of the original video is selected so that it is properly fit to the channel bandwidth. We consider that the data offloading to the edge of all the video sets created is an inefficient technique that will consume most of the edge resources. As a result, a different offloading technique should be introduced to handle the data offloading of interactive video applications. Previous work in [11] suggests approaching such scenario by adapting the caching strategy specifically for interactive applications. We adopt such an approach and we combine it with a data offloading signaling procedure to provide QoE with respect to the network resource utilization.

4 DEMONSTRATION

We present here two scenarios that demonstrate the MoDASH functionality and effectiveness: the *regular* and *interactivity* use cases.

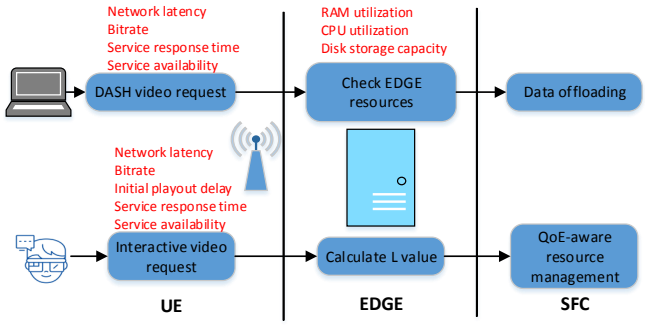

Figure 5: Regular (top) and interactivity (bottom) use cases

We discuss each in detail and provide Key Performance Indicator (KPI) measurements that reflect the QoE perceived by users according to 3GPP [1]. Fig. 5 depicts the MoDASH methodology for the regular and interactivity use cases.

During the regular use case, the UE creates a DASH video request. Initially, the remaining free disk storage of the edge server is inspected: if sufficient to host this video, the data offloading procedure will run to offload the video to the edge; Otherwise, any previously stored content should be deleted to free space. Afterwards the new content is offloaded to the edge and the DASH service starts delivering the content to the user. We demonstrate the video streaming from the edge of the network as well as monitor KPIS related to 3GPP standards for VR applications [2] that indicate the performance of our techniques.

In the interactivity use case scenario the video is pre-transcoded to a large set of videos with different rates and resolutions and stored to the DC. First, the user creates an interactive video request and the edge calculates the value of L, as described by Eq. 1. Second, the QoE-aware resource management procedure is invoked utilizing the calculated L value. The L-lowest transcoded content versions are then offloaded to the edge while the rest remains stored in the DC. Finally, the DASH service delivers the appropriate video segments to the user either from the edge (if the segment belongs to the L-lowest segments) or from the DC. During the demo, it is possible to stream content along with monitoring the measured KPIs obtained via the CLMC component.

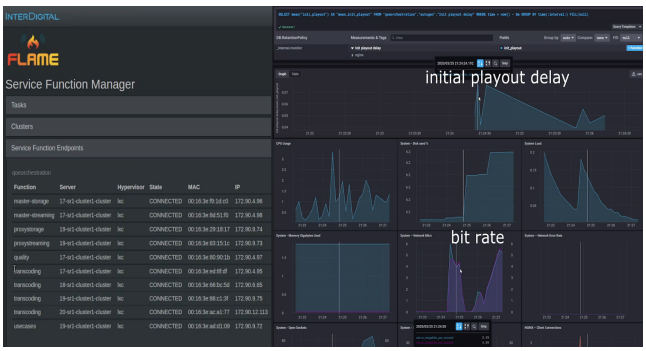

Figure 6: Orchestrator and CLMC dashboards

The aforementioned KPIs and measurements are highlighted in red in Fig. 5 and can be monitored from the CLMC as shown in Fig. 6. Throughout the demonstration we explain how the SFC implements the described uses cases, show the distributed deployment using the FLAME platform, stream the transcoded videos from different emulated UEs and show when the defined triggers are met, leading to automatic offloading procedures.

5 CONCLUSION

We present the MoDASH solution as part of our FLAME trial experiment, carried out in Barcelona during December 2019. This demonstration paper presents the detailed concept, the novel techniques it leverages on and the demonstrated use cases.

ACKNOWLEDGMENTS

This work has been funded by the EU-funded H2020 project FLAME with Grant agreement no. 731677, under its Open Call 2.

REFERENCES

[1] 3GPP 2017. *3GPP TR 26.909, Study on improved streaming Quality of Experience (QoE) reporting in 3GPP services and networks, v14.0.0.* 3GPP. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3041.

[2] 3GPP 2018. *3GPP TR 26.929, QoE parameters and metrics relevant to the Virtual Reality (VR) user experience, v.0.6.0.* 3GPP. https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3327.

[3] K. A. Harras Ab. Mtibaa and A. Fahim. 2013. Towards Computational Offloading in Mobile Device Clouds. In *IEEE 5th International Conference on Cloud Computing Technology and Science.* 331–338. https://doi.org/10.1109/CloudCom.2013.50

[4] K. Bilal and A. Erbad. 2017. Edge computing for interactive media and video streaming. In *Proceedings of the 2017 Second International Conference on Fog and Mobile Edge Computing (FMEC).* 68–73. https://doi.org/10.1109/FMEC.2017.7946410

[5] M. Boniface S. Phillips A. Betzler C. Fernandez D. Guija M. Catalan D. Trossen, S. Robitzsch and P. S. Khodashenas. 2019. A 5G Platform for Future Interactive Media Systems. (2019). https://ict-flame.eu/wp-content/uploads/sites/3/2019/01/5g-platform-future.pdf

[6] Y.Jia A.Kabir E.J.Kitindi, S.Fu and Y.Wang. 2017. Wireless Network Virtualization With SDN and C-RAN for 5G Networks: Requirements, Opportunities, and Challenges. *IEEE Access* 5 (2017), 19099–19115. https://doi.org/10.1109/ACCESS.2017.2744672

[7] X. Li H. Zhou, H. Wang and V. C. M. Leung. 2018. A Survey on Mobile Data Offloading Technologies. *IEEE Access* 6 (2018), 5101–5111. https://doi.org/10.1109/ACCESS.2018.2799546

[8] V. Haghighi and N. S. Moayedian. 2018. An Offloading Strategy in Mobile Cloud Computing Considering Energy and Delay Constraints. *IEEE Access* 6 (2018), 11849–11861. https://doi.org/10.1109/ACCESS.2018.2808411

[9] S.-T. Hong and H. Kim. 2016. QoE-Aware Computation Offloading Scheduling to Capture Energy-Latency Tradeoff in Mobile Clouds. In *Proceedings of the 2016 13th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON).* 1–9. https://doi.org/10.1109/SAHCN.2016.7733009

[10] M.F.Zolkipli M. Ali, J.M.Zain and G.Badshah. 2015. Battery efficiency of mobile devices through computational offloading: A review. In *2015 IEEE Student Conference on Research and Development (SCOReD).* 317–322. https://doi.org/10.1109/SCORED.2015.7449347

[11] J. Liang W. Wang X. Que Y. Guo M. Zhao, X. Gong and S. Cheng. 2017. QoE-driven optimization for cloud-assisted DASH-based scalable interactive multiview video streaming over wireless network. *Signal Processing: Image Communication* 57 (2017), 157 – 172. https://doi.org/10.1016/j.image.2017.05.015