Microprocessors and Microsystems 77 (2020) 103178

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

The CASPER user-centric approach for advanced service provisioning in mobile networks

Eirini Liotou[a,*], Dimitris Tsolkas[a], Giorgos Kalpaktsoglou[b], Stefano Tennina[c], Luigi Pomante[d], Nikos Passas[a]

[a] Department of Informatics & Telecommunications, National and Kapodistrian University of Athens, Greece
[b] ADAPTERA, Athens, Greece
[c] WEST Aquila S.r.l., L'Aquila, Italy
[d] Università degli Studi dell'Aquila, L'Aquila, Italy

ARTICLE INFO

Article history:
Received 20 January 2020
Revised 19 April 2020
Accepted 14 June 2020
Available online 24 June 2020

Keywords:
Quality of experience
SDN
Orchestration

ABSTRACT

This paper presents an overview of the project CASPER,[1] a 4-year Marie Curie Research and Innovation Staff Exchange (RISE) project running between 2016 and 2020, describing its objectives, approach, architecture, tools and key achievements. CASPER combines academic and industrial forces towards leveraging the expected benefits of Quality of Experience (QoE) exploitation in future networks. In order to achieve that, a QoE orchestrator has been proposed which implements the basic functionalities of QoE monitoring, estimation and management. With means of simulation and testbed emulation, CASPER has managed to develop a proprietary SDN Controller, which implements QoE-based traffic rerouting for the challenging scenario of HTTP adaptive video streaming, leading to more stable and higher QoE scores compared to a state-of-the-art SDN Controller implementation.

1. Introduction

CASPER ("*A user-Centric middleware Architecture for advanced Service Provisioning in futurE netwoRks*") is a Research and Innovation Staff Exchange (RISE) project under Marie Skłodowska-Curie Actions (2016–2020), which has just recently been completed. CASPER has aimed at bringing together experts from industry and academia, from cross-sectorial research areas having complementary background, with the goal of analysing and exploiting Quality of Experience (QoE) towards developing efficient schemes for managing multimedia digital services. In particular, efforts have been devoted to the development of innovative QoE estimation mechanisms and QoE monitoring protocols for optimizing service delivery and providing key input to Customer Experience Management (CEM) policies. CASPER has created a fully-integrated and multi-disciplinary program on the development of QoE network optimisation modules towards the creation of a programmable, reconfigurable and adaptable architecture.

QoE is defined by the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) Rec. P.10 (Amendment 2, 2008), as "*the overall acceptability of an application or service, as perceived subjectively by the end-user*". This concept has emerged due to the inefficiency of the traditional Quality of Service (QoS) term to fully imprint the actual experience of the end-users, as well as its inability to fully capture application-specific and user-specific characteristics. The purpose of the QoE concept is, therefore, to provide means to track the degree of user satisfaction of a network's performance in a qualitative or quantitative manner and to try to improve it in order to meet or exceed the users' expectations. QoS provisioning policies can therefore be improved [1]. In order to guarantee QoE in a network, three procedures need to be implemented [2]:

1.1. QoE modelling

It refers to an estimation method that attempts to quantify the level of user experience and imprint it into a Mean Opinion Score (MOS) or a collection of values for Key Performance Indicators (KPIs). There are various different approaches for estimating QoE. Their primary classification is into objective and subjective models. Subjective models are based on real life experiments with human participants who evaluate their experience of an application or service. Objective models, on the other hand, try to automat-

* Corresponding author.
E-mail addresses: eliotou@di.uoa.gr (E. Liotou), dtsolkas@di.uoa.gr (D. Tsolkas), gkalpaktsoglou@adaptera.gr (G. Kalpaktsoglou), stefano.tennina@westaquila.com (S. Tennina), luigi.pomante@univaq.it (L. Pomante), passas@di.uoa.gr (N. Passas).

[1] http://gain.di.uoa.gr/casper/.

https://doi.org/10.1016/j.micpro.2020.103178
0141-9331/

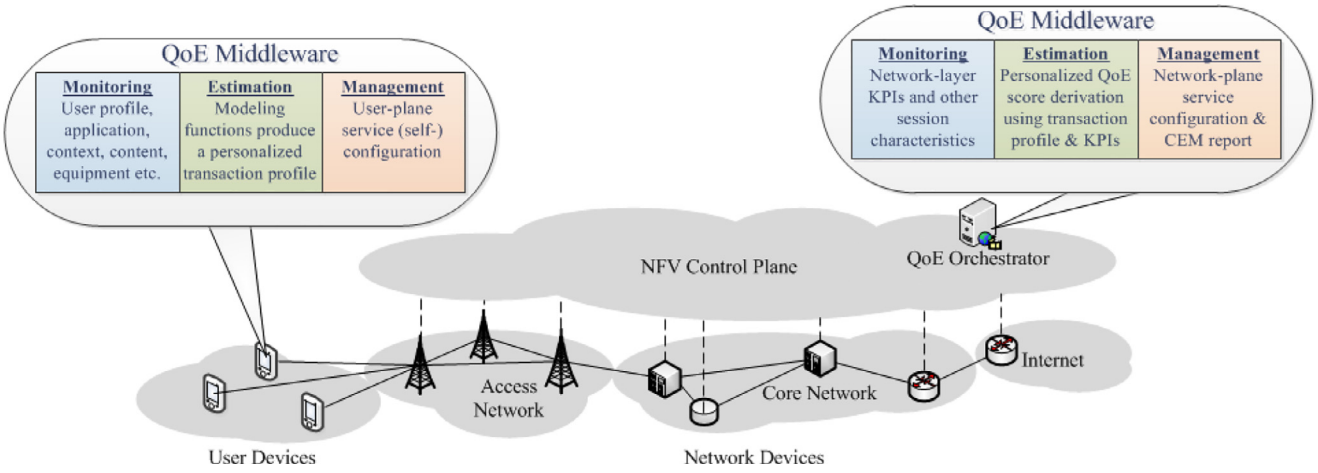


Fig. 1. A mapping of the key functional modules for QoE provisioning on the mobile network infrastructure. QoE monitoring, estimation and management functions can be implemented at user level and core network level.

ically measure or predict the quality perceived by the end-users, without their intervention and are, therefore, required for real-time service evaluation.

1.2. QoE monitoring

It refers to the key functionality of gathering network, application and user-related factors for a reliable estimation of the end-user's satisfaction level. These factors refer to service evaluation feedback from the end-user and from parameters available in diverse network entities (user device, access network, service provider) in all protocol layers. For the gathering of these factors, active (intrusive) and passive (non-intrusive) monitoring schemes can be used, while only a specific set of factors is monitored depending on the specifications of the used estimation model.

1.3. QoE management

It refers to methods, protocols and techniques that aim to enhance the customer experience, while in parallel making efficient use of network resources (e.g. [3]). The available service management procedures for QoE management include but are not limited to routing, scheduling, rate adaptation, and network selection. Such procedures, especially for multimedia services, are currently bounded by QoS limitations that rely on network constraints (e.g., available spectrum). The QoE factor is, therefore, the key for moving beyond QoS thresholds and acquiring at the same time the levels of end-users' satisfaction for a service.

A QoS-aware middleware for collaborative multimedia streaming and caching has been presented in [4] (for computing environments, though). CASPER proposes a QoE-based middleware paradigm in order to integrate the three aforementioned procedures and to guarantee the user experience in mobile networks. It consists of three interlinked and optimised modules applied both in user and network plane, dealing with the QoE monitoring, estimation, and management functionalities. The QoE provisioning logic exploits the flexibility and re-configurability provided by the Network Functions Virtualisation (NFV) and Software-Defined Networking (SDN) technology in a bidirectional manner to acquire QoE influence factors from anchor points in the network and to apply QoE-driven service management policies. A twofold target has been set for the three interlinked functionalities: i) to personalise service delivery through acquiring and combining knowledge available in distributed places, such as various network elements in the access/core/backhaul network, as well as in databases, human subjects and surrounding environments, and ii) to motivate providers to use enhanced QoE-driven service management policies as well as advanced CEM techniques. A mapping of the key functional modules for QoE provisioning on the mobile network infrastructure, as proposed by CASPER, is depicted in Fig. 1.

The rest of the paper is organized as follows: Section II presents the project objectives, while Section III describes its concept and targeted scenarios. Then, Section IV provides the CASPER implementation methodology. Section V presents the main achievements of the project, namely: a) a user-centric orchestration architecture, b) a QoE modelling tool, c) a QoE monitoring methodology for managing multimedia flows, as well as d) a system level simulator and e) a testbed for experimentation of user-centric management scenarios. Finally, Section VI briefly describes the project's consortium, while Section VII concludes the paper.

2. Research and technological objectives

CASPER has three main research objectives:

1. To study, design and optimise QoE estimation mechanisms for multimedia services. This main scientific objective includes mechanisms for subjective and objective evaluation of a provided service. The target is to enhance and optimise QoE estimation models, to thoroughly study and evaluate key indicators that affect the QoE, and to design efficient and dynamic schemes for QoS to QoE translation.
2. To study, design, and optimise QoE monitoring protocols. This objective includes the thorough study of next generation network architecture and protocol stack towards defining the anchor points from where key QoE indicators can be acquired and the communication protocols for gathering those at network and user plane. The achievement of this objective is expected to provide an essential module to operators for exploiting the knowledge of their customers' experience.
3. To analyse, design and optimise advanced QoE-driven service management policies. An innovative and stochastic representation of the future networks, including the recent advances in SDN and NFV for orchestrating service provisioning is used for designing cross-layer QoE-driven service management policies. Additionally, by capitalizing on the QoE-awareness at operators/service providers and end-devices, application layer enhancements are provided at user plane for more personalised service delivery.

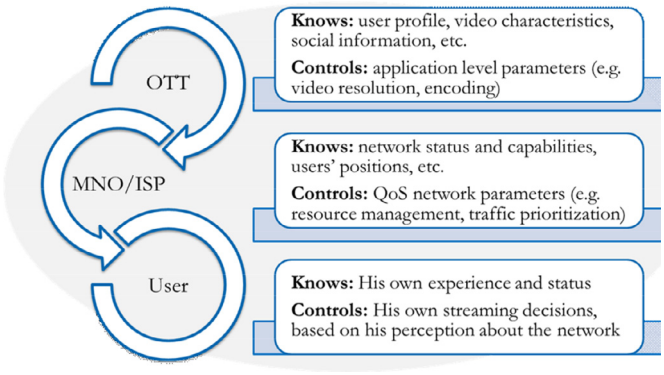


Fig. 2. The main stakeholders in CASPER. OTTs provide their services through the MNO/ISP infrastructure to the end-users.

In addition, CASPER's main technological objectives are:

1. To develop a System Level Simulator (SLS), where the QoE estimation and monitoring schemes, and the QoE-driven service management policies will be evaluated under realistic scenarios.
2. To develop a hardware proof-of-concept testbed, where the most promising schemes identified via the SLS will be integrated and tested in a real-working environment.
3. To integrate QoE exploitation functions into real devices, where a meaningful subset of the middleware architecture will be implemented in commercial devices and network entities for real-life experimentation.

3. Concept and approach

The main objective of CASPER is the design and implementation of a middleware architecture for QoE-driven service provisioning. The first goal towards this objective is the identification of the distinct stakeholders involved in this architecture who can, therefore, benefit from it. Three main stakeholders can be distinguished in the investigated service provisioning paradigm, namely a) the Over-The-Top Service Providers (OTT), b) the Mobile Network Operators (MNO) / Internet Service Providers (ISP) and, finally c) the end-users (Fig. 2). The end-users are the final consumers of a service, which is provided either by the MNO directly or by an OTT service provider through the MNO's infrastructure. CASPER defines and studies three main scenarios which involve the aforementioned stakeholders, while different service types are considered per scenario.

The first CASPER scenario (Scenario A) refers to real-time communication between two parties that is end-to-end supported by the infrastructure of the MNO. The involved parties (end-users) have connectivity to the same or different operators (intra-operator and inter-operator cases, respectively) and they may be located in the same or different cells (Fig. 3– Scenario A). Table 1 summarises the major principles of scenario A.

Scenario B concerns cases of video content delivery, namely video streaming from an OTT service provider. The common paradigm nowadays is that OTT parties provide access to video streaming services, which the users can access only through their MNO or ISP connection. (Note that MNOs or ISPs may also provide video streaming services by themselves to their subscribers). The second scenario of CASPER is depicted in Fig. 3 as well. The main service that may be applied to this scenario is video streaming (Non-adaptive or HTTP adaptive video streaming), i.e., a non-

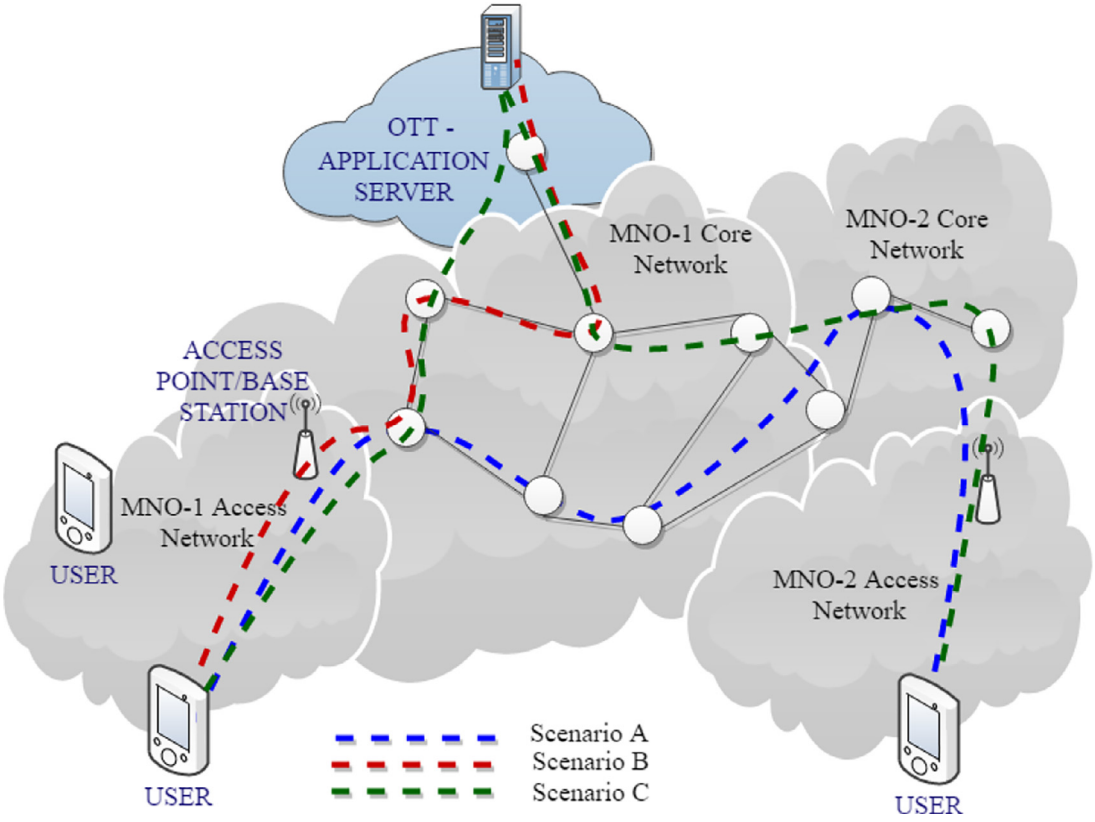


Fig. 3. Abstract representation of CASPER scenarios A, B and C (only inter-operator cases are presented). Two users communicate directly (A) or via an OTT-server (C), or a user consumes an OTT-service (B).

Table 1
Scenario A.

Scenario A	
Modules involved	SDN modules, QoE manager
Service type	Real time (VoIP or video)
Involved users	Two communicating peers
Technical challenge	Apply network-side optimisations to guarantee high level of QoE at both peers
Approach	Advanced routing of the data flows
Interface(s) / Components used to apply the policy	OpenDaylight Application Programming Interface (API)
Parameters to be monitored - service level	Packet loss ratio, Latency, Jitter, Codec, and Coding rate
Parameters to be monitored - network level	SDN data and control plane parameters (e.g., Packet Latency, Transmit/Receive Throughput Rate, Connected OpenFlow Switches)
QoE manager feeding approach	ELK module (querying the elasticsearch) (Note: "ELK" is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana)

Table 2
Scenario B.

Scenario B	
Modules involved	SDN modules, NFV modules, QoE manager
Service type	Video Streaming
Involved users	A user that consumes video on demand
Technical challenge	a) Apply end-to-end service optimizations b) Apply content delivery optimizations
Approach	a) Proper transcoding of video content b) Content instantiation using Content Delivery Network (CDN) and caching techniques
Interface(s) / Components used to apply the policy	OpenStack and OpenDaylight
Parameters to be monitored - service level	Data rate, Video bit rate, Media bit rate, Number of stalling events, Duration of stalling events, Initial delay, Video start failure, Time on highest layer, Frequency of quality switches, Amplitude of switches
Parameters to be monitored - network level	SDN data and control plane parameters (e.g., Packet Latency, Transmit/ Receive Throughput Rate, Connected OpenFlow Switches)
QoE manager feeding approach	ELK module (querying the elasticsearch), User application layer

real-time service. Caching video content is an important feature of this scenario. Table 2 summarises the major principles of Scenario B.

The third scenario (Scenario C) is the most challenging one since it includes an OTT provider, one or more MNO/ISP providers and multiple users. Applications that fall into this category are online meetings, video conferencing and in general shared collaborative platforms. The traffic in this case is accumulated at a server in the possession of the OTT party, passing however through the network infrastructure of one or more MNO/ISPs (intra-operator and inter-operator case, respectively). This scenario is summarised in Table 3. More information about the CASPER scenarios are available at [5].

Table 3
Scenario C.

Scenario C	
Modules involved	SDN modules, NFV modules, Big Data analytics, QoE manager
Service type	Online meeting (e.g., WebEx with independent voice and video streams), or many diverse service types of multiple competing users
Involved users	Multiple users at different locations
Technical challenge	Global flow optimizations
Approach	Apply proper but differentiated routing of voice and video streams (e.g., WebEx) Functions/Content instantiation based on KPI requirements (e.g., users streaming simultaneously a very popular video)
Interface(s) / Components used to apply the policy	Custom API, OpenStack, and OpenDaylight
Parameters to be monitored - service level	All parameters of scenarios A and B
Parameters to be monitored - network level	SDN data and control plane parameters (e.g., Packet Latency, Transmit/ Receive Throughput Rate, Connected OpenFlow Switches, Memory utilization)
QoE manager feeding approach	ELK module (querying the elasticsearch)

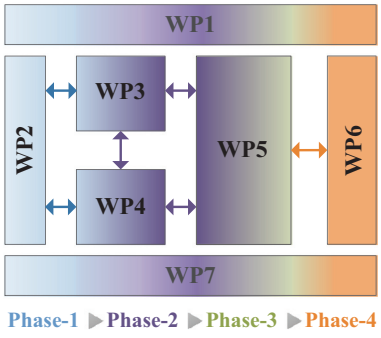


Fig. 4. CASPER workpackages and implementation phases. WP2-WP6 progress the technical work of the project.

4. Implementation and work packages

CASPER methodology encompasses four main phases. These phases, together with their encompassed Work Packages (WP) are depicted in Fig. 4 and are further described below.

4.1. Phases

4.1.1. Phase-1

Middleware architecture and QoE analysis and exploitation schemes. During this phase, academic and industrial beneficiaries have worked in synergy to design the middleware architecture for QoE service delivery. In particular, all beneficiaries were responsible to define the requirements of the middleware architecture, the key parameters for QoE estimation/monitoring and the reconfigurable parameters used in the QoE-aware service management. Attention was given on defining potential hardware limitations and interoperability/energy/privacy issues, while innovative QoE estimation mechanisms, QoE monitoring protocols, and service management algorithms were developed. The methodology for this phase consisted of two steps: i) define the middleware architecture and a set of quality indicators used for QoE estimation/monitoring;

and ii) design QoE estimation mechanisms and monitoring protocols, and QoE-driven service management policies.

4.1.2. Phase-2

Analysis and optimisation of the QoE provision schemes using simulations. This phase included the development of a System Level Simulator (SLS) based on the QoE architecture designed in Phase-1. Using this simulator, the algorithms and protocols proposed in Phase-1 have been evaluated and optimised. The methodology adopted in this phase consisted of three main steps: i) the translation of the algorithms and protocols developed in Phase-1 in fundamental modules written in the programming language of the SLS; ii) the integration of the modules into the SLS in order to realise the proposed middleware architecture; and iii) the performance assessment in the project scenarios and the comparison with benchmark solutions by taking into account the project's service performance evaluation metrics.

4.1.3. Phase-3

Implementation, proof-of-concept study, and experimental assessment. During the third phase, the modules defined in Phase-2 have been applied in a testbed and their performance has been evaluated through a measurement campaign in a realistic working environment. This phase has been considered critical before the implementation in real devices, in order to gain: i) a very clear understanding of the requirements for applying the proposed solutions to real environments, ii) an understanding of the capabilities and limitations of the new solutions, iii) the ability to assess design decisions early in the process, and iv) the ability to a-priori visualise the look-and-feel of the solutions. The main goal has been to clarify the maturity of the new system including: i) what can be supported or not by the implemented middleware architecture, ii) if the technology is strong enough to be deployed and achieve foreseen performances, iii) if these solutions can be used on existing wireless and mobile networks. The methodology for this phase consisted of the following steps: i) the testbed implementation and upgrade with the innovative algorithms and protocols; ii) the definition of test scenarios, metrics and targeted values; iii) the execution of scenarios and extraction of measurements; and iv) the evaluation of results and conclusions.

4.1.4. Phase-4

Product integration in commercial devices and realistic experiments. In the fourth and last phase, a meaningful subset of the middleware architecture with the QoE estimation and service management modules has been integrated into a testbed with real devices that were made available by the industrial beneficiaries. Real-life measurement scenarios have been defined, covering reliability, real-time applicability and performance issues under a wide range of conditions. The methodology for this phase consisted of the following steps: i) integration of a meaningful subset of the middleware architecture into real devices; ii) execution of real-life scenarios and experiments; iii) assessment of the developed system.

The project has been organised into 7 work packages (WPs), as shown in Fig. 4. WP1 and WP7 are concerned with management and dissemination respectively, running for the whole duration of the project. The rest WPs, from WP2 to WP6, are the technical WPs of the project. More specifically, WP2 is concerned with the definition of the overall middleware architecture and the requirements for exploiting QoE in future networks; WP3 focuses on the analysis of key quality indicators, and the design of QoE estimation mechanisms and QoE monitoring protocols; WP4 performs the design and optimisation of the QoE-driven service management policies; WP5 deals with the implementation and validation of algorithms/protocols in the system level simulator and the testbed; and finally WP6 handles the final integration in commercial devices and proof-of-concept in a real-working environment. More details per WP are presented next.

4.2. Work packages

4.2.1. WP2 (Middleware architecture definition and requirements description) objectives

To provide an end-to-end description of the middleware architecture and ensure its smooth and realistic applicability in commercial products by taking into consideration real-life systems' requirements, such as performance, complexity, hardware, and energy constraints. The tasks of WP2 include the description of various communication scenarios and the definition of target performance metrics and key parameters. Moreover, WP2 includes the design of the complete middleware architecture that thoroughly describes the new modules, methods and functionalities required to be implemented inside real devices and providers' infrastructure.

4.2.2. WP3 (QoE estimation mechanisms and QoE monitoring protocols) objectives

To set the ground for the implementation of the enhanced QoE support framework, by providing the theoretical findings regarding QoE estimation and monitoring options. The tasks in this WP include the identification of the QoE influence factors and the investigation of possible QoE modelling options. Moreover, research focuses on the monitoring options in smart devices using for instance software agents or present capabilities of the devices as well as deal with the available monitoring options at the service provider's side (routing devices or provider's management nodes). Finally, this WP defines and simulates a protocol for the dissemination of information from the various monitoring and estimation locations to the service provider's service management nodes.

4.2.3. WP4 (Design and optimisation of QoE-driven service management policies) objectives

To define the basic QoE management framework and to design optimisation and control algorithms for enhanced service provisioning. This WP designs a management framework targeted for service providers and operators in order to exploit the acquired QoE-related information for their own as well as for their customers' benefit. The framework is designed on top of NFV/SDN technologies under the control of a centric QoE orchestrator. Moreover, WP4 defines the CEM scenarios to be applied to the QoE management support, and investigates advanced policies based on acquired QoE measurements.

4.2.4. WP5 (Development and validation of proposed modules) objectives

To integrate simulation modules of WP3 and WP4 and conduct joint simulations, followed by the development of building blocks for the respective algorithms and protocols. In WP5, the separate simulation modules, constructed during WP3 and WP4 and offering measurements on the performance of individual solutions, are integrated into one overall simulation model. Next, WP5 translates these algorithms and protocols into modules /building blocks which are integrated into the nodes of the testbed in order to conduct detailed experimental activities for a preliminary proof-of-concept of the proposed middleware architecture.

4.2.5. WP6 (System integration and proof-of-concept) objectives

To integrate the proposed architecture into real devices and to test the performance in real working environments. This will provide a clear assessment of advantages/disadvantages of the proposed solutions. In this WP, a meaningful set of the modules/building blocks identified, developed, and tested in WP5 are integrated into real

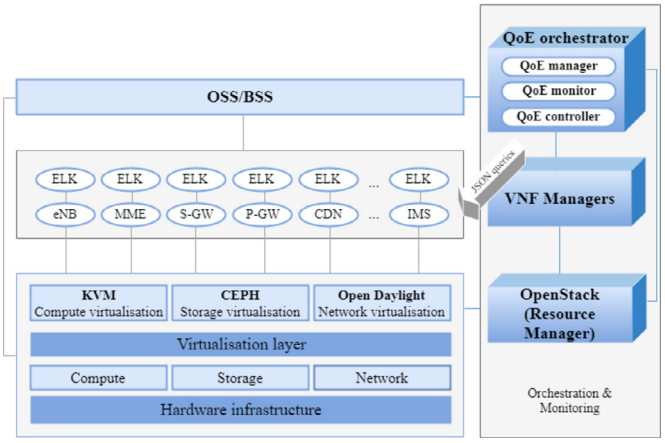


Fig. 5. QoE orchestration in CASPER [8]. The QoE NFVO controls the flow of the QoE service management logic.

devices. Moreover, WP6 performs a comprehensive experimental assessment of the developed devices improved with the new middleware and evaluates the performance improvement in realistic environments where the devices are expected to operate.

5. Main achievements

5.1. CASPER orchestration architecture

CASPER orchestration logic is integrated into the European Telecommunications Standards Institute (ETSI) NFV-MANO architecture (Management and Orchestration Architecture) [6] as shown in Fig. 5. The proposed QoE orchestrator acts as the NFV Orchestrator (NFVO) and is logically divided into three components: a) the QoE manager which implements the service and network management logic, b) the QoE monitor which includes the basic functionalities of QoE modelling and monitoring, and c) the QoE controller, which, as its name implies, controls the collection of monitored information from the various network entities into the QoE orchestrator. The flow of QoE service management in CASPER is as follows:

1. The QoE monitor requests KPIs from the QoE controller.
2. Each ELK (Elasticsearch, Logstash, Kibana) element collects appropriate information from the Virtual Network Function (VNF) that it is in charge of. In the context of Long-Term Evolution (LTE) networks [7], these VNFs may represent either access or core network entities, such as: the evolved Node B (eNB), Mobility Management Entity (MME), Serving Gateway (S-GW), Packet Data Network Gateway (P-GW), or even a Content Delivery Network (CDN) and IP Multimedia Subsystem (IMS), among others.
3. The QoE controller creates respective queries to these ELKs.
4. These requests are handled by the appropriate ELK.
5. Monitored parameters are reported back to the QoE controller.
6. The QoE monitor translates collected information to QoE "language" (i.e. MOS or KPI values vs. appropriate thresholds).
7 Network and/or application management mechanisms are triggered by the QoE manager, using for instance OpenStack (which acts as the Virtualized Infrastructure Manager) or OpenDaylight (which acts as the SDN controller and Virtual Networking Manager).

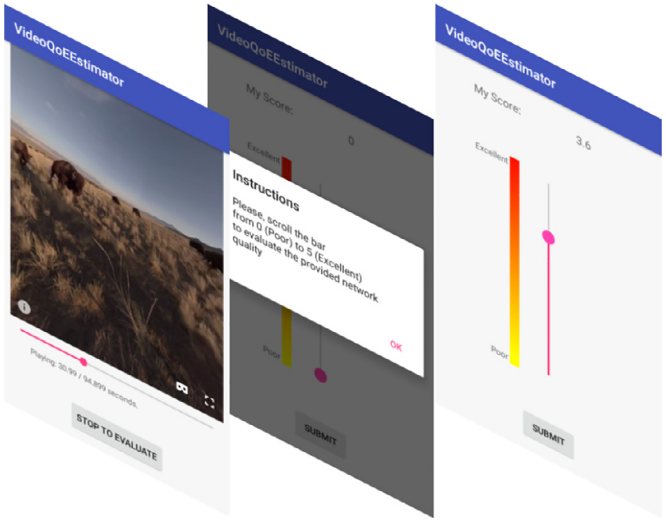


Fig. 6. Storyboard of the CASPER application. After the user watches a video, he/she is instructed to evaluate its quality using a MOS sliding scale.

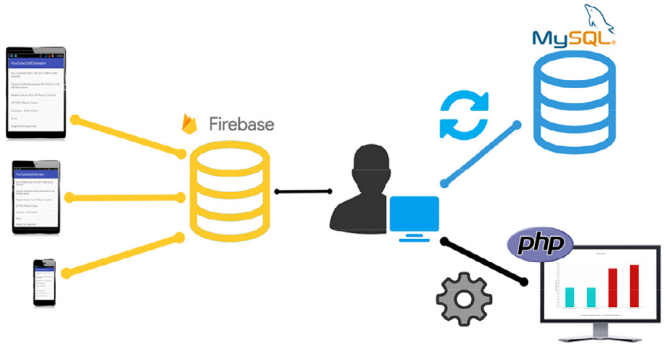


Fig. 7. The ingredients of the CASPER QoE modelling tool. QoE scores are stored in the MySQL database.

5.2. CASPER QoE modelling tool

To facilitate the analysis on QoE estimation, in the context of CASPER we developed a tool that uses, as a key interface with the end-users, a mobile application. A storyline for the usage of this tool is as follows:

- The mobile application is used to monitor and store to a database, video delivery characteristics (such as initial delay, stalling events etc.) and network characteristics (such as the connection type).
- The video consumer is requested by the application to evaluate the presented video by giving a MOS. The value provided by the user is also stored to the database.
- The collected data (user's feedback and monitored values) are then analyzed using machine learning techniques to set a mathematic formula that translates monitored parameters to a QoE score.

The procedure to be used for the data collection is based on specifications for subjective QoE tests and current work on crowdsourcing practices. A storyboard of the application is presented in Fig. 6. The CASPER QoE modelling tool is composed of the following components (see Fig. 7): a) an Android application, b) a google firebase synchronised with the application, c) a web service to download the collected data to an editable local MySQL base, and d) a php/matlab tool to elaborate on the collected data and apply data analysis.

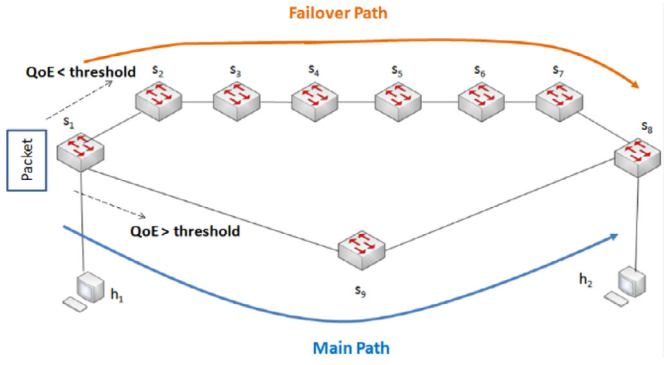


Fig. 8. Traffic forwarding with QoE monitoring [9]. When QoE is below a predefined threshold, traffic is redirected from the main shortest path (s1->s9->s8) to the failover longest path (s1->s2->s3->s4->s5->s6->s7->s8).

5.3. Network-side QoE monitoring and signalling

Moreover, CASPER has proposed a QoE-based SDN controller on top of a Mininet network topology, presented in Fig. 8. The main functionality of the controller is to guarantee a minimum QoE level for the end-users, at situations of high network congestion that lead to intense packet losses. This guarantee is achieved by triggering traffic redirection from the main (shortest) to a failover path. In order to achieve that, the controller periodically collects statistics from the network switches in order to estimate MOS as follows:

- For VoIP applications, the delay and packet loss are necessary, in order to be able to use the ITU-T G.107 E-model (a parametric QoE estimation model for VoIP traffic).
- For video applications, the bit rate, frame rate and packet loss are necessary, in order to use the ITU-T G.1070 E-model ((a parametric QoE estimation model for video traffic).

Then, if MOS is lower than a predefined threshold, the controller will provoke the traffic redirection to the failover path through the insertion of appropriate rules to the OpenFlow switches. Indicative results in terms of MOS that present the above scenario are presented in Fig. 9. At this figure, we can observe that with the proposed SDN controller, the experienced MOS can recover and reach very high values after a situation of great congestion that led to a momentary MOS reduction.

5.4. System level simulator - The OMNeT++ framework

CASPER simulator builds upon INET/SimuLTE over the OMNeT++ Discrete Event Network Simulator [10]. OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily developed for building network simulators.

All CASPER scenarios have been simulated in this environment, while here we present indicative results of Scenario B, i.e. a video streaming service that runs at the application layers of a variable number of users and the OTT server. Each user first sends a request to the server, which then starts transmitting the chunks of the video. The download can be executed simultaneously by one or more users or in different periods. The reference topology for Scenario B is shown in Fig. 10.

The QoE policy for video streaming aims at reacting to a decrease in the MOS value, due to the incipient network congestion, which increases the end-to-end delay and the Packet Loss Rate (PLR). In this case, a path's change from the main path to the backup path is triggered if the value of MOS persistently falls below a given threshold. In parallel, we apply a second policy, aiming at increasing the throughput and then its corresponding MOS value. We can change the value of the packets' size or the interval at which the OTT server sends the packets, i.e., we simulate the effects of adaptive streaming (namely, the packet's size or the send interval are adjusted at run-time continuously trying to cope with changing network conditions). In Fig. 11, we present simulation results for this scenario, where the vertical red lines are the markers indicating when a streaming flow has changed the path over the network, while black vertical lines indicate an increase in packet size.

5.5. CASPER testbed

In this section, we present the necessary components and tools that are required in order to prepare the CASPER testbed from scratch (Fig. 12 and Table 4). Specifically, to create the CASPER testbed environment, we have setup the following implementation

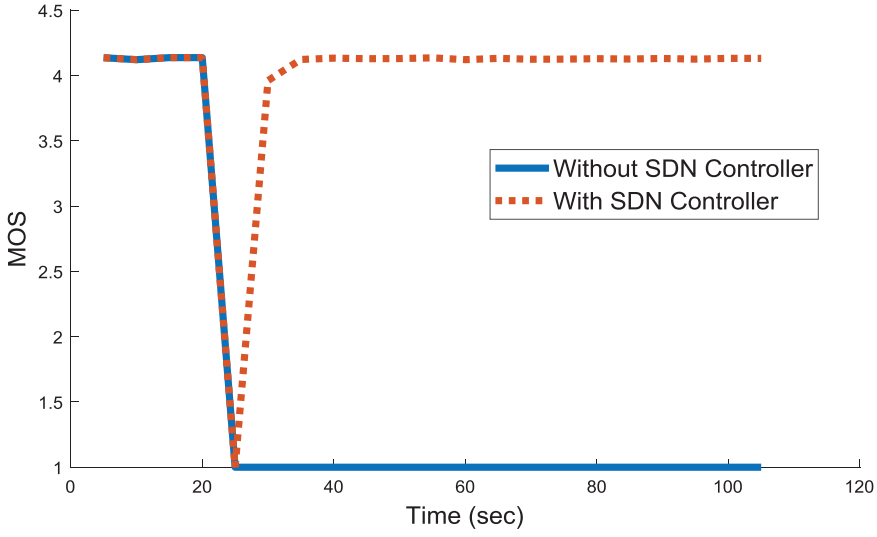


Fig. 9. MOS comparison between cases with (red plot) and without (blue plot) the SDN controller in VoIP traffic generation, when a link failure occurs (adopted from [9]). With the SDN Controller, MOS values bounce back to high values. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

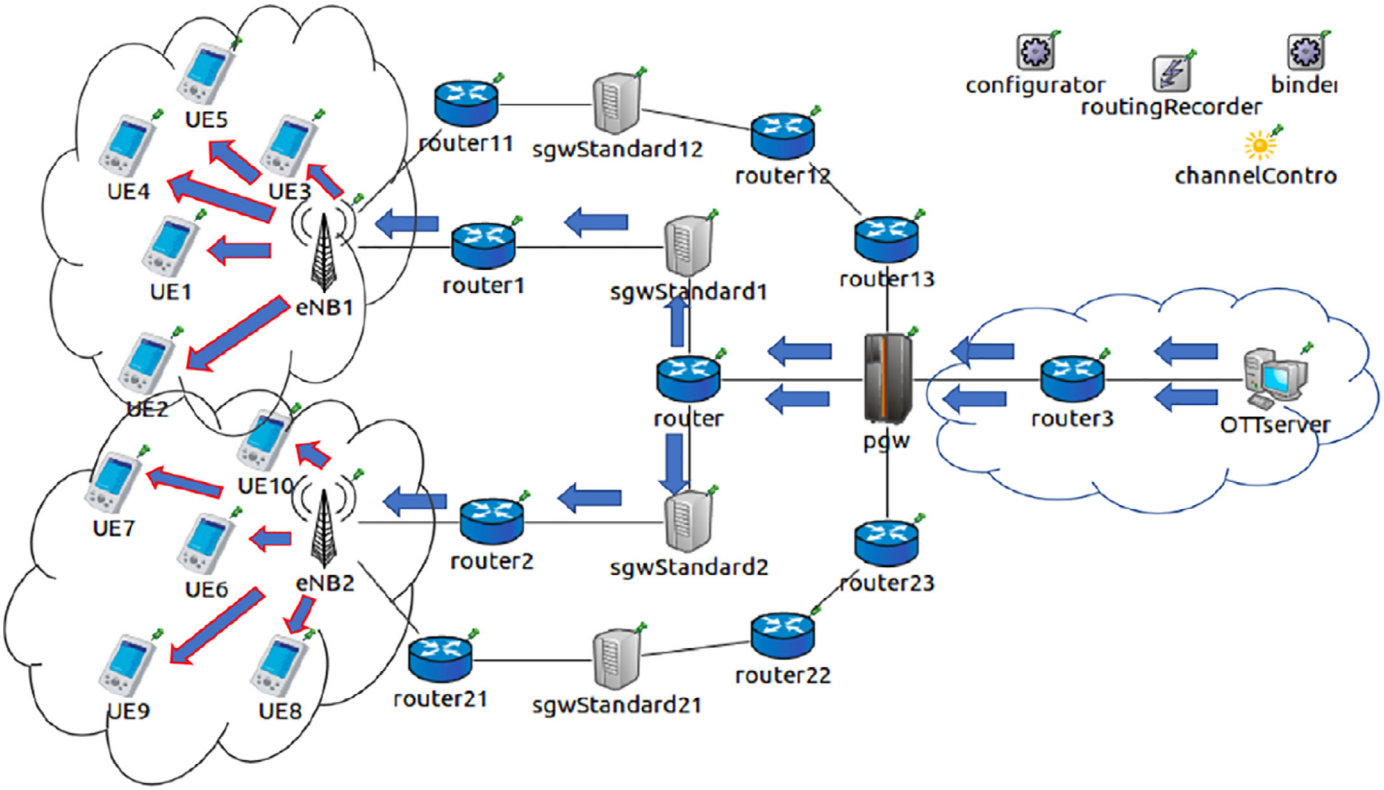


Fig. 10. Scenario B – Network Topology [11]. All users (UEs) simultaneously download a video from an OTT server via the LTE access network and transport network.

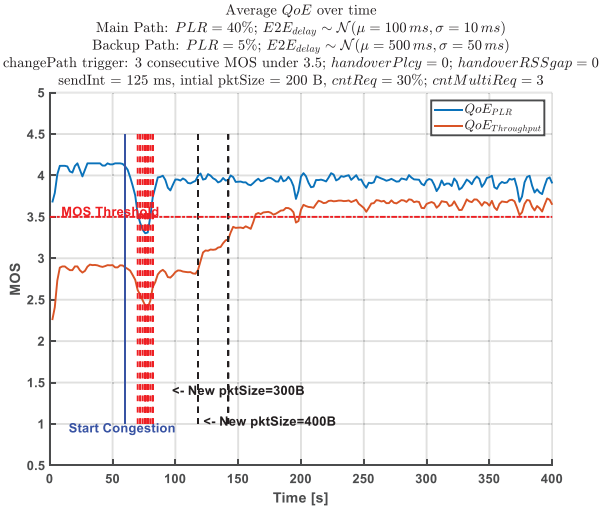


Fig. 11. Average QoE metrics over time for PLR and Throughput [11]. When congestion starts, since MOS is below a predefined threshold, traffic is redirected to the backup path leading to a MOS increase.

components, associated with the functionality that we should provide to the testbed users:

1. **Mininet**: An emulator platform to setup the virtual network. Within Mininet we setup our network topology consisting of switches, hosts and the controller [12].
2. **POX controller**: An SDN controller to control the virtual network. POX is a Python based open source OpenFlow/SDN controller used for faster development and prototyping of new network applications [13].
3. **QoE orchestrator modules**: When we initiate an experiment or scenario in CASPER testbed, all available network measurements are stored and queried in real-time through the QoE controller. The QoE Monitor checks these KPIs against acceptable thresholds as well as estimates the overall QoE score. Every time a metric crosses a certain threshold, the QoE Manager is informed. Specifically, if the quality of the link is under pressure or problematic due to high delay, packet loss or jitter, the QoE Manager will instruct POX to select the fail-over path which provides better conditions for such a communication.
4. **HTTP Adaptive Streaming server**: The HTTP Adaptive Streaming server used in CASPER comes in two versions. One version is Node.js – based (i.e., written in Javascript) and it is installed in Ubuntu environment [14]. This allows us to experiment with the network emulation and virtual hosts in parallel, and it has been implemented as a first step, mainly for a proof of concept of the server-client successful streaming. The alternative server version used in CASPER is the Microsoft Internet Information Services (IIS)

Table 4
Tools used.

TOOLS	VERSION
Ubuntu	16.0
Windows	10
Mininet (incl. POX controller)	2.3.0d6
ELK	7.5.0
Node.js	13.2.0
VLC	3.0.8 Vetinari
DASH-IF	3.0.1
FFmpeg	3.4.6–0ubuntu0.18.04.1
MP4Box – GPAC	0.5.2-DEV-revVersion
Wireshark	2.6.10
MiniEdit	2.2.0.1

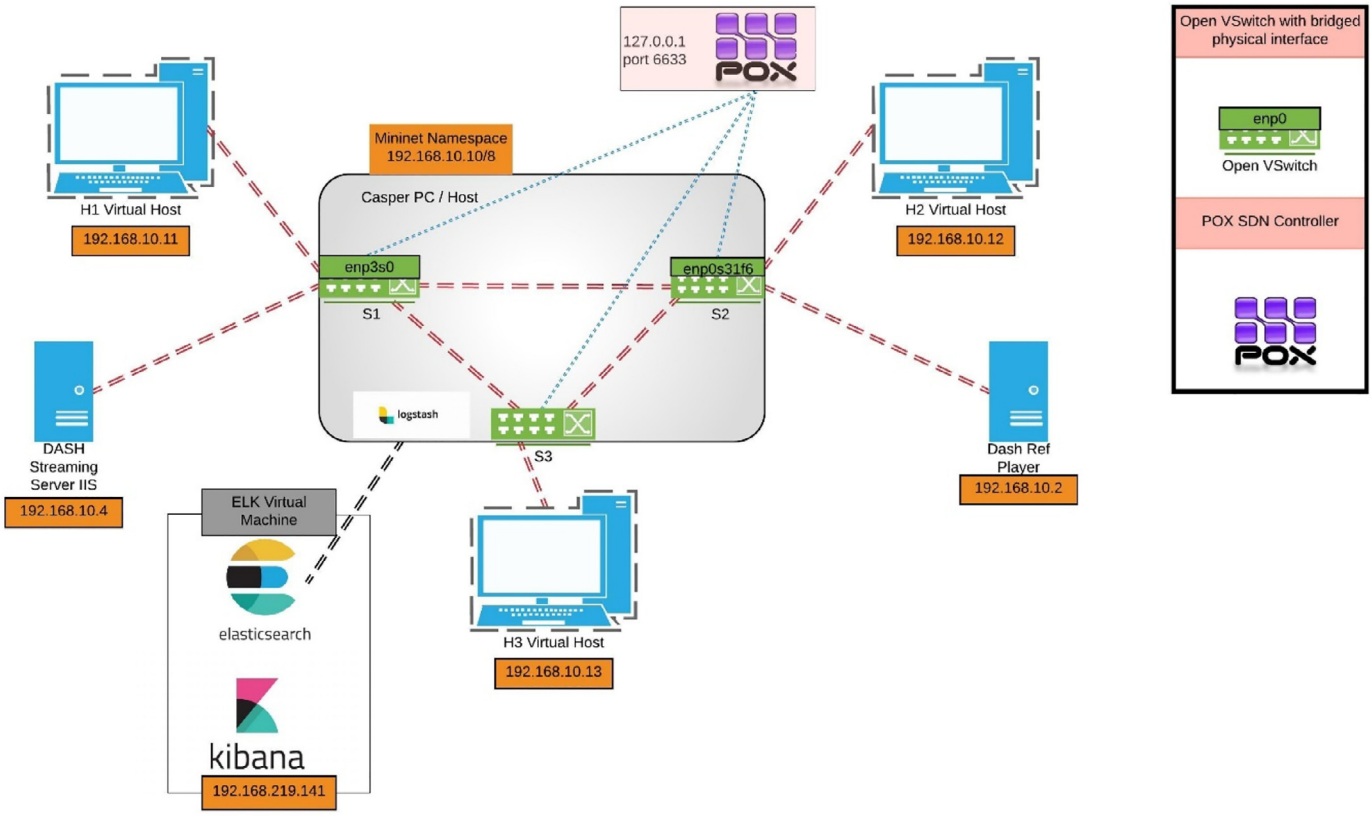


Fig. 12. CASPER testbed topology. The testbed consists of the main CASPER PC (Mininet network emulator, POX and ELK), three virtual hosts within the main CASPER PC, and two physical hosts (DASH server and client).

server installed in a physical host [15]. This allows us to experiment a) with a widely well-known server, adopted in order to support the HTTP Adaptive Streaming logic and b) with physical hosts, making the experiment very realistic.

5. **ELK (Elasticsearch, Logstash, Kibana)**: An analytics platform to store, visualize and push processed/aggregated data to the SDN controller for performing the network functionality over the virtual network [16].
6. **HTTP Adaptive Streaming client**: Except for VLC [17], we also use for the implementation of the video client a well-known adaptive streaming player called Dash.js, which provides a web interface. This tool allows us to monitor adaptive streaming-specific KPIs, such as client buffer, download rate, etc. [18].
7. **FFMPEG** (HTTP Adaptive Streaming library): FFmpeg is a free and open-source project which contains a wide software set of libraries and programs which was extensively used in CASPER for handling video, audio, and other multimedia files and streams [19].
8. **MP4Box – GPAC**: In CASPER, we used MP4Box for the preparation of the HTTP Adaptive Streaming content, specifically to create the Dynamic Adaptive Streaming over HTTP (DASH) manifest and associated files [20].
9. **MiniEdit** (Mininet's graphical user interface): The Mininet network simulator includes MiniEdit. It is a simple Graphical User Interface (GUI) editor for Mininet. MiniEdit is an experimental tool in order to show how Mininet can be extended [21].
10. **Wireshark**: Used for network monitoring purposes. It provides network troubleshooting, analysis, software and communications protocol development [22].

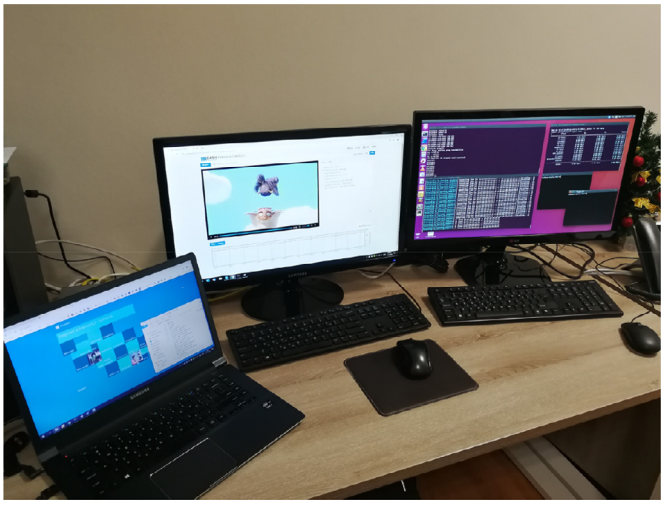


Fig. 13. The CASPER testbed consists of the main CASPER PC and two physical hosts (DASH server and client).

5.6. Hardware setup

In our CASPER testbed we are experimenting both with virtual and physical hosts. Virtual hosts provide greater flexibility in terms of manipulation, however physical hosts help reach to more realistic, near-the-market conclusions.

As shown in the photo of the testbed in Fig. 13, it consists of three physical devices:

- Mininet and POX controller run inside an OptiPlex 7050 PC in Ubuntu environment. It needs to be emphasized that

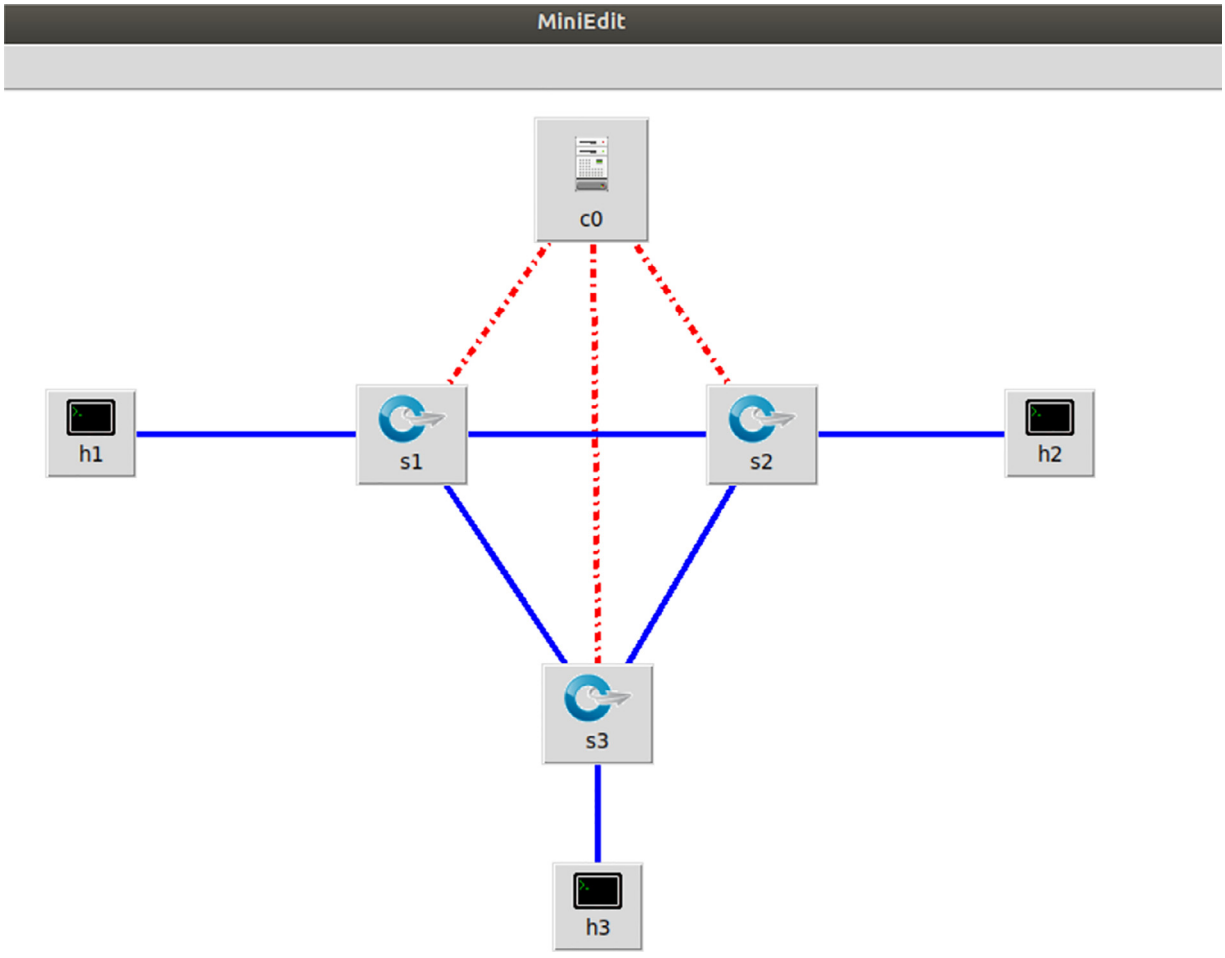


Fig. 14. Mininet virtual topology consists of three virtual hosts, three switches and the SDN Controller.

Mininet is not installed as a virtual machine; otherwise bridging between the two physical hosts would not work, which is mandatory for the HTTP Adaptive Streaming scenario. In the case of the Node.js – based HTTP Adaptive Video Server, both the server and the client run inside the same PC. The IP addresses of this system are 192.168.10.3 (first network card) and 192.168.10.5 (second network card). In Fig. 13, this corresponds to the screen on the right.

- In the case of Microsoft IIS, the HTTP Adaptive Video server runs inside a Samsung Series 9 laptop in Windows 10. The IP of this system is 192.168.10.4. In Fig. 13, this corresponds to the screen on the left.
- Also, for the Microsoft IIS case, the Dash.js HTTP Adaptive Video client runs inside an OptiPlex 3050 PC in Windows 10 environment. The IP of this system is 192.168.10.2. In Fig. 13, this corresponds to the screen in the middle.

In Fig. 13, we capture a moment during actual HTTP Adaptive Video Streaming from the server to the client, via Mininet, of the Big Buck Bunny video.

5.7. Performance evaluation

Our testing methodology for performance evaluation is based on the idea of comparing the CASPER-based controller with a default POX controller implementation, which comes pre-installed with Mininet and is the current status quo (e.g. [23]). For these experiments, we use the Microsoft IIS HTTP Adaptive Video server and the Dash.js HTTP Adaptive Video client. The steps in order to run the experiments are the following:

1. We open Mininet and activate the default POX controller.
2. We start running pings among all hosts in order to create traffic.
3. We then start the HTTP Adaptive Streaming session between the two physical hosts, while their traffic is passing through Mininet, and is controlled by the default controller.
4 We record the viewing experience at the client in terms of index being downloaded, bitrate being downloaded, etc. using the DASH reference client web interface.
5. We then initiate extra traffic within Mininet in order to create congestion in the network, as this creates a common bottleneck problem. We do this using the VLC player.
6. We check again how the viewing experience of the client is affected using the DASH reference client web interface.
7. We run the same experiment from scratch for the CASPER controller, record the user experience and compare. We run the experiment for three different monitored video KPIs: delay, packet loss and jitter.
8. In parallel, we record the bandwidth and delay values as time progresses using Kibana software.

The virtual topology within Mininet is presented in Fig. 14, while the full testbed topology including also the physical hosts is already depicted in Fig. 12.

Assisting in performance evaluation, the DASH reference client provides the possibility to select among a plethora of KPIs related to the quality of the video stream. These are the following [18]:

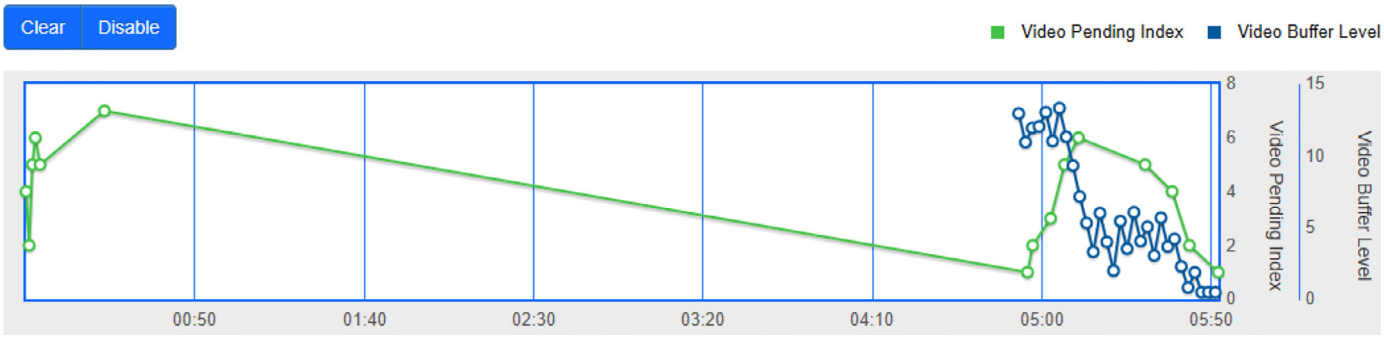


Fig. 15. Experiment progress – default controller (snapshot from dash.js enviornment). When congestion is added in the network, the video index is reduced to 1, while the video buffer keeps decreasing leading to a stalling. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

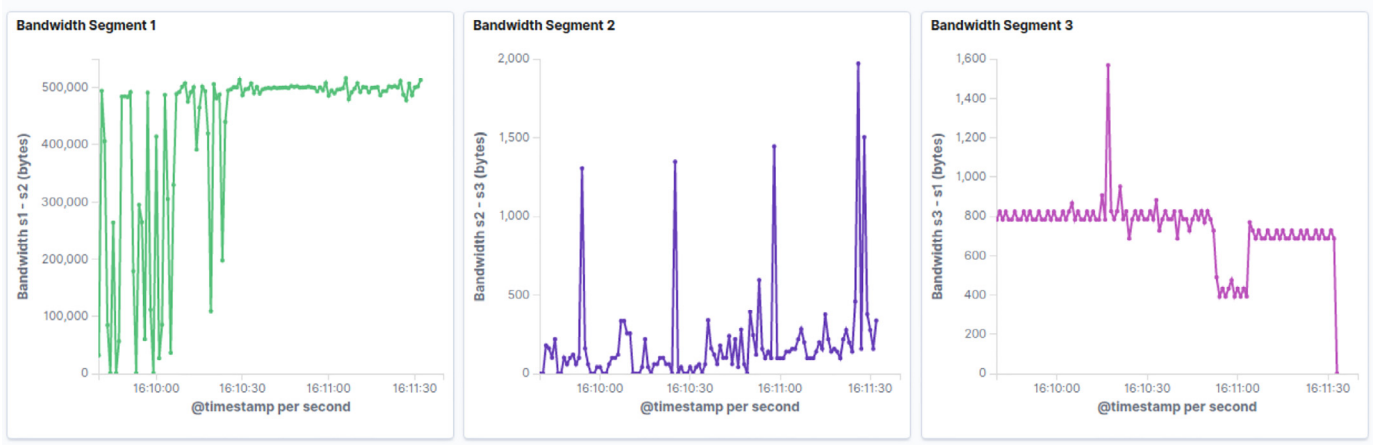


Fig. 16. HTTP Adaptive Streaming traffic and VLC traffic on top – default controller (snapshot from Kibana). All video traffic is passing through the s1->s2 path (green plot) throughout the whole duration of the experiment. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

- Buffer length: The length of the forward buffer, in seconds.
- Bitrate downloading: The bitrate of the representation being downloaded.
- Index downloading: The representation index being downloaded and appended to the buffer.
- Current Index / Max Index: The representation index being rendered / The maximum index available.
- Dropped Frames: The absolute count of frames dropped by the rendering pipeline since play commenced.
- Latency (min|avg|max): The minimum, average and maximum latency over the last 4 requested segments. Latency is the time in seconds from request of segment to receipt of first byte.
- Download (min|avg|max): The minimum, average and maximum download time for the last 4 requested segments. Download time is the time in seconds from first byte being received to the last byte.
- Ratio (min|avg|max): The minimum, average and maximum ratio of the segment playback time to total download time over the last 4 segments.

Any KPIs that are selected from this list will appear as plots in time at the dash.js web interface.

5.7.1. Default controller case

We first run the experiment with the default controller. We initiate traffic only by pings. Then we initiate the HTTP Adaptive Streaming session. Traffic is passing through the s1->s2 path (as declared in Fig. 14), while the bandwidth is greatly increased for this path. At the same time, the downloaded index at the DASH.js video client fluctuates until it reaches index=7 (the highest available) (Fig. 15). At the beginning of the 3rd minute of the video, we suddenly initiate a simple (non-adaptive) video stream between two Mininet virtual hosts (h1 and h2) (Fig. 16). All traffic continues to pass through the s1->s2 path (i.e., "bandwidth segment 1" in Fig. 16). The impact of this parallel stream is immediately shown in Fig. 15 where the buffer length is shown to progressively decrease, until a point that it reaches zero and the video is stalled. From that point onwards, the downloaded index keeps fluctuating at lower values as demonstrated in Fig. 17. Note that in Fig. 15 the blue plot which corresponds to the video buffer level is refreshed periodically and only the latest values are kept. On the contrary, the green plot which corresponds to the video pending index keeps its values since the video started playing.

5.7.2. CASPER controller case

In this scenario, we configure as the monitored KPI the average delay, while packet loss and jitter threshold are deliberately set to high values so that they do not have an impact on the controller behaviour, which allows us to understand what impact the delay KPI has on the controller decisions, and in sequence on the user viewing experience.

The HTTP Adaptive Streaming session goes through the path s1->s2, similarly as in the default case. While this happens, the user's buffer will start increasing. When the extra video stream is initiated via VLC, traffic starts flowing through the same path. However, as the CASPER QoE-based controller monitors the delay KPI, when this reaches a predefined threshold, the controller decides to reroute this traffic that relates to the virtual hosts to the alternative path s1->s2->s3, as shown in Fig. 18 (bandwidth segment 2 and segment 3). The traffic that is related to the HTTP Adaptive

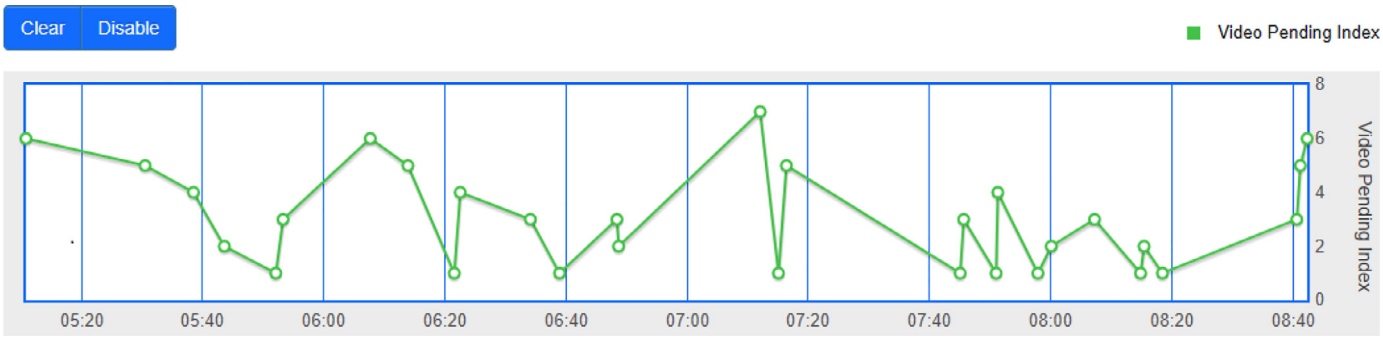


Fig. 17. Index downloaded – default controller (snapshot from dash.js environment). The index is fluctuating causing video quality fluctuations, which degrade the user viewing experience. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

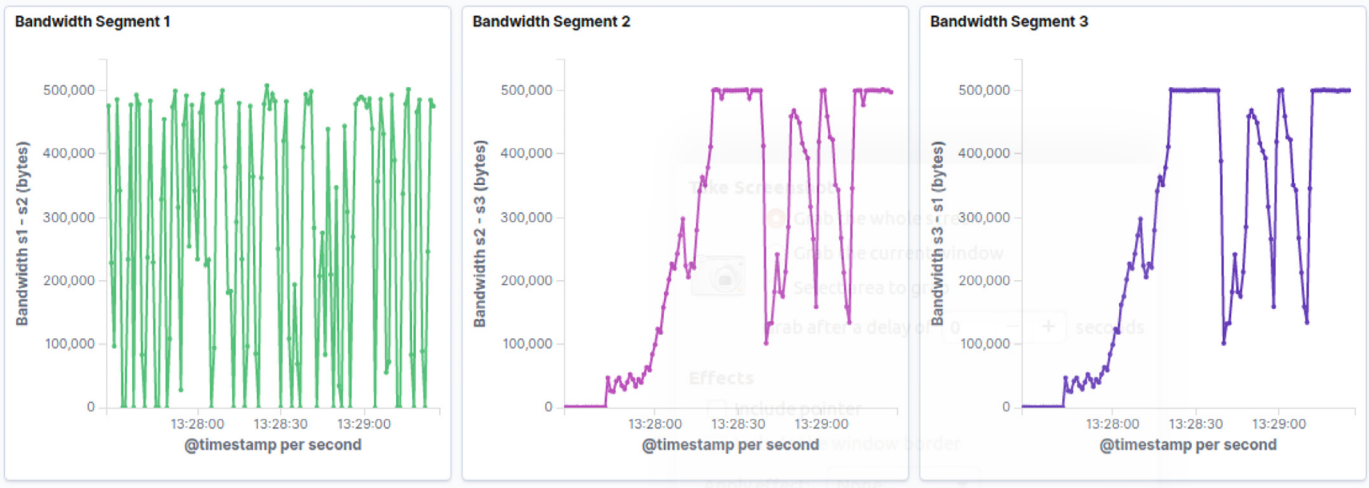


Fig. 18. HTTP Adaptive Streaming traffic and VLC traffic on top – CASPER controller (snapshot from Kibana). When congestion is added in the network, HTTP adaptive video traffic is passing through the s1->s2 path, while VLC non-adaptive video traffic is passing through s1->s2->s3.

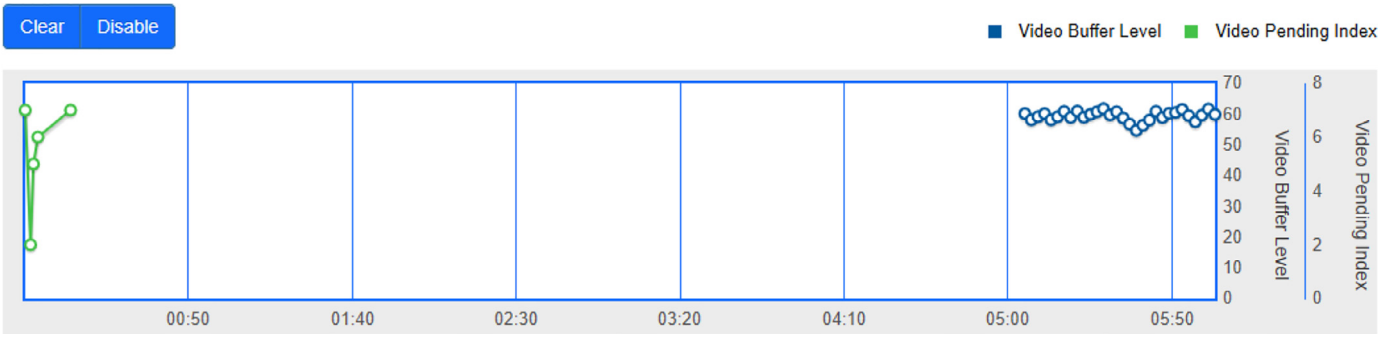


Fig. 19. Index downloaded – CASPER controller (snapshot from dash.js enviornment). The video index is preserved at the highest value 7 throughout the whole duration of the experiment.

Streaming session continues flowing through the s1->s2 path. In this way, the index downloaded is preserved at a high value, having a respective impact also to the viewing experience of the user (Fig. 19). Note that at this figure, the index is shown to be "7″ before 00:50 playtime; since the value is preserved until the end of the video, the "green plot" is not extended.

5.7.3. QoE evaluation

Since in our experiments we focus on HTTP Adaptive Streaming scenarios, in order to evaluate the QoE of the end users we need a standard QoE model for this specific type of applications. For this purpose, we use the ITU-T Rec. P.1203.1: "*Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport - Video quality estimation module*" [24].

The procedure in order to obtain video quality MOS scores for the video streamed to the client's device is the following: We keep track of the video segments been played out at the client, as well as the timestamps of video quality switches. This can be possible e.g. using the DASH.js reference player, since it provides plots with the indexes been downloaded in time (seconds). Afterwards, we run the itu-p1203 tool downloaded from [25] and presented at [26,27] for each video quality been played out, which will provide MOS values per second. Based on the seconds that each video quality has been actually played out we collect the

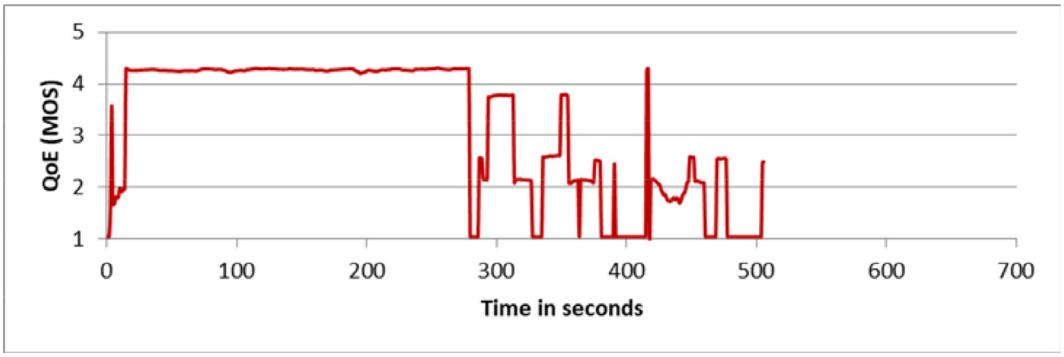


Fig. 20. QoE – Default controller. When congestion is added in the network, MOS values are fluctuating at low values.

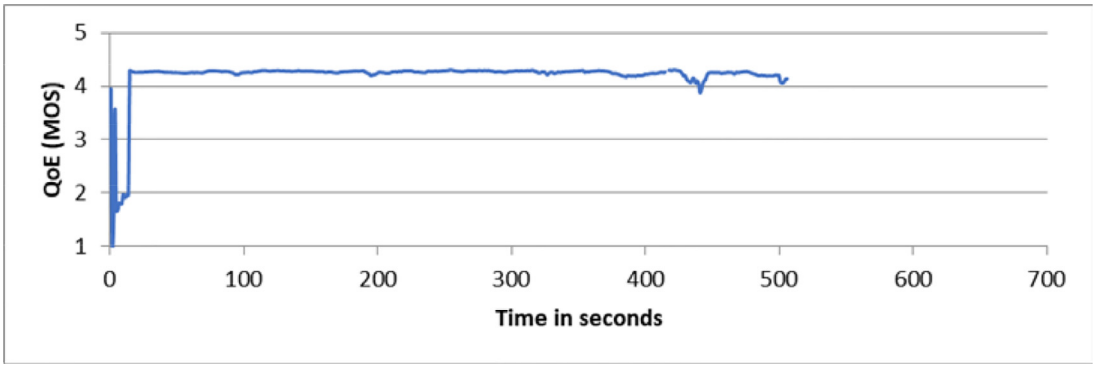


Fig. 21. QoE – CASPER controller. Even though congestion is added in the network, MOS values are preserved at stable and high values.

Fig. 22. CASPER consortium map.

respective values and create a plot of video streaming MOS over time.

Following this procedure, we therefore present the MOS plot for the default and CASPER controller cases. In Fig. 20, for the default controller case, we observe that the MOS starts with some fluctuations in the beginning, until the buffer is sufficiently filled, and then it is stable at very high value of approximately 4.3. However, afterwards, as the new VLC video stream is also initiated, the MOS score is significantly dropped and keeps fluctuating until the video ends. It is in fact, not only the low index values that will deteriorate the user viewing QoE, but also these continuous video resolution fluctuations, called switches, which have a negative impact on the user experience. Moreover, it needs to be noted that the video also presented a first stalling of 8 s, followed by multiple stallings of less than 2 s duration, as well as another stalling of 10 s. Such stalling events have an extremely negative impact on the user's experience. On the contrary, in Fig. 21, we observe that the MOS values for the CASPER controller case are preserved at the highest index available, even after the VLC stream start. The reason is that the controller reroutes the VLC traffic to the alternative path, causing much less congestion to both (as a load balancer).

6. Consortium

CASPER consortium consists of 2 academic and 3 industrial partners across 3 countries: Greece, Italy and Spain (Fig. 22). In this way, CASPER fosters the fruitful collaboration of academia and industry with the objective of developing innovative QoE-aware service management solutions and bridging the fundamental gap between academic (i.e., more theoretical, and less application-specific) and industrial (i.e., more system- and application-specific and devoted to solutions that can be implemented in real devices) research. In particular, CASPER builds a bridge among experts in: i) designing advanced theoretical algorithms, ii) developing and implementing middleware architectures in testbeds, and iii) developing and implementing software applications and hardware drivers. The industrial beneficiaries (WEST Aquila, IQUADRAT and ADAPTERA) have strong technical background on the development of advanced middleware architectures for service provisioning and CEM schemes. Also, the academic beneficiaries (University of Athens and University of L'Aquila), have experience in design methodologies and algorithms for service and network management, as well as in QoE analysis. Apart from the technical impact of CASPER, the consortium also commits on disseminating the

project results to the technical as well as the non-expert audience.[2]

7. Conclusion

This paper has presented the CASPER project. We have provided an overview of the project's mission and technical objectives, its structure and implementation logic, as well as main results in terms of QoE modelling, monitoring and management. As discussed in the paper, CASPER takes into consideration recent technological advances towards designing a practical, applicable, and integrated middleware architecture for personalised QoE provisioning by orchestrating the management of service flows. The project follows a human-centred approach, designing solutions that target at the satisfaction of the end-user when consuming a service. This approach entails a degree of personalisation in service delivery, which enforces an era of more "fair" telecommunication services. Moreover, the holistic approach of CASPER is expected to incentivize a collaboration paradigm between MNOs and OTTs, by providing a technologically feasible realization where feedback from MNOs is enabled and application-awareness is enforced. Furthermore, the final CASPER middleware architecture is delivered as a software solution enabling its exploitation from smaller telecom players and start-ups, who by taking advantage of the results of the project are enabled to create new business activities, such as specialized products for quality monitoring and benchmarking. The presented results depict that the user experience can be significantly improved by flexibly manipulating the network flows using a QoE-based SDN-based orchestrator.

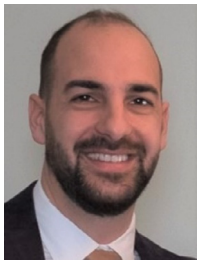
Acknowledgment

This work has been funded by the EC under the auspices of the H2020-MSCA-RISE CASPER project (grant 645393).

References

[1] N. Wang, D.C. Schmidt, A. Gokhale, C.D. Gill, B. Natarajan, C. Rodrigues, et al., Total quality of service provisioning in middleware and applications, Microprocess. Microsyst. 27 (2) (2003) 45–54, doi:10.1016/S0141-9331(02)00096-0.
[2] E. Liotou, D. Tsolkas, N. Passas, L. Merakos, Quality of experience management in mobile cellular networks: key issues and design challenges, IEEE Commun. Mag. 53 (7) (2015) 145–153.
[3] D. Tsolkas, E. Liotou, N. Passas, L. Merakos, The need for QoE-driven interference management in femtocell-overlaid cellular networks, in: Proceedings of the Tenth International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, Mobiquitous, 2014, pp. 588–601. 2013.
[4] W.J. Jeon, K. Nahrstedt, QoS-aware middleware support for collaborative multimedia streaming and caching service, Microprocess. Microsyst. 27 (2) (2003) 65–72, doi:10.1016/S0141-9331(02)00098-4.
[5] A. Colarieti, A. Marotta, M. Mpervarakis, L. Pomante, V. Tsolkas, QoE provisioning over mobile networks: The CASPER perspective, in: Proceedings of the IEEE Twenty-Second International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2017, pp. 1–5.
[6] ETSI GS NFV-MAN 001 V1.1.1 - Network Functions Virtualisation (NFV); Management and Orchestration.
[7] 3GPP TS 23.002 V15.0.0 - Technical Specification Group Services and System Aspects; Network architecture.
[8] E. Liotou, A. Marotta, L. Pomante, K. Ramantas, A middleware architecture for QoE provisioning in mobile networks, in: Proceedings of the IEEE Twenty-Second International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2017, pp. 1–5.
[9] M.-E. Xezonaki, E. Liotou, N. Passas, L. Merakos, An SDN QoE monitoring framework for VoIP and video applications, in: Proceedings of the IEEE Nineteenth International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), 2018, pp. 1–6.
[10] A. Virdis, G. Stea, G. Nardini, SimuLTE – A modular system-level simulator for LTE/LTE-A networks based on OMNeT++, in: Proceedings of the Fourth International Conference on Simulation and Modeling Methodologies, Technologies and Applications, 2014, pp. 59–70.
[11] D. Ciambrone, S. Tennina, M. Boschi, D. Tsolkas, L. Pomante, Assessing QoE–driven management policies for VoIP and video streaming service provisioning, in: Proceedings of the IEEE Twenty-Third International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2018, pp. 1–6.
[12] http://mininet.org/.
[13] https://noxrepo.github.io/pox-doc/html/.
[14] https://nodejs.org/en/.
[15] https://www.iis.net/.
[16] https://elasticsearch-py.readthedocs.io/en/master/.
[17] https://www.videolan.org/vlc/.
[18] https://github.com/Dash-Industry-Forum/dash.js.
[19] https://www.ffmpeg.org/.
[20] https://github.com/gpac/gpac/wiki/mp4box-dash-opts.
[21] https://github.com/mininet/mininet/blob/master/examples/miniedit.py.
[22] https://www.wireshark.org/.
[23] R. Jmal, L. Chaari Fourati, Implementing shortest path routing mechanism using Openflow POX controller, in: Proceedings of the 2014 International Symposium on Networks, Computers and Communications, 2014, pp. 1–6.
[24] P.1203: Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport (https://www.itu.int/rec/T-REC-P.1203-201710-I/en).
[25] https://github.com/itu-p1203/itu-p1203.
[26] A. Raake, M.-N. Garcia, W. Robitza, P. List, S. Göring, B. Feiten, A bitstream-based, scalable video-quality model for HTTP adaptive streaming: ITU-T P.1203.1, in: Proceedings of the Ninth International Conference on Quality of Multimedia Experience (QoMEX), 2017.
[27] W. Robitza, S. Göring, A. Raake, D. Lindegren, G. Heikkilä, J. Gustafsson, P. List, B. Feiten, U. Wüstenhagen, M.-N. Garcia, K. Yamagishi, S. Broom, HTTP Adaptive Streaming QoE Estimation with ITU-T Rec. P.1203 – Open databases and software, in: Proceedings of the Nineth ACM Multimedia Systems Conference, 2018.

Eirini Liotou received the diploma degree in electrical and computer engineering from the National Technical University of Athens, the M.Sc. degree in communications and signal processing from the Imperial College of London, and the Ph.D. degree from the Department of Informatics and Telecommunications, University of Athens in 2017. She was a Senior Software Engineer in the Research and Development Department, Siemens Enterprise Communications. She has authored more than 25 peer-reviewed publications in conferences and high-impact journals, while she has served as Technical Program Committee member and Technical Reviewer in flagship IEEE conferences and international journals. She is currently a Post-Doc Researcher in the Communication Networks Laboratory, University of Athens. Her research interests are in the area of Quality of Experience in mobile cellular networks.



Dimitris Tsolkas holds a Ph.D. degree from the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens (NKUA). He has extensive experience in R&D and project management, working in a plethora of EC-funded research and development projects in association with institutions from academia (Department of Informatics and Telecommunications - NKUA, Computer Technology Institute and Press "Diophantus", and Department of digital systems - University of Piraeus) and industry (LINK Technologies S.A., WEST L'Aquila, and MOBICS S.A.). He has teaching experience as a lecturer at the Business College of Athens (BCA) and as an instructor of postgraduate and graduate courses in the Department of Computer Science and Engineering, University of Ioannina, and the Department of Informatics and Telecommunications - NKUA. He has published more than 35 articles in peer-reviewed journals, international conferences and book chapters. His research interests are in the areas of 5 G RAN, D2D communications, and QoE provisioning.

Giorgos Kalpaktsoglou is a System Engineer in Adaptera Ltd. He received his Bachelor's degree in Mechanical Engineering and Mechatronics from the Technological Educational Institute of Crete and Brunel University London. His research interest is on SDN networks, network analytics as well as system integration and cloud computing. He previously worked in the Maritime section as well as a Research Engineer in Robotics and Software Intelligence lab, NCSR Demokritos, Greece.

[2] http://gain.di.uoa.gr/casper/dissemination.html.

Stefano Tennina received the Laurea degree (cum laude) in Electronic Engineering from the University of L'Aquila, Italy, in 2003 and the Ph.D. degree in Electrical and Information Engineering from the same Institution in 2007. From 2002 to 2009 he has been with the Centre of Excellence in Research DEWS as a post-doctoral researcher. From 2009 to 2012 he was a Research Scientist in the CISTER Research Unit, involved in two projects: EMMON (ARTEMIS program), aiming at large-scale and dense real-time WSNs and SENODS (PT-CMU program), targeting energy efficiency in large Data Centers. Since 2004 he is a co-founder of WEST Aquila s.r.l., where he holds now a position of Senior Researcher and is full time employee since 2012. His research focus is in the area of wireless communication protocols/systems, with particular emphasis on experimentation with real COTS testbeds. His current research activity is mainly focused on fully distributed positioning algorithms and energy efficient wireless communications through network coding and distributed source coding algorithms. As professional member of IEEE and ACM since 2011, he is author or co-author of more than 30 journal and conference papers in technical journals and conference proceedings.

Luigi Pomante has received the "Laurea" (i.e., BSc+MSc) Degree in Computer Science Engineering from "Politecnico di Milano" (Italy) in 1998, the 2nd Level University Master Degree in Information Technology from CEFRIEL (a centre of Excellence of "Politecnico di Milano") in 1999, and the Ph.D. Degree in Computer Science Engineering from "Politecnico di Milano" in 2002. He had been a Researcher at CEFRIEL from 1999 to 2005 and, in the same period, he had been also a Temporary Professor at "Politecnico di Milano". From 2006, he is an Academic Researcher at centre of Excellence DEWS ("Università degli Studi dell'Aquila", Italy). From 2008 he is also Assistant Professor at "Università degli Studi dell'Aquila" (he is responsible of the "Embedded Systems" course). His activities focus mainly on Electronic Design Automation (in particular Electronic System-Level HW/SW Co-Design) and Networked Embedded Systems (in particular Wireless Sensor Networks). In such a context, he has been author (or co-author) of more than 100 articles published on international and national conference proceedings, journals, and book chapters. He has been also reviewer and member of several TPCs related to his research topics. From 2010, he has been in charge of scientific and/or technical issues on behalf of DEWS in more than 10 funded European and national research projects.

Nikos Passas received the Diploma (Hons.) from the University of Patras, Patras, Greece, in 1992 and the Ph.D. degree from the University of Athens, Athens, Greece, in 1997. He is currently a member of the teaching staff with the Department of Informatics and Telecommunications, University of Athens, and a Group Leader of the Green, Adaptive and Intelligent Networking Research Group inside the department. Over the years, he has participated and coordinated a large number of national and European research projects. He has authored/co-authored more than 140 papers in peer-reviewed journals and international conferences. His research interests include mobile network architectures and protocols. He is particularly interested in quality of service provision for wireless networks, medium access control, and mobility management. He was a Guest Editor and Technical Program Committee Member in prestigious magazines and conferences, such as IEEE Wireless Communications Magazine, Wireless Communications and Mobile Computing Journal, IEEE Vehicular Technology Conference, IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, and IEEE Global Communications Conference.