Timber: An SDN based emulation platform for QoE Management Experimental Research

Arslan Ahmad, Alessandro Floris, and Luigi Atzori
DIEE, University of Cagliari, UdR CNIT of Cagliari, Italy
{*arslan.ahmad, alessandro.floris*}*@diee.unica.it, l.atzori@ieee.org*

***Abstract*—In this paper, we present an open source Software-Defined Networking (SDN) based emulation platform called *Timber*. It is aimed at providing the research community with a tool for experimenting new Quality of Experience (QoE) management procedures and tools in multimedia service delivery. *Timber* is developed on the top of Mininet SDN emulator and Ryu SDN controller, which provides the major functionalities of the traffic engineering abstractions in SDN environment. Moreover, the platform provides an actual complete video streaming application including the implementation of the server side and client side probes for QoE measurements which have functionalities to store the quality measurements into the cloud database accessible to the SDN controller application. In this paper, we first discuss the general architecture and framework of *Timber*. Secondly, we provide the implementation details and major functionalities of the platform. Thirdly, we provide experimental results to highlight the major functionalities of *Timber* by 4 different scenarios which include traffic shaping through DiffServ and dynamic resource allocation by queuing strategies.**

***Index Terms*—Quality of Experience (QoE), QoE Management, Software-Defined Networking (SDN), Mininet, Ryu controller.**

I. Introduction

Today the Internet traffic is mainly composed of multimedia traffic due to significant advancement in relevant technologies that are now exploited by the Over-The-Top service providers (OTTs) [1]. Such a huge rise in consumption of the multimedia services over Internet has raised an issue of delivering quality to the end users, i.e., the Quality of Experience (QoE). Hence, it is a matter of fact that QoE-aware service management approaches are gradually replacing classical QoS-aware service management to consider the subjective perception of the users. Accordingly, many studies proposed QoE-aware management approaches for different services and especially video streaming services [2]–[4].

The rise of Software-Defined Networking (SDN) provided new insights into network and service management. Being based on traffic engineering abstractions and virtualization techniques, SDN allows to simplify the implementation of traffic engineering techniques and the communication between network and service providers. SDN provides the ease to maintain a global view of the network to perform the network-wide management operations with the centralized programmable control plane separated from the data plane. SDN architectures are dynamic, manageable, cost-effective, and adaptable, and for these reasons have been widely used in the last years for the QoE-aware management of multimedia services [5], [6].

Currently, different open source SDN based network management tools are available such as Ryu controller[1]/Mininet[2]; however their use for QoE monitoring and management is not straightforward as those tools are not particularly designed for QoE monitoring/management purposes. Moreover, the learning curve of those tools is quite steep. Though some SDN-based QoE management solutions particularly designed for the specific purposes can be found in literature, on the top of which new algorithms cannot be evaluated. Hence, there is a need for an open source and general purpose SDN-based platform, which may assist the research community to evaluate QoE management and monitoring strategies.

In this work, we propose *Timber*, an open source SDN based emulation platform to provide the research community with an emulation tool for experimenting new QoE management procedures and techniques. *Timber* is developed on the top of the Mininet SDN emulator and Ryu SDN controller, which provides the major functionalities of the traffic engineering abstractions in SDN environment. Furthermore, *Timber* provides a complete video streaming application: the server side implementation includes the content hosting whereas the client-side implementation provides probes at the user end for the collection of QoE measurements, which are stored in a cloud database which is accessible to the SDN controller application for network-wide QoE management. The current implementation of *Timber* is equipped with QoE-aware traffic engineering techniques in SDN environment which include traffic shaping, dynamic resource allocation and prioritization of flows by DiffServ implemented through SDN controller applications. Finally, we provide experimental results to highlight the major functionalities of *Timber* by 4 different scenarios which include traffic shaping through DiffServ and dynamic resource allocation by queuing strategies.

The paper is structured as follows: Section II presents some of the most relevant QoE-aware service management studies considering SDN based approaches. Section III highlights

The work on this paper was funded from the European Unions Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 643072, Network QoE-Net (http://qoenet-itn.eu) and from Italian Ministry of University and Research (MIUR), within the Smart Cities framework (Project Netergit, ID: PON04a200490).

[1]https://osrg.github.io/ryu/
[2]http://mininet.org/

the motivations behind the implementation of *Timber* and the considered use cases. Section IV presents the proposed SDN based architecture including the implementation details, whereas in Section V preliminary experimental results are presented. Section VI concludes the paper.

II. Background

SDN based network management approaches are widely used for QoE optimization. In [5], an SDN-based function is presented, which is supposed to enable OTT players to monitor QoE and ensure QoE requirements by imposing policies/rules by the SDN controller in specific points of the network, as the means to perform traffic management. However, besides the proposed architecture, no implementation/experiment results is provided. [7] leveraged on SDN to implement an in-network QoE Measurement Framework (IQMF) that provides live network-wide QoE measurements. IQMF offers network measurements and the respective analysis as a service to the network provider or content distributor via an API, which can use it for QoE-aware service optimization. A combined architecture of LTE and SDN is proposed in [8], where an SDN controller was included into an LTE network as the QoS manager. APIs were implemented to allow the SDN controller to interact with the PCRF entity so that dynamic network changes could be made in accordance with LTE policy rules.

Most of the SDN-based architectures are focused on video streaming services. In [9], an SDN-based architecture is adopted for dynamic bandwidth allocation across multiple competing HTTP Adaptive Streaming (HAS) streams. A video QoE optimization application (VQOA) is implemented to collect information from various points in the network, by interacting with the SDN controller, to analyze the network state and provide a more accurate estimate of the end user QoE. Also, the VQOA is in charge of coordinating rate adaptation decisions in HAS clients and bandwidth allocation at the bottleneck link. Similarly, [6] proposed an intelligent streaming architecture, called SDNHAS, which leverages SDN capabilities of assisting HAS players in making better adaptation decisions. SDNHAS formulates the bitrate and quality decisions and then uses a per-cluster decision algorithm (Q-learning based) to successfully avoid sub-optimal decisions and to maximize QoE without any bandwidth violations and affecting other players QoE. To mitigate bottlenecks and improve video QoE, [10] leveraged an SDN platform from OTT video service providers point of view. The proposed SDN application is designed to monitor network conditions of streaming flow in real time and dynamically change routing paths using multi-protocol label switching (MPLS) traffic engineering to provide reliable video watching experience. By selecting better routing paths to provide higher resolution video during a download, simulations results show that the proposed QoE-aware mechanism allows for a 55.9% improvement in enhancing the viewing experience, especially during busy hours. In [11], a QoE-driven dynamic task allocation scheme is proposed, which makes use of SDN/NFV technologies to find and provide the best combination of network nodes that can cooperate during the execution of the tasks and in the same time to improve the overall QoE level of the end users. Preliminary results based on the Mininet network emulator and the OpenDaylight controller have shown that the proposed approach can significantly improve the QoE of a transmitted video by selecting the best path with normalized QoS values. [12] used an SDN-controller for a video streaming service to perform several tasks within the ISP network: i) communicate with the network devices in order to enable monitoring of the required data; ii) monitor the network; iii) analyze the data to perform the MOS estimation for the different users; iv) define an action plan to improve the quality of the delivered video.

However, all the aforementioned studies are particularly designed for specific objectives. Our objective is to provide the research community a platform for experimenting QoE management techniques in a realistic environment. To the best of author's knowledge, the only study proposing a framework to facilitate rapid experimentation of QoE models in SDN environments is [13], which is still in its early stage.

III. Motivations and use cases

Though some SDN-based QoE management solutions can be found in literature which are particularly designed for the specific purposes, as highlighted in Section II, one of the major drawbacks is that new algorithms cannot be evaluated on the top of these QoE management solutions. Hence, there is a need for an open source and general purpose SDN-based platform, which may assist the research community to evaluate QoE management and monitoring strategies.

When defining a QoE-aware management approach, the following functionalities should be considered: 1) *QoS monitoring*– collection of network-related parameters and measurement of the network state; 2) *QoS management*– traffic engineering actions aimed at optimizing network load and traffic flows; 3) *Service management*– optimization of service-related parameters; 4) *QoE monitoring*– collection of quality perceived by the user, user-related information and user's opinions and feedback; 5) *QoE prediction*– mathematical model which predicts user's QoE as a function of user-related parameters, network state and service state and; 6) *QoE controller:* software that, on the basis of the current predicted QoE, decides whether correcting actions should be taken on network and service parameters to optimize QoE.

It is evident that the QoE-aware management of multimedia services is not an easy task, especially due to different information and services in the hands of different providers. Indeed, whereas QoS monitoring and QoS management are in the hands of network providers, service providers may count on QoE monitoring and prediction. But the QoE controller must take decisions on the basis of the combined information owned by both the providers to efficiently optimize end users' QoE [14]. However, joint service management may be impeded by different un-interoperable systems which may discourage the providers towards collaborative approaches.

The proposed SDN based platform, *Timber*, may provide the solution to the need of collaboration among the providers

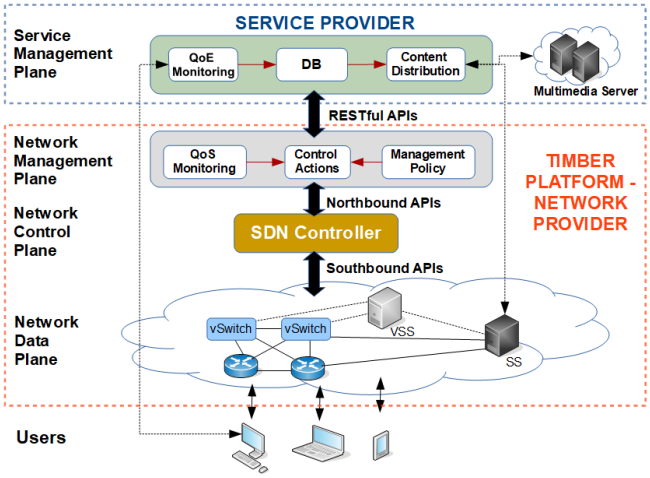

Fig. 1. Reference architecture.

to implement a QoE-aware joint service management. Indeed, thanks to traffic engineering abstractions and virtualization techniques applied by SDN, it is possible to abstract the different system's hardware and to define virtual functions aimed at the monitoring and sharing of information as well as combined controlling decisions. *Timber* has been mainly implemented to test different algorithms and collaboration approaches aimed at QoE-aware management in an open source emulated SDN environment. Hence, the proposed platform is based on the Mininet to create virtual networks that emulate real networks, as well as on the Ryu SDN controller, which allows the application of the traffic engineering solutions in the emulated SDN environment. Also, we implemented further modules to permit joint QoE-aware service management, such as client-side probes for QoE measurements, which have functionalities to store the quality measurements into the cloud database accessible to the SDN controller application and to both the providers. *Timber* is presented in Section IV.

IV. Proposed SDN based platform: Timber

In this Section, we present our proposed SDN based platform called *Timber*. First, we describe the *Timber* architecture and then the implementation details.

A. Architecture

Fig. 1 represents the reference architecture, which is composed of four planes: the Network Data Plane, the Network Control Plane, and the Network Management Plane are the planes concerning *Timber*, whereas the Service Management Plane is a cloud space owned by the service provider.

In the following, we provide the details for each plane.

- *Network Data Plane*: the physical and virtual network, which includes all network devices (switches, vswitches and routers) as well as Surrogate Servers (SSs) and Virtual Surrogate Servers (VSSs) for content replication applied by Content Delivery Networks (CDN), which can be owned by both the network and the service providers.
- *Network Control Plane*: the SDN controller through which it is possible to manage the network devices.
- *Network Management Plane*: the application layer of the SDN architecture controlled by the network provider. It includes: a QoS Monitoring module to monitor the performance of the network; a Management Policy module to take into account Service Level Agreements (SLA); a Control Actions module that decides, on the basis of specific algorithms and objectives, which network control actions should be implemented by the SDN controller to optimize the network resources and improve the quality.
- *Service Management Plane*: a cloud space owned by the service provider. It includes: a QoE Monitoring module to estimate the user's QoE on the basis of service parameters acquired at the client side; a DB where QoE measurements are stored and may be shared with the network provider; a Content Distribution service to deliver multimedia contents with a CDN.

In the proposed approach, the role of the service provider is to perform the QoE monitoring of the delivered service by acquiring service information from the client side based on passive measurements of service-related parameters. On the basis of these measurements, specific QoE models may be used to predict the user experience. Moreover, having its application installed in the user's device, the service provider may predict the user's perceived quality also as a function of human and context parameters which are not available to the network provider. The QoE measurements of active clients sessions are then stored in the DB, which can be also accessed by the network provider through RESTful APIs.

The network provider makes use of SDN technologies to dynamically manage the network in collaboration with QoE measurements performed by the service provider. The SDN application is responsible for QoS monitoring and management to implement the network-wide policies by communicating with the controller through the Northbound RESTful APIs. Furthermore, the SDN controller performs network-wide operations such as link-aggregation, the addition of new flows, traffic reroute and placement of the VSSs. For example, in case of network congestion events, the SDN controller may perform traffic engineering techniques on the congested part of the network. The SDN controller may also make use of Virtual Network Functions (VNFs) to place the VSSs on the network edge near to locations where cache miss handling/flash crowd occurrences are reported by the service provider. The use of Network Function Virtualization (NFV) allows for dynamic resource allocation in terms of content retrieval by enhancing the total capacity of surrogate servers to serve more clients in case of a flash crowd. Moreover, the content retrieval time is also reduced because the client is served by the VSSs instead of the peered CDNs outside the network. Hence, cache miss handling can be easily avoided.

B. Implementation details

Timber is based on several open source software packages which are listed in Table I. The implementation is broadly cate-

TABLE I
SOFTWARE PACKAGES USED IN THE CURRENT IMPLEMENTATION OF TIMBER.

Software Package	Purpose
Mininet	Creation of virtual networks
Ryu Controller	SDN controller for network management
Apache HTTP Server	Server side implementation of content hosting server
MongoDB 3.4	Implementation of cloud database for exchange of information between network and service providers
Video.js	Server and client side implementation of the video streaming application

gorized into three categories: 1) Server side implementation; 2) Client-side implementation and; 3) Virtual network and SDN controller. The rest of the section provides the implementation details of the aforementioned categories.

1) Server side implementation: The server-side video streaming service is implemented by the combination of the Apache HTTP server[3] and Video.js[4], which implements the HTML5 video player. The videos encoded in different resolutions and the web pages to play the hosted video contents are hosted on the Apache server. MySQL server and PHP libraries for apache2 server are also installed as dependencies for the web/content hosting service.

2) Client-side implementation: The client-side implementation includes the client-side video player and the monitoring probe. The client-side video player is implemented by the combination of the Video.js HTML5 player, Javascript library, Jquery, and Ruby script. The major functionality of the back-end side is the implementation of the user end probe which is inspired by the works in [15], [16]. The back-end implementation is mainly responsible for: 1) Loading the video from the server to the front-end player; 2) Front-end player state observation; 3) Computation of QoE application level Key Quality Indicators (KQI) through the probe and; 4) Storing of the results of the probe in the MongoDB[5] database (shared between network and service providers). The computation and storage of video player parameters in the MongoDB database are performed by the Ruby script. Video player parameters include *initial buffering time*, *stalling duration* and *number of stalling events*. These parameters have been considered due to higher influence on the quality perceived by the user as investigated by the studies [17]. The initial buffering time is computed as the time difference between the video loading time and the starting time of the video playback. The number of stalling events is computed throughout the video playback by counting the occurrence of the buffering events while the stalling duration is calculated as the time difference between the occurrence of a buffering event and the video playback event after the buffering. The computed video parameters are sent to the MongoDB where they are stored in the database.

[3]https://httpd.apache.org
[4]http://videojs.com
[5]https://www.mongodb.com/

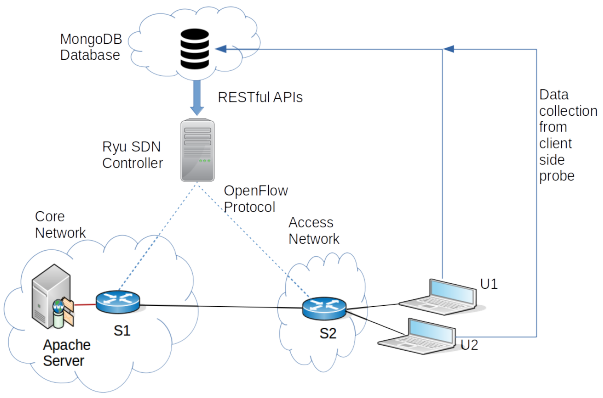

Fig. 2. *Timber* implementation details and Mininet topology used for the experiments.

3) Virtual network and SDN Controller: Timber is built on the top of Mininet emulator, which is used to create the virtual network environments. The virtual host of the Mininet acts as a client by running the client-side application in the Google Chrome browser. The Mininet Open vSwitches (OvS) are connected to the Ryu controller through the Southbound interface with the OpenFlow 1.3 protocol in the bridged configuration. The Ryu controller is used as the SDN controller to perform the network-wide monitoring and management tasks through the OpenFlow 1.3 protocol, which provides the following functionalities: 1) Configuration of OvSs through *ovs-vsctl* while keeping the global view of the network and; 2) Monitoring and administration of flow statistics and flow tables through *ovs-ofctl* respectively. The SDN controller application runs on the top of the Ryu controller which communicates through Northbound RESTful APIs. In the current implementation, the controlling operations are divided into two categories based on traffic engineering concepts: 1) Dynamic resource allocation for flows by setting the queues and; 2) Flow prioritization by setting DSCP prioritization classes and remarking of the flows (DiffServ). The SDN application is connected to the cloud database (implemented by MongoDB 3.4) through the RESTful APIs which enable the SDN application to acquire the QoE oriented information (stored by the client side probe) from the cloud database. The SDN application retrieves the information from the cloud database through the secured HTTP GET response after the authentication. The retrieved information follows the JSON format which is later sorted by the SDN controller application. The SDN controller application performs the network level operations on the basis of retrieved information. Fig. 2 shows the implementation details of *Timber* as well as the Mininet topology used for the experiments conducted in Section V.

V. EXPERIMENTS SETUP AND SCENARIOS

The objective of the experiments is to highlight some of the major functionalities of *Timber* for the QoE management of multimedia services in the SDN paradigm. Note that *Timber*

platform is not limited to only these experiments and the focus of experiments is better QoE provisioning to video streaming.

The experiments are conducted on the *Timber* platform built in the Linux Ubuntu server 14.04, 64 bits environment installed on the Virtualbox with 3 GB RAM, 8GB Hard drive and core 2 Duo @2.4 GHz CPU resources. The Big Buck Bunny 60-second long video sequence encoded with H.264 codec in 1280×720 resolution with 30 fps is used for all the experiments. The web page for video streaming contains only simple HTML5 player without any other content. The web page and video content are hosted on the Apache server. The Apache server is run as a service in one of the Mininet hosts. All the experiments follow the same topology illustrated in Fig. 2. For the experiment purposes, we fix the maximum link capacity of $S1 - S2$ to 4 Mbps, as the 720p HD video streaming requires $1.5 - 4$ Mbps throughput[6]. Hence, the $S1 - S2$ link can provide a good video streaming service only if one HD 720p video streaming session traffic is going through it. In order to create a network bottleneck at $S1 - S2$ link, HD video streaming sessions are started in the $U1$ terminal while TCP protocol based web traffic of 4 Mbps is generated through the $U2$ terminal in all experiments settings. The video streaming session is started in the $U1$ terminal by visiting the link of a video streaming web page hosted by Apache server. We considered four different scenarios to highlight major functionalities of *Timber* to improve quality in case of network congestion:

- *Best Effort (BE)*– In this scenario no optimization/traffic engineering concepts are applied, i.e., the network delivers the traffic on the best effort.
- *Differentiated Resource Distribution (DRD)*– This scenario provides different network level resources to video streaming and normal web traffic. The queues are set up in $S1$ and $S2$ OvS switches by the controller application to avoid QoE degradation by congestion. The queues have the throughput variance $2.5 - 3$ Mbps and $1 - 1.5$ Mbps for video streaming and web traffic respectively. The SDN controller application assigns the traffic flows generated by the video streaming and web traffic to the specific queues through flow matching. These queues provide the minimum guaranteed throughput of 2.5 Mbps and 1 Mbps to video streaming and web traffic respectively.
- *Fair Resource Distribution (FRD)*– This scenario is similar to DRD scenario in terms of traffic engineering concepts; however the throughput is divided among the video streaming and web traffic by setting two different queues of the same throughput, i.e., throughput of each queue is 2 Mbps.
- *Diffserv Prioritization (DP)*– In this scenario, the MPLS network is created between $S1$ and $S2$. The video streaming flow is prioritized as DSCP class $AF11$ at $S1$ while marking is done at $S2$ by setting the flow matching criteria through controller application. The web traffic is served as best effort in this scenario.

[6]https://support.google.com/youtube/answer/2853702?hl=en

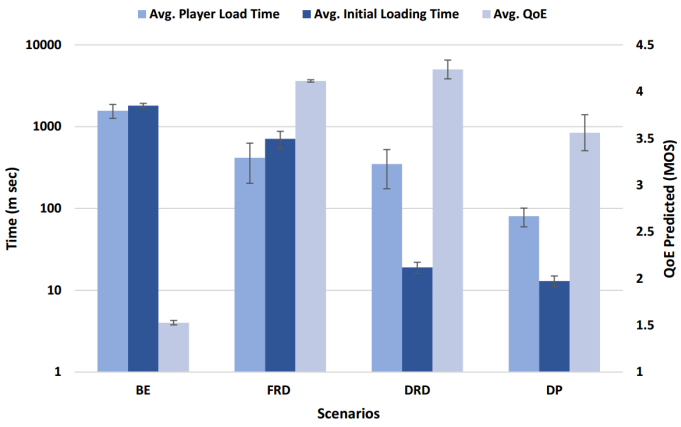

Fig. 3. Player loading, initial loading time and predicted QoE for each scenario.

In all scenarios, the video sequence is played 5 times (5 runs). The Fig. 3 represents the average page loading time and initial loading time of the video in milliseconds (ms) on a logarithmic scale in each scenario on the primary vertical axis. The *BE* scenario shows the highest average page loading time 1568 ms while *DP* has the minimum average page loading time 80.33 ms. The *BE* scenario has the highest average initial loading time 1809 ms among all scenarios while the *DP* has the minimum 13 ms. It is worth noticing that significant decrease in page loading time and initial loading time is observed after by QoE-aware traffic engineering techniques performed through *Timber* controller application as compared to BE case.

Fig. 3 shows the average predicted QoE for all scenarios on Mean Opinion Score (MOS) scale (1-5) on the secondary vertical axis. For the computation of the QoE, we considered the model (which is validated by subjective assessment) presented in [17]. Note that the selection of a particular QoE model is not the prime focus of the work and it can be replaced with the more complex models. The *DRD* scenario has the highest average predicted QoE 4.23 in all scenarios while *BE* has the lowest score 1.52. Hence, significant improvements in the predicted QoE is observed even in the case of network congestion by QoE management through *Timber*. Fig. 4 represents the throughput utilization behavior of video streaming and web traffic in one of the 5 runs observed by traffic engineering application of Timber platform in each scenario. Note that Fig. 4 does not represent the average throughput utilization of the scenarios in 5 runs. This is because to avoid the curve smoothness due to aggregation. The throughput fluctuations in *BE* scenario represent that client player continuously fetched the video segments from the server due to limited throughput in order to fill the buffer while in *DRD* scenario most of the video segments are downloaded till 38 s because video traffic is provided dedicated queue. In case of *FRD*, the video traffic is shaped to 2 Mbps which is not enough for $720p$ video so continuous download of video can be observed while in case of *DP*, the effect of priority on video and web traffic can be observed. The Table II provides the comparison summary for

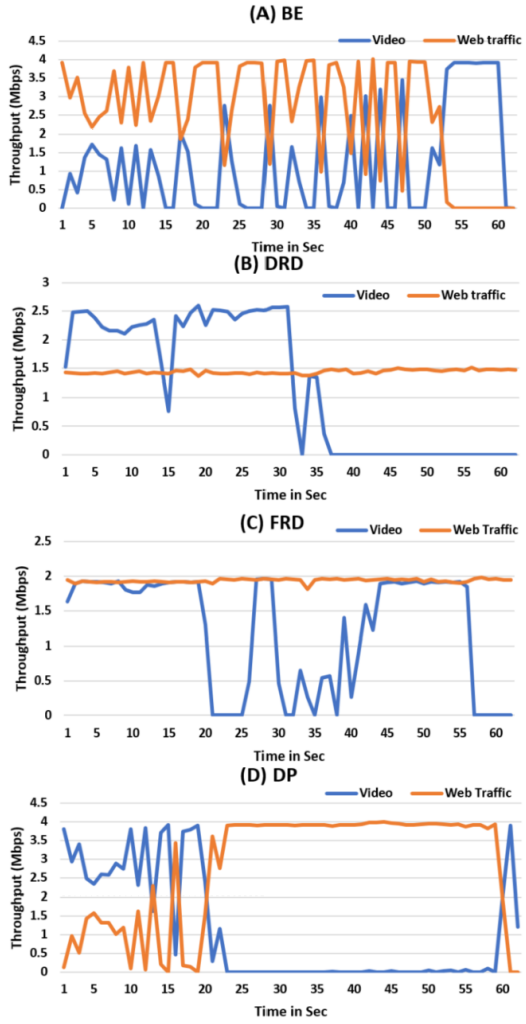

Fig. 4. Throughput utilization behavior for each scenario in one run.

TABLE II
COMPARISON SUMMARY FOR ALL SCENARIOS IN TERMS OF KQIS AND PREDICTED AVERAGE QOE.

Scenario	Avg. Player Loading Time (ms)	Avg. Initial Loading Time (ms)	Avg. Stalling Duration (ms)	Max. No. of Stalling Events	Min. No. of Stalling Events	Avg. QoE (MOS)
BE	**1568**	**1809**	**4758.08**	**7**	**4**	**1.528**
FRD	416.33	711.33	677.33	1	1	4.114
DRD	350	19	**372**	1	1	**4.237**
DP	**80.33**	**13**	2264.66	1	1	3.560

all the scenarios in terms of the measured KQIs and average predicted QoE.

VI. CONCLUSION

In this work, we provide an SDN based emulation platform called *Timber* for QoE management experimental research purposes. *Timber* is developed on the top of Mininet SDN emulator using Ryu controller. *Timber* has the capability to not only monitor the KQIs of video streaming session through client side (user end probe) but also to perform QoE-aware traffic engineering tasks through SDN controller application. This work provides the architectural and implementation details of *Timber*, which can be used by the research community to evaluate future QoE management and monitoring strategies. Moreover, we provide experimental results to highlight the major functionalities of *Timber* by 4 different scenarios which include traffic shaping through DiffServ and dynamic resource allocation by queuing strategies. The future work will be focused on adding more multimedia services in the platform.

REFERENCES

[1] Cisco, "Cisco Visual Networking Index: Forecast and Methodology, 2016-2021 - White paper," 2017.
[2] J. Seppänen, M. Varela, and A. Sgora, "An autonomous QoE-driven network management framework," *Journal of Visual Communication and Image Representation*, vol. 25, no. 3, pp. 565–577, 2014.
[3] S. Baraković and L. Skorin-Kapov, "Survey and Challenges of QoE Management Issues in Wireless Networks," *Journal of Computer Networks and Communications*, vol. 2013, 2013.
[4] E. Liotou, D. Tsolkas, N. Passas, and L. Merakos, "Quality of Experience Management in Mobile Cellular Networks: Key Issues and Design Challenges," *IEEE Communications Magazine*, vol. 53, no. 7, pp. 145–153, 2015.
[5] E. Liotou, G. Tseliou, K. Samdanis, D. Tsolkas, F. Adelantado, and C. Verikoukis, "An SDN QoE-service for dynamically enhancing the performance of OTT applications," in *Seventh Int. Workshop on Quality of Multimedia Experience (QoMEX), 2015*. IEEE, 2015.
[6] A. Bentaleb, A. C. Begen, R. Zimmermann, and S. Harous, "SDNHAS: An SDN-Enabled Architecture to Optimize QoE in HTTP Adaptive Streaming," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2136–2151, 2017.
[7] A. Farshad, P. Georgopoulos, M. Broadbent, M. Mu, and N. Race, "Leveraging SDN to Provide an In-network QoE Measurement Framework," in *IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS), 2015*. IEEE, 2015, pp. 239–244.
[8] S. Khairi, M. Bellafkih, and B. Raouyane, "QoS management SDN-based for LTE/EPC with QoE evaluation: IMS use case," in *Fourth Int. Conf. on Software Defined Systems (SDS), 2017*. IEEE, 2017, pp. 125–130.
[9] S. Ramakrishnan, X. Zhu, F. Chan, and K. Kambhatla, "SDN Based QoE Optimization for HTTP-Based Adaptive Video Streaming," in *IEEE Int. Symposium on Multimedia (ISM), 2015*. IEEE, 2015, pp. 120–123.
[10] H. Nam, K.-H. Kim, J. Y. Kim, and H. Schulzrinne, "Towards QoE-aware video streaming using SDN," in *IEEE Global Communications Conference (GLOBECOM), 2014*. IEEE, 2014, pp. 1317–1322.
[11] E. Grigoriou, A. A. Barakabitze, L. Atzori, L. Sun, and V. Pilloni, "An SDN-approach for QoE management of multimedia services using resource allocation," in *IEEE Int. Conf. on Communications (ICC), 2017*. IEEE, 2017.
[12] M. García-Pineda, S. Felici-Castell, and J. Segura-García, "Adaptive SDN-based architecture using QoE metrics in live video streaming on Cloud Mobile Media," in *Fourth Int. Conf. on Software Defined Systems (SDS), 2017*. IEEE, 2017, pp. 100–105.
[13] L. Fawcett, M. Mu, M. Broadbent, N. Hart, and N. Race, "SDQ: Enabling rapid QoE experimentation using Software Defined Networking," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2017, pp. 656–659.
[14] A. Ahmad, A. Floris, and L. Atzori, "QoE-centric service delivery: A collaborative approach among OTTs and ISPs," *Computer Networks*, vol. 110, pp. 168–179, 2016.
[15] A. Ahmad, L. Atzori, and M. G. Martini, "Qualia: A multilayer solution for QoE passive monitoring at the user terminal," in *IEEE Int. Conf. on Communications (ICC), 2017*. IEEE, 2017, pp. 1–6.
[16] F. Wamser, M. Seufert, P. Casas, R. Irmer, P. Tran-Gia, and R. Schatz, "Yomoapp: A tool for analyzing qoe of youtube http adaptive streaming in mobile networks," in *2015 European Conference on Networks and Communications (EuCNC)*, 2015, pp. 239–243.
[17] T. Hoßfeld, C. Moldovan, and C. Schwartz, "To each according to his needs: Dimensioning video buffer for specific user profiles and behavior," in *IFIP/IEEE Int. Symposium on Integrated Network Management (IM), 2015*. IEEE, 2015, pp. 1249–1254.