IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 23, 2020, accepted January 11, 2021, date of publication January 14, 2021, date of current version January 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3051570

Follow the User: A Framework for Dynamically Placing Content Using 5G-Enablers

DAVID SANTOS[1], RUI SILVA[1], DANIEL CORUJO[1], (Senior Member, IEEE), RUI L. AGUIAR[1], (Senior Member, IEEE), AND BRUNO PARREIRA[2]

[1]Instituto de Telecomunicações and Universidade de Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal
[2]Altran Portugal, R&D Department, 4400-012 Vila Nova de Gaia, Portugal

Corresponding author: David Santos (davidsantos@av.it.pt)

This work was supported by the Fundação para a Ciência e Tecnologia (FCT) / Ministério para a Educação e Ciência through the National Funds under Project PTDC/EEI-TEL/30685/2017.

ABSTRACT This article presents a framework for improved and efficient video delivery in scenarios featuring users moving at high speed (e.g., trains), leveraging on dynamic Multi-access Edge Computing (MEC) enabled 5G network capabilities. The framework is location-aware and it allows the content to efficiently follow the users, conserving load usage on network and computational resources, by placing virtualized Content Delivery Network (vCDN) nodes at edge sites. The nodes are controlled by the framework's centralized control unit, which is able to dynamically and preemptively deploy virtulized resources, as the train moves. The framework is capable of segmenting video content and placing the specific portion of content that a user is likely to consume across a set of dynamically deployed vCDN nodes, associated to the coverage section the train is currently passing. A proof of concept was implemented and evaluated, where the benefits of using this framework are assessed. Results showed that the proposed system was able to reduce the load on the core network by 10.9 percent and maximize the cache hit ratio to a value of 99.8 percent.

INDEX TERMS vCDN, 5G, MEC, location-awareness, video delivery optimization, NFV, SDN.

I. INTRODUCTION

The amount of mobile data that networks need to deal with is rapidly and constantly increasing, posing various challenges to network operators and their capability to provide service with acceptable Quality of Experience (QoE) while keeping the CAPEX and OPEX as low as possible. In an Internet report from 2020 [1], it is predicted that mobile broadband speeds are expected to reach an average of 59 Mbps in 2023, as compared to 15 Mbps in 2018, an almost 5 fold increase. This increase is mainly due to the introduction of the 5G mobile network which is expected to offer downlink performance in the order of 575 Mbps by 2023 [1], which associates mobile devices and users in mobility scenarios to this network data flow increase. As the network performance capability increases, allied to smartphone's developments in terms of screen resolution and the ability to display VR content, the mobile end user is able to consume highly demanding traffic such as Ultra High-Definition (UHD) and Virtual Reality (VR) content. However, these impose considerate levels of load to the network, as the associated content generates large amounts of traffic over the network, proportional to the popularity of the services. These will add and complement to existing services such as Video on Demand (VoD), motivating the need to optimize how video content flows through the network and is delivered to the user. As a result, current networks are being overwhelmed by new service requirements and greater amount of connected devices. This leads to the need for advancements in various areas such as 5G [2], Machine Learning [3] or Block Chain [4], to name a few.

Despite the fact that 5G enables (or improves) scenarios which will contribute to a significant increase in the network load, particularly over core links where a cumulative traffic effect prevails, it also provides technologies that optimize the networks' behaviour, such as its native support for Multi-access Edge Computing (MEC) and Management and Orchestration (MANO) [5]. MEC allows the content and network procedures to be placed closer to users, reducing the amount of interactions with the core network and video servers, as well as reducing latency [6]. This technology, combined with state of the art Content Delivery Networks (CDN), currently used by video service providers such as

The associate editor coordinating the review of this manuscript and approving it for publication was Mingjun Dai.

Netflix and Amazon Prime Video, allows for an optimized content distribution. Another feature that also contributes to this is the 5G system's capability to easily integrate computational systems, such as prediction systems [7]. The predicted knowledge of a user's mobility path allows for content to be distributed in CDN Nodes across a set of available MEC hosts in the 5G network, further optimizing content delivery. MEC hosts, placed on edge sites, are usually small datacenters with less computing resources, making it important to optimize their usage. The MANO framework [8] plays a key role here as it allows for CDN nodes to be dynamically deployed as Virtualized Network Functions (VNF), scaled and terminated as required, lowering OPEX costs (e.g., by reducing electrical costs).

Users in mobility scenarios can cross different mobile network infrastructure contexts, transiting from dense metro centers with a high number of cells and access to high computational capacity, and into rural areas where there is less capable infrastructure both at radio transmission and at computational capacity levels. Taking this into consideration, mobile network operators need to be able to provide solutions that do not rely on other vehicles as data mules. Our focus is in capitalizing the networks' ability to instantiate virtualized network functions allowing a cloud-enabled mobile network operator to instantiate mechanisms that assist on the delivery of data and content in the different contexts it encounters.

This article, an ample evolution of our previous work [9], presents a framework that takes advantage of the enablers and features offered by the novel 5G network, using them to optimize the delivery of video content to users in high mobility scenarios, aiming to lower the costs for network operators and increase the QoE perceived by the users. The framework incorporates a monitoring module that is able to obtain location data from the 5G core network's APIs and use it to predict the path of movement of users and compute what content is predicted to be consumed by the user throughout its journey. The placement of video data across the movement path of users is achieved by placing centrally-controlled vCDN nodes in available MEC hosts (using Software-defined Networking (SDN) principles), taking into consideration what content the user will consume while traversing a specific MEC host, making it possible for our framework to have the content following a user or a set of users. The centralized control part of this framework is able to use the 5G MANO's framework to dynamically deploy and remove virtualized CDN nodes in MEC hosts, thus optimizing computational resources. It also offers another capability that helps reducing the amount of data crossing the core network and increases reliability in the face of user's unpredictable behaviour: peer-to-peer caching. This capability allows vCDN nodes to retrieve content not only from the video server but also from peer caches placed in adjacent MEC hosts. 5G also exposes an API that enables applications to influence the network's traffic, allowing this framework to dynamically forward video related traffic to desired MEC hosts.

Although extensive work has been conducted in the field of efficient edge caching in 5G networks, the majority relates to optimization algorithms that only present numerical results. The works that address caching frameworks are mainly evaluated using simulations, with no testbed implementation or testing. In this line, the contribution of our work is as follows:

- It provides a framework that is able to dynamically instantiate cache nodes in available MEC platforms and de-instantiate the same nodes when they are not needed, thus saving resources in the usually resource-constrained edge node;
- The proposed framework is able to interact with both the 5G network and the MEC platform via interfaces standardized by 3GPP and ETSI, respectively, allowing for a smooth transition between our testbed and a real life MEC enabled 5G network;
- A Monitoring module is introduced that collects and stores monitoring information about the different procedures of the framework. Other novel modules, such as decision algorithms discussed in section II, can use this monitoring data for dynamic configuration of network procedures. In this work, this monitoring ability was exploited by retrieving information about user location from the 5G network, via standard interfaces, and acting according to the user position, making this a location aware framework;
- A Control module is able to configure the cache nodes in a SDN-like manner, triggering the nodes to load parts of content from either the origin server or peer caches or even to evict content from its storage;
- This is a mobility-aware framework, which only loads the necessary content for the time span a particular user set will be served by a specific MEC platform;
- Finally, the framework was assessed through an experimental implementation, with emulated train and passenger movements (using real life travel data), and results showed that it was able to maximize the cache hit ratio (measurement of how many content requests a cache is able to fulfil successfully, compared to how many requests it receives) in CDN nodes while reducing the usage of network resources (both networking and computational) and maintaining an user QoE level where no video buffering occurs. In order to accomplish that, the delivery throughput per user for a 1080p video must be equal or higher than 5 Mbps as per [10].

The remainder of the paper is structured as follows. Section II presents the state of the art, followed by a description of the framework in Section III and its behaviour in Section IV. The description of the experimental deployment of the framework is presented in Section V, with evaluation results showcased in Section VI. Finally, the paper concludes in Section VII and presents future work guidelines in Section VIII.

II. BACKGROUND AND STATE OF THE ART

The fifth generation (5G) cellular network architecture promises not only to deliver higher data rates, extremely low latency and reliability to mobile users (similarly to the evolution of previous cellular technologies) but also to provide a framework to support new verticals (i.e., new business) and new applications with specific requirements. To handle these challenges, this architecture relies on key emerging technologies that are helpful in improving the architecture and meeting the demands of users, such as Network Functions Virtualization (NFV), SDN and MEC.

A. ENABLING TECHNOLOGIES

1) NETWORK FUNCTION VIRTUALIZATION (NFV)

The main idea behind NFV is to virtualize network functions, and migrate these functions from stand-alone boxes on dedicated hardware, to appliances running on cloud systems [11]. NFV separates the network functions from proprietary hardware appliances so they can run in software. Once software functions are independent of the underlying physical machines, specific functions can be packaged together into a new network and assigned to an environment [12]. This approach allows Virtual Network Functions (VNFs) to be dynamically instantiated, relocated or destroyed, without necessarily requiring the purchase and installation of new hardware, giving flexibility and reducing costs to the network operator [13].

2) SOFTWARE DEFINED NETWORKS (SDN)

Conventional computer networks can be classified in three different planes: data, control and management. Usually the control plane (which is responsible for making the decision of how to handle the network traffic) and the data plane (which is responsible for forwarding the traffic network following the decisions made by the control plane) have always been wrapped together on the majority of network devices such as routers, switches, firewalls, etc.

SDN essentially decouples the control plane from the data plane and moves it to a centralized controller. By doing this, all complexity of network control is moved into this software-based entity which is directly programmable and manageable in a centralized manner [14]. Thus, the underlying infrastructure can be abstracted from the applications and network services, enabling the network to be treated as a logical entity [15].

3) MULTI ACCESS EDGE COMPUTING (MEC)

In order to support low latency requirements, 5G networks, have native support for MEC [5], deploying a cloud at the edge of the network, closer to the users, providing lower latency times and lower backhaul traffic and core load. Despite low latency being one of the most recognized key requirements for the introduction of edge servers and enabling new services, it is not the only Key Performance Indicator (KPI) for MEC. Mobile operators and service providers are in fact progressively defining their strategy for the adoption of MEC, expecting it to be a technology providing two kinds of benefits: revenue generation and cost saving. Particularly, as infrastructure owners, they are interested in other MEC KPIs such as network utilization and cost savings (given by shorter data traffic paths), energy efficiency, total cost of ownership (TCO) and the management of networking and computation resources in virtual environments [16]

MEC can satisfy the communication requirements of high reliability and low latency, since it provides Internet service environment and cloud computing capability for access network [17].

4) CONTENT DELIVERY NETWORKS (CDN)

Content Delivery Networks (CDNs) have evolved to improve user perceived Quality of Service (QoS) when accessing Web content. A CDN creates replicas of the original content into distributed cache servers closer to users, allowing content to be delivered to end-users in a more reliable and timely manner [18].

By deploying a distributed system positioned throughout the network that disseminates popular content, such as streaming video at locations closer to the user, MEC is poised to make CDN up to 40% more efficient for cellular communications service providers [19]. When instantiated as a vCDN, it can operate across a range of virtualized infrastructure, from Core data center to every single Edge data center, and it can fully support all types of virtualized network function deployment at different operator network vantage points.

One of the major aspects of a cellular network, when compared to a traditional wired network, is subscriber mobility. The negative impact of user mobility is amplified with the decrease of the cell coverage radius [20], usually coupled with the high frequencies necessary to attain the higher downlink performances to be provided by 5G. Consequently, there is substantial interest in academia and industry in finding solutions for mobility optimizations, such as handover, offered traffic, dimensioning of signaling network, user location updating, registration, paging and multi-layer network management [21].

B. RELATED WORK

Regarding the use of CDN's, [22] presented a prototype for a cost-aware, cloud-based vCDN suitable for a federated cloud scenario. The authors used a virtual CDN controller that spawns and releases virtual caching proxies according to variations in user demand, by leveraging a OpenStack-based [23] federated cloud. The deployed caching proxies, which are virtual machines (VMs) with associated storage, consist of NGINX [24], a well-known load-balancer, web server and reverse proxy. Despite that results indicate that spawning virtual proxies is an elegant solution to address user demand, the presented prototype does not account for user mobility and the deployed caches do not have peer-to-peer capabilities. It is worth mentioning that the authors also

present a cost-based heuristic algorithm used for selecting the data centers where proxies should be spawned.

The authors of [25] point the key benefits when merging the two complementary technologies: CDNs and P2P. When compared with conventional CDNs, peer-assisted CDNs have higher scalability and throughput, since workload and content can be balanced and exchanged among multiple peers, thus reducing backhaul traffic. The results of the survey suggest a significant contribution of peer-assisted content delivery in reducing infrastructure costs for content providers and CDNs. The measurement results presented in [26], [27] and [28] demonstrate that only 8,8% of Spotify's music traffic is delivered from their own servers, while the remaining 91.2% are delivered from either peer-assistance (35.8%) or from local caches (55.4%). The authors in [29] reported that up to 98.0% of the video content in Kankan is distributed in a peer-to-peer fashion, since edge nodes are responsible for handling a long-tail of unpopular videos. In a similar way, in [30] the authors have analyzed user trace data collected from the Tudou platform and reported a traffic saving from 36% to 96% for popular video content.

In [31], the authors exploit the node mobility for vehicles to serve as relays to improve network performance of Vehicular content centric networks (VCCNs). Yao *et al.* propose a scheme called cooperative caching based on mobility prediction (CCMP) for VCCNs. The main idea of CCMP is to cache popular contents at a set of mobile nodes that may visit the same hotspot areas repeatedly. Utilizing the trajectory history records of different vehicles, CCMP is able to predict the probability of their next visits to different hot regions in the area. Caching nodes are chosen based on the sojourn time and content caching is decided according to popularity. The authors only used simulations to demonstrate that the scheme has a higher success ratio and lower access delay when compared with several other state-of-the-art schemes.

In [32], Niu *et al.* focus on the problem of mobility-aware transmission scheduling for caching at edge nodes near hotspots and utilize the multi-hop relaying and concurrent transmissions to achieve better performance. A mobility-aware caching scheduling scheme is presented, called Multi-Hop Relaying-based Caching, where multi-hop D2D paths are established for edge nodes, and concurrent transmissions are exploited in the scheduling of caching at edge nodes. After performance evaluations, the authors demonstrate that the scheme achieves more than 1x higher expected cached data amount, compared with other existing schemes. There is no physical implementation on the proposed scheme.

Dutta *et al.* [33] proposed an algorithm to cache the contents in the user device, maximizing local hit rate with minimal power consumption in high-speed trains (HST). Although the simulation evaluation presents beneficial results, it does not include the possibility of introducing a cache mechanism in a device commercially available off-the-shelf (COTS) and is limited to the users devices' capacity.

In [34] a MEC-based video caching mechanism is proposed, where only the highest available bit-rate video is cached and, by using the processing power available at the MEC, is transcoded to the requested lower bit-rate version. The authors developed a testbed to evaluate the performance of the proposed caching mechanism in real time and demonstrate that the proposed method reduces the backhaul traffic load and video load time, while increasing the cache hit-rate as compared to traditional store and forward caching mechanism. In the suggested mechanism the authors do not take into account user mobility, the cached data is populated considering only video popularity and the different caches do not exploit peer-to-peer communications to lower the backhaul traffic even more. The proposed method could be deployed on top of our framework and be an asset to cope with different bitrate requests by users.

Reference [35] proposes a peer-to-peer caching solution for multi-virtual operator environments, where the caches can retrieve content from peer caches belonging to other virtual operators that are co-located at the same physical edge node. The solution was implemented and tested in a cloud environment and results show a cache hit ratio increase of approximately 15.3% by retrieving content from the mentioned peer caches.

In [36] the authors propose a content-centric mobile network framework for edge caching in 5G networks. In this framework, content can be cached in the 5G core network, in base-band unit (BBU) pools and in the terminal itself being, in the latter case, exchanged between terminals via device-to-device (D2D) communication. The authors also point out some implementation challenges alongside solutions, but no actual implementation is assessed, and only simulation results are presented. There are two main advantages of our solution when compared with a Content-Centric Network (CCN) solution: (i) CCNs are disruptive for current networks. Because in CCN, endpoints communicate based on named data instead of IP addresses, modifications to the network architecture are required. Our solution uses standard interfaces and APIs. (ii) Our solution loads, across edge nodes distributed through the movement path of the users, only the content that will effectively be consumed.

Reference [37] proposes a CaaS framework placed at the mobile operator's cloud which can be controlled by third parties via a API. The paper then focuses on a caching policy that maximizes the return on caching investment.

In [38] the authors introduce a dynamic caching framework for MEC which predicts the content demands of mobile users by using a machine learning approach. The framework takes into consideration the users' location and requested content in order to efficiently place the necessary content in the MEC nodes. The authors present numerical results showcasing the effectiveness of their solution.

There is extensive work in the field of algorithms regarding efficient content caching. Reference [39] presents a Stackelberg game-based optimization model for the distribution and coverage of edge nodes, traffic models and time span in a proactive edge caching scenario. Reference [40] proposes a novel cache replacement strategy based on user location

and their content preference, which promises 23% more traffic saving relative to traditional methods. Reference [41] presents a multi-tier caching and resource sharing optimization problem for high performance video streaming in 5G networks. Reference [42] proposes a mobility-aware handoff priority-based request admission control policy for cached content request management in 5G networks. Reference [43] proposes a theoretical caching design focusing on the placement and retrieval of popular chunk-based contents for efficient utilization of the available storage capacity. In this work, the authors considered that content is placed at the RAN nodes. Reference [44] uses real-world datasets to assess the request patterns and user behaviours video streaming behaviours in a mobile network alongside the effectiveness of current caching solutions. The authors propose a caching strategy based on the traces and evaluate its performance through simulations, comparing the solution with current caching strategies. In [45] the authors propose a content placement algorithm based on optimal bandwidth allocation. The algorithm is then evaluated, presenting simulation results.

Leveraging SDN in mobile scenarios, an important mobility aspect in future mobile network systems is the ability to predict a user's next cell or even the path it will traverse [46], and take appropriate steps to prepare the network accordingly. Usually, the applications of mobility prediction in mobile networks can be classified as handover management, location-based applications and resource management [17].

To guarantee continuous service without retaining substantial amounts of resources across the whole network, passive resource reservation policies are used [47], by pre-reserving a certain amount of resources in the coverage areas that will probably be visited by users. Furthermore, when it is possible to accurately predict the future movements of users, some location-based contents which are highly demanded can be distributed according to users consumption and network conditions.

Concretely, in [48], the authors propose the concept of software defined virtual CDN (SDvCDN) as a virtual cache network fully deployed in software, over a programmable distributed cloud network infrastructure that can be optimized, using global information about network conditions and service requirements. This solution takes into consideration both the placement and routing of content objects along with the allocation of the required virtual storage and transport resources. Although this is a promising and interesting concept, it does not take into account mobility scenarios nor does it present evaluation results of a practical implementation.

Likewise, in [49] the authors built a prototype in which a CDN surrogate server is deployed on-the-fly upon request of the CDN provider, using NFV and micro-service architecture principles. Although the authors use a cache node as a new deployed CDN component in their experiment, the content of the cache node is not dynamic, but a sample downloaded from Dropbox. The authors did not focus on content placement and request redirection.

Another way to cope with an increasing demand of video transmissions is to adapt the video quality according to the network conditions. In [50] the authors propose a QoE management model for video streaming service in a SDN context. This article aims to introduce/develop a method based on machine learning algorithm to predict, estimate Mean Opinion Score (MOS) and adapt video under specific network conditions. By providing different scenarios, through the change of quality parameters (e.g., buffering time, resolution, bitrate, number of frames per second) and measuring real MOS values, the learning algorithm can predict an estimated MOS. The authors prove that the algorithm works correctly by comparing the estimated MOS value with the current network conditions by measuring the Video Quality Monitoring (VQM), Structural Similarity Index (SSIM) and Peak signal-to-noise ratio (PSNR). The video parameters are then modified (i.e., number of frame per second, bitrate, resolution) in function of the value of estimated MOS and Quality of Service (QoS) parameters such as (RTT, jitter, bandwidth, buffering and delay), and the traffic is steered using SDN capabilities.

III. SYSTEM FRAMEWORK

This section presents the design principles and architecture of the system, illustrated in figure 1, which aims to minimize the challenges identified in section I in a Video on Demand service context. This is a location-aware vCDN framework for network operators (or third-party CDN providers that subscribe to operator services) and it is an evolution of a previously presented work [9]. The design takes advantage of 5G's capabilities and enablers, namely its high throughput and QoS capabilities, monitoring and traffic influence APIs and its native support for MEC, allowing the system to place content as close to the users as possible. Furthermore, the system is fully virtualized and can be deployed as a single or a collection of VNFs. The vCDN system uses 3GPP and ETSI MEC standard interfaces to communicate with the 5G core and the MEC framework, respectively, meaning that it can be integrated in an actual 5G network with minimal overhead.

The framework was devised to support users consuming VoD services via a 5G network in a high speed mobility scenario. In the scope of the design of this, it was considered that the various **5G Access Nodes** spread across the movement path of the users have an associated MEC in a mapping of 1:*n* (one MEC per *n* Access Nodes). This means, as an example, that when a user handovers from cell_*n* of MEC0 to cell_*0* of MEC1, a MEC handover occurs. The MEC placement paradigm is a work in progress and it is out of the scope of this article, The reader can refer to [51] for more information on that matter.

vCDN Nodes, the dataplane components of the vCDN system, are placed in the MEC becoming closer to the end user, and will hold video content to be delivered to end users. These nodes, which can be considered as distributed caches, are connected to a central controller (**vCDN Engine**) which controls the cache contents of each node, having a general

view of the vCDN topology. This concept is similar to that of SDN, where several SDN switches are connected to a central controller. The vCDN Engine is capable of communicating with the infrastructure's **MANO** in order to deploy nodes in available MEC hosts along the path of movement of users, thus taking advantage of the NFV capabilities.

The proposed framework includes active and passive monitoring capabilities by subscribing to monitoring data in the 5G network (active monitoring) and receiving monitoring parameters from other vCDN framework components without an explicit request (passive monitoring). The main monitoring component of this framework is the **Monitoring Manager (MM)**. It is also able to use the monitoring data to apply algorithms, using the **User Location Predictor (ULP)** and the **Content Placement Planner (CPP)**, that plan ahead where to place content in order to minimize access time and video origin load, while maximizing the cache hit ratio. By using this form of planning, there is no need to load an entire video to a MEC cache, but only the portion that will effectively be consumed while the user is in the vicinity of that MEC cache. This will significantly reduce the load on the core network and video origin while providing the content with a lower latency, since it is placed closer to the end user. Making the analogy with SDN, and considering figure 1, one can compare the vCDN Nodes to SDN switches, the vCDN Engine to a SDN Controller and the ULP and CPP to SDN Applications.

This location-aware system follows a design principle that ''your content will follow you as you move''.

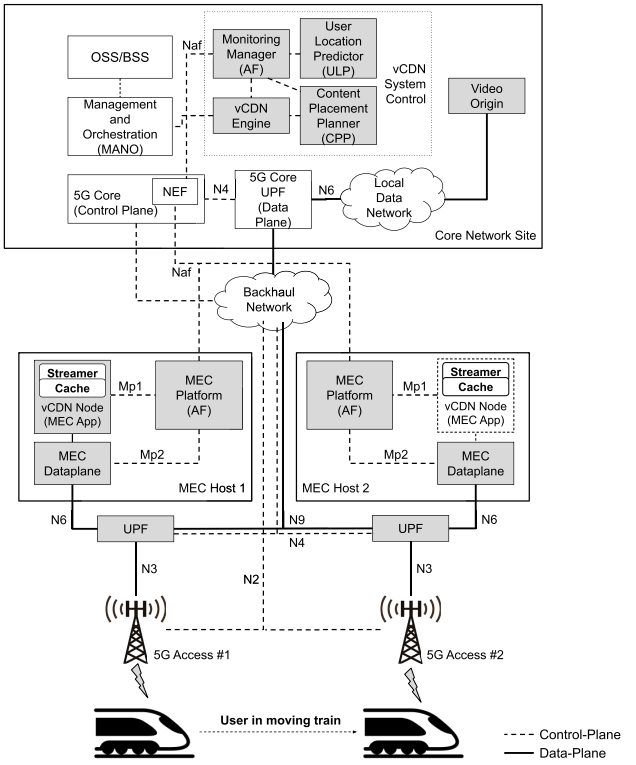

FIGURE 1. vCDN System Framework and its interaction with a 5G network. The MEC hosts are distributed across the movement path of the user. The MEC Platform, which functions as an Application Function, steers the traffic to the MEC Dataplane by interacting with the 5G Core's NEF. The NEF also provides monitoring data to the Monitoring Manager (also a AF), which delivers it to other modules in order to employ various control algorithms.

A. DESIGN PRINCIPLES

The following subsections identify the design principles that gave rise to the proposed framework.

1) ALLEVIATION OF DEMAND ON THE CORE NETWORK AND VIDEO ORIGIN

To minimize the load on both the core network and the video origin, the framework is able to segment content (i.e., a video) and load only the necessary portion in the distributed caches. The system uses a monitoring module to obtain information about the content being consumed by a user in any given time as well as its location and, using a prediction algorithm, predicts what portion of content should be loaded into the distributed caches along the user's path. This means that only the content that is predicted to be consumed is loaded from the video origin and traverses the core network, instead of proactively loading the entire video. Furthermore, the framework implements another important capability that allows the minimization of the amount of traffic that comes from the video origin and crosses the core network: peer-to-peer (P2P) caching. P2P caching enables the distributed caches to load content not only from the video origin but also from other caches. This dramatically diminishes the amount of content that comes from the video origin: given that the distributed caches do not implement content eviction, i.e., the loaded content will remain in cache indefinitely, a particular chunk of content comes from the video origin only once, being then sent back and forth between the peer caches as needed. The information of where the caches should load content from (video origin vs peer cache) comes from a centralized controller with knowledge of which content each distributed cache holds.

2) CONTENT ACCESS TIME REDUCTION

The reduction of content access time is achieved by placing content physically closer to the end users. That is possible by deploying cache nodes in the edge. The system presented in this article takes advantage of MEC technology and its native support in the 5G network. Using MEC and 5G APIs, the system is able to divert selected traffic flows to and from the MEC where the content is placed. Furthermore, the P2P capability also contributes to the content access time reduction, since the peer caches are usually geographically closer to users than the video origin server.

3) CACHE HIT RATIO MAXIMIZATION AND RESOURCE OPTIMIZATION

The resource optimization challenge is addressed by dynamically deploying cache nodes in the distributed MECs and by releasing resources when they are not needed. This is

accomplished by taking advantage of NFV which allows for the dynamic instantiation of network functions in a fully virtualized environment. The proposed framework is able to communicate with a MANO platform to indicate when and where virtual cache nodes should be deployed and to load the content that the user is likely to consume. To further optimize the restricted MEC resources, the framework also scales the cache nodes as the usage increases or decreases. The system's capability to segment content also takes an important role in the MEC resource optimization as only the necessary content is loaded to the MEC storage. Because the content is loaded beforehand, based on a prediction, the cache hit ratio will tend to 100 percent as the system maximizes the probability of a required content chunk being available in a given cache node.

B. INVOLVED ENTITIES

A detailed description of the involved entities of the considered framework is now provided.

1) MEC DATA PLANE

The data plane is the building block responsible for the forwarding of traffic, either from users to MEC applications or between MEC applications, according to traffic rules received from the MEC Platform. According to [52], the data plane can be realized as a Physical Network Function (PNF) or as a VNF. Because our work has a strong focus on virtualization, the MEC data plane was considered as being a VNF.

2) MEC PLATFORM (MECP)

The MECP is responsible for providing an environment where MEC applications can offer MEC services. It can receive traffic rules from MEC applications or the platform manager and configure the datapath accordingly, receive DNS records and configure a DNS proxy/server and host MEC services such as Radio Network Information and Location. The MECP reference architecture is defined in [53]. In [54], the authors discuss the integration of the ETSI MEC platform in the 5G system, being one of the options to place the MECP as an Application Function (AF), which was similarly pursued in this work. The MECP exposes the *Mp1* interface to MEC applications. In order to forward selective traffic flows to the MEC as per application needs, the MECP uses the *Naf* interface, *trafficInfluence* API [55] of the 5G system to influence its datapath configuration. In an NFV environment, the MECP is seen as a VNF, as for [52].

3) MANAGEMENT AND ORCHESTRATION (MANO)

The MANO block encompasses the MECP manager and the MEC orchestrator entities. Therefore, the main responsibility of the MANO block is to manage the elements of the MECP, application and requirements rules and, finally, the lifecycle of applications. It is also responsible for the deployment and configuration of MEC applications [53].

4) USER LOCATION PREDICTOR

The ULP is an analytical component which can integrate different algorithms to make use of geographical data obtained from the Monitoring Manager and provided by the 5G Core APIs. This can be added with map data, MEC host location and other available parameters, to identify the location and/or predict the mobility paths of mobile users. Depending on the scenario or service, the geographical data may be aggregated in groups of users for scalability purposes (e.g., the module could predict that a group of users will enter a certain geographical region and scale certain components beforehand in order to accommodate the increase of active users). After predicting the path of users, the ULP sends that prediction to the Monitoring Manager, where it can be used by other modules. Our framework is flexible enough to encompass different kinds of algorithms, considering tools such as Machine Learning, but these are out of scope from the paper.

5) CONTENT PLACEMENT PLANNER (CPP)

This component has the knowledge regarding available MEC hosts and combines information about active user sessions and user location prediction (the latter two obtained from the Monitoring Manager) to plan which content should be placed at a given moment in each of the MEC hosts available along the users itinerary. The CPP sends the planning suggestion to the vCDN engine which, in turn, will enforce it.

6) vCDN NODE

The vCDN node is composed by a cache, for local storage of content, and a module called the Streamer, which can establish connections with the video server and the client. This element is deployed as a MEC application VNF, thus using the *Mp1* interface to communicate with the MECP. It also sends monitoring data to the Monitoring Manager such as active sessions, video information and cache status. This module is the point of contact for users, acting as a proxy. However, since all traffic management is made dynamically by using MEC and 5G, there is no need to configure a proxy in the use's device, which is transparently connected to it.

7) MONITORING MANAGER (MM)

This is the central monitoring component of the vCDN system. It subscribes to network information in the 5G core network via the *MonitoringEvent* API [56] and also receives monitoring parameters from other framework components. In the 5G system, this module is seen as an Application Function. The Monitoring Manager has a message broker that allows multiple framework components to subscribe to information in a plug and play manner, meaning that new components can be added to the framework and use the available data with minimal effort.

8) vCDN ENGINE

The vCDN engine is the central control plane component of this system. Similarly to a SDN controller, the engine has a general view of the entire vCDN system, such as which nodes are deployed and the cache contents of each node. The module enforces content placement suggestions received by the CPP, triggering the nodes to load the necessary content. Because the vCDN engine has the knowledge of which content is placed in each node, besides triggering the nodes to load content from the video origin, it can also signal the nodes to load content from other nodes (peer-to-peer caching). When a content placement suggestion is received, the vCDN engine checks if a node is already deployed in the suggested MEC host and, in case it is not, it signals the MANO to instantiate the necessary node. This module is the first point of contact for nodes that have been instantiated and it provides them with configuration parameters.

9) VIDEO ORIGIN

This component holds all the videos available in the system and it receives requests from users or vCDN nodes. It supports H264, thus delivering video in chunks via byte range requests, allowing it to deliver only the requested range of a particular content. It also sends information about user sessions and video information to the Monitoring Manager.

The framework presented in this work extends the previous framework in [9] by incorporating the monitoring module, capable of obtaining monitoring data from the 5G core APIs through a standard interface, as well as holding and delivering monitoring data to other framework components. Furthermore, an extensive set of evaluation trials was conducted and the results extracted, considering a real life mobility scenario emulation. The vCDN Node was separated into its streamer and storage components, allowing for a more optimized usage of resources and its implementation was revised in order to be able to accommodate multiple simultaneous user connections.

The framework presented in this article can be deployed in 4G networks and Non-Standalone 5G networks (which both use the 4G Core Network) and Standalone 5G networks (which uses the 5G Core Network). In order to retrieve monitoring data from the network, the MonitoringEvent API [56] is used, which is available in both 4G and 5G. As for the network traffic influence, the 5GC offers the TrafficInfluence API [55]. As for the 4G's EPC, our framework is able to influence the traffic by using the Packet Flow Description Function (PFDF) and the Traffic Steering Support Function (TSSF) [57]. Summarizing, our framework targets a deployment in a Standalone 5G network but it is flexible enough to be deployed in a network that uses the EPC (4G or Non-Standalone 5G).

IV. FRAMEWORK BEHAVIOUR

The signalling chart presented in figure 2 represents the signalling between the different modules of the proposed architecture, and the MEC and 5G core components, addressing

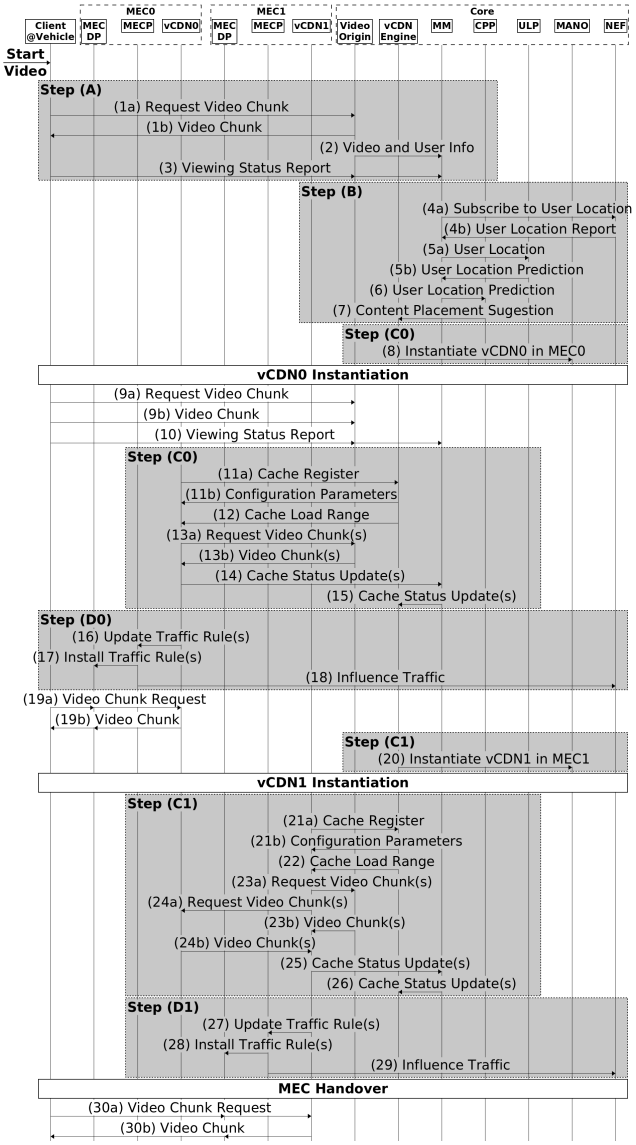

FIGURE 2. vCDN System procedures signalling chart.

the case of a single user starting to watch a video made available by the operator. That user is travelling in a vehicle (e.g., a train) and it will view a video using a web client (HTTP-based video). The framework was designed to react to external triggers, such as a user starting to view a video. The client video player periodically reports the viewing status with information about the current viewing time. For simplicity, the MEC App discovery and registration procedure, as well as module subscription procedures both in the 5G Core and in the Monitoring Manager, are not considered. Also for the sake of simplicity, only two MECs are represented, as well as the signalling for a single user. However, the system supports multiple users as will be shown in section V.

Stage (A): Detection of a new User

1) A user starts watching a video. The video client requests a chunk to the video origin which returns it, since there are no deployed vCDN Nodes.

2) The video origin extracts information about the video being watched (e.g., total size, duration and aspect ratio) and proactively sends that information to the Monitoring Manager, alongside the user session information (association between a user session and the video being watched).
3) When the video client starts playing the video, it periodically reports the viewing status (e.g., current viewing time) to the video origin. The origin then relays that information to the Monitoring Manager. The viewing status report period is a configurable parameter.

Stage (B): Subscription to Monitoring Data, User Movement Path Prediction and Content Placement Calculation

4) After the Monitoring Manager receives information that a new user session is in place, it communicates with the 5G core network's NEF, via the *Naf* interface, in order to subscribe to the user location. If the subscription is successful, the NEF will periodically report the user location back to the Monitoring Manager which will store it. The location report period can be configured by the requesting module.
5) The ULP, which subscribed to user location information, receives the location report from the Monitoring Manager. With that information, the ULP executes an algorithm in order to predict the user's itinerary. That prediction will be sent back and stored by the Monitoring Manager. Despite not being in the scope of this article, the system also encompasses the possibility of dynamic adjustments and re-calculation of the itinerary, if the used prediction algorithm supports it.
6) The CPP, which subscribed to the location prediction information in the Monitoring Manager, receives the information and, combining it with the knowledge of available MECs, computes the content that should be available in each MEC along the user's itinerary.
7) The CPP sends a *Content Placement Suggestion* message to the vCDN engine containing the content portion to be placed in the distributed MECs along the user's itinerary.

Stage (C0): Instantiation and Cache Loading of vCDN0

8) The vCDN engine analyses the content placement suggestion provided by the CPP and verifies if the next MEC in the user's path already has an instantiated vCDN Node (vCDN0). If it does not, the vCDN engine sends a command to the MANO in order to instantiate the missing node. In the meantime, the vCDN Engine, which has knowledge about the cache contents of every vCDN Node in the system, verifies if the content to be loaded is present in any peer cache or if it has to be loaded from the video origin. This information is saved to be sent to the vCDN0 when it becomes available.
9) During the vCDN0 instantiation, the video continues playing and the video client continues to request video chunks to the video origin, which delivers them.
10) The video continues playing and the video client keeps sending viewing status reports to the video origin, which in turn relays the information to the Monitoring Manager.
11) When the instantiation of the vCDN0 is complete, it sends a cache register request towards the vCDN Engine, replying with the necessary configuration parameters such as the Monitoring Manager's IP address and the time period for cache status reporting.
12) After the vCDN Engine sends the configuration parameters to the vCDN node, it sends a command to the node with the information about the content that it must load to its cache, saved in step 8, and where to obtain the content (video origin or peer caches). In the scope of this flow chart, the vCDN0 must retrieve all the necessary content, indicated by the vCDN Engine, from the video origin.
13) The vCDN0 requests the content range from the video origin, which delivers it.
14) After the vCDN0 loads the content that the vCDN Engine requested, it sends a *Cache Status Update* to the Monitoring Manager with information about the newly cached content. This report is periodically sent, but it only happens when new content is either cached or released, meaning that the report follows an incremental reporting approach.
15) The vCDN Engine, which is a subscriber of Cache Status information, receives the cache status update and stores that information.

Stage (D0): Video Traffic Steering to vCDN0

16) Now that the vCDN0 has loaded the necessary content, it sends *Update Traffic Rule(s)* message(s) to its MEC Platform, via the *Mp1* interface, so that the MEC dataplane can be configured to forward video requests originated by the user and destined to the video origin, to the vCDN0.
17) The MEC Platform sends *Install Traffic Rule(s)* message(s) to the MEC dataplane, via the *Mp2* interface, to implement the traffic rules that will forward the desired traffic to the vCDN0 (MEC Application). After this step, the MEC dataplane is configured to handle the vCDN traffic.
18) After the configuration of the MEC dataplane, it is necessary to influence the user's video traffic in the 5G network so that said traffic can be forwarded to and from the MEC dataplane. In order to accomplish that, the NEF's *Traffic Influence* API is used. The MEC Platform (an Application Function in the scope of this article) uses the *Naf* interface to send a request to the NEF to forward all the video requests originated by the user and destined to the video origin to the MEC dataplane, triggering the 5G core to reconfigure the UPF closest to the MEC host.
19) The web client requests a new video chunk to the video origin. This time, when the request reaches the

UPF closest to the MEC host, it forwards the request to the MEC0's dataplane. The MEC dataplane, which was previously configured, forwards the user request to the vCDN0. In turn, the vCDN0 which contains the requested chunk in its cache, answers with the chunk without communicating with the video origin.

Stage (C1): Instantiation and Cache Loading of vCDN1

20) The user is now approaching a point where it will be handed over to an access node that is served by MEC1, thus performing a MEC handover. The vCDN Engine sends a command to the MANO in order to instantiate a vCDN node (vCDN1) in MEC1.
21) The vCDN1's instantiation is complete and it sends a *Cache Register* request to the vCDN Engine, which answers with configuration parameters.
22) After sending the configuration parameters, the vCDN Engine sends a *Cache Load Range Request* to the vCDN1 with information about the content that it must load in its cache. In this message, the vCDN Engine indicates where the vCDN1 should obtain the necessary contents.
23) The vCDN1 requests video chunks to the video origin, as indicated by the vCDN Engine. The video origin then returns the requested content.
24) Because a portion of the content that must be loaded by vCDN1 was already loaded by vCDN0, the peer-to-peer caching capability of this system is used in order to save bandwidth in the core network as well as to reduce the load on the video origin. The vCDN Engine indicated to the vCDN1 that it must load a portion of the content from vCDN0. The vCDN0 then returns the requested content.
25) After the vCDN1 loads the content indicated by the vCDN Engine, it sends a *Cache Status Report* message to the Monitoring Manager.
26) The vCDN Engine receives the cache status update, as it subscribed to that information in the Monitoring Manager.

Stage (D1): Video Traffic Steering to vCDN1

27) Now that the vCDN1 has loaded the content that the users will consume while being served by vCDN1, it sends an *Update Traffic Rules* message to its MEC Platform (MEC Platform in MEC host 1), through the *Mp1* interface, so that when the user is handed over to this MEC all the video requests originated by the user and destined to the video origin will be forwarded to the vCDN1.
28) The MEC Platform configures its dataplane to forward traffic as instructed by the vCDN1 using the *Mp2* interface.
29) The MEC Platform in MEC1 then sends a *Influence Traffic Message* to the 5G Core's NEF so that the desired traffic is offloaded to the MEC1's dataplane.
30) The MEC handover takes place and the users are now connected to the vCDN1 when they make requests to the video origin. The video client requests a video chunk and the vCDN1 returns it without communicating with the video origin.

Using this mechanism, the only cache misses that occur are in the beginning of the video on demand session, when a user starts viewing a video, and it is not yet in the cache. This means that, if a client views a video from the beginning until the end, the cache hit ratio tends to 100 percent with the increase of the video duration in the considered scenario. Due to the ability that the framework has to move content from peer-to-peer, the maximum volume of data delivered by the video origin is approximately equal to the size of all the videos available in the video origin, since after a portion of content is loaded in any vCDN node, that content can be provided to other vCDN nodes in the system, assuming that the content will remain in the vCDN node's caches for the entirety of the user's journey (e.g., through an entire train trip).

V. PROOF OF CONCEPT DEPLOYMENT

The following section presents the test scenario used to evaluate the proposed framework. It also provides insight on implementation aspects and simplifications made in order to have a working proof-of-concept.

A. EVALUATION SCENARIO

To perform evaluation tests to the proposed framework a train journey was considered between two cities in Portugal, Aveiro and Lisbon, with an approximate extension of 266km (165 miles) of railway. The train is equipped with a 5G Costumer Premises Equipment (CPE) with two interfaces: (i) a Wi-Fi interface that commuters will connect to and (ii) a 5G interface, that will act as the Internet Gateway for connected clients. The commuters will enter and leave the train at train stations and will start to consume video content provided by the network operator some time after boarding the train. Various 5G access nodes (gNBs) were distributed along the path of movement of the train as illustrated in Fig. 3, with each connected to one of five available MECs. For simplification, the gNB distribution was based on real 4G eNB locations. The set of eNB locations where extracted from CellMapper,[1] considering a set of 4G cells of a Portuguese Network Operator along the considered path. An average movement speed for the train of 110kmh/h (68 mph) was used, with a total journey time of approximately 2 hours and 27 minutes. A network operator offers a mobile VoD service to its 4/5P subscribers and these subscribers are the considered users for testing purposes. Hereafter, a commuter that is a subscriber of the considered network operator's 4/5P plan is referred to as user. The estimation of the number of users that enters or leaves the train at each train station was performed using the following method:

1) The online ticket sales of a Portuguese train operator was monitored during the entirety the considered

[1]CellMapper: https://www.cellmapper.net

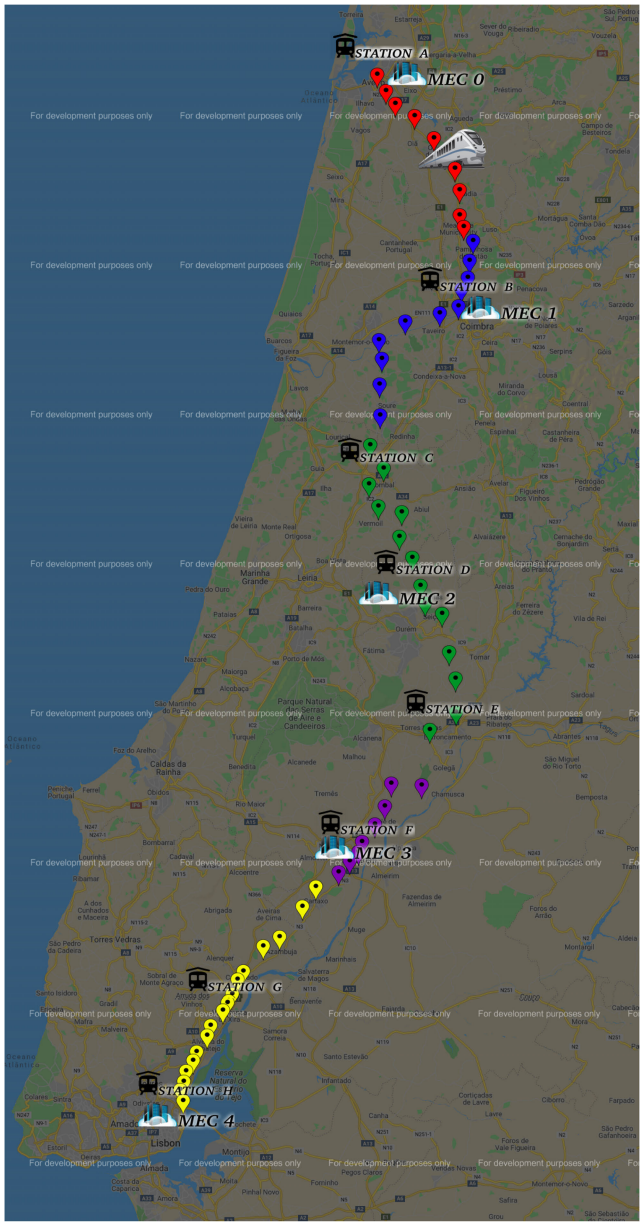

FIGURE 3. Access Node and MEC host Distribution in the considered scenario.

journey (analysed on November 21, 2019 during morning rush hour, thus data is related to a pre-COVID-19 usage without any passenger seat occupation restrictions, which makes for a denser scenario). This provides approximate information about the occupied seats and thus, the number of commuters inside the train on the stretch between two stations;

2) For all the commuters inside the train, an estimate was made in order to obtain the probability of a commuter being a user. To obtain this probability, both the probability of a commuter being a client of the considered network operator and the probability of it being a subscribed to a 4P or 5P plan where extracted from a market share report from the Portuguese Telecommunications Regulator: Anacom [58]. The report states that 4/5P subscriptions account for around 47 percent of all subscriptions and that the market share of the considered operator is approximately 40 percent. With these two values, the combined probability was applied, obtaining a probability of 19 percent for the probability of a commuter being a user;
3) In order to obtain the number of commuters that leave the train at each station, an estimation was performed, for simplification purposes, based on the population density of the city where each train station is located. The estimation was performed in two steps: first, subtracting the number of commuters exiting the train from the total number of current travelers. Then, subtracting this value from the initial number of commuters (i.e. before stopping at the station. Table 1 presents the number of inhabitants of each city and its corespondent percentage. The probability of a commuter leaving the train at each train station is then correspondent to the population percentage. Note that Lisbon is not represented as we consider that all passengers leave at that final station.

TABLE 1. Inhabitant percentage of each of the considered cities.

St.	City	No. Inhabitants	Percentage
A	Aveiro	78450	14
B	Coimbra	143396	26
C	Pombal	55217	10
D	Ourém	45932	9
E	Entroncamento	20206	4
F	Santarém	62200	12
G	Vila Franca de Xira	136886	25

4) Finally, the number of commuters entering the train at each station was obtained by subtracting the number of commuters inside the train after it reaches the station to the number of commuters that leave the train at the same station. Then, that value is subtracted to the number of commuters that leave the station inside the train.

Using the above estimated probabilities, table 2 presents the number of commuters entering and leaving the train at each station and, out of those commuters, how many are users. The *No. Commuters* column represents the number of passengers inside the train after it leaves the station.

Table 3 presents the timing of considered events of the experience, namely the time that the train arrives at each train station and handover times, all being relative to the start time (t=0s).

B. PoC SIMPLIFICATIONS

Our proof of concept focused on the development of the new modules for enabling the dynamic video content placement, along with the MEC platform, and a SDN/NFV substract that handles the NFV MANO, as well as virtual links management. The deployment architecture is presented in Fig. 4.

TABLE 2. Number of Commuters and Users Entering and Leaving the train at each train station.

St.	No. Commuters	Entering	Leaving	Users Entering	Users Leaving	Total Users
A	302	302	0	57	0	57
B	251	28	80	5	15	47
C	293	67	5	12	5	55
D	310	41	24	8	4	59
E	364	65	11	12	2	69
F	372	49	41	9	7	70
G	374	95	93	18	18	70
H	0	0	374	0	70	0

TABLE 3. Scenario Timings.

Event	Runtime (s)
START/Station A	0
Handover to MEC1	1250
Station B	1860
Handover to MEC2	2840
Station C	3480
Station D	4500
Station E	5640
Handover to MEC3	5379
Station F	6720
Handover to MEC4	6870
Station G	7980
Station H/END	8760

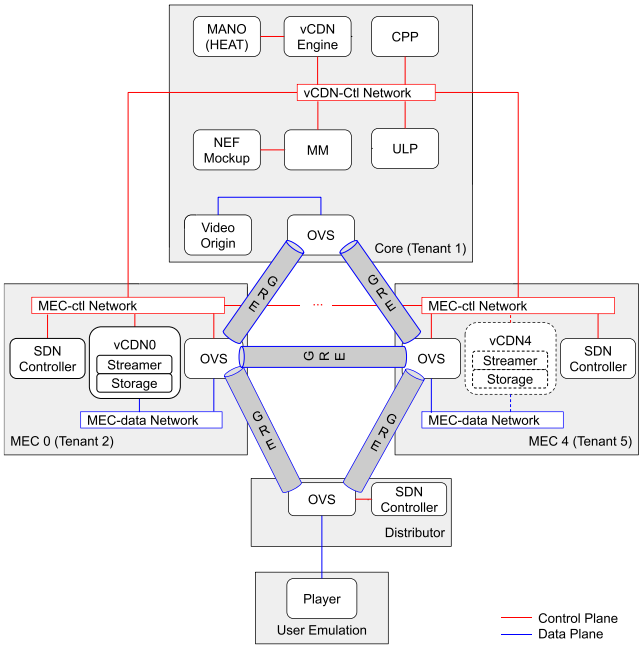

FIGURE 4. vCDN System architecture used for testing.

The next subsections describe the simplifications made to some of the architectural functional blocks.

1) NETWORK EXPOSURE FUNCTION

The presented system leverages on the 5G core network Northbound APIs, to obtain user location data and influence user traffic. As such information is provided by the NEF, we focused only on these mechanisms of this specific element, in which the information generated by the NEF emulates the location of a train moving along the defined region. The *MonitoringEvent* API was implemented in order to provide the emulated train location to subscribers. The traffic influence part of this framework was implemented using SDN, meaning that the NEF does not take part in the traffic redirection.

2) SOFTWARE DEFINED NETWORK TRAFFIC FLOW DISTRIBUTION

The mechanisms provided by our framework are mainly deployed considering the MEC platform and a single element from the 5G Core network (i.e., the NEF as explained above). As such, our PoC did not deploy a Radio Access Network, relying instead on a network deployment virtualized over a the same cloud environment where our modules were deployed. A SDN-based mechanism was implemented to simulate MEC handovers. As the event related with an handover between two access nodes that are connected to different MECs is the main radio-related information that we need to trigger our procedures (as shown in IV, only the handover between two border cells is of concern, which was simulated using SDN. As shown in Fig. 4, the Distributor switches the path between the different MECs. The SDN controller receives a message from the NEF (component that emulates the train's movement) in the handover moment, triggering the path switch from MECx to MECy, as an example. The Distributor establishes a GRE tunnel with the current MEC. The MEC handover is processed as follows:

1) The Distributor SDN Controller receives a command from the NEF in order to switch the serving MEC.
2) The Distributor SDN Controller deletes the flow entries of the switch that establish a tunnel to the current MEC and waits 10.5ms before installing the new ones, which will establish the tunnel to the next MEC. According to the MGMN 5G white paper [59], in an handover situation, the data plane interruption time should not surpass 10.5ms in order to maintain connectivity transparency. The delay introduced between the deletion of the flow entries and the installation of new ones has the purpose of simulating this handover delay.
3) After the new flow entries are installed in the switch and the tunnel is established, the user is now connected to the next MEC.

3) END USER

With the objective of emulating the train movement, the end user was implemented as a software that emulates the behaviour of a user watching a video on a train. The number of users that are using the VoD service at any given time was presented in Table 2. When a user enters a train it does not immediately start watching a video from the set available in the video origin. For that reason, a random time between the user entering the train and the video starts playing was assumed. The end user script uses one process per user inside

the train and it guarantees that the video chunks are requested at the appropriate time.

C. TEST DEPLOYMENT IMPLEMENTATION

All the modules represented in Fig. 4 are virtual machines, with specifications presented in table 4, running on Openstack (Openstack Queens). For the dynamic deployment of vCDN Nodes (playing the MANO part), the Openstack HEAT project[2] was used. The Core network modules and the MEC modules where implemented in different tenants thus emulating different points of presence. For the video origin, the Streama software[3] release 1.6.0-RC9 was used. Ten 1080p videos with a framerate of 30 fps were loaded and made available to the users with total video duration ranging from 10 to 40 minutes. The combined size of all the available videos is 6.8 GB.

TABLE 4. VM Resource Allocation for vCDN System Components.

Component	No. vCPUs	RAM (GB)
Skeleton NEF	1	2
CPP	1	2
ULP	1	2
vCDN Engine	1	2
SDN Controller	1	2
vCDN Node Storage	1	2
MM	2	4
Video Origin	2	4
OVS	2	4
vCDN Node Streamer	2	4
User Emulation	4	8

The Monitoring Manager is composed by a module that enables it to subscribe to monitoring data in the NEF and by a message broker. The broker used in this deployment was Kafka[4] Release 3.6.0. Every module that connects to this broker to either produce or consume monitoring data must implement a specific Kafka producer or consumer, respectively. In order to optimize the transactions with the broker, all data was serialized using Apache Avro.[5]

The vCDN Node Streamer and Storage part were implemented as two separate modules that will be deployed by the system's MANO platform. This modules connect to the MEC's internal network for communication to the outside and between each other.

The distributor was deployed using a SDN Controller (RYU SDN Controller[6] in this case) and an associated SDN Switch (OVS[7] Version 2.7). The distributor module receives traffic from the User Emulation and encapsulates it in a GRE tunnel, established between the distributor and the current MEC. The distributor's SDN controller exposes an API that enables external modules to trigger a MEC handover, which will be used by the skeleton NEF, for simulating the user movement.

The MEC Platform was implemented using a SDN Controller, exposing the MEC's Traffic Rules API developed within the scope of this work. This SDN Controller controls a SDN Switch that provides a dynamically configurable MEC dataplane. Two internal networks were created, *mec-ctrl* and *mec-data*, where MEC applications will connect their control and data interfaces, respectively.

The ULP implemented a simple algorithm that takes into consideration the current location of a user and it computes an estimate for the user's handover time to the next MEC.

The CPP reads the location prediction provided by the ULP and it calculates the video chunks that should be available at each MEC, taking into consideration the current viewing time of each video being watched by each user.

The skeleton NEF was implemented in order to emulate a real 5G environment. The Location Reporting part of the 3GPP-compliant *MonitoringEvent* API was implemented allowing the Monitoring Manager to also use a standard interface. This component provides location information at the level of cell identifier (Cell ID). This component also implements an emulation of the user movement based on the real train journey presented in Fig. 3. When the emulated train reaches a location when there is a MEC handover, a message is sent to the distributor triggering that same handover.

The user emulation was implemented in a way that it allows multiple emulated users to be simultaneously connected and requesting video chunks. The emulation considers that users will watch a video until it ends or until the user leaves the train. If the user stays and the video ends, a new video, that the user has not watched yet during the trip, is selected and the video playback starts again. The video chunk size considered for this PoC was of 10MB. This value can be configured and optimized but that optimization falls out of the scope of this work.

Because all the tenants were placed in the same geographical region (the same datacenter located at our research institute's premises), a fixed RTT delay of 50ms between the MECs and the core tenant was introduced so that the MEC to core latency could be emulated.

Table 4 presents the computational resources used by each component of the test deployment.

VI. EVALUATION RESULTS

The following sections present the results of various tests conducted to the system, namely vCDN Node Instantiation times, the impact that multi users have on the framework, the throughput generated by the framework for both data and signalling, and the cache hit ratio in the considered scenario.

A. vCDN NODE INSTANTIATION

The vCDN Node instantiation time was measured by instantiating a full node (video streamer plus the storage needed for videos) and measures the amount of time that each component takes to be available after a deployment message is

[2]Openstack Heat: https://wiki.openstack.org/wiki/Heat
[3]Streama: https://github.com/streamaserver/streama
[4]Apache Kafka: https://kafka.apache.org/
[5]Apache Avro: https://avro.apache.org/
[6]RYU SDN Controller: https://osrg.github.io/ryu/
[7]OVS: https://www.openvswitch.org/

sent to the virtualization platform controller (i.e., the HEAT framework). The tests were conducted 15 times and present a confidence interval of 95 percent. Table 5 presents the average instantiation time for each component of the vCDN node and also for the full node. This vCDN Node instantiation was presented in a previous work [60]. The reason why the instantiation time of the full node is not the sum of the time it takes to instantiate each component separately is due to the fact that the instantiation occurs almost simultaneously. The vCDN Engine needs to obtain the IP address of the storage part of the node so that it can provide it to the streamer, and it only sends a command to instantiate the streamer when it receives that IP address.

TABLE 5. vCDN Node Instantiation and Configuration Time.

Module	Instantiation Time (s)
Storage	157.2 +/- 1.7
Streamer	160.9 +/- 1.0
Storage + Streamer	178.1 +/- 1.4

B. MULTI-USER IMPACT ON QoE

In order to determine the maximum number of users that the vCDN Nodes can serve while maintaining the QoE, a test was conducted where the number of users connected to a single vCDN Node was progressively increased, until the average chunk delivery throughput dropped below a reference value. That reference value was extracted by playing a video in the web client and progressively reducing the throughput of the video origin server until the video client started buffering. It was determined that, for a 1080p video, buffering occurred when the throughput dropped below 5 Mbps, making this the reference value. Fig. 5 shows the result of this test, presenting the average delivery throughput in function of the number of connected users. The results where obtained by launching a new user every second and taking 100 measurements when all the considered users were connected. The results present a confidence interval of 95 percent. Analysing the graph, it can be seen that the intersection of the QoE Reference Throughput with the Chunk Delivery Throughput happens for a number of users of about 125. This means that a single vCDN Node is capable of serving 125 simultaneous users, viewing a 1080p video, while maintaining its perceived QoE.

C. DATA DELIVERY THROUGHPUT

This test aims to measure the throughput generated by the vCDN System to deliver video over the course of the experiment. Three different scenarios where considered to illustrate the advantages of the proposed system:

1) Proactive Caching Without our Proposed Framework (hereafter refered to as Proactive Caching): A scenario where the vCDN Nodes are placed in the considered MEC hosts but act as proactive caches. This means that the caches will load the video from the point that the user is watching until the end as soon as a user starts requesting it. This scenario does not consider the control part of our proposed system.
2) Proposed System Without Peer-to-Peer: A scenario where our proposed framework is in place, but without peer-to-peer caching capabilities.
3) Proposed System: A scenario where our system is fully in place.

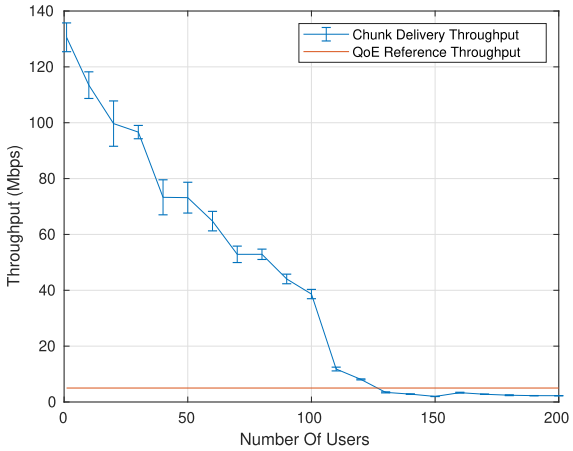

FIGURE 5. Average Chunk Delivery Throughput measured by each user.

The following subsections present the results obtained from a randomly selected run. The tests consider that the initial vCDN Node (vCDN0) is already instantiated and only the next vCDN Nodes will be dynamically instantiated.

1) PROACTIVE CACHING

This test measured the throughput generated by each deployed vCDN Node while delivering video data to the users. The results of a selected run can be seen in Fig. 6, alongside information about the number of active users at each moment. It can be seen that users receive data from the distributed vCDN Nodes placed in their movement path. The total amount of data delivered to users was of 276.8 GB in this scenario. The drops in the graph of Fig. 6 represent users leaving the train and happen when it stops at a train station. In addition to the measurement of the throughput generated by each vCDN Node, the throughput generated by the video origin and thus, traversing the core network, was also measured and it is presented in Fig. 7.

The graph shows that there are moments in time where are spikes of data being sent. These spikes correspond in time to MEC handovers, since when a user transitions to a different node, that node will load to its cache the video from the point that the user is watching, until the end of the video. The total amount of video data that came from the video origin and thus, traversed the core network was of 38.3 GB. This represents 13.9 percent of the total video data delivered to the users.

2) PROPOSED SYSTEM WITHOUT PEER-TO-PEER

The measurements for this test were extracted as previously mentioned but with the proposed system running without

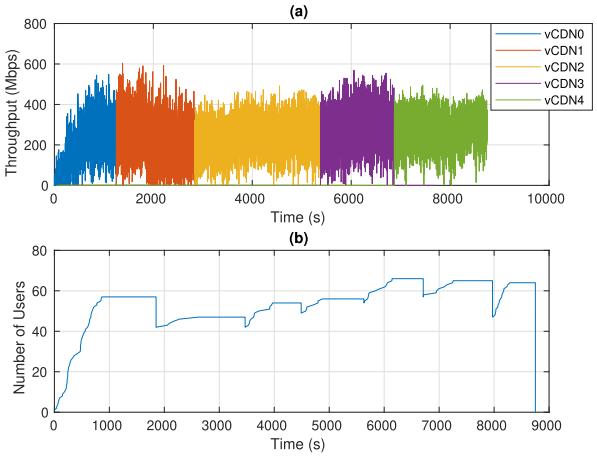

FIGURE 6. (a) Throughput generated by the distributed vCDN Nodes delivering video chunks to users (Proactive caching scenario); (b) Number of active users during the course of the experience.

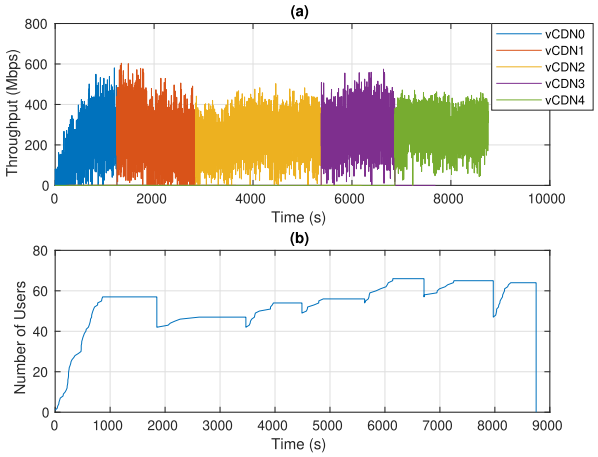

FIGURE 8. (a) Throughput generated by the distributed vCDN Nodes delivering video chunks to users (Proposed System without peer-to-peer); (b) Number of active users during the course of the experience.

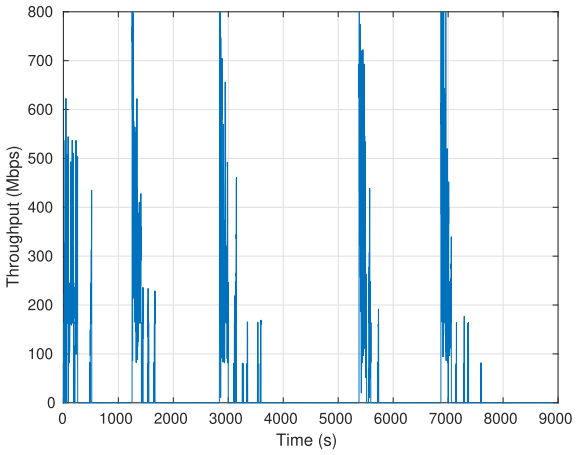

FIGURE 7. Throughput generated by the video origin when delivering video chunks (Proactive Caching).

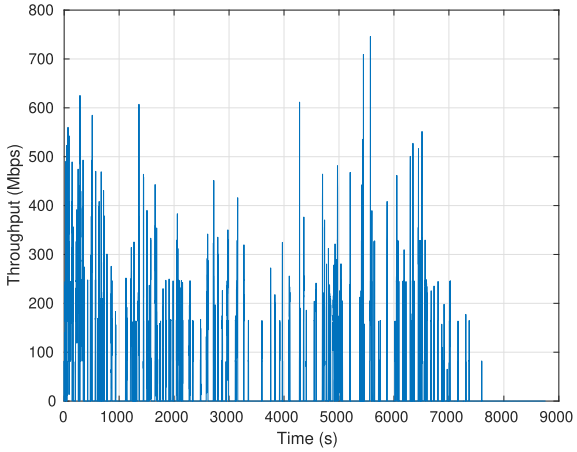

FIGURE 9. Throughput generated by the video origin when delivering video chunks (Proposed System without peer-to-peer).

peer-to-peer caching capabilities. The generated throughput of each vCDN Node as well as the number of active users at each moment (equal to the test above) are shown in Fig. 8. It can be observed that the values are very similar to the one obtained from the previous subsection, meaning that the introduction of the location aware control part of the system does not negatively influence the rate at which users receive video data. The total amount of data delivered to users by vCDN Nodes was of 276.9 GB.

Fig. 9 shows the throughput generated by the video origin and consequently passing through the core network. It can be seen that the behaviour is different than in the previous test since, instead of loading an entire video after a user transitions to a vCDN Node, the control part of the proposed framework signals the nodes which portion of content they need to load to their cache as the users are moving. The total amount of data originated at the video origin was of 35.8 GB, composing 12.9 percent of the total data delivered to users and 1.0 percent less than in the previous scenario. This small reduction in comparison with the previous scenario is due to the fact that in every load command received by the vCDN Engine, the data is still retrieved from the video origin. The introduction of the P2P capability will help reduce the amount of data that comes from the video origin.

3) PROPOSED SYSTEM

Fig. 10 shows throughput generated by the vCDN Nodes while delivering video data to users as well as the number of active users at each moment, in a selected run. Analysing the graph it can be seen that it is similar to the one in the previous scenario, meaning that the introduction of the peer-to-peer capability in the system did not impact the rate at which data is sent to users. In this scenario, 276.9GB of video data were delivered to users. Fig. 11 shows the throughput generated not only by the video origin but also the throughput generated between each storage node of the system, since in this scenario they exchange data between each other. Analysing the graphs presented, it can be seen that

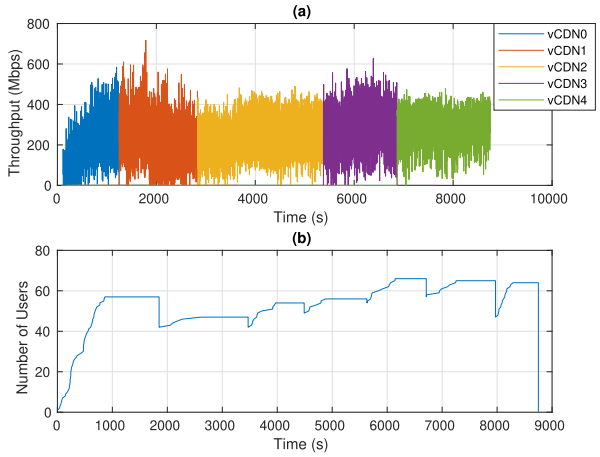

FIGURE 10. (a) Throughput generated by the distributed vCDN Nodes delivering video chunks to users (Proposed System); (b) Number of active users during the course of the experience.

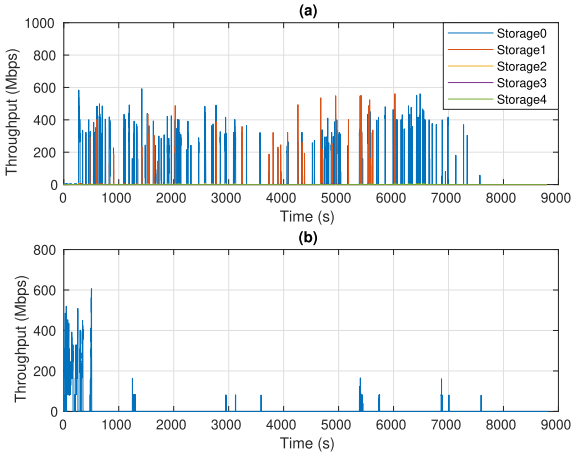

FIGURE 11. (a) Throughput generated by the various vCDN Storage Nodes exchanging data between each other (Proposed System); (b) Throughput generated by the video origin (Proposed System).

the amount of data that comes from the video origin has been dramatically reduced and condensed at the initial moments of the test, since after a video chunk is loaded by any of the available vCDN Nodes, the node itself can provide the chunk to peer nodes, not needing to retrieve it from the video origin again. This lead to a total amount of video data traversing the core network of 8.2 GB, just 3 percent of the total data delivered to users and 10.9 percent less than in the scenario considering proactive caching and 9.9 percent less than in the scenario considering our proposed framework without P2P.

D. CACHE HIT RATIO

The following section presents the cache hit ratio results for the three scenarios considered earlier.

1) PROACTIVE CACHING

Fig. 12 presents the evolution of the cache hit ratio along the course of the experiment for the scenario of proactive

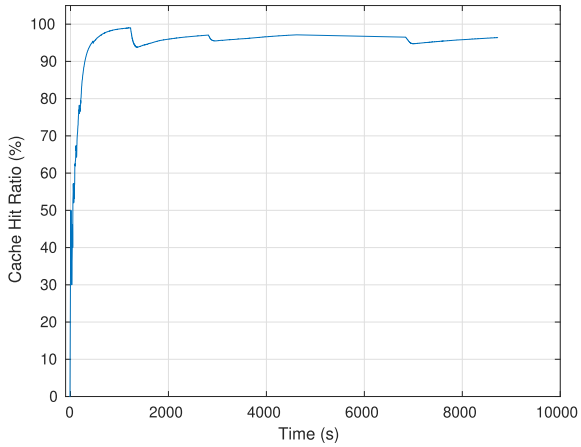

FIGURE 12. Cache Hit Ratio evolution over time (proactive caching scenario).

caching. The cache hit ratio drops can be identified in three time moments, corresponding to the timing of MEC handovers, since when users transition to a new vCDN Node, their cache is empty and it will only start to load necessary data afterwards. The final cache hit ratio value for this scenario was of 96.4 percent.

2) PROPOSED SYSTEM WITHOUT PEER-TO-PEER

Fig. 13 presents the evolution of the cache hit ratio over time for the scenario where our proposed system is in place but with its peer-to-peer caching capabilities disabled. It can be seen that the cache hit ratio is maintained close to 100 percent for about three quarters of the total run time. The value of the cache hit ratio at the end of the test was of 99.8 percent.

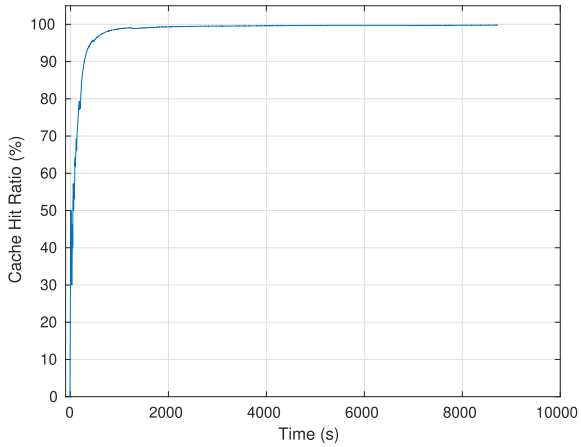

FIGURE 13. Cache Hit Ratio evolution over time (proposed system without peer-to-peer caching scenario).

3) PROPOSED SYSTEM

The last cache hit ratio test was performed with the full proposed system working and the result for a selected run is presented in Fig. 14. The profile of the hit ratio is similar to the one in the scenario where the peer-to-peer caching

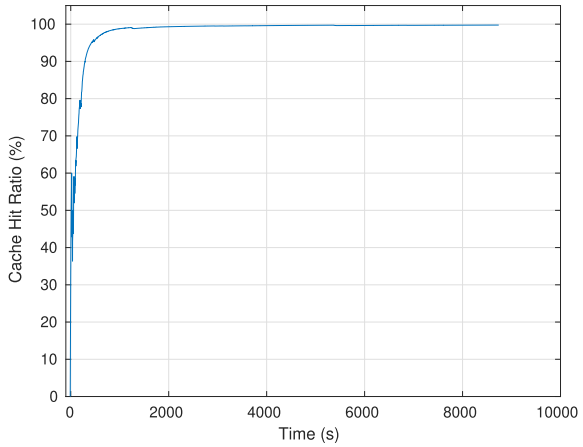

FIGURE 14. Cache Hit Ratio evolution over time (full proposed system scenario).

capability is disabled meaning that the feature has no impact in the overall cache hit ratio. The final cache hit ratio value for this test was of 99.8 percent.

E. SIGNALLING IMPACT

The following section aims to assess the impact of the signalling data introduced by our proposed system. The following subsections present the signalling data generated by running the tests for each of the considered scenarios.

1) PROACTIVE CACHING

In this scenario the only signalling involved is the signalling between users and the vCDN Nodes, requesting chunks and reporting the viewing status. The throughput profile of this data is presented in Fig. 15. In the graph it can be seen that the average throughput fluctuates as the number of active users increases or decreases. The test generated 3.0 GB of signalling data.

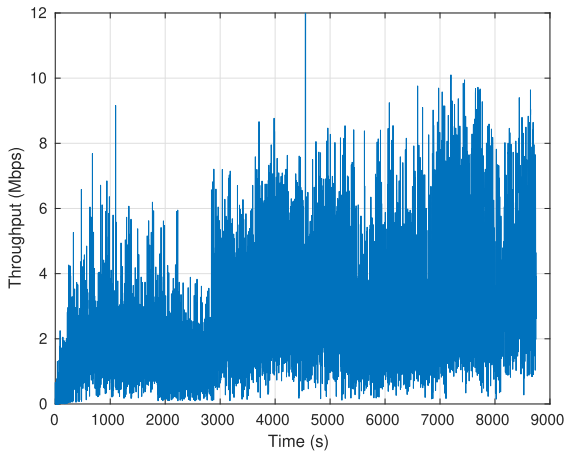

FIGURE 15. Signalling Data generated by the users in a scenario of proactive caching.

2) PROPOSED FRAMEWORK WITHOUT PEER-TO-PEER

In this scenario, there are other components generating signalling data such as the Monitoring Manager, vCDN Engine, CPP and ULP. Fig. 16 presents the throughput graphs of each component.

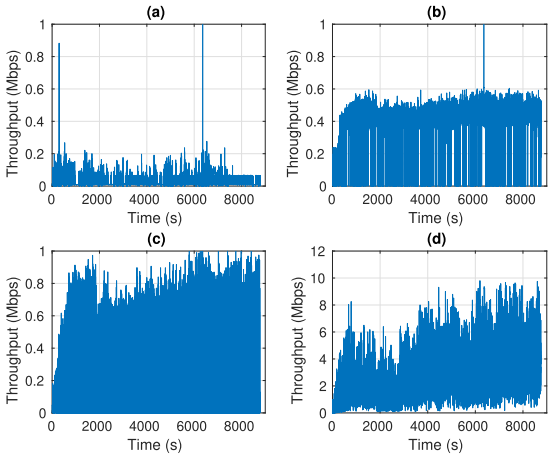

FIGURE 16. Signalling data generated by (a) vCDN Engine; (b) Monitoring Manager; (c) ULP and CPP; (d) Users (proposed framework without peer-to-peer capability).

The total amount of data generated by each module is presented in Table 6. The signalling data presented for each component corresponds to the data that is both received and transmitted by the considered component.

TABLE 6. Signalling data generated by the vCDN System Components (proposed framework without peer-to-peer caching scenario).

Component	Signalling (MB)
vCDN Engine	39.6
Monitoring Manager	432.7
ULP and CPP	376.1
Users	3000.0

3) PROPOSED FRAMEWORK

Fig. 17 presents the throughput generated by the components of the proposed framework when the peer-to-peer caching feature is enabled.

Table 7 presents the total amount of signaling data generated by the different components.

F. MEC HOST RESOURCES UTILIZATION

All the previous tests where the vCDN Nodes are dynamically instantiated consider that the CPP sends a *Content Placement Suggestion* for all the needed MECs as soon as it calculates the necessary chunks. This leads to the vCDN Nodes being all instantiated at the beginning of the experience. This section assesses the impact of deploying the necessary nodes only when the train is near the MEC handover moment. For that effect, the CPP calculates the necessary chunks that need to be present at each MEC but it sends the recommendation to the vCDN Engine only when necessary, saving the remaining

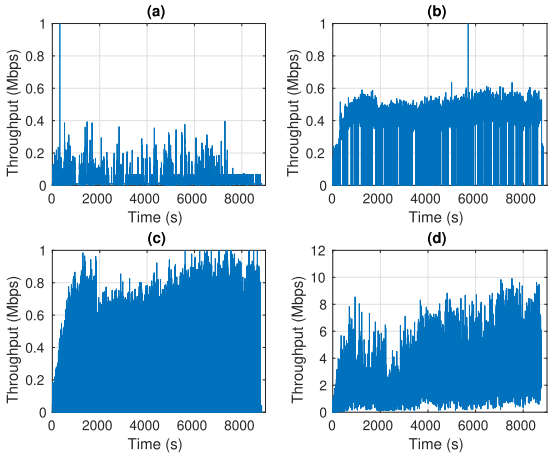

FIGURE 17. Signalling data generated by (a) vCDN Engine; (b) Monitoring Manager; (c) ULP and CPP; (d) Users (full proposed framework scenario).

TABLE 7. Signalling data generated by the vCDN System Components (Full proposed framework).

Component	Signalling (MB)
vCDN Engine	32.6
Monitoring Manager	434.4
ULP and CPP	389.5
Users	3000.0

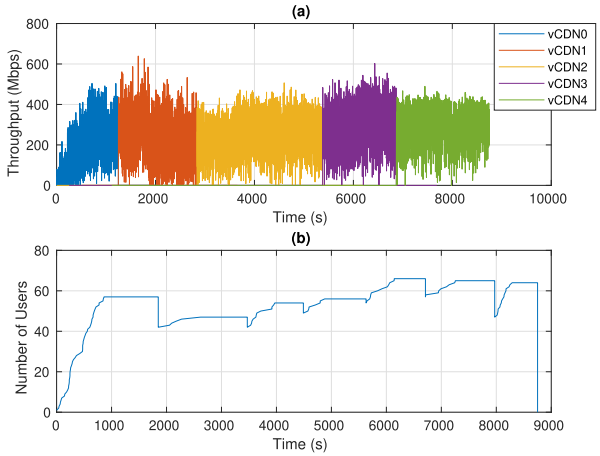

FIGURE 18. (a) Throughput generated by the distributed vCDN Nodes delivering video chunks to users (Proposed Framework); (b) Number of active users during the course of the experience.

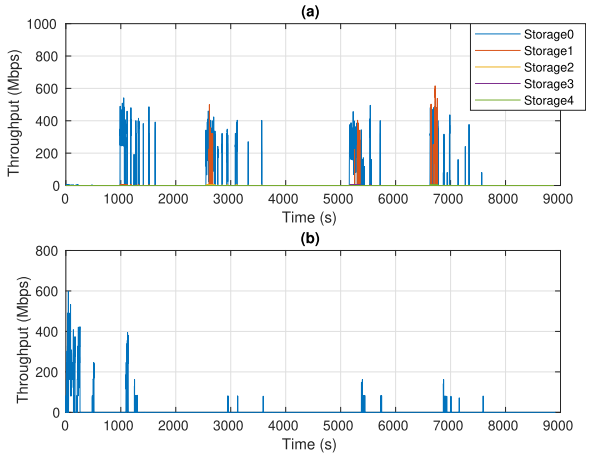

FIGURE 19. (a) Throughput generated by the data exchanged between storage nodes (Proposed System); (b) Throughput generated by the video origin (Proposed System).

suggestions for the right moment. As was shown in Table 5, the total instantiation time for a full vCDN Node is around 178 seconds, or around 3 minutes. The CPP was configured to send the content placement suggestion 6 minutes before the handover, leaving time for the instantiation and for the Nodes to load the necessary content. This time value can be programmed, as it varies with the amount of data that a particular node needs to load, predicted train speed, current network conditions, and it poses an optimization problem that falls out of the scope of this work.

Fig. 18 shows the throughput generated by the vCDN Nodes in the considered scenario and, analysing it, it can be seen that it is very similar to the one where the nodes are deployed at the beginning, meaning that the fact that this scenario does not influence the rate at which users receive video data. The total amount of data sent to the users was 276.9 GB.

Fig. 19 shows the throughput generated between the storage nodes and by the video origin, the latter being also the throughput generated in the core network. It can be seen that the throughput generated in the core network is more condensed when compared to the scenario where all the nodes were instantiated at the beginning, meaning that the resource usage in the core network was optimized. The total amount of data that originated at the video origin was 8.0GB, 2.9 percent of the total data send to users. These results show that the impact of this scenario was positive, optimizing the available core network bandwidth.

Regarding cache hit ratio, Fig. 20 presents its evolution over the course of the selected run and it can be seen that it is very similar to the one where the nodes are deployed at the beginning of the experience, meaning that this scenario does not impact the cache hit ratio.

The next evaluation has the purpose of estimating how much MEC host resources are saved by instantiating the nodes only when the users are approaching a MEC handover. Regarding RAM and disk usage, in Openstack, both these resources are allocated as soon as the VM is created according to the size specified in the flavor used for the instance, meaning that these resources are reserved for the vCDN node's VM even if it is using only a fraction of the resources. The vCPUs are shared so, there is minimal impact if the instance is running but the vCPU it uses are idle,[8,9]

Fig. 21 illustrates the resources used by the vCDN System in terms of RAM and Storage, considering the cases

[8]https://docs.openstack.org/project-deploy-guide/openstack-ansible/draft/overview-storage-arch.html

[9]https://docs.openstack.org/glance/rocky/admin/troubleshooting.html

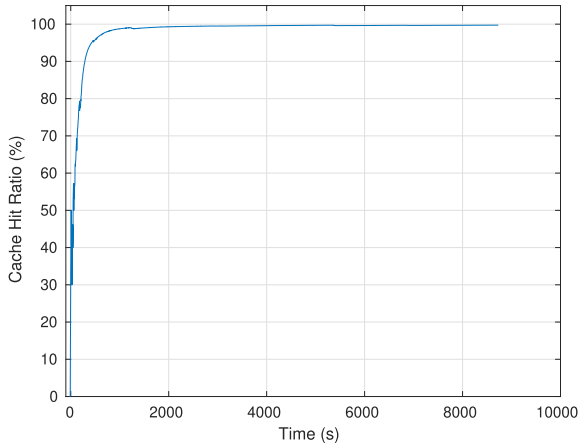

FIGURE 20. Cache Hit Ratio Evolution over the course of the selected run (full system scenario).

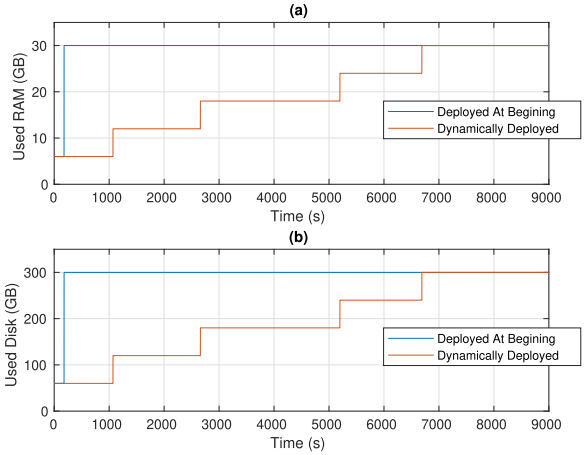

FIGURE 21. (a) RAM Usage by the vCDN System when all the nodes are instantiated at the beginning vs instantiated when approaching a MEC handover; (b) Disk Usage by the vCDN System when all the nodes are instantiated at the beginning vs instantiated when approaching a MEC handover.

when all the nodes are instantiated at the beginning of the experience and when they are instantiated on the verge of a MEC handover. By integrating the graphs individually and dividing one by the other, it is possible to obtain an estimate for the resource savings of both RAM and storage. The value obtained was 35 percent resource savings when the nodes are instantiated near a MEC handover versus when they are instantiated at the beginning of the test.

G. COMPARISON WITH STATE OF THE ART SOLUTIONS

In order to further assess the contributions of the proposed framework, table 8 presents a comparison between the proposed system and other state of the art solutions. In this table, four key features are compared:

1) Location aware: it compares if the solution is aware of the user's location, allowing it to optimize the data delivery.
2) P2P Caching: it compares if the solution is able to both retrieve and provide data from and to peer caches.
3) Disruptive: it compares if the solution is disruptive, i.e., if it requires significant changes to the current networks.
4) Mobility: it compares if the solution supports or is aware of user mobility.

In table 8, most solutions are considered disruptive in the way that they consider that the user devices act as peer cache nodes, uploading data to be consumed by other users and that strategy is not employed in today's networks. From the table it also can be seen that the proposed system extends the other solutions by providing location at the Cell ID level in a way that it is not disruptive to how those CDN nodes operate.

TABLE 8. Comparison between the presented framework and some state of the art solutions.

Solution	Location Aware	P2P Caching	Disruptive	Mobility
Proposed Framework	cell Id	Yes	No	Yes
[61]	Regional	Yes	Yes	No
[62]	Regional	Yes	Yes	No
[63]	Regional	Yes	Yes	No
[64]	Regional	Yes	Yes	No

VII. CONCLUSION

The results shown that the dynamic vCDN System proposed in this work is capable of maximizing the cache hit ratio, by introducing a location aware system, while minimizing both the traffic in the core network and the MEC resource usage, through the usage of peer-to-peer caching and dynamic instantiation, respectively. Regarding the vCDN Node instantiation time, the total instantiation time for a full vCDN Node was of around 3 minutes. This time could be greatly reduced by instantiating a vCDN node as a container, enabling greater dynamicity and efficiency in the way resources are used, e.g. faster booting and freeing up resources. The system reduced the core network load by 10.9 percent when compared with a caching solution with proactive caching and it increased the cache hit ratio by 3.4 percent, with a final value 0.2 percent away from a perfect 100 percent cache hit ratio. This cache hit ratio is an improvement of 12 percent when compared with a previously presented work on a previous version of the vCDN System. Not only was the load on the core network and video server reduced, it was also optimized by aggregating the periods of time that the system uses the core network's available bandwidth, reducing the time that the system is using core network bandwidth resources. Furthermore, 35 percent of usually constrained MEC resources were saved when compared with a solution where cache nodes are deployed all the time. By comparing the presented framework with existing others, it can be concluded that it contributes by providing a non disruptive, location aware framework that optimizes network resources.

VIII. FUTURE WORK

This work leaves some points for improvement. The usage of the 5G Core network's APIs could be extended to identify, as an example, low coverage zones where the QoE of the users drops below an acceptable level. By using this feature combined with the ability to place a cache in the trains CPE, content could be placed in the train when it crosses those low coverage areas. The presented system has many configurable parameters that can be tuned to further optimize the results, such as the location prediction algorithm, the node instantiation timings, video chunk size and viewing status report period. Also, the authors hope to test this system in a commercial MEC enabled 5G deployment and analyse its behaviour.

REFERENCES

[1] Cisco Annual Internet Report (2018–2023). (2020). *Cisco*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf

[2] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1617–1655, 3rd Quart., 2016, doi: 10.1109/COMST.2016.2532458.

[3] S. Shrivastava, B. Chen, C. Chen, H. Wang, and M. Dai, "Deep Q-network learning based downlink resource allocation for hybrid RF/VLC systems," *IEEE Access*, vol. 8, pp. 149412–149434, 2020, doi: 10.1109/ACCESS.2020.3014427.

[4] J. Wu, M. Dong, K. Ota, J. Li, and W. Yang, "Application-aware consensus management for software-defined intelligent blockchain in IoT," *IEEE Netw.*, vol. 34, no. 1, pp. 69–75, Jan. 2020, doi: 10.1109/MNET.001.1900179.

[5] *5G;System Architecture for the 5G System (5GS)*, document 3GPP TS 23.501 version 15.11.0 Release 15, Oct. 2020.

[6] *Multi-access Edge Computing (MEC); Phase 2: Use Cases and Requirements*, document GS MEC 002, ETSI, 2018.

[7] *Study of Enablers for Network Automation for 5G*, document 3GPP TR 23.791 V16.2.0, pp. ,–2019.

[8] *Network Functions Virtualisation (NFV);Architectural Framework*, document ETSI GS NFV 002, 2014.

[9] R. Silva, D. Santos, D. Corujo, R. L. Aguiar, S. Figueiredo, and B. Parreira, "Mobility-optimized dynamic content placement for fast vehicles in 5G networks," in *Proc. IEEE 30th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Istanbul, Turkey, Sep. 2019, pp. 1–7.

[10] Netflix. *Internet Connection Speed Recommendations*. Accessed: Apr. 6, 2020. [Online]. Available: https://help.netflix.com/en/node/306

[11] European Telecommunications Standards Institute (ETSI). *Network Functions Virtualisation—Introductory White Paper*. Accessed: Apr. 22, 2020. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper.pdf

[12] Red Hat Enterprise Linux. *What is Virtualization*. Accessed: Apr. 22, 2020. [Online]. Available: https://www.redhat.com/en/topics/virtualization/what-is-virtualization

[13] M. Richart, J. Baliosian, J. Serrat, and J. L. Gorricho, "Resource slicing in virtual wireless networks: A survey," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 3, pp. 462–476, Dec. 2016, doi: 10.1109/TNSM.2016.2597295.

[14] S. Rowshanrad, S. Namvarasl, V. Abdi, M. Hajizadeh, and M. Keshtgary, "A survey on sdn, the future of networking," *J. Adv. Comput. Sci. Technol.*, vol. 3, no. 2, p. 232, 2014. [Online]. Available: http://www.sciencepubco.com/index.php/IJPE/article/view/3754, doi: 10.14419/jacst.v3i2.3754.

[15] W. Stallings, "SDN and openflow," *The Internet Protocol J.*, vol. 16, no. 3, pp. 1–40, 2013. [Online]. Available: https://ipj.dreamhosters.com/wp-content/uploads/issues/2013/ipj16-1.pdf.

[16] D. Sabella, A. Reznik, and R. Frazao, *Multi-Access Edge Computing in Action*. Boca Raton, FL, USA: CRC Press, 2020.

[17] X. Ma, J. Zhao, Y. Gong, and Y. Wang, "Key technologies of MEC towards 5G-enabled vehicular networks," in *Quality, Reliability, Security and Robustness in Heterogeneous Systems. QShine* (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), vol. 234, L. Wang, T. Qiu, and W. Zhao, Eds. Cham, Switzerland: Springer, 2018, pp. 153–159, doi: 10.1007/978-3-319-78078-8_16.

[18] Al-Mukaddim, K. Pathan, and R. Buyya, "A taxonomy and survey of content delivery networks," Grid Comput. Distrib. Syst. (GRIDS), Lab. Dept. Comput. Sci. Softw. Eng., Univ. Melbourne, Parkville, VIC, Australia, Tech. Rep., 2012.

[19] *Wireless Network Optimization: Mobile Edge Computing (MEC) and Content Delivery Network (CDN) Market Outlook, Forecasts, and the Path to 5G Enabled Apps and Services*. Accessed: May 14, 2020. [Online]. Available: https://www.researchandmarkets.com/reports/4335566/wireless-network-optimization-mobile-edge

[20] X. Ge, J. Ye, Y. Yang, and Q. Li, "User mobility evaluation for 5G small cell networks based on individual mobility model," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 3, p. 528–541, Mar. 2016.

[21] M. M. Zonoozi and P. Dassanayake, "User mobility modeling and characterization of mobility patterns," *IEEE J. Sel. Areas Commun.*, vol. 15, no. 7, pp. 1239–1252, Sep. 1997, doi: 10.1109/49.622908.

[22] D. Ilie and V. V. K. S. Datta, "On designing a cost-aware virtual CDN for the federated cloud," in *Proc. Int. Conf. Commun. (COMM)*, Bucharest, Romania, Jun. 2016, pp. 255–260, doi: 10.1109/ICComm.2016.7528255.

[23] *Openstack—Open Source Software for Creating Private and Public clouds*. Accessed: May 21, 2020. [Online]. Available: www.openstack.org

[24] NGINX. *High Performance Load Balancer, Web Server*. Accessed: Mar. 18, 2020. [Online]. Available: www.nginx.com

[25] N. Anjum, D. Karamshuk, M. Shikh-Bahaei, and N. Sastry, "Survey on peer-assisted content delivery networks," *Comput. Netw.*, vol. 116, pp. 79–95, Apr. 2017, doi: 10.1016/j.comnet.2017.02.008.

[26] M. Goldmann and G. Kreitz, "Measurements on the spotify peer-assisted music-on-demand streaming system," in *Proc. IEEE Int. Conf. Peer Comput.*, Kyoto, Japan, Aug. 2011, pp. 206–211, doi: 10.1109/P2P.2011.6038737.

[27] G. Kreitz and F. Niemela, "Spotify—Large scale, low latency, P2P music-on-demand streaming," in *Proc. IEEE 10th Int. Conf. Peer Comput. (P2P)*, Delft, The Netherlands, Aug. 2010, pp. 1–10, doi: 10.1109/P2P.2010.5569963.

[28] B. Zhang, G. Kreitz, M. Isaksson, J. Ubillos, G. Urdaneta, J. A. Pouwelse, and D. Epema, "Understanding user behavior in spotify," in *Proc. IEEE INFOCOM*, Turin, Italy, Apr. 2013, pp. 220–224, doi: 10.1109/INFCOM.2013.6566767.

[29] G. Zhang, W. Liu, X. Hei, and W. Cheng, "Unreeling xunlei kankan: Understanding hybrid CDN-P2P video-on-demand streaming," *IEEE Trans. Multimedia*, vol. 17, no. 2, pp. 229–242, Feb. 2015, doi: 10.1109/TMM.2014.2383617.

[30] Z. Liu, Y. Ding, Y. Liu, and K. Ross, "Peer-assisted distribution of user generated content," in *Proc. IEEE 12th Int. Conf. Comput. (P2P)*, Tarragona, Spain, Sep. 2012, pp. 261–272, doi: 10.1109/P2P.2012.6335807.

[31] L. Yao, A. Chen, J. Deng, J. Wang, and G. Wu, "A cooperative caching scheme based on mobility prediction in vehicular content centric networks," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5435–5444, Jun. 2018, doi: 10.1109/TVT.2017.2784562.

[32] Y. Niu, Y. Liu, Y. Li, Z. Zhong, B. Ai, and P. Hui, "Mobility-aware caching scheduling for fog computing in mmWave band," *IEEE Access*, vol. 6, pp. 69358–69370, 2018, doi: 10.1109/ACCESS.2018.2880031.

[33] L. K. Dutta, J. Xiong, L. Gui, B. Liu, and Z. Shi, "On hit rate improving and energy consumption minimizing in cache-based convergent overlay network on high-speed train," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jeju, South Korea, Jun. 2019, pp. 1–6, doi: 10.1109/BMSB47279.2019.8971853.

[34] S. Kumar, D. S. Vineeth, and A. F. A, "Edge assisted DASH video caching mechanism for multi-access edge computing," in *Proc. IEEE Int. Conf. Adv. Netw. Telecommun. Syst. (ANTS)*, Indore, India, Dec. 2018, pp. 1–6, doi: 10.1109/ANTS.2018.8710106.

[35] K. V. Katsaros and V. Glykantzis, "Experimenting with cache peering in multi-tenant 5G networks," in *Proc. 21st Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Paris, Feb. 2018, pp. 1–5, doi: 10.1109/ICIN.2018.8401623.

[36] T. Zhang, X. Fang, Y. Liu, and A. Nallanathan, "Content-centric mobile edge caching," *IEEE Access*, vol. 8, pp. 11722–11731, 2020, doi: 10.1109/ACCESS.2019.2962856.

[37] S. E. Ghoreishi, D. Karamshuk, V. Friderikos, N. Sastry, M. Dohler, and A. H. Aghvami, "A cost-driven approach to caching-as-a-service in cloud-based 5G mobile networks," *IEEE Trans. Mobile Comput.*, vol. 19, no. 5, pp. 997–1009, May 2020, doi: 10.1109/TMC.2019.2904061.

[38] D. T. Hoang, D. Niyato, D. N. Nguyen, E. Dutkiewicz, P. Wang, and Z. Han, "A dynamic edge caching framework for mobile 5G networks," in *IEEE Wireless Commun.*, vol. 25, no. 5, pp. 95–103, Oct. 2018, doi: 10.1109/MWC.2018.1700360.

[39] Z. Zheng, L. Song, Z. Han, G. Y. Li, and H. V. Poor, "A stackelberg game approach to proactive caching in large-scale mobile edge networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5198–5211, Aug. 2018, doi: 10.1109/TWC.2018.2839111.
[40] A. Soltani, B. Akbari, and N. Mokari, "User profile-based caching in 5G telco-CDNs," in *Proc. IEEE 8th Int. Conf. Cloud Netw. (CloudNet)*, Coimbra, Portugal, Nov. 2019, pp. 1–6, doi: 10.1109/CloudNet47604.2019.9064113.
[41] N.-S. Vo, M.-P. Bui, P. Q. Truong, C. Yin, and A. Masaracchia, "Multi-tier caching and resource sharing for video streaming in 5G ultra-dense networks," *IEEE Commun. Lett.*, vol. 24, no. 7, pp. 1500–1504, Jul. 2020, doi: 10.1109/LCOMM.2020.2983408.
[42] M. F. Ahmad and M. Arif Hossain, "Mobility aware cache management in 5G future generation wireless communication system," in *Proc. 1st Int. Conf. Adv. Sci., Eng. Robot. Technol. (ICASERT)*, Dhaka, Bangladesh, May 2019, pp. 1–6, doi: 10.1109/ICASERT.2019.8934710.
[43] M. Furqan, C. Zhang, W. Yan, A. Shahid, M. Wasim, and Y. Huang, "A collaborative hotspot caching design for 5G cellular network," *IEEE Access*, vol. 6, pp. 38161–38170, 2018, doi: 10.1109/ACCESS.2018.2852278.
[44] G. Ma, Z. Wang, M. Zhang, J. Ye, M. Chen, and W. Zhu, "Understanding performance of edge content caching for mobile video streaming," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1076–1089, May 2017, doi: 10.1109/JSAC.2017.2680958.
[45] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative edge caching in user-centric clustered mobile networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 8, pp. 1791–1805, Aug. 2018, doi: 10.1109/TMC.2017.2780834.
[46] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, p. 2392–2431, 4th Quart., 2017.
[47] F. De Rango, P. Fazio, and S. Marano, "Utility-based predictive services for adaptive wireless networks with mobile hosts," *IEEE Trans. Veh. Technol.*, vol. 58, no. 3, pp. 1415–1428, Mar. 2009, doi: 10.1109/TVT.2008.924989.
[48] J. Llorca, C. Sterle, A. M. Tulino, N. Choi, A. Sforza, and A. E. Amideo, "Joint content-resource allocation in software defined virtual CDNs," in *Proc. IEEE Int. Conf. Commun. Workshop (ICCW)*, London, U.K., Jun. 2015, pp. 1839–1844, doi: 10.1109/ICCW.2015.7247448.
[49] N. T. Jahromi, R. H. Glitho, A. Larabi, and R. Brunner, "An NFV and microservice based architecture for on-the-fly component provisioning in content delivery networks," in *Proc. 15th IEEE Annu. Consum. Commun. Netw. Conf. (CCNC)*, Las Vegas, NV, USA, Jan. 2018, pp. 1–7, doi: 10.1109/CCNC.2018.8319227.
[50] A. B. Letaifa, "Adaptive QoE monitoring architecture in SDN networks: Video streaming services case," in *Proc. 13th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2017, pp. 1383–1388.
[51] L. Zhao, J. Liu, Y. Shi, W. Sun, and H. Guo, "Optimal placement of virtual machines in mobile edge computing," in *Proc. IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–6.
[52] *Mobile Edge Computing (MEC);Deployment of Mobile Edge Computing in an NFV environment*, document ETSI GR MEC 017, 2018.
[53] *Multi-access Edge Computing (MEC); Framework and Reference Architecture*, document ETSI GS MEC 003, 2019.
[54] S. Kekki *et al.*, "MEC in 5G networks," 1st ed., Eur. Telecommun. Standards Inst., Sophia Antipolis, France, ETSI White Paper 28, Jun. 2018.
[55] *5G; 5G System; Network Exposure Function Northbound APIs*, document TS 29.522 version 15.5.0 Release 15, 3GPP, 2020.
[56] *Universal Mobile Telecommunications System (UMTS);LTE; 5G; T8 reference point for Northbound APIs*, document TS 29.122 version 15.6.0 Release 15, 3GPP, 2020.
[57] *Digital Cellular Telecommunications System (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Policy and Charging Control Architecture*, document TS 23.203 version 16.2.0 Release 16, 3GPP, 2020.
[58] Anacom. (2019). *Pacotes de Serviãs de Comunicaães Eletrãnicas*. [Online]. Available: https://www.anacom.pt/streaming/PacotesServicos1S19_rev.pdf?contentId=1480249&field=ATTACHED_FILE
[59] N. Alliance. (2015). *5G White Paper*. [Online]. Available: https://www.ngmn.org/wp-content/uploads/NGMN_5G_White_Paper_V1_0.pdf
[60] J. Aires, P. Duarte, B. Parreira, and S. Figueiredo, "Phased-vCDN orchestration for flexible and efficient usage of 5G edge infrastructures," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Dallas, TX, USA, Nov. 2019, pp. 1–6, doi: 10.1109/NFV-SDN47374.2019.9040097.
[61] J. Wu, Z. Lu, B. Liu, and S. Zhang, "PeerCDN: A novel P2P network assisted streaming content delivery network scheme," in *Proc. 8th IEEE Int. Conf. Comput. Inf. Technol.*, Sydney, NSW, Australia, Jul. 2008, pp. 601–606.
[62] M. Garmehi and M. Analoui, *Envy-Free Resource Allocation and Request Routingin Hybrid CDN–P2P Networks*. Cham, Switzerland: Springer, 2015.
[63] H. Shen, Z. Li, Y. Lin, and J. Li, "SocialTube: P2P-assisted video sharing in online social networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2428–2440, Sep. 2014.
[64] L. Chen, Y. Zhou, M. Jing, and R. T. B. Ma, "Thunder crystal: A novel crowdsourcing-based content distribution platform," in *Proc. 25th ACM Workshop Netw. Oper. Syst. Support Digit. Audio Video*, 2015, pp. 43–48.

DAVID SANTOS received the M.Sc. degree in electronics and telecommunications engineering from the Electronics, Telecommunications and Informatics Department, University of Aveiro, Portugal, in 2018. From 2017 to 2020, he was a Researcher with the Institute of Telecommunications, Portugal. He contributed to the 5G Mobilizer (5GO) and Smart Green Homes (SGH) projects. His current research interests include 5G networks exploiting software defined networks (SDNs) and network function virtualization (NFV) mainly focusing on multi-access edge computing (MEC) capabilities.

RUI SILVA received the M.Sc. degree in electronics and telecommunications engineering from the Electronics, Telecommunications and Informatics Department, University of Aveiro, Portugal, in 2018. He is currently working as a Researcher and a Developer with the Telecommunications and Networking Group, Instituto de Telecomunicações, Aveiro, participating in the 5GO Project. His main research interests include virtualization and enhancement of mobile networks' core (4G and 5G) using software defined networks (SDNs) and network function virtualization (NFV), network slicing, and multi-access edge computing (MEC) applied to mobile networks.

DANIEL CORUJO (Senior Member, IEEE) received the Ph.D. degree in communication middleware for the future mobile Internet from the University of Aveiro, in 2013. He was the Coordinator of the Telecommunications and Networking Research Team, Instituto de Telecomunicações, Aveiro, Portugal, a team of more than 50 people, from 2017 to 2018. He has been an active Researcher and a Contributor to standardization in the fields of mobility management, through the IETF/IRTF, and Media Independent Handovers, through the IEEE. He has pursued such concepts under the scope of a broad range of EU FP7 research projects, since 2007, such as DAIDALOS, OneLab2, 4WARD, MEDIEVAL, OFELIA, and 5GROWTH, where he also played key roles from proposal elaboration to task and workpackage co-leading. He is currently an Assistant Professor with the University of Aveiro. He is also the WP Leader in the National 5G Mobilizer Project. Parallel to his 13 years of experience on mobility management research, he has been more recently developing work on the areas of the 5G, network function virtualization, software defined networking, and information centric networking, deploying new visions and enhancements of such concepts over wireless networks in national and international research projects. He is the Secretary of the IEEE ComSoc PT Chapter.

RUI L. AGUIAR (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the University of Aveiro, in 2001. He was an Adjunct Professor with INI, Carnegie Mellon University. He is currently a Full Professor with the University of Aveiro. He is also a Visiting Scholar with the Universidade Federal de Uberlandia. He is also leading the national Co-Head of Networks and Multimedia inside ITAV. His current research interest includes the implementation of advanced wireless networks and systems, with a special emphasis on QoS and mobility aspects. He has more than 400 published articles in those areas. He has served as the Technical and the General Chair for several conferences from IEEE, ACM, and IFIP, and is regularly invited for keynotes on 5G and FI networks. He sits on the TPC of all major IEEE ComSoc conferences. He has extensive participation in national and international projects, of which the best example is his position as the Chief Architect of the IST Daidalos Project, and has extensive participation in industry technology transfer actions. He is also associated with the 5G PPP Infrastructure Association and is on the steering board of the Networld2020 ETP. He is a Chartered Engineer, the Portugal ComSoc Chapter Chair, and a member of ACM. He is also an Associate Editor of ETT (Wiley), *Wireless Networks* (Springer), and of the recently launched *ICT Express* (Elsevier).

BRUNO PARREIRA is currently an Expert with Altran, providing expertise in telecommunications in the Research and Development Division. During his master thesis, he received the part-time Scholarship under the Program ETAPAS (a joint effort between the Instituto de Telecomunicações and PT Inovação, which is now called Altice Labs), where he contributed to the FP7 4WARD Project. After finishing his master thesis in electronic engineering and telecommunications from the University of Aveiro, in 2012, he received the Full Scholarship under the same program, ETAPAS, where he participated in several FP7 projects, such as SAIL, MCN, and TNOVA. Afterwards, he started working at Altice Labs in the OSS Division, where he helped to evolve their products to support new technologies and processes related with SDN and NFV. Apart from additional internal Research and Development projects, he also provided technical leadership to H2020 projects SELFNET and SLICENET, and participated in ONF Project Boulder and ETSI ISG ENI, where he was Rapporteur to the Context-aware Policy Management Gap Analysis work item. More recently at Altran, he is also working on 5G mobilizer, HAL, and NARVID. During his professional career, he has made more than ten publications in conference venues (mostly organized by IEEE), journals, and magazines; and contributed to some open-source projects such as Openstack and OpenDayLight.