Received March 30, 2020, accepted April 20, 2020, date of publication April 24, 2020, date of current version May 8, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2990292

SDN Enhanced Multi-Access Edge Computing (MEC) for E2E Mobility and QoS Management

SYED DANIAL ALI SHAH[1], MARK A. GREGORY[1], (Senior Member, IEEE), SHUO LI[1], (Member, IEEE), AND RAMON DOS REIS FONTES[2]
[1]School of Engineering, RMIT University, Melbourne VIC 3000, Australia
[2]Federal Institute of Education, Science and Technology of Bahia, Salvador 40301-015, Brazil

Corresponding author: Mark A. Gregory (mark.gregory@rmit.edu.au)

ABSTRACT Multi-access Edge Computing (MEC) is a key enabler of the fifth-generation (5G) mobile cellular networks. MEC enables Ultra-reliable and Low-latency Communications (URLLC) by bringing the data and computational resources closer to the mobile users. As 5G deployments commence in earnest, researchers have turned their attention to various aspects of edge computing in an effort to leverage the new capabilities offered by 5G. In this paper, we propose the integration of Software Defined Networking (SDN) and cloud-native virtualization techniques, such as containers, with the MEC architecture, to facilitate the orchestration and management of Mobile Edge Hosts (MEH). The proposed architecture focuses on the end-to-end mobility support required to maintain service continuity when mobile users relocate from one MEH to another. SDN is proposed as a reliable, programmatic paradigm to provide mobile edge orchestration and dynamic configuration of the underlying network for improved service continuity and quality of experience. The proposed architecture is validated through vehicle-to-everything simulations that highlight the advantage of the centralized network intelligence and the modularity and portability offered by SDN and containers.

INDEX TERMS Software defined networking, mobility management, multi-access edge computing, 5G, cloud-native, containerization, URLLC.

I. INTRODUCTION

Multi-access Edge Computing (MEC) has evolved to complement cloud computing and align with the future of the telecommunications industry, i.e., to support the explosive growth of the Internet of Things (IoT) devices and the diverse service requirements of emerging vertical industries. The MEC paradigm aims to provide Ultra-reliable and Low-latency Communications (URLLC), a reduction in the traffic to centralized cloud facilities and a reduction in the complexity of cloud computing related to IoT and mobility devices. This is achieved by pushing the cloud computing resources, e.g., storage and computing, closer to the access network where the IoT and mobility devices are located and data is generated [1]. Apart from the benefits of mitigating communication latency, MEC also enables services such as Location Services (LS) and Radio Network Information Services (RNIS) that can be leveraged to make intelligent network-related decisions for effective service management.

MEC is expected to play a key role in the upcoming 5G ecosystem by providing network flexibility, scalability, and offering optimized diverse services that will positively impact on various vertical markets [2], [3]. However, to fully realize the benefits of MEC for vertical market segments, several major issues are yet to be addressed, e.g., the MEC architecture for an automotive domain. There is a need for end-to-end (E2E) mobility support to maintain service continuity when mobile users migrate from one Mobile Edge Host (MEH) to another. To provide mobility support, Mobile Edge (ME) systems should facilitate user, service and application state information transfer from one MEH to another, in a seamless manner [4].

The service relocation policies should be designed for the application, e.g., a dedicated application or shared application. A dedicated application instance serves a specific user, and as the user moves from the service area of a MEH to another, the ME application instance should be relocated

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

between MEH to ensure that there is service continuity. A shared application may serve multiple users (e.g., a multicast application). For this shared application, the application instance for a mobility user may not need to be relocated between MEH and service continuity would be maintained by moving the user-context between MEH.

The objective of the service relocation policies should be to reduce service downtime. The European Telecommunications Standards Institute (ETSI) proposes the concept of service pre-relocation in [5]. This approach aims to make use of the vehicle trajectories to relocate the application between MEH before the actual handover takes place. As defined by the ETSI in [4] there are multiple challenges involved in the service migration process such as prediction of the optimal time to initiate the service relocation process, identification of a target MEH with resources available to support the application, seamless migration of the application to the target MEH, real-time configuration of updated traffic rules and setup of the new communication paths to the device.

To elaborate on the concept of the service migration and application relocation policies provided by the ETSI, we propose a Software Defined Networking (SDN) enhanced edge computing architecture that integrates cloud-native technologies and an advanced form of virtualization, the open-source docker container platform, to perform seamless service migration. In our proposed architecture, SDN uses the RNIS and LS of the ME system to predict the timing of the service migration, selects optimal target MEH for service relocation depending upon the availability of the required resources, performs handover to the underlying target MEH network, and allocates the bandwidth required for the application at the target MEH. The MEC bandwidth manager is an essential feature of MEC. The MEC platform assigns the requested bandwidth, if it is available, and priority to the application or session [6].

In the proposed architecture, SDN evolves as a Mobile Edge Orchestrator (MEO) that enables coordination between MEH and makes efficient network-related decisions and routing updates. The integration of docker container platform and its services in the MEC architecture would allow new features such as application portability and immutability, scalability, Continuous Integration and Continuous Delivery (CI/CD), and fast deployment speeds. Because of these features offered by the docker, it is considered as one of the key enablers of the MEC [7]–[9]. We made use of these features offered by the docker container platform and proposed a new docker service in the MEC architecture for service migration called as a Docker Registry Service (DRS). The DRS stores the ME applications filesystem also called as Docker images that can be executed in real-time as ME applications running inside docker containers. By utilizing the operating system independent virtualization and portability offered by the docker platform, these ME applications can be relocated between MEH in real-time with improved service continuity.

We provide Proof-of-Concept (PoC) experiments for the proposed architecture for a Vehicle-to-Everything (V2X) use case using Mininet-WiFi and Containernet. Mininet-WiFi is an extension of Mininet emulator for Software-Defined Wireless Networks (SDWN) [10] and Containernet allows the use of docker containers as hosts in an emulated network topology [11]. Both emulators are actively being used by the research community that focuses on SDN, cloud computing, edge computing and Network Function Virtualization (NFV). We overhauled Mininet-WiFi and Containernet to resemble the main components and features of the reference architecture and conceived framework of MEC. To the best of the author's knowledge, this is one of the first works which integrates the SDN paradigm into an edge computing environment to support inter-operator interactions for E2E mobility and Quality of Service (QoS) management. The main contributions are summarized as follows:

1. We emphasized the importance of the synergy between the SDN and MEC, by realizing the benefits of the new SDN paradigm in coping with the mobility challenges of MEC.

2. We proposed an SDN enhanced edge computing architecture that is fully aligned with the ETSI MEC reference architecture for E2E mobility and QoS management. We focus on the service migration between MEH of different networks with guaranteed service continuity, which is still an open issue [12].

3. We integrated the advanced form of virtualization offered by the docker container platform into the MEC architecture. We also introduced DRS as a new ME service to store and relocate the ME applications with minimal service disruptions. We considered the dedicated application scope, which demands frequent relocation of ME applications.

4. We integrated the features of widely used emulators, Mininet-WiFi and Containernet, to simulate the functionalities of MEC and provided PoC experiments to support the V2X use case.

The rest of the paper is organized as follows. A review of related works is presented in Section II. In Section III, we briefly introduce the MEC architecture and its mobility aspects. In Section IV, we propose our SDN enhanced MEC architecture for E2E mobility and QoS management. In Section V, we provide an experimental evaluation of our proposal for the V2X mobility scenario followed by conclusion and future directions in Section VI.

II. RELATED WORKS

MEC supports low-latency, mission-critical, and emerging vertical services in 5G networks. Recently, researchers have investigated MEC to exploit different use-cases and deal with the associated challenges. In [13], Zhou *et al.* proposed a dynamic energy-efficient workload offloading algorithm that takes into account vehicle mobility to achieve energy savings for battery constrained in-vehicle User Equipment (UEs). Liao *et al.* in [14], proposed an optimal learning-based channel selection framework to maximize the long-term throughput of edge computing by incorporating service reliability, energy, and backlog awareness. Zhou *et al.* in [15], proposed an air-ground integrated MEC architecture that makes use of the centralized network intelligence and flexible resource management offered by SDN for connectivity enhancement, adaptive resource allocation, and mobility management. In [16], Li *et al.* introduced deep learning into the edge computing environment and propose algorithms to optimize network performance by maximizing the number of tasks executed at the edge nodes. Balasubramanian *et al.* in [17], proposed an intra-vehicle resource sharing model that allows in-vehicle servers with idle computation resources to collaborate and form low-latency Vehicular Service Clouds (VSC) to complete vehicular service requests with minimal delay. Balasubramanian *et al.* in [18], proposed a unified service architecture and introduced an Identifier-Locator (I-L) concept to enable seamless handover between 4G and 5G services by using the network slicing paradigm. While these works address essential issues in MEC, seamless service migration, and mobility management between edge clouds of different networks to ensure service continuity still remains a challenge [12], [19].

Few studies deal with service migration and mobility management in edge computing. In [20], Bellavista *et al.* proposed an edge computing architecture for application-aware and application-agnostic service migration between MEC servers leveraging the container characteristics. Addad *et al.* in [21], proposed parallel migration of containers to facilitate service and network slice mobility across edge clouds. In [22], Machen *et al.* proposed a layered framework for service migration leveraging the container and Virtual Machine (VM) characteristics. In [23], Campolo *et al.* evaluated the potential of containers to support the MEC for 5G-V2X use cases by using the copy and transfer techniques between MEH, e.g., export, SCP, and import. In [24], Farris *et al.* proposed a container-based framework for service replication in mobile edge networks by using conventional copy techniques.

Nevertheless, the related works mentioned target the service migration between two MEC nodes using conventional copy and transfer techniques and do not take into account the different underlying networks associated with the MEH. However, to enable low-latency services, the MEH is co-located with the Radio Access Network (RAN), and seamless service continuity can only be maintained when there is interoperability between edge clouds of different Mobile Network Operators (MNOs). The existing works on service migration also don't support the procedures required to support the seamless service migration between MEH, e.g., prediction of the accurate timing to initiate the service migration process and identification of capable target MEH that has idle computational resources to ensure service continuity. Therefore, our proposed SDN enhanced MEC architecture supports inter-operator interactions to share the network status of MEH and its underlying network, e.g., computational load and channel conditions, that assist the MEC system to select the optimal MEH for service relocation to enhance the user quality of experience. Our proposed architecture is an improvement to the existing solutions for service migration, and provides a SDN Bandwidth Management Service (SDN-BWMS), which gives the relocated ME application session its configured bandwidth allocation at the target MEH and its underlying network. The SDN-BWMS allows fast flow resumption that supports updating the bandwidth allocation to the ME application session upon request if the MEH has resources available.

III. ETSI MEC ARCHITECTURE AND MOBILITY ASPECTS

ETSI launched an Industry Specialization Group (ISG) for MEC in 2014, to develop a standardized and open edge environment that would enable seamless interaction of applications from different stakeholders across heterogeneous computing platforms. A reference architecture for MEC and a set of Application Programming Interfaces (APIs) for interaction among different MEC components have been specified in [25]. The main components of the MEC reference architecture along with their functionalities can be seen in Fig. 1 and are briefly introduced as follows:

A. ME HOST (MEH)

MEH is an important entity in the MEC architecture that consists of MEC Platform (MEP) and a virtualization infrastructure that provides computing, networking, and storage resources to the ME applications. The ME application is a realized software program running on the virtualization infrastructure as a Virtual Machine (VM) or container. The application instances can interact with the MEP to consume or offer ME services.

B. MEC PLATFORM (MEP)

MEP offers an environment where the applications can discover, consume and offer ME services. Some of the most important services offered by the MEC platform are the RNIS and LS that provide information about the radio network, e.g., network computational load, available network resources, channel conditions, position of the user in the network and its serving radio base station. The services can be used by ME applications to carry out efficient network mobility management. In addition to these services, MEP is also responsible for instantiation and termination of ME applications, if requested by the MEC platform manager (MEPM).

C. MEC ORCHESTRATOR (MEO)

MEO is a system-level management entity in the MEC architecture that maintains an overall view of the MEH deployed in the MEC system. It contains all of the topology information of the deployed MEH, such as available resources and ME services. MEO is also responsible for triggering the process of application relocation between MEH when the mobility feature is supported. The MEO should select an appropriate MEH to perform application relocation based on constraints, e.g., latency, available resources, and available services.

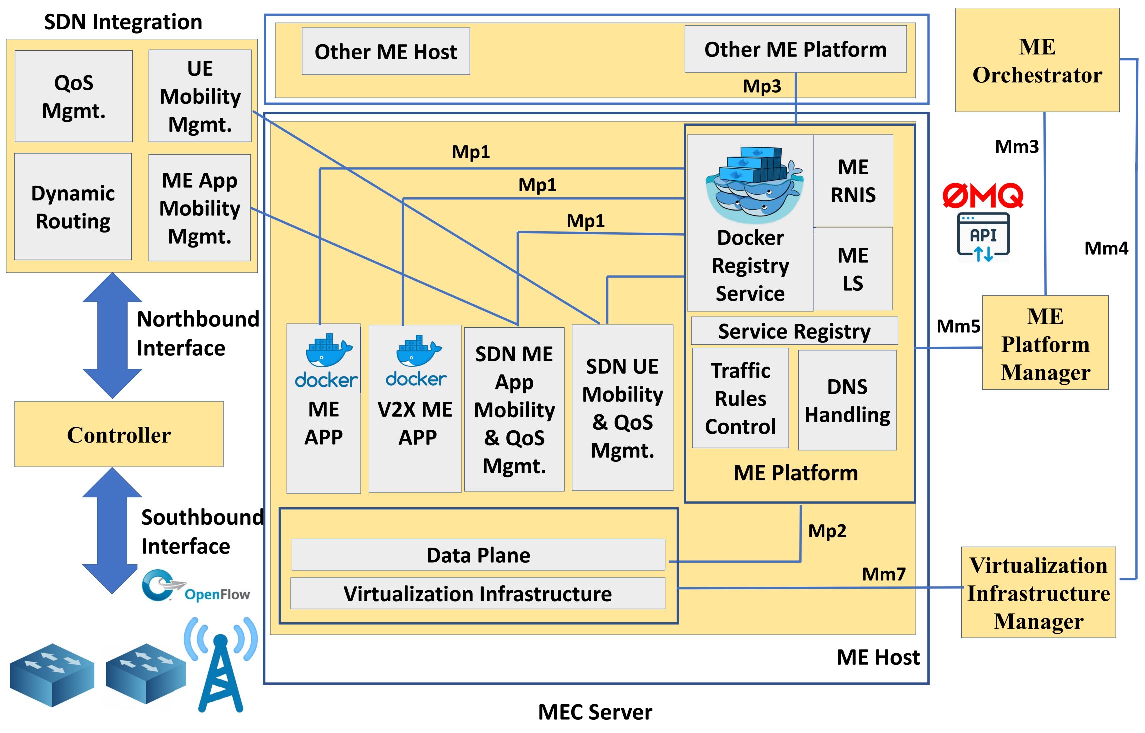


FIGURE 1. SDN enhanced MEC architecture.

D. APPLICATION INSTANCE RELOCATION

Application instance relocation is the MEC mobility capability that supports moving an application instance between MEH, to provide service continuity. Depending on the type of ME application, e.g., stateful or stateless, the operational state of the application instance may also need to be transferred between MEH for stateful applications.

E. SERVICE PRE-RELOCATION MANAGEMENT

To provide MEC mobility support, it is important that there is an accurate prediction of the handover to the target MEH and its associated network [4]. Relocation failure such as too late, too early or relocation to a wrong MEH can affect the user QoE and increase service disruptions. Therefore, efficient pre-relocation strategies should be designed so that the application and its state (in the case of a stateful application) can be pre-relocated between MEH before the actual handover takes place.

F. ME HOSTS DEPLOYMENT OPTIONS

There are multiple deployment options for MEH, e.g., co-located with the radio base stations (eNodeB) or deployed at a location that is close to the radio base stations. When applications and services are co-located with the radio base stations, traffic to the user devices traverses the RAN and latency is reduced significantly. This type of deployment option adds complexity by requiring the MEC system to relocate the ME application instance at every user handover. On the other hand, when the MEH is deployed closer to the eNodeBs, a single MEH can serve a geographical area covering adjacent cells, thus reducing the application relocation frequency. In this paper, we considered the first deployment scenario in which the MEH is co-located with the base station, to enable URLLC and deal with the added complexity of frequent application relocation.

IV. SDN ENHANCED EDGE COMPUTING ARCHITECTURE

In this section, we propose an SDN enhanced framework for supporting the mobility aspects of MEC. The proposed framework leverages SDN and containerization engine to optimize the relocation of ME applications between MEH associated with different networks.

The proposed high-level architecture of SDN enhanced MEC builds upon the reference model of MEC as defined by the ETSI in [25]. Within the conceived general design of the reference architecture of MEC, we customized its specific functionalities and procedures to support the mobility features of mobile cellular networks. These customized functions and architectural components can be seen in Fig. 1 and are described as follows:

A. SDN MOBILITY MANAGEMENT

The mobility features in MEC demand fast path resource allocation algorithms in the fronthaul and backhaul segments to migrate a service between MEH, with minimal service disruptions. Real-time service migration is required for the mobility aspects of MEC [26], which further adds to the complexity of the handover management. Therefore, we propose an SDN mobility management application that is transferred to the MEH along with other ME applications, to ensure low-latency operation. This SDN application makes use of the ME services offered by the MEP such as RNIS and LS to collect up-to-date information of the UE position, available radio base stations and MEH, the channel conditions, and the load on the MEH. The SDN controller continuously monitors and tracks the trajectory of the mobile user and remains up-to-date with the network status of MEH. Based on the updates and depending upon the application requirements and constraints, e.g., latency and bandwidth, the SDN controller assists MEO to select an optimal MEH for service migration by performing handover to its underlying network.

B. SDN ME APPLICATION/SERVICE MIGRATION MANAGEMENT

The SDN application for service migration management is also pushed to the MEH to facilitate low-latency operation for managing the application relocation procedures. As the SDN controller continuously monitors the user trajectories and radio network information by using RNIS and LS services, the controller predicts the need for the relocation of a dedicated application and triggers the application relocation procedure. The controller notifies the MEP of the target MEH to instantiate the dedicated application. For example, the SDN controller keeps track of the user trajectory through the Received Signal Strength Indicator (RSSI) from different networks/radio base stations as retrieved from the RNIS and LS. As it detects the RSSI from the source MEH and its associated network falls below a certain threshold, it triggers the process of application relocation in the target MEH by notifying the target MEP. There could be several ways in which the inter-process communication between the SDN controller and target MEP can be established, such as by using the lightweight messaging library and connectivity protocols, e.g., ZeroMQ and MQ Telemetry Transport (MQTT). The lightweight messaging protocols are designed to minimize the network bandwidth, reduce resource consumption for resource-constrained devices, and ensure reliability. In this paper, we use the request/reply messaging pattern of the high performance asynchronous ZeroMQ API for inter-process communication between the SDN application and the target MEP. The ZeroMQ API provides a lightweight socket API that can carry the message across various transports like TCP, inter-process, and in-process.

C. SDN BANDWIDTH MANAGER SERVICE (SDN-BWMS)

Multiple ME applications run in parallel on the same MEH, and compete for the shared bandwidth resources. All of the ME applications may have their specific static or dynamic bandwidth demands. In particular, in the 5G network slicing framework, a dedicated application has its own specific bandwidth demands to fulfill the diverse service requirements of its customers. The ME system should be able to assign the specific static or dynamic bandwidth demands of the ME applications to enable diverse services and support the network slicing framework. This issue becomes even more complicated when a mobile user relocates from one MEH to another. To maintain service continuity for the mobile user, the ME application should be relocated and assigned its required bandwidth at another MEH that may be associated with some other network. Therefore, we propose a central bandwidth resource allocation service in the MEC system, that uses SDN and the latest features of the OpenFlow switch including per-flow meter-tables to enable the differentiated services and maintain service continuity.

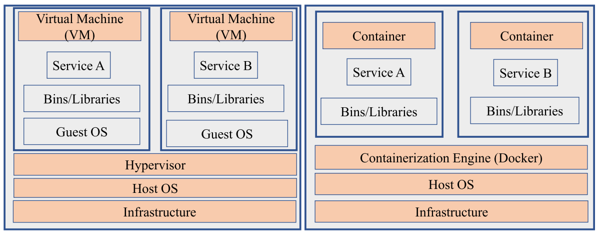


FIGURE 2. **Containers vs VMs.**

D. CONTAINERIZATION ENGINE

Containers are a light weight alternative to VMs that leverage two Linux kernel features: namespace and cgroups. The namespace is used to enable isolation of an application by providing it a limited view of the underlying operating system environment, i.e., network resources (IP addresses, routing tables and interfaces, etc.). The cgroups provide the capability to enforce limitations and prioritization of system resources, e.g., CPU and memory.

The MEC architecture relies heavily on virtualization techniques, however, VM based virtualization adds complexity and overhead to the system as the VMs require packaging of the entire operating system (OS) along with the application or functions. When compared to VMs, containers can reduce the overhead to a great extent by packaging only the application/functions and the application-specific OS dependencies. This lightweight form of virtualization and modularity features offered by the microservices architecture of containers makes them highly portable and scalable with fast deployment speeds. We propose the integration of the Docker Engine that is an open-source containerization technology for building and containerizing the applications. Docker Engine runs on the MEH and uses the resources provided by the virtualization infrastructure to build and containerize the ME applications. The advantage of using containers as compared to the VMs can be seen in Fig. 2.

E. DOCKER REGISTRY SERVICE AS A ME SERVICE

In our proposed architecture, the DRS runs in the MEH MEP in addition to the conventional services offered, e.g., RNIS and LS. The DRS manages the information about the docker images and provides an HTTP API as a protocol to facilitate the distribution of the images to the Docker Engine upon its interaction/request. With the DRS running in the MEH

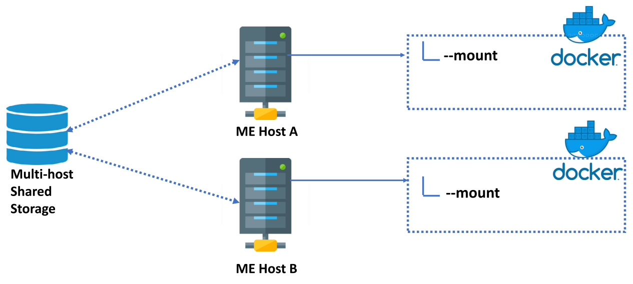
MEP the remote hosts can interact with it and pull images from it. For example, consider the case where the mobile user moves from the service area of one MEH to another, and the target MEH does not have the dedicated application image for the mobile user to maintain service continuity. The target MEH can pull the image from the DRS running in the source MEH, which can be used by the Docker Engine in target MEH to build and containerize the application in real-time.

F. MULTI-HOST SHARED PERSISTENT STORAGE FOR STATEFUL ME APPS

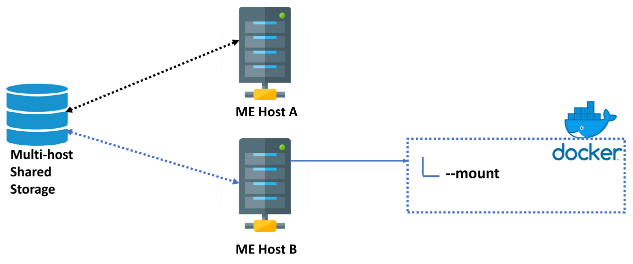
Not all of the ME applications are stateless, and the stateful applications require persistent storage for application data and instance state. In a mobile environment stateful application storage accessible by the mobile user device is critical for successful operation. To ensure service continuity, the container running the relocated application in the target MEH should be able to access the application data and instance state.

Initially, containers did not provide the capability to store state information as there was no option for persistent storage. Recently, advanced features have been introduced to containerization technology such as a Docker volume that provides persistent storage to save application data and instance state. Based on the new features, three types of persistent storage implementations can facilitate the containerization of stateful applications in MEC. They are container storage, shared storage on the same MEH and multi-host shared storage.

Per-container storage provides storage to the container only for the container life span. All data created inside the container within the volume is cleaned up when the container exits. This option is not feasible as the containers spin up and down dynamically depending upon the load, and the application data stored in the container storage might be lost. In shared storage on the same MEH, all of the containers running inside the host have access to the same underlying volume. This option is useful when multiple containers work in cooperation and read the application data from the same volume, but it is not feasible for MEC mobility support as multiple MEH may require access to the same application data and instance state. A multi-host shared persistent storage implementation provides scalable capabilities that are required to support the mobility features in MEC. There could be multiple options to deploy shared storage such as using Network File System (NFS) or the open-source object storage platform Ceph, lying externally, or on another MEH. As a result, multiple MEH running the same containerized application have access to the application data, instance state and configuration files. For example, a volume created within one host maps back to the shared storage to write application data and instance state, and the same volume can be mounted in other hosts to access the application data, instance state and configuration files. This feature can support the MEC mobility aspects, where any changes written to the mount directory will be reflected onto all of the containers running the same application in the MEH. Other significant advantages of this approach include it being very resource-efficient, as the link to the shared storage is only established when the container is running, and the container can still have access to the up-to-date application data when it restarts after exiting as seen in the Fig. 3(a) and Fig. 3(b).



(a) Connections established only when the containers are running



(b) Link disconnection when the container cleans up after service migration to MEH B

FIGURE 3. **Implementation of multi-host shared persistent storage.**

G. FLOW FOR INTER-ME HOST MOBILITY SUPPORT

Inter-ME host mobility support demands execution of the following procedures to ensure service continuity:

- Accurate timing prediction to initiate the service relocation process.
- Relocation of application instance between MEH, depending upon the scope of application (dedicated or shared).
- Service activation at the target MEH to ensure service continuity.
- Reconfiguration of traffic routing rules for the mobile user.
- Termination of the service at the source MEH to release network resources, depending upon the application scope.

Fig. 4 shows the detailed message flow of the proposed SDN enhanced edge computing architecture that integrates containerization technology for seamless service migration between MEH of different networks.

V. PROOF-OF-CONCEPT EXPERIMENTAL EVALUATION

In this section, we provide a practical realization of our proposed architecture using Mininet-WiFi and Containernet.

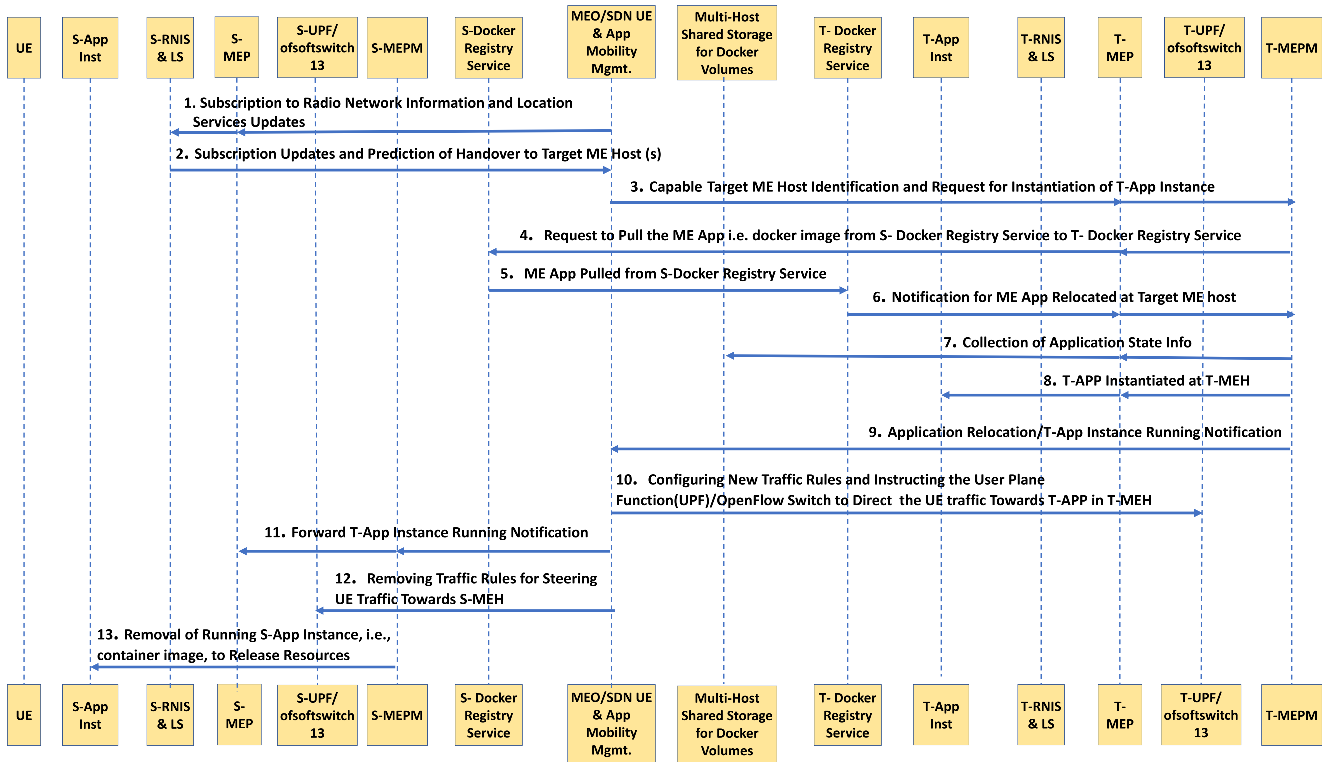

UE
S-App Inst
S-RNIS & LS
S-MEP
S-UPF/ ofsoftswitch 13
S-MEPM
S-Docker Registry Service
MEO/SDN UE & App Mobility Mgmt.
Multi-Host Shared Storage for Docker Volumes
T- Docker Registry Service
T-App Inst
T-RNIS & LS
T-MEP
T-UPF/ ofsoftswitch 13
T-MEPM
1. Subscription to Radio Network Information and Location Services Updates
2. Subscription Updates and Prediction of Handover to Target ME Host (s)
3. Capable Target ME Host Identification and Request for Instantiation of T-App Instance
4. Request to Pull the ME App i.e. docker image from S- Docker Registry Service to T- Docker Registry Service
5. ME App Pulled from S-Docker Registry Service
6. Notification for ME App Relocated at Target ME host
7. Collection of Application State Info
8. T-APP Instantiated at T-MEH
9. Application Relocation/T-App Instance Running Notification
10. Configuring New Traffic Rules and Instructing the User Plane Function(UPF)/OpenFlow Switch to Direct the UE traffic Towards T-APP in T-MEH
11. Forward T-App Instance Running Notification
12. Removing Traffic Rules for Steering UE Traffic Towards S-MEH
13. Removal of Running S-App Instance, i.e., container image, to Release Resources

FIGURE 4. **Message flow of proposed architecture for E2E mobility support in MEC.**

Both of these network emulators are extensions of popular Mininet emulator. We selected Mininet-WiFi for experimental evaluation because it permits fast prototyping of wireless networks and supports OpenFlow-enabled Wireless Access Points (APs) [10]. It also supports well-known propagation models for the emulation of a wireless medium. Containernet, on the other hand, permits docker container emulation as network hosts. We integrated the features of Mininet-WiFi and Containernet to provide the experimental evaluation of our proposed architecture. More details on the implementation and capabilities of Mininet-WiFi and Containernet can be found in [10] and [11], respectively.

We conducted experiments with 802.11, as provided by Mininet-WiFi; however, the proposed architecture is independent of the underlying wireless technology and can be implemented in 5G networks. Research on multi-access edge-dominated networks has already been conducted and PoC experiments have been presented that use the SDN control features and wireless emulation medium supported by Mininet-WiFi [27]–[30].

The main objective of our reproducible PoC experiments was to showcase the viability and potential benefits of our proposal to support E2E mobility. We consider the V2X scenario where mobility should be handled as a norm. Our experimental evaluation exhibits the potential benefits of our proposal to deal with the seamless service migration between MEH of different MNOs, with guaranteed service continuity. Implementation choices used to emulate the MEC environment for V2X communication were identified based on a common environment.

Vehicles were emulated as wireless hosts capable of connecting with eNodeBs and APs. Radio Access Technology (RAT) is modeled by configuring data rates and the RAN node range to resemble eNodeB and AP functionalities. eNodeBs and APs are OpenFlow-capable switches, connected to their respective MEH and remote cloud. MEH is modeled as a virtual instance that is a single unit having one or more Docker containers linked together by a single switch. The V2X ME application can be hosted as a Docker container at either MEH or in the remote cloud. For the SDN controller, RYU was chosen for our experimental evaluation because it provides support to the latest OpenFlow 1.3 features such as meter-tables. The meter-tables can be used to enforce rate limiting and enable differentiated services. To showcase the potential benefits of our proposal for service migration, we realize realistic scenarios where DRS runs as a container running over two workstations that use the Ubuntu operating system, equipped with Intel Core i5 and 8GB RAM. To evaluate the performance of our proposal in different settings, the system bandwidth is varied and taken as 50 and 100 Mbps to resemble congested network conditions. The two nodes are physically separated from each other by introducing delay using the Linux Traffic Control (TC) utility. The Linux TC utility provides the ability to configure the kernel packet scheduler, to reproduce a realistic MEH deployment scenario.

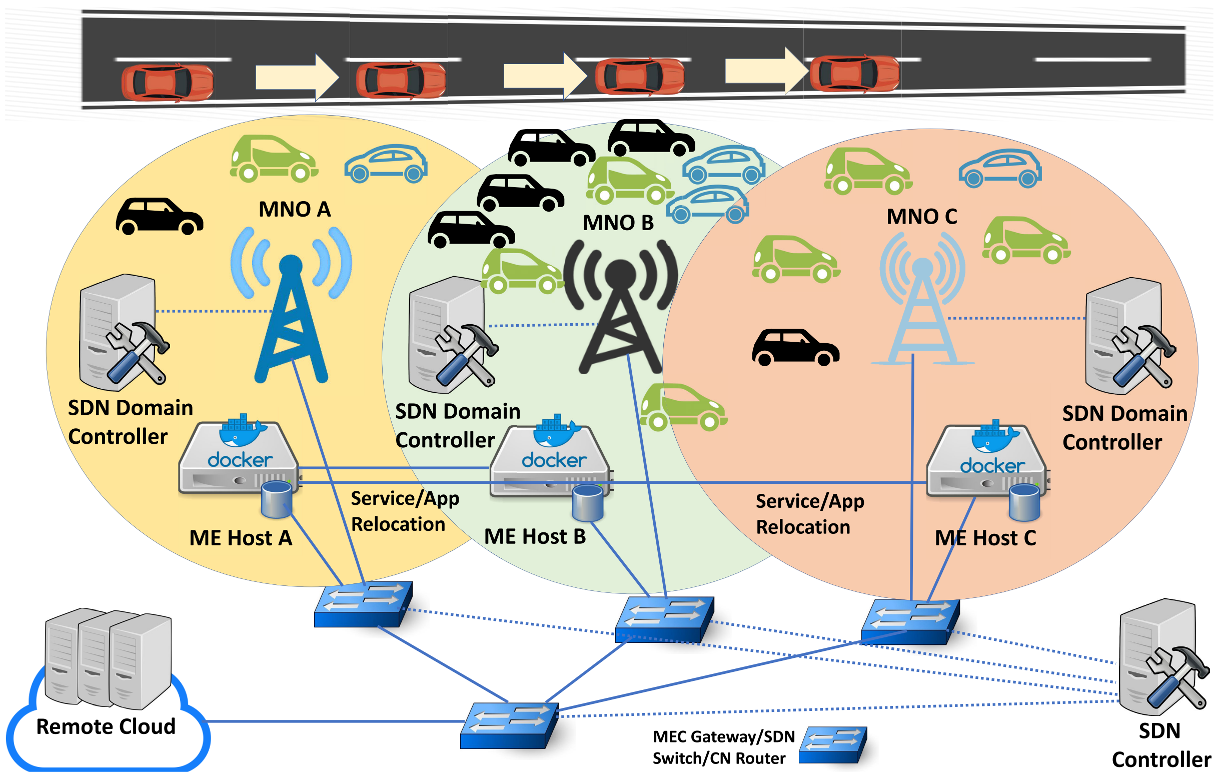


FIGURE 5. **V2X mobility use case.**

A. V2X MOBILITY SCENARIO

Fig. 5 represents the reference topology for our V2X mobility scenario that is intended to show the effectiveness of our proposal. In the topology, each network has its own SDN domain controller that makes use of the RNIS and LS to predict the need for service relocation, identify capable MEH for service relocation, and trigger handover actions to the underlying network of target MEH. It is important to note that for RNIS and LS, we used a packet manipulation tool for sniffing/sending beacon frames called python scapy. Therefore, the vehicle assists the process by sending the network information packets, e.g., RSSI, available networks and MEH. The handover management is triggered when the RSSI falls below a certain threshold. This threshold is defined and managed by Background Scanning (bgscan), that is responsible for requesting background scans for the purpose of roaming within an ESS. In particular, we use the learn module provided by bgscan, a module that stores the list of all discovered Basic Service Sets (BSS) and their operating frequencies in the Extended Service Set (ESS). The centralized SDN controller is responsible for the configuration of the corresponding flow tables, traffic routing updates and performing ME application bandwidth management to maintain service continuity.

Three networks are considered in the reference topology, as seen in Fig. 5, where the target vehicle is served by the containerized ME application in MEH A of MNO A. We considered a heterogeneous RAT deployment where MNOs B and C have overlapping coverage areas. The target vehicle moves from the serving area of MEH A and its underlying network towards the other MEH. In our proposed case, the SDN domain controller of the respective network predicts the need for service relocation, as it observes the RSSI of the current network is falling consistently. It initiates the request for service migration towards the target MEH and its network. As multiple candidate MEH are available, e.g., MEH B and C, the SDN controller selects the MEH and its network for service relocation considering its computational load and available resources to enhance the user QoE. We assume that MEH B associated with MNO B has more computational load by emulating multiple vehicles that are currently being served by it. Upon receiving the request for service relocation, the target MEH pulls the ME application from the DRS running at the source MEH and captures the application state by mounting the Docker volume from the multi-host shared storage. As the SDN domain controller initiates the process of service migration before the actual handover takes place, the service can be migrated without any performance degradation. As the vehicle enters the overlapping coverage areas of MEH B and C, the SDN domain controller triggers handover to the underlying network of the target MEH. The centralized SDN controller reconfigures the traffic routing updates to reroute the vehicle traffic from the source MEH towards the relocated application in target MEH. The centralized SDN controller also assigns the bandwidth as required by the relocated application by using the per-flow meter-tables functionality of OpenFlow 1.3 protocol. In this paper, it is assumed that the ME application serving the vehicle requires a consistent bandwidth of 5 Mbps to ensure service continuity.

Fig. 6 presents the results obtained in terms of latency, bandwidth, and service pre-relocation latency. In our proposed case, as the vehicle moves away from the coverage area of MEH A and its underlying network towards others at $t = 30$ sec, it maintains significantly lower and stable latency as compared to the conventional cases. In our proposal, unlike the traditional RSSI-based handover and service migration, the SDN controller initiates the service migration process towards MEH C and triggers handover to MNO C as it has less computational load as compared to the MNO B which has slightly better RSSI. In the conventional case 1, the service is not migrated, and the vehicle is served by the application hosted in the remote cloud following a traditional RSSI-based handover to MNO B. In the conventional case 2, SDN doesn't take into account the availability of the required resources and initiates the service migration process towards MEH B of MNO B as it has better RSSI as compared to MNO C. This results in increased latency as the vehicle travels towards MNO B at $t = 30$ sec, as can be seen in Fig. 6(a). Fig. 6(b) represents the performance of our proposal in terms of bandwidth, whereas the vehicle moves from the serving area of MEH A, the SDN controller performs the afore-mentioned mobility management process towards MEH C of MNO C as it has less computational load as compared to MEH B of MNO B. As a result, the vehicle achieves a consistent bandwidth performance of 5 Mbps as required by the V2X ME application to maintain service continuity. In the conventional case, when there are no inter-operator interactions, the service is migrated towards MEH B of MNO B, which has better RSSI but not enough resources to support the required bandwidth for service continuity. As a result, MNO B limits the bandwidth assigned to the relocated ME application session to 3 Mbps by using the OpenFlow v1.3 metering rules, as seen in Fig. 6(b). Fig. 6(c) represents the performance of the SDN-BWMS, where the ME application instance relocates to MEH C of MNO C and then requests a bandwidth update to 8 Mbps at $t = 40$ sec. The SDN controller responds to this request and allocates the requested bandwidth by installing a new flow rule as MEH C has idle computational resources. We assume that at $t = 50$ sec, a group of mobile users arrive within the coverage area of MEH C and request a dedicated network slice, that creates a significant load on MEH C. The SDN controller reacts to this change and installs a new flow for the relocated ME application session to limit its allocated bandwidth to 5 Mbps to maintain service continuity. The service pre-relocation latency is represented in Fig. 6(d), for the migration of sample container images of size 123 MB and 296 MB. The latency induced from service migration by our proposed DRS is much lower when compared to cloud-based service migration and traditional copy and transfer techniques supported by Docker and Linux.

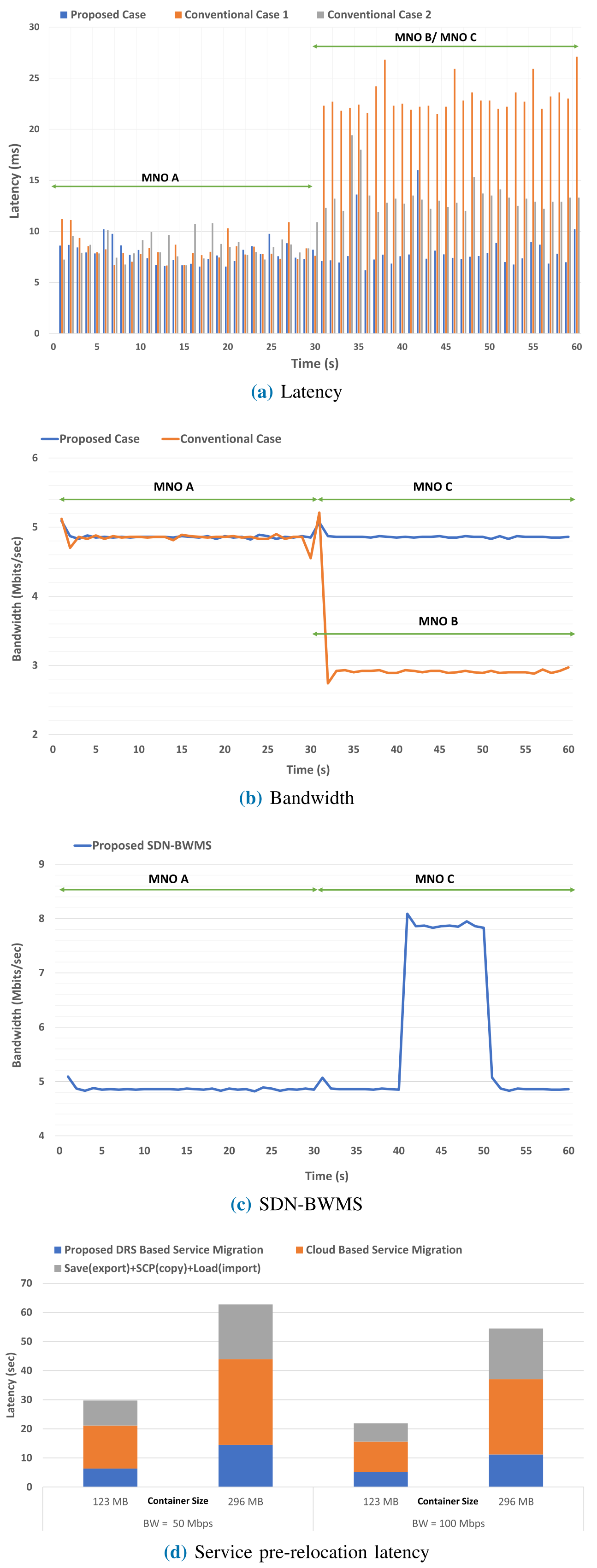


FIGURE 6. Performance evaluation of proposed architecture.

The service pre-relocation latency is further reduced in our proposal and for the cloud-based case when the system bandwidth is increased to 100 Mbps. This increase in system bandwidth does not impact significantly on the latency caused by the traditional copy and transfer techniques, because of the dependencies on the system hardware resources as they involve compression and extraction processes. The copy and transfer techniques used for the comparison with the proposed DRS are ''docker save'' for exporting the container image as a tar file, SCP for securely copying the file from one host to the other, and ''docker load'' to import the tar file to create an image for the container. The cloud, in this case, is assumed to be the official cloud-based service provided by the Docker platform and is a ''docker hub'' for managing and distributing containerized images. The service pre-relocation latency induced by our proposal is also significantly lower than the similar containerization-based approach for service migration reported in [23].

VI. CONCLUSION AND FUTURE WORK

We have introduced an SDN enhanced architecture for MEC that integrates containers, which are a lightweight form of virtualization, for E2E mobility support and QoS management. As with the development of auto-driving and unmanned aerial vehicles, the provision of mobility and reliable network connectivity becomes imperative for service continuity, which is an exacting challenge that needs to be addressed. In particular, for the case of long-distance movements, where the service migration and interoperability between edge clouds of different mobile network operators are yet to be addressed. To deal with these challenges, our proposal customized the specific functionalities and architectural components of the MEC, such as integrating SDN, introducing a new ME DRS, and integrating containerization technology. Our proposed architecture is fully aligned with the reference architecture of MEC, as defined by the ETSI. To showcase the viability of our proposed framework, we conducted PoC experiments for the V2X mobility scenario. The results show that near-seamless service migration and handover operation between MEH of different networks is achieved. As the development of the role of SDN in the edge computing framework is still in its infancy, our proposed framework emphasizes the importance of conducting more in-depth investigations and extensive evaluation in this field.

For future work, we will extend the proposed system to support the network slicing framework for mobility management between heterogeneous network slices across edge clouds. We plan to investigate and develop methods to support different slice mobility patterns across edge clouds, e.g., full slice mobility for a slice serving a group of mobile users and partial slice mobility for a sub-set of slice resources. As the edge clouds have limited resources, efficient slice mobility management procedures should be developed to ensure seamless service continuity across different edge clouds. We intend to study the applicability of the edge clouds interoperability offered by our proposed architecture to develop an inter-slice resource sharing and federation model for different slice mobility patterns across edge clouds of different networks.

REFERENCES

[1] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, and J. P. Jue, ''All one needs to know about fog computing and related edge computing paradigms: A complete survey,'' *J. Syst. Archit.*, vol. 98, pp. 289–330, Sep. 2019.

[2] T. Taleb, K. Samdanis, B. Mada, H. Flinck, S. Dutta, and D. Sabella, ''On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration,'' *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[3] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, ''Survey on multi-access edge computing for Internet of Things realization,'' *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2961–2991, Jun. 2018.

[4] MEC ETSI ISG for Multi-Access Edge Computing (MEC) and ETSI Industry Specification Group (ISG), ''Mobile edge computing (MEC); end to end mobility aspects,'' ETSI, Sophia Antipolis, France, Tech. Rep. GR MEC 018, Oct. 2017.

[5] MEC ETSI ISG for Multi-Access Edge Computing (MEC) and ETSI Industry Specification Group (ISG), ''Multi-access edge computing (MEC); study on MEC support for V2X use cases,'' ETSI, Sophia-Antipolis, France, Tech. Rep. GR MEC 022, Sep. 2018.

[6] MEC ETSI ISG for Multi-Access Edge Computing (MEC) and ETSI Industry Specification Group (ISG), ''Multi-access edge computing (MEC); phase 2: Use cases and requirements,'' ETSI, Sophia-Antipolis, France, Tech. Rep. GS MEC 002, Oct. 2018.

[7] R. Morabito, V. Cozzolino, A. Y. Ding, N. Beijar, and J. Ott, ''Consolidate IoT edge computing with lightweight virtualization,'' *IEEE Netw.*, vol. 32, no. 1, pp. 102–111, Jan. 2018.

[8] G. Avino, M. Malinverno, F. Malandrino, C. Casetti, and C. F. Chiasserini, ''Characterizing docker overhead in mobile edge computing scenarios,'' in *Proc. Workshop Hot Topics Container Netw. Netw. Syst. (HotConNet)*, 2017, pp. 30–35.

[9] J. Islam, E. Harjula, T. Kumar, P. Karhula, and M. Ylianttila, ''Docker enabled virtualized nanoservices for local IoT edge networks,'' in *Proc. IEEE Conf. Standards Commun. Netw. (CSCN)*, Oct. 2019, pp. 1–7.

[10] R. R. Fontes, S. Afzal, S. H. B. Brito, M. A. S. Santos, and C. E. Rothenberg, ''Mininet-WiFi: Emulating software-defined wireless networks,'' in *Proc. 11th Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2015, pp. 384–389.

[11] M. Peuster, J. Kampmeyer, and H. Karl, ''Containernet 2.0: A rapid prototyping platform for hybrid service function chains,'' in *Proc. 4th IEEE Conf. Netw. Softwarization Workshops (NetSoft)*, Jun. 2018, pp. 335–337.

[12] F. Giust, V. Sciancalepore, D. Sabella, M. C. Filippou, S. Mangiante, W. Featherstone, and D. Munaretto, ''Multi-access edge computing: The driver behind the wheel of 5G-connected cars,'' *IEEE Commun. Standards Mag.*, vol. 2, no. 3, pp. 66–73, Sep. 2018.

[13] Z. Zhou, J. Feng, Z. Chang, and X. Shen, ''Energy-efficient edge computing service provisioning for vehicular networks: A consensus ADMM approach,'' *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 5087–5099, May 2019.

[14] H. Liao, Z. Zhou, X. Zhao, L. Zhang, S. Mumtaz, A. Jolfaei, S. H. Ahmed, and A. K. Bashir, ''Learning-based context-aware resource allocation for edge computing-empowered industrial IoT,'' *IEEE Internet Things J.*, early access, Dec. 31, 2020, doi: 10.1109/JIOT.2019.2963371.

[15] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong, ''An air-ground integration approach for mobile edge computing in IoT,'' *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 40–47, Aug. 2018.

[16] H. Li, K. Ota, and M. Dong, ''Learning IoT in edge: Deep learning for the Internet of Things with edge computing,'' *IEEE Netw.*, vol. 32, no. 1, pp. 96–101, Jan. 2018.

[17] V. Balasubramanian, S. Otoum, M. Aloqaily, I. Al Ridhawi, and Y. Jararweh, ''Low-latency vehicular edge: A vehicular infrastructure model for 5G,'' *Simul. Model. Pract. Theory*, vol. 98, Jan. 2020, Art. no. 101968.

[18] V. Balasubramanian, F. Zaman, M. Aloqaily, I. A. Ridhawi, Y. Jararweh, and H. B. Salameh, ''A mobility management architecture for seamless delivery of 5G-IoT services,'' in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.

[19] H. Abdah, J. P. Barraca, and R. L. Aguiar, ''QoS-aware service continuity in the virtualized edge,'' *IEEE Access*, vol. 7, pp. 51570–51588, 2019.

[20] P. Bellavista, A. Corradi, L. Foschini, and D. Scotece, "Differentiated Service/Data migration for edge services leveraging container characteristics," *IEEE Access*, vol. 7, pp. 139746–139758, 2019.

[21] R. A. Addad, T. Taleb, H. Flinck, M. Bagaa, and D. Dutra, "Network slice mobility in next generation mobile systems: Challenges and potential solutions," *IEEE Netw.*, vol. 34, no. 1, pp. 84–93, Jan. 2020.

[22] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 140–147, Feb. 2018.

[23] C. Campolo, A. Iera, A. Molinaro, and G. Ruggeri, "MEC support for 5G-V2X use cases through docker containers," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2019, pp. 1–6.

[24] I. Farris, T. Taleb, A. Iera, and H. Flinck, "Lightweight service replication for ultra-short latency applications in mobile edge networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[25] MEC ETSI ISG for Multi-Access Edge Computing (MEC) and ETSI Industry Specification Group (ISG), "Multi-access edge computing (MEC); framework and reference architecture," ETSI, Sophia-Antipolis, France, Tech. Rep. GS MEC 003, Jan. 2019.

[26] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A scalable and quick-response software defined vehicular network assisted by mobile edge computing," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 94–100, Jul. 2017.

[27] R. Dos Reis Fontes, C. Campolo, C. Esteve Rothenberg, and A. Molinaro, "From theory to experimental evaluation: Resource management in software-defined vehicular networks," *IEEE Access*, vol. 5, pp. 3069–3076, 2017.

[28] C. Campolo, R. Fontes, A. Molinaro, C. E. Rothenberg, and A. Iera, "Slicing on the road: Enabling the automotive vertical through 5G network softwarization," *Sensors*, vol. 18, no. 12, p. 4435, Dec. 2018.

[29] J. Al-Badarneh, Y. Jararweh, M. Al-Ayyoub, R. Fontes, M. Al-Smadi, and C. Rothenberg, "Cooperative mobile edge computing system for VANET-based software-defined content delivery," *Comput. Electr. Eng.*, vol. 71, pp. 388–397, Oct. 2018.

[30] J. Al-Badarneh, Y. Jararweh, M. Al-Ayyoub, M. Al-Smadi, and R. Fontes, "Software defined storage for cooperative mobile edge computing systems," in *Proc. 4th Int. Conf. Softw. Defined Syst. (SDS)*, May 2017, pp. 174–179.

SYED DANIAL ALI SHAH received the B.S. degree in telecommunication engineering from the University of Engineering and Technology, Taxila, Pakistan, in 2016, and the master's degree in electronics engineering from Incheon National University (INU), South Korea, in 2018. He is currently pursuing the Ph.D. degree with the School of Engineering, RMIT University, Melbourne, VIC, Australia. He was a Research Assistant with INU, for a period of two years. His research interests include 5G/6G, cloud/edge computing, software-defined wireless networks, vehicular networks, and network design.

MARK A. GREGORY (Senior Member, IEEE) received the Ph.D. degree from RMIT University, Melbourne, VIC, Australia, in 2008. He is currently an Associate Professor with the School of Engineering, RMIT University. His research interests include telecommunications, network design, 5G/6G, and technical risk management. He is a Fellow of the Institute of Engineers Australia. In 2009, he received the Australian Learning and Teaching Council Citation for an Outstanding Contribution to Teaching and Learning. He is the General Co-Chair of ITNAC. He is also a Managing Editor of the two international journals *AJTDE* and *IJICTA*.

SHUO LI (Member, IEEE) received the B.S and Ph.D. degrees from the City University of Hong Kong, in 2009 and 2014, respectively. She was a Lecturer with Tianjin University, China, from 2014 to 2017. She is currently a Lecturer with the School of Engineering, RMIT University, Australia. Her research interests include telecommunications, analysis and design of optical networks, 5G/6G core networks, and underwater sensor networks.

RAMON DOS REIS FONTES received the Ph.D. degree in electrical and computer engineering from the State University of Campinas, Brazil, in June 2018. He was a Visiting Ph.D. Student with the DIANA Team, Sophia Antipolis, France, in 2016. He is currently a Professor in computer science with the Federal Institute of Education, Science and Technology of Bahia (IFBA), Brazil. His main research interests include software defined wireless networks (SDWN), vehicular networking, and future internet architecture.