On Enabling 5G Automotive Systems Using Follow Me Edge-Cloud Concept

Abdelkader Aissioui, Adlen Ksentini, Abdelhak Mourad Gueroui, and Tarik Taleb

***Abstract*—One of the key targets of the upcoming 5G system is to build a mobile network architecture that supports not only classical mobile broadband applications (i.e., Internet and IMS), but also vertical industry services, such as those of automotive systems, e-health, public safety, and smart grid. Vertical industry is known to have specific needs that cannot be sustained by the current cellular networks. More notably, automotive systems require strict quality of service in terms of ultrashort latency for vehicle-to-infrastructure/network (V2I/N) communications. In this paper, we introduce the Follow Me edge-Cloud (FMeC) concept, leveraging the mobile edge computing (MEC) architecture to sustain requirements of the 5G automotive systems. Assuming that automotive services are deployed on MEC entities, FMeC ensures low-latency access to these services by guaranteeing that vehicles (i.e., as well as user equipment on board vehicles) always connect to nearest automotive service. Besides the FMeC architecture, our contribution in this paper consists in presenting a projection of the FMeC solution on an automated driving use case that integrates automotive and Telco infrastructures, to realize the vision of future 5G automotive systems. We introduce the envisioned software defined networking/OpenFlow-based architecture and our mobility-aware framework based on a set of building blocks that permit achieving the automated driving requirements within 5G network. The evaluation results, obtained conjointly through theoretical analysis and computer simulation, show that our proposed solution outperforms baseline approaches in meeting the automated driving latency requirement and minimizing the incurred global cost.**

***Index Terms*—MEC, SDN, automotive driving, verticals, 5G, service mobility, follow me edge, follow me cloud.**

I. Introduction

CLOUD Computing has been gaining lots of momentum for its flexibility, elasticity, and cost-efficiency. The basic tenet of cloud computing is that users do not need to be concerned about the placement of their services nor the provisioning of the required resources; principally offered following the pay-per-use model. While for most elastic web applications, the relative distance between end-users and service end-points does not affect the perceived Quality of Experience (QoE), highly-interactive applications are sensible to latency and jitter. In the absence of an explicit Quality of Service (QoS) control mechanism in the network, the only way to improve QoE for such applications is by locating corresponding servers in the vicinity of end users. Such an approach, largely exploited by Content Delivery Networks (CDNs), can be further advanced in the era of cloud computing. Assuming that several federated small-scale data centers are deployed at the edges of the Internet (i.e., Federated Edge Cloud), service providers may leverage them to optimally locate service instances as close as possible to their respective users. In such context, the mobility of mobile users makes such location decisions/planning difficult. To cope with such issue, the Follow Me Cloud (FMC) principle was introduced in [1], wherein mobile user equipment are always connected via the optimal data anchor gateway to access data and services from the optimal and geographically nearest data centers.

To ensure an optimal end-to-end connection to the cloud for mobile users, Virtual Machines (VM) (i.e., service) can be migrated between data centers when deemed appropriate [2], [3]. Accordingly, services are always provided from data centers that are geographically optimal for the current locations of end-users. It is worth noting that VM migration shall be seamless and transparent to users. Thus, on-going sessions between user equipment and servers shall not be interrupted and connections do not need to be reestablished, even if users and/or servers (i.e., hosting services) change location. Besides improving users' Quality of Service/Quality of Experience, FMC allows preserving operators' network resources by offloading network traffic to data centers through the nearest points compared with users' locations.

Mobile Edge Computing (MEC) [4] has recently emerged as a promising technique, the core idea of MEC is to move computation closer to users, whereby small servers or micro-data centers that can host cloud applications are distributed across the network and connected directly to entities, such as cellular base stations, at the mobile network edge. MEC is also expected to be more robust than traditional centralized Cloud computing systems [5], because it is distributed and is thus less impacted by failures at a centralized point. The idea of distributing cloud servers at the mobile network edge is also known as cloudlets [5], edge computing [6], and fog computing [7]. In all these techniques, each set of servers or each micro-data center

Manuscript received November 12, 2016; revised May 27, 2017 and November 28, 2017; accepted January 16, 2018. Date of publication February 12, 2018; date of current version June 18, 2018. This work was supported in part by the Academy of Finland Project CSN under Grant 311654 and in part by the European Union's Horizon 2020 research and innovation program under the 5G!Pagoda project under Grant 723172 and EU H2020 5G-Transformer Project under Grant 761536. The review of this paper was coordinated by Dr. B. Canberk. *(Corresponding author: Tarik Taleb.)*

A. Aissioui and A. M. Gueroui are with the LI-PaRAD/University of Versailles Saint-Quentin-En-Yvelines, Versailles Cedex 78035, France (e-mail: aissiouikader@gmail.com; mourad.gueroui@uvsq.fr).

A. Ksentini is with the EURECOM, Biot Sophia Antipolis Cedex 06410, France (e-mail: adlen.ksentini@eurecom.fr).

T. Taleb is with the Computer and Information Security Department, Sejong University, Seoul 143-747, South Korea, and also with the School of Electrical Engineering, Aalto University, Aalto 02150, Finland (e-mail: tarik.taleb@aalto.f).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2018.2805369

0018-9545 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

is responsible for a small geographical area, although some servers/micro-data center may not be directly connected to the base station. The MEC paradigm permits offering environments characterized by low latency, high bandwidth and location-awareness that can be leveraged by applications; opening the way for the development of several new applications.

It is generally agreed that the 5G mobile system [8] will largely benefit from MEC in order to enable novel services from Vertical Industries, particularly those covering automotive industry; which have several constraints that cannot be accommodated by the current 4G mobile networks. Knowing the benefit to fulfill automotive and particularly vehicular needs, in term of business market, recently, the 3rd Generation Partnership Project's (3GPP) has defined the specification of Cellular Vehicle-to-Everything (C-V2X) communication for the next version of the LTE standard – Release 14 [9]. The C-V2X communication is designed to operate in several modes: Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I) and Vehicle-to-Pedestrian (V2P) for shorter-range direct communications, as well as Vehicle-to-Network (V2N) for longer-range network-based communications. Among those new envisioned services, the support of relevant applications is highly challenging, mainly due to the ultra low-latency access requirement. For instance, automotive driving needs to access to computing services in a range of milliseconds. Though 5G system aims for ensuring a latency of 1 ms (by using new techniques at the physical and MAC layers), supporting automotive services remains challenging, if the latter are hosted in a centralized Cloud. In this context, locating services at the edge of the mobile network will permit to maintain low-latency access to services. Moreover, UEs are highly mobile, which leads to increase in the communication path (i.e., latency) to the service even if the latter is hosted in MEC (i.e., moving far from the current service location). To overcome these limitations, we are proposing the FMeC framework (i.e., covering the architecture and enabling algorithms), which couples the Follow Me Cloud (FMC) and Mobile Edge Computing (MEC) concepts. FMeC ensures low-latency access by: (i) considering that cloud services are located at the edge; (ii) enabling services migration among the edge, ensuring that users are always connected to the optimal edge. FMeC is dedicated to cover automotive vertical industry, and specifically, the automated driving use case within the 5G context. The adopted methodology aims at presenting all the components enabling FMeC: (i) the architecture and its elements; (ii) the enabling algorithms (i.e., metrics monitoring, mobility pattern updating, QoS evaluation, optimal edge-Cloud selection and service migration); (iii) the traffic management using SDN/OpenFlow paradigm.

The main challenge addressed by FMeC is to ensure low-latency access to applications, which is considered as the main requirement of the 5G Ultra Reliable Low Latency Communications (URLLC) services. We recall that most of existing contributions optimize the lower layers of the 5G systems (MAC and physical). These optimizations are able to reduce only the Radio Access Network (RAN) access delay to less than 5 ms, but if the service is still hosted in the cloud, the latency cannot be reduced to less than 10 ms as required by the URLLC service. Therefore, we believe that FMeC can be a complementary solution to the RAN optimization solutions to reduce the latency access to services. Thanks to hosting the service in MEC, while ensuring connection to the optimal Edge via service migrations, FMeC allows to reduce the latency to less than 10 ms.

The remainder of this paper is organized as follows. Section II discusses some related work. In Section III, we present the system description and functioning. The Section IV studies a use case implementation for 5G automotive systems. While Section V provides an evaluation of the solution, Section VI discusses the obtained results. The paper concludes in Section VII.

II. Related Work

A. The FMC Concept

The FMC concept was initially proposed in [1]. It was dedicated to the case where all mobility management procedures are handled at the 3GPP domain. In [2], an analytical model is presented to evaluate the performance of the FMC mechanism, while in [3] a Markov-Decision Process (MDP) was introduced for the service migration procedure. In [10], an OpenFlow-enabled implementation of FMC was proposed. The paper describes the components needed to enable FMC, in particular the detection of users' movements, the decision logic for migrating services and the method for making migration seamless. The authors presented a proof-of-concept of FMC based on VMware (i.e., VMotion for live VM migration), a NOX-based FMC controller and OpenFlow switches. As the latter have to handle multiple per-flow rules, scalability of FMC rules became an issue. A remedy could be the distributed architecture of FMC controller presented in [11] or the elastic distributed SDN controller for follow-me cloud proposed in [12], [13], which is a two-level hierarchical architecture. A first level includes the global controller, and second level that integrates several local controllers deployed on-demand via NFV; created on demand and depending on the global system load. In [15], the authors use the concept of identifier/locator separation of edge networks to support service continuity in FMC. Effectively, in case of a VM migration, the old IP address serves as an identifier and the new IP address serves as a locator for the mobile node. Whilst this operation ensures somehow service continuity, it incurs an important overhead for manipulating the locator/identifier values on the edge networks. In [16], the authors proposed another implementation of FMC based on LISP (Locator/Identifier Separation Protocol), whereby the main goal is to render FMC independent from the underlying radio access technology. Thanks to LISP features, both users' mobility and VM migration are jointly managed at the same control plane. Besides the LISP entities, all FMC entities were implemented as virtualized network functions running on VMs, facilitating further the concept of carrier cloud [17]. The results obtained from a real-life testbed of the proposed LISP-based FMC architecture showed that the architecture achieved its main design goals, transferring users, services in the order of milliseconds and with very minimal downtime. In [18] the authors proposed a Proxy Mobile IPv6 (PMIPv6)-based architecture for Follow Me Cloud (FMC). The envisioned approach consists of two parts, the first is a PMIPv6-based inter-domain mobility management support based on Distributed Mobility Management (DMM), and the second is a control plane based on Software Defined Networking – SDN/OpenFlow, which exploits the mobility information delivered by the PMIPv6 inter-domain mobility management support to decide on triggering the migration of services within the cloud. Integrating SDN/Openflow rules with PMIPv6 permits removing the complexity and workload associated with tunneling in mobility management. Results obtained via analysis were encouraging and showed the advantages of PMIPv6-based FMC in comparison to the state of the art mobility management protocols.

B. The MEC Paradigm

The user mobility becomes a key factor in MECs, a preliminary work on mobility-driven service migration based on Markov Decision Processes (MDPs) is given in [3], which mainly considers one-dimensional (1-D) mobility patterns with a specific defined cost function. Standard solution procedures are used to solve this MDP, which can be time consuming especially when the MDP has a large number of states. Due to real-time dynamics, the cost functions and transition probabilities of the MDP may change rapidly over time, thus it is desirable to solve the MDP in an effective manner. With this motivation, a more effective solution to the 1-D mobility case was proposed in [19], where the transmission and migration costs are assumed to be constant whenever transmission/migration occurs. In [20], the authors present one-dimensional (1-D) and two-dimensional (2-D) mobility service migration solutions based on Markov Decision Process (MDP). The solutions' formulation captures general cost models and provides a mathematical framework to design optimal service migration policies. The authors showed that the resulting MDP is exact for uniform one-dimensional mobility, while it provides a close approximation for uniform two-dimensional mobility with a constant additive error term. They also propose a new algorithm and a numerical technique for computing the optimal solution which is significantly faster in computation than traditional methods based on value or policy iteration.

C. The Connected Vehicles

The Cloud-based processing of connected vehicles' data has attracted broad interest in both automotive and telecom industries. In [21], different architectures and approaches have been developed. Moreover, several challenges, solutions and advantages have been discussed regarding the integration of connected vehicles within the internet of things ecosystems. In [22], the randomized transmission of Floating Car Data (FCD) was proposed in order to reduce the communication costs in traffic information sharing systems. The solution relies on an Information Cost Model to quantify a trade-off relationship between the communication cost of the system and the accuracy of information, in addition to a randomized method to avoid redundant transmissions. In [23], the collection, dissemination and multi-hop forwarding of vehicle FCD for LTE-based car-to-car as well as car-to-infrastructure communication has been analyzed and evaluated, in terms of efficiency and packet losses, via computer simulation. The results show that the multi-hop extension leads to an improvement of the LTE4V2X framework performances, for applications based on both data collection and data dissemination. In [24], authors studied the impact of car-to-cloud communication on LTE infrastructure and network capacity. They presented a simulation-based analysis of car-to-cloud data traffic, with a detailed environment model including a precise road map and cell location. The mobility of vehicles was simulated for different traffic states (i.e., free flow, traffic jam) using SUMO simulator, while the proposed LTE-based car-to-cloud network communication was founded on empirical measurements. Simulation results indicated an average data rate of 4821.1 kbps in free flow traffic, which is reduced by a factor of seven in traffic jam state. The upper and lower data rates estimation provided by this work serve for network resource planning in terms of Resource Blocks (RB) and resource scheduling for car-to-cloud type services.

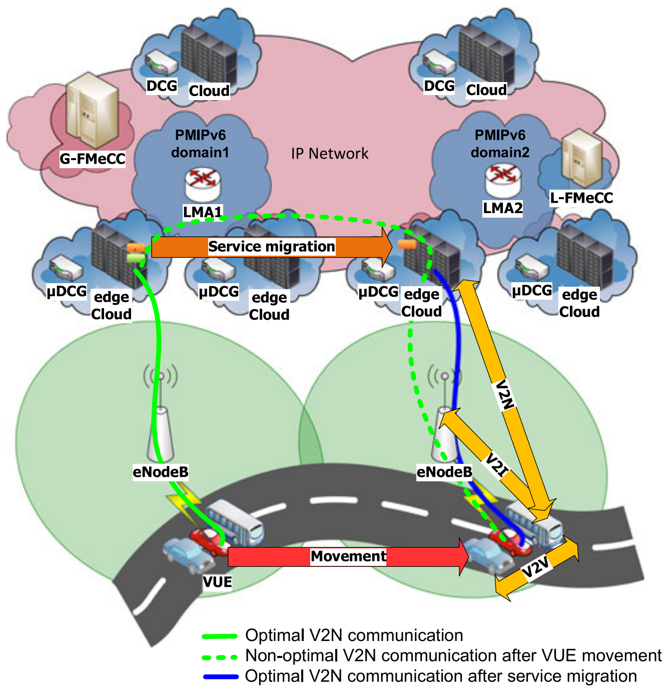


Fig. 1. The envisioned PMIPv6-based FMeC architecture.

TABLE I
THE FMEC ARCHITECTURE ELEMENTS

Element	Description
G-FMeCC (Global FMeC Controller)	SDN/OpenFlow controller for all domains
L-FMeCC (Local FMeC Controller)	SDN/OpenFlow controller for a specific domain
eNodeB (Evolved Node B)	The cellular base station
VUE (Vehicle User Equipment)	The vehicle user equipment
DCG (Data Center Gateway)	The gateway of the datacenter or Cloud
μDCG (micro Data Center Gateway)	The gateway of the micro-datacenter or edge-Cloud

III. FMEC SYSTEM DESCRIPTION AND FUNCTIONING

Our proposed Follow Me edge-Cloud (FMeC) architecture is depicted in Fig. 1. Table I introduces the acronyms of all architecture elements along with their description. FMeC takes advantage of both MEC and FMC by joining these two concepts together into a unique approach tailored for environments where mobile users and cloud-based services are subject to constraints of movements and migrations, permitting thereby the cloud resources to be brought to the mobile network edge, closer to mobile users on one hand, and the cloud-based services to follow mobile users movement over the mobile network on the other hand.

Firstly, we begin by addressing the elements permitting the integration of FMC within MEC. Secondly, we project our solution on a SDN/NFV architecture implementation dedicated for the future 5G automotive systems and more specifically for automated driving systems. Our proposed solution considers a multi-PMIPv6 domains environment, whereby each PMIPv6 domain comprises three parts: (i) the mobile network operator part, (ii) the cloud service part, (iii) and the edge-Cloud service part. The SDN/OpenFlow and NFV architecture of the system is constructed of a Global Follow Me edge-Cloud Controller (G-FMeCC), multiple Local Follow Me edge-Cloud Controllers (L-FMeCCs) and a set of OpenFlow-enabled devices which are Local Mobility Anchor gateways LMAs (i.e., P-GWs), Data Center Gatways (DCGs) and micro-Data Center Gateways (μ DCG). The network access part consists of a set of base stations (i.e., eNodeBs). The details of the architecture and the functionality of the distributed SDN/NFV controllers for FMC (and FMeC) are outside the scope of this paper. Interested readers may refer to the authors' work on distributed FMCC control-plane architecture in [12] and [13].

The goal of our solution is to propose a framework based on the introduced FMeC concept, which is the merging result of both FMC and MEC paradigms. This framework permits the provisioning of ultra-low end-to-end delay/latency between the mobile end-user and the hosted services by finding a tradeoff between reducing the global cost of Cloud/edge-Cloud service migration and meeting the requirement of end-users in terms of the perceived end-to-end QoS. As illustrated in Fig. 1, the target network consists of multiple PMIPv6 domains. The number of these domains is denoted by N. In each domain, a pool of eNodeBs is geographically deployed covering the domains area. A set of federated edge-Cloud services is deployed in geographically distributed micro-data centers, placed on the edge of the mobile network domains. The number of micro data centers is denoted by M. We assume that each eNodeB of a domain pool can access each of the federated edge-Cloud services. The edge clouds are assumed to be interconnected via the backhaul network.

IV. 5G Automotive Systems—Automated Driving Service

In this section, we describe how 5G automotive systems can be enabled using our envisioned FMeC architecture. Automated driving services are delay-sensitive and require ultra-low latency. The envisioned architecture is fully distributed and is based on SDN, NFV and MEC as described in Section III. In addition, we assume that eNodeBs offer automotive Vehicle-to-Infrastructure/Network (V2I/N) communication under their LTE covered areas [9]. In our work, each eNodeB element is assumed to be OpenFlow-enabled and provides a monitoring module that provides information on key performance indicators (e.g., delay, throughput, workload). In this paper, we focus only on the delay performance indicator. Information on current delays are collected in real time and at regular $T_{monitoring}$ time intervals by the eNodeBs monitoring modules. They are sent to the respective domain's FMeCC SDN controller of the eNodeB entity. Regarding the vehicular network communication, and particularly for this work, we mainly focus on the Vehicle-to-Infrastructure/Network (V2I/N) communication network, while Vehicle-to-Vehicle (V2V) and Vehicle-to-Pedestrian (V2P) communication networks are left for future works.

A. V2I/N Automated Driving Service—First Connection

We denote by D_j the domain of the eNodeB pool *eNodeB*$_j$, and by R_j the number of eNodeBs in this pool. For monitoring the federated edge-Cloud latency, the FMeCC of D_j maintains an updated Global Delay Indicator Monitoring Table (G-DIMTab as shown in Table III) for each pool member *eNodeB*$_j^k$ of domain D_j.

The acronyms of the system defined tables, table fields and global time/velocity parameters along with their description are summarized in Table II. When a vehicle end user requests an automated driving service session for the first time in a domain D_j, it sends an AUTO-DRIVE-SESSION-REQ message to its current serving eNodeB. This message contains the following information fields: *VID*, *EED*, *LOC*, *OAV* and *AV*. Upon the reception of the message, the eNodeB adds the current domain and eNodeB (i.e., *DID* and *eNID* fields) information to the original AUTO-DRIVE-SESSION-REQ message and sends it, in turn, as an OpenFlow *PacketIn* message to the D_j-FMeCC of the domain to request the most appropriate edge-Cloud destination to place the automated driving service. Based on the information extracted from the OpenFlow PacketIn message and the instantaneous information retrieved from its local D_j-FMeCC global delay indicator monitoring table (Table III), the SDN controller (D_j-FMeCC) permits the selection of the most appropriate edge-Cloud for this automated driving service session request by applying Algorithm 1. Thereafter, D_j-FMeCC updates its Global Vehicle Automated Driving Information Table (G-VADITab, see Table IV) with the original VUE AUTO-DRIVE-SESSION-REQ message information completed with the created automated driving instance and the selected edge-Cloud service (i.e., *SID* and *eCID* fields). It sends a AUTO-DRIVE-SESSION-REP as an OpenFlow *FlowMod* message including the created automated driving instance and the selected edge-Cloud service to the source eNodeB; meanwhile, it relays to the selected edge-Cloud the AUTO-DRIVE-SESSION-REQ message completed with the created automated driving instance. Upon receiving the AUTO-DRIVE-SESSION-REP message, the eNodeB installs the appropriate SDN/OpenFlow rules permitting to forward the subsequent traffic of the automotive driving session to the selected edge-Cloud, and sends the AUTO-DRIVE-SESSION-REP message to the VUE. Once received, the VUE updates its Local Vehicle Automated Driving Information Table (L-VADITab, see Table V) with the created automated driving session instance (i.e., SID) along with the

TABLE II
THE DEFINITION OF SYSTEM TABLES AND PARAMETERS

Element	Description
G-DIMTab (Global Delay Indicator Monitoring Table)	A table maintained by the SDN controller to monitor the latency between eNodeBs and the federated edge-Cloud. Its fields are: DID, eNID, eCID$_1$,..., eCID$_M$
G-VADITab (Global Vehicle Automated Driving Information Table)	A table maintained by the SDN controller to register/update VUEs' automated driving service key information. Its fields are: SID, DID, VID, eNID, eCID, EED, LOC, AV, OAV, VMP
L-VADITab (Local Vehicle Automated Driving Information Table)	A VUE local table of automated driving service key information. Its fields are: SID, DID, VID, eNID, eCID, EED, LOC, AV, OAV
SID (Service instance ID)	The instance ID of the VUE automated driving service
DID (Domain ID)	The domain ID of the automated driving service instance
VID (Vehicle ID)	ID of VUE's using the automated driving service session
eNID (eNodeb ID)	ID of the current VUE's eNodeb
eCID (edge-Cloud ID)	ID of the current VUE's edge-Cloud hosting the automated driving service
EED (End-to-End Delay)	The E2E delay from the VUE to the current edge-Cloud service
LOC (LOCation)	The VUE's current location
OAV (Old Average Velocity)	The VUE's old average velocity
AV (Average Velocity)	The current VUE's average velocity
VMP (Vehicle Mobility Pattern)	The current VUE's mobility pattern
$T_{monitoring}$	The time interval used by eNodeBs to monitor the federated edge-Cloud delay
T_{update}	The time interval used by VUEs to update the eNodeBs and the SDN controller with the mobility and QoS information
VUE-EED (Vehicle User Equipment End-to-End Delay)	The E2E delay from a specific VUE to its current edge-Cloud
CV-Thr	The velocity constant serving to predict the mobility pattern when compared with VUEs Δ velocity
ED-Thr	The maximum authorized end-to-end delay for the automated driving service

TABLE III
G-DIMTAB: D_j-FMECC GLOBAL DELAY INDICATOR MONITORING TABLE

DID	eNID	eCID$_1$	...	eCID$_M$
1	1	–	...	–
...	...	...	...	...
1	R_1	–	...	–
2	1	–	...	–
...	...	...	...	...
2	R_2	–	...	–
...	...	...	...	...
N	1	–	...	–
...	...	...	...	...
N	R_N	–	...	–

TABLE IV
G-VADITAB: D_j-FMECC GLOBAL VEHICLE AUTOMATED DRIVING INFORMATION TABLE

SID	DID	VID	eNID	eCID	EED	LOC	AV	OAV	VMP
1	1	1000	2	2	70	10	70	60	accel
2	3	2000	4	4	80	20	65	55	const
...	...	...	...	...	...	...	...	...	...

TABLE V
L-VADITAB: VUE LOCAL VEHICLE AUTOMATED DRIVING INFORMATION TABLE

SID	DID	VID	eNID	eCID	EED	LOC	AV	OAV
1	1	1000	2	2	70	10	70	60

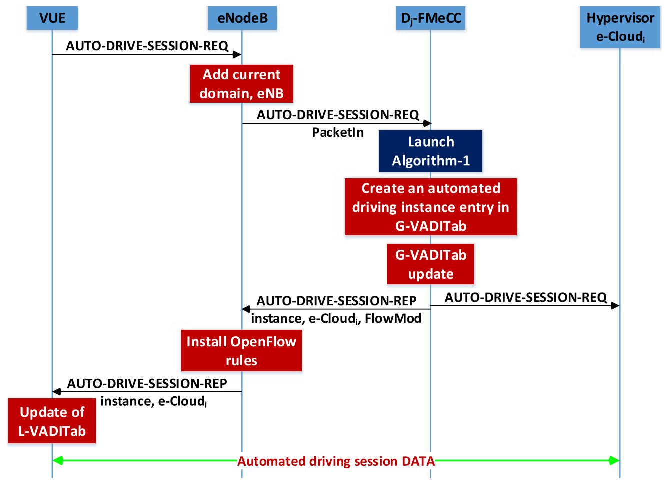


Fig. 2. Signaling flow for the request of an automated driving service.

```
Algorithm 1: First edge-Cloud Selection Algorithm (FeCSA).
Input: domain d, eNodeB eNB
Output: selected-edge-Cloud seC
1: edge-Cloud ← 0, delay ← $\infty$
2: for $i = 1$ to $M$ do
3:    if G-DIMTab[d][eNB][$eCID_i$].delay < delay then
4:       delay ← G-DIMTab[d][eNB][$eCID_i$].delay
5:       edge-Cloud ← $eCID_i$
6:    end if
7: end for
8: seC ← edge-Cloud
```

selected edge-Cloud service (i.e., eCID) information; allowing to establish the automotive driving service session. Fig. 2 shows the signaling flow corresponding to the connection setup request of the automated driving service.

Algorithm 1, dubbed as First edge-Cloud Selection Algorithm (FeCSA), is executed by the current D_j-FMeCC of domain D_j and given as follows:

Algorithm 2: VUE Mobility Pattern Updating Algorithm (VMPUA).

```
Input: instance s
Output: G-VADITab Updated VUE mobility pattern values
1:  initVel ← 0, constVelCount ← 0
2:  while new AUTO-DRIVE-SESSION-UPDATE for G-VADITab[s] do
3:      $\Delta$ velocity ← G-VADITab[s].AV − G-VADITab[s].OAV
4:      if $\Delta$ velocity $> +$ CV-Thr then
5:          G-VADITab[s].VMP ← acceleration
6:      end if
7:      if $\Delta$ velocity $< -$ CV-Thr then
8:          G-VADITab[s].VMP ← deceleration
9:      end if
10:     if $|\Delta$ velocity$| \leq +$ CV-Thr then
11:         constVelCount ← constVelCount +1
12:         if constVelCount = 1 then
13:             G-VADITab[s].VMP ← constant
14:             initVel ← G-VADITab[s].OAV
15:         else
16:             if G-VADITab[s].AV− initVel $>+$ CV-Thr then
17:                 G-VADITab[s].VMP ← acceleration
18:                 constVelCount ← 0
19:             end if
20:             if G-VADITab[s].AV− initVel $<-$ CV-Thr then
21:                 G-VADITab[s].VMP ← deceleration
22:                 constVelCount ← 0
23:             end if
24:         end if
25:     end if
26: end while
```

B. V2I/N Automated Driving Service—Dynamic QoS Adaptation

During an established automated driving service session, the vehicle end-user sends, at regular T_{update} time intervals (note that $T_{update} > T_{monitoring}$) an AUTO-DRIVE-SESSION-UPDATE message about the perceived performance of the service session to its current serving eNodeB. This QoS information is computed locally at the VUE, under the Local Vehicle Automated Driving Information Table (L-VADITab, see Table V) which, consists of the following fields: *SID*, *DID*, *VID*, *eNID*, *eCID*, *EED*, *LOC*, *OAV* and *AV*. Upon receiving the AUTO-DRIVE-SESSION-UPDATE message, the eNodeB inserts the domain and the eNodeB values and sends the resulted message to the D_j-FMeCC (which may be G-FMeCC or L-FMeCC, depending on the global system state), which updates consequently its G-VADITab.

In order to detect and update the VUE mobility pattern over time, the D_j-FMeCC uses Algorithm 2 named as VUE Mobility Pattern Updating Algorithm (VMPUA). This algorithm is executed per automated driving service session instance, and kept running during the entire lifetime of the session. For each new AUTO-DRIVE-SESSION-UPDATE message, the D_j-FMeCC analyses the VUE's mobility pattern; based on the velocity variation values (i.e., AV − OAV), and updates accordingly its G-VADITable *VMP* field to one of these velocity variation pattern values: *acceleration*, *constant* or *deceleration*. We note the adoption of a Constant Velocity Threshold parameter noted CV-Thr, which is used to compare the Δ velocity (i.e., Δ velocity = AV OAV) variations at each T_{update} epoch; two cases are distinguished.

Case 1: $|$ AV OAV$| >$ CV-Thr

Or even: $-$ CV-Thr $> \Delta$ velocity $> +$ CV-Thr

Observation: − The VUE is considered in a acceleration phase with: Δ velocity $> +$ CV-Thr

− The VUE is considered in a deceleration phase with: Δ velocity $< -$ CV-Thr

Case 2: $|$ AV OAV$| \leq$ CV-Thr

Or even: $-$ CV-Thr $\leq \Delta$ velocity $\leq +$ CV-Thr

Observation: The VUE is considered in relatively constant high velocity phase.

Particularity: In order to detect and avoid the acceleration/deceleration phases with consecutive Δ velocity values satisfying the inequality "$|\Delta$ velocity$| \leq$ CV-Thr" which the system may consider as constant but which, in fact, are not (i.e., false constant phases). The system maintains the OAV value of each first detection of a constant velocity phase (i.e., $|\Delta$ velocity$| \leq$ CV-Thr) as initial velocity value (i.e., initVel), and observes at each T_{update} epoch the velocity variation value of $|$ AV initVel$|$. If this value exceeds the defined constant velocity threshold CV-Thr (i.e., $|$ AV initVel$| >$ CV-Thr), then the system will react accordingly and correct as necessary the current velocity phase aspect to acceleration phase if "AV initVel $> +$ CV-Thr" or deceleration phase if "AV initVel $< -$ CV-Thr". The VMPUA performs the operations in Algorithm 2.

In addition, based on this received AUTO-DRIVE-SESSION-UPDATE, the global real-time information tables G-DIMTab, G-VADITab and the global End-to-End Delay Automated Driving QoS Threshold (ED-Thr), D_j-FMeCC triggers the execution of Algorithm 3. The latter permits assessing the system evolution in order to ensure the required QoS for the smooth running of the ongoing automated driving service session. Thus the system will be in one of the following two cases.

Case 1: $VUE\text{-}EED_{current} \leq$ ED-Thr

Observation: The minimum QoS requirement is ensured for VUE.

Action: No service migration is needed; continue with the current edge-Cloud service.

Case 2: $VUE\text{-}EED_{current} >$ ED-Thr

Observation: The minimum QoS requirement is not ensured for VUE.

Action: The migration of the automated driving session is needed to more appropriate edge-Cloud service.

Fig. 3 shows the signaling flow for an automated driving service update when $VUE\text{-}EED_{current}$ is smaller than the threshold *ED-Thr*. Fig. 4 illustrates the signaling flow for an automated driving service update when the $VUE\text{-}EED_{current}$ exceeds the threshold *ED-Thr*.

Let's $A = delay(VUE,\ eNodeB_{current})$, $B = delay(eNodeB_{current},\ edge\text{-}Cloud_{current})$, and $VUE\text{-}EED_{current} = A + B$. where $eNodeB_{current}$ is the VUE's current eNodeB, $edge\text{-}Cloud_{current}$ is the VUE's current

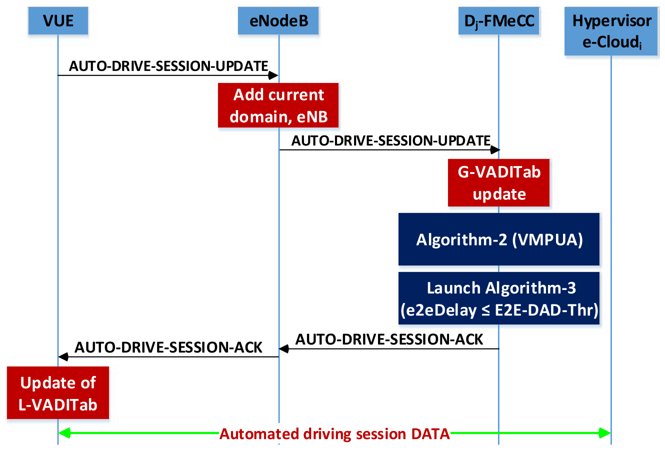


Fig. 3. Signaling flow for an automated driving service update in case of VUE-EED$_{current}$ $\leq$ ED-Thr.

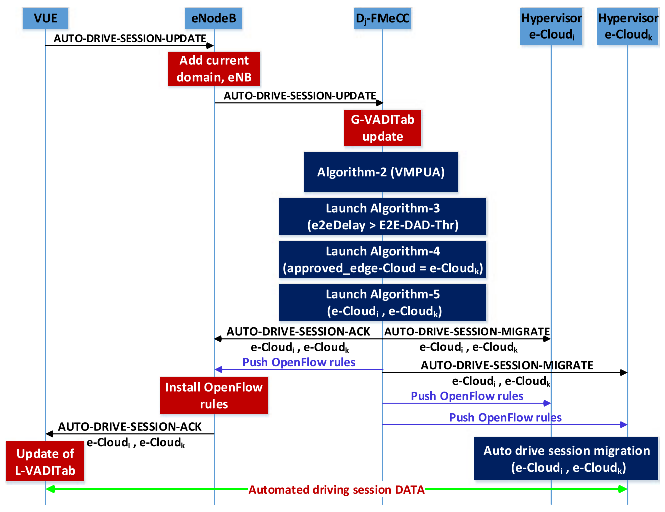

VUE
eNodeB
Dj-FMeCC
Hypervisor e-Cloudi
Hypervisor e-Cloudk
AUTO-DRIVE-SESSION-UPDATE
Add current domain, eNB
AUTO-DRIVE-SESSION-UPDATE
G-VADITab update
Algorithm-2 (VMPUA)
Launch Algorithm-3 (e2eDelay > E2E-DAD-Thr)
Launch Algorithm-4 (approved_edge-Cloud = e-Cloudk)
Launch Algorithm-5 (e-Cloudi , e-Cloudk)
AUTO-DRIVE-SESSION-ACK e-Cloudi , e-Cloudk
AUTO-DRIVE-SESSION-MIGRATE e-Cloudi , e-Cloudk
Push OpenFlow rules
AUTO-DRIVE-SESSION-MIGRATE e-Cloudi , e-Cloudk
Install OpenFlow rules
Push OpenFlow rules
Push OpenFlow rules
AUTO-DRIVE-SESSION-ACK e-Cloudi , e-Cloudk
Update of L-VADITab
Auto drive session migration (e-Cloudi , e-Cloudk)
Automated driving session DATA

Fig. 4. Signaling flow for and automated driving service update in case of $VUE\text{-}EED_{current}$ > ED-Thr.

edge-Cloud and $VUE\text{-}EED_{current}$ is the VUE's current E2E delay to the hosted automated driving service. If the minimum QoS requirement (delay) is no longer ensured for the VUE automated driving service session, we then have:

$$VUE\text{-}EED_{current} = A + B > ED\text{-}Thr \tag{1}$$

Under this condition, the automated driving service session should be migrated to a more appropriate edge-Cloud of the federated edge-clouds pool in order to preserve the minimum QoS requirement. This is achieved by establishing a *candidate edge-Cloud List*, denoted by $edge\text{-}Cloud_{candidate}$, which meets the following condition:

$$\begin{aligned} &VUE\text{-}EED_{current} delay(eNodeB_{current},\\ &\quad edge\text{-}Cloud_{current}) + delay(eNodeB_{current},\\ &\quad edge\text{-}Cloud_{candidate}) \leq ED\text{-}Thr \end{aligned} \tag{2}$$

Using the notation $B' =$ *delay(eNodeB$_{current}$, edge-Cloud$_{candidate}$)*, the condition becomes:

$$A + B \leq ED\text{-}Thr \tag{3}$$

Algorithm 3: edge-Cloud QoS Evaluating Algorithm (eCQEA).

```
Input: domain d, eNodeB eNB, edge-Cloud eCl, instance s
Output: candidate-edge-Cloud-List
 1: list edge-Clouds ← null
 2: if G-VADITab[s].EED ≤ ED-Thr then
 3:     {minimum requested QoS is ensured, session
        migration is not need}
 4: end if
 5: if G-VADITab[s].EED > ED-Thr then
 6:     {minimum requested QoS is not ensured, session
        migration to more appropriate edge-Cloud service
        is need}
 7:     $AB$ ← G-VADITab[s].EED
 8:     $B$ ← G-DIMTab[d][eNB][eCl].delay
 9:     for $i = 1$ to $M$ do
10:         if $AB - B+$ G-DIMTab[d][eNB][$eCID_i$].delay<
            ED-Thr then
11:             edge-Clouds ← edge-Clouds, $eCID_i$
12:         end if
13:     end for
14: end if
15: candidate-edge-Cloud-List ← edge-Clouds
```

Algorithm 3, edge-Cloud QoS Evaluating Algorithm (eCQEA), is executed by the current D_j-FMeCC of domain D_j, upon the reception of an AUTO-DRIVE-SESSION-UPDATE message (i.e., every T_{update} time interval) and it is given as follows:

The decision of selecting one candidate edge-Cloud as the target edge-Cloud for service session migration from the Candidate edge-Cloud List is approved by Algorithm 4. As the studied use case relates to highways environment, the VUE's mobility/velocity is the key factor to consider when selecting the target edge-Cloud for service session migration. To achieve this result, we envisioned a *mobility-aware migration* approach in this decision-making algorithm. Our objective is to reduce to the minimum the incurred cost related to successive services migrations and data transmission by optimizing the service placement with respect to the VUE's mobility/velocity. We recall that the global system cost at time t is given as follows:

$$Cost(t)_{global} = Cost(t)_{migration} + Cost(t)_{transmission} \tag{4}$$

The global system cost over time is then given as:

$$Cost_{global} = \int_0^T (Cost(t)_{migration} + Cost(t)_{transmission})\,\mathrm{d}t \tag{5}$$

where T is the total observation time period, $Cost(t)_{migration}$ denotes the instantaneous incurred cost if a service migration occurs at time t, and $Cost(t)_{transmission}$ is the instantaneous incurred cost resulting from the data transmission via the backhaul network when the VUE's serving eNodeB and the edge-Cloud hosting the service are in different locations (e.g., this cost is set to zero when the VUE's serving eNodeB and the edge-Cloud hosting the service are in the same location).

We distinguish two cases regarding the VUE velocity variation pattern Δ velocity (i.e., Δ velocity = AV OAV) as presented and computed in Algorithm 2.

Case 1: Δ velocity $\geq -$ CV-Thr

Observation: The VUE is in acceleration phase or in constant high velocity phase.

Action: From the candidate edge-Cloud List, select the edge-Cloud with the farthest location compared to the VUE's current location to place the migrated service.

Rational: As the VUE is moving with positive acceleration (acceleration phase) or with constant high velocity, it is optimal to place the service in the candidate edge-Cloud with farthest location compared to the VUE current location, permitting thus, the VUE to travel a maximum distance before the next service migration becomes needed (i.e., when $VUE\text{-}EED_{current} > ED\text{-}Thr$), and to minimize the number of successive service migrations permitting therefore to minimize the migrations' incurred cost ($Cost_{migration}$).

Case 2: Δ velocity $< -$ CV-Thr

Observation: The VUE is in a deceleration phase.

Action: From the candidate edge-Cloud List, select the edge-Cloud with the nearest location compared to the VUE's current location to place the migrated service.

Rational: As the VUE is moving with negative acceleration (deceleration phase), it is optimal to place the service in the candidate edge-Cloud nearest to the VUE's current location, allowing to offer the best QoS in terms of delay to the VUE, and to minimize the cost associated with the data ($Cost_{transmission}$), and avoiding therefore all possible waste of network resources (e.g., when the VUE stops moving).

Algorithm 4, edge-Cloud Service Migration Approving Algorithm (eCSMAA), is triggered by the edge-Cloud QoS Evaluating Algorithm (eCQEA), marking the end of its execution. It is executed by the current D_j-FMeCC of domain D_j and performs the operations in Algorithm 4.

Once the candidate edge-Cloud is approved by the eCSMAA Algorithm, the latter triggers the execution of Algorithm 5, the edge-Cloud Service Migration eXecuting Algorithm (eCSMXA). The eCSMXA Algorithm is executed by the current D_j-FMeCC and permits accomplishing the automotive driving service session migration from the current edge-Cloud to the candidate edge-Cloud. This is achieved by sending an AUTO-DRIVE-SESSION-MIGRATE message to both the source and target micro-datacenter hypervisors to initiate the service migration among the edge-Clouds. It is then followed by generating and installing the OpenFlow rules on the data path elements of the SDN/OpenFlow architecture (i.e., current VUE's eNodeB, source μ DCG, and destination μ DCG); aiming at ensuring a seamless traffic migration, and hence an optimal data communication in the new path The eCSMAA performs the operations in Algorithm 5.

Algorithm 4: edge-Cloud Service Migration Approving Algorithm (eCSMAA).

```
Input: domain d, eNodeB eNB, instance s,
       candidate-edge-Cloud-List ceCL
Output: approved-edge-Cloud
1: edge-Cloud ← 0, location ← 0, delay ← $\infty$
2: if G-VADITab[s].VMP = acceleration or constant then
3:     for $i = 1$ to length(ceCL) do
4:         if G-DIMTab[d][eNB][ceCL[i]].location >
           location then
5:             location ← G-DIMTab[d][eNB][ceCL[i]].
               location
6:             edge-Cloud ← ceCL[i]
7:         end if
8:     end for
9: end if
10: if G-VADITab[s].VMP = deceleration then
11:     for $i = 1$ to length(ceCL) do
12:         if G-DIMTab[d][eNB][ceCL[i]].delay < delay
            then
13:             delay ← G-DIMTab[d][eNB][ceCL[i]].delay
14:             edge-Cloud ← ceCL[i]
15:         end if
16:     end for
17: end if
18: approved-edge-Cloud ← edge-Cloud
```

Algorithm 5: edge-Cloud Service Migration eXecuting Algorithm (eCSMXA).

```
Input: domain d, eNodeB eNB, instance s,
    src-edge-Cloud $eCl_i$, dst-edge-Cloud $eCl_k$
Output:
1: send(AUTO-DRIVE-SESSION-MIGRATE,
   $hypervisor_i$, $eCl_i$, $eCl_k$)
2: send(AUTO-DRIVE-SESSION-MIGRATE,
   $hypervisor_k$, $eCl_i$, $eCl_k$)
3: send(AUTO-DRIVE-SESSION-ACK, eNB, $eCl_i$, $eCl_k$)
4: send(OpenFlow Rules, eNB, $eCl_i$, $eCl_k$)
5: send(OpenFlow Rules, $\mu$ $DCG_i$, $eCl_i$, $eCl_k$)
6: send(OpenFlow Rules, $\mu$ $DCG_k$, $eCl_i$, $eCl_k$)
7: G-VADITab[s].eCID ← $eCl_k$
```

V. Evaluation and Performance Analysis

To validate our solution two evaluation approaches are proposed: (i) a theoretical analysis based approach; (ii) a simulation experiments based approach.

A. Theoretical Analysis

In this section, we evaluate the performance of our proposed Follow Me edge-Cloud solution in the context of future 5G automated driving systems. The system evaluation was divided into three parts: (i) the street and mobile network environment part; (ii) the vehicle traffic flow simulation part; and (iii) the LTE-based network communication model part.

1) Street and Mobile Network Environment: The street environment is based on a linear highway scenario of 60 km long with three road segments; each segment is two-lane and 20 km long. For the mobile network environment, we assume that eNodeBs are located along the highway and are separated by a

TABLE VI
Mobility Simulation Parameters

Parameter	Value
Simulated Vehicles	200, 3000
Car-following model	Krauss
Lane changing model	LC2013
Maximum acceleration	2.6 m/s^2
Maximum deceleration	4.5 m/s^2
Maximum speed	70.0 m/s^2
Speed factor	1.0
Vehicle length	5.0 m
Driver imperfection	0.5
Driver reaction time	1.0 s
Minimum gap	2.5 m
Free flow traffic density	200 Vehicles/h
Traffic jam traffic density	3000 Vehicles/h

distance of 4 km in such a way that each eNodeB covers a zone of 4 km of the highway ($eNodeB_0$ covers from $[0-4]$ km, $eNodeB_1$ from $[4-8]$ km, etc.) and it is located in the middle of the covered zone.

2) Vehicle Traffic Simulation: For a traffic mobility model, we use SUMO (Simulation of Urban MObility) [25], an open microscopic traffic flow simulation platform, which allows to generate vehicles mobility traces (e.g., time, vehicle-id, lane, speed, position, etc.). For this SUMO simulation, VUEs are assumed to move according to Krauss car-following model [25] along the three highway road segments. To obtain different traffic states, the traffic densities of road segments are variable, and two points of traffic jam have been added at the end of the first two highway road segments. Thus the studied VUE goes through the following phases with respect to its observed velocity during simulation time:

- The first is the acceleration phase during which the VUE's velocity increases, this phase characterises the beginning of highway road segment 1, as well as highway road segment 2 and 3 after passing trafic jam zones;
- The second represents a constant phase whereby the VUEs remains relatively in constant high velocity with $|\Delta\ velocity| \leq CV\text{-}Thr$;
- The third is the deceleration phase during which the VUE's velocity decreases, this phase characterises the end of highway road segment 1 and 2 just before traffic jam zones.

The above-mentioned simulation scenario is run for 21 minutes, with the edge-Cloud QoS Evaluating Algorithm (eCQEA) runs every 5 seconds ($T_{update} = 5$ sec). The choice of T_{update} depends on the current state of the system (e.g., VUE's mobility model, E2E delay sensitivity, and edge-Cloud/cloud local load) and can be further tuned through a more detailed analysis.

Through this analysis, the VUE is assumed to follow a one-dimensional $(1-D)$ mobility model with one direction of traffic flow. Table VI lists all SUMO parameters we used in the simulation.

Fig. 5 shows the distance travelled by the VUE during the simulation time, including the traffic jam points at the end of the highway road segment 1 (i.e., distance = 20 km) and segment 2 (i.e., distance = 40 km). While Fig. 6 illustrates the VUE velocity variation over the simulation time with the different

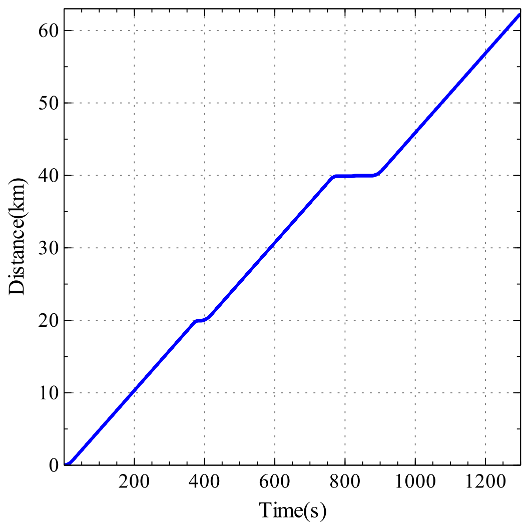


Fig. 5. The distance traveled by the VUE over time.

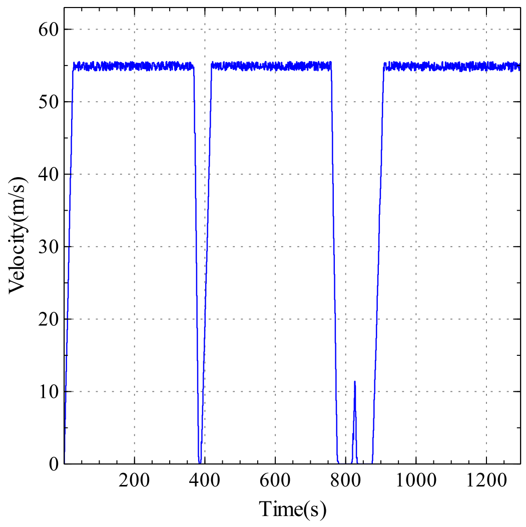


Fig. 6. The velocity variation of the VUE over time.

acceleration and deceleration phases, as well as the relatively constant high velocity phases (i.e., $|\Delta\ velocity| \leq CV\text{-}Thr$).

3) Network Communication Model: The network communication is composed by: (i) the V2X communication based on LTE and (ii) the backhaul communication. We used an analytical approach, based on the delay and cost parameters described in Table VII, to model the network communication.

4) Location Computation: We define the location of each eNodeB$_i$ as the index i of the eNodeB along the highway, starting from a referential eNodeB$_0$ (i.e., $index$ 0). For the sake of simplicity, we also assume a one-to-one mapping between eNodeBs and edge-Cloud services, so that each eNodeB$_i$ is collocated with only one edge-Cloud$_i$ service in the same domain and vice versa. Thus, the location (offset) associated to the ith (eNodeB, edge-Cloud) pair and the highway portion covered by the ith eNodeB are given as follow:

$$Location\text{-}eNodeB_i = Location\text{-}edge\text{-}Cloud_i = 4i + 2 \tag{6}$$

TABLE VII
NETWORK COMMUNICATION PARAMETERS

Parameter	Description
A	A variable representing the delay between the VUE and its current eNodeB
B	A variable representing the delay between the VUEs current eNodeB and the current edge-Cloud service
α	A constant representing the backhaul additional delay between an eNodeB$_i$ and an edge-Cloud$_j$ resulting from a non-optimal service delivery situation with $\|i-j\|=1$. More generally, the backhaul additional delay between an eNodeB$_i$ and an edge-Cloud$_j$ equals to $\|i-j\| * \alpha$.
β	A constant representing the data transmission cost resulting from a non-optimal service delivery situations (i.e., $\|i-j\| \neq 0$)
φ	A constant representing the service migration cost resulting from an edge-Cloud service migration
z	A variable representing the number of VUEs edge-Cloud service migrations during simulation time
k	A variable representing the number of VUEs eNodeB handovers during simulation time

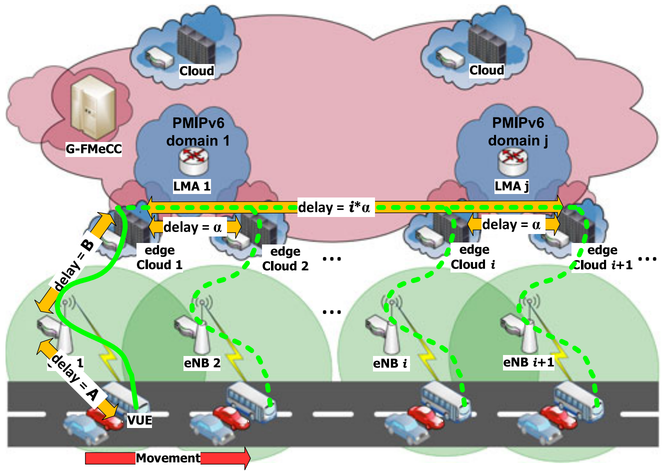


Fig. 7. The FMeC network architecture envisioned for the simulation.

$$Segment\text{-}eNodeB_i = [4i, 4(i+1)] \tag{7}$$

5) VUE e2e Delay Computation: As shown in Fig. 7, the e2e perceived delay by a VUE attached to eNodeB$_i$ and served by an edge-Cloud$_j$ is calculated as follows:

$$e2e\text{-}delay\text{-}\mathrm{VUE}_{ij} = A + B + |i-j| \cdot \alpha \tag{8}$$

where A, B and α are assumed fixed delay values. We compare our proposed edge-Cloud service placement solution against two baseline counterparts which are:

- *Always Migrate (AM):* the service is migrated in the edge-Cloud side every VUEs eNodeB handover in the LTE access network side.
- *Migrate If Needed (MIN):* the service is migrated in the edge-Cloud side whenever the VUEs E2E delay exceeds ED-Thr threshold.

6) Cost Computation: The cost incurred by the system depends directly on: (i) the edge-Cloud service migration decision taken at the beginning of each T_{update} slot (i.e., migrate/do not migrate); (ii) the VUE's handover from an eNodeB to another during its move along the highway, which leads to increase the E2E delay and consequently an increase of the data transmission cost. Denoting the current edge-Cloud service location by *edge-Cloud*$_{cl}$ and the current serving eNodeB location by $eNodeB_{cl}$, the following formulas can be used to calculate the instantaneous system cost $c(t)$ at time t.

$$c(t) = m(t) + h(t) \tag{9}$$

$$m(t) = \begin{cases} \varphi, & \text{if migration at } t \text{ and} \\ & \quad edge\text{-}Cloud_{cl} = eNodeB_{cl} \\ \varphi + \beta, & \text{if migration at t} \\ & \quad \text{and } edge\text{-}Cloud_{cl} \neq eNodeB_{cl} \\ 0, & \text{Otherwise} \end{cases}$$

$$h(t) = \begin{cases} \beta, & \text{if handover at t and } edge\text{-}Cloud_{cl} \neq eNodeB_{cl} \\ 0, & \text{Otherwise} \end{cases}$$

From the previous formulas, we can deduce the global sum cost C over the simulation time T_s.

$$C = \int_0^{T_s} (m(t) + h(t))\,\mathrm{d}t = \int_0^{T_s} m(t)\,\mathrm{d}t + \int_0^{T_s} h(t)\,\mathrm{d}t \tag{10}$$

Denoting by z the number of edge-Cloud service migrations and by k the number of VUE eNodeB handovers during the evaluation time, the global sum cost, for the two approaches (i.e., always migrate and migrate if needed), is given as:

$$C_{always\ migrate} = C_{migrate\ if\ needed} = z \cdot \varphi + k \cdot \beta \tag{11}$$

For our introduced approach Mobility Aware Migration, we put $z = z_1 + z_2$; we denote by z_1 the number of migrations with transmission cost incurred ($edge\text{-}Cloud_{cl} \neq eNodeB_{cl}$), and by z_2 the number of migrations without transmission cost incurred ($edge\text{-}Cloud_{cl} = eNodeB_{cl}$). The global sum cost can be given as:

$$\begin{aligned} C_{mobility\ aware\ migration} &= z_1(\varphi + \beta) + z_2\varphi + (k - z_1)\beta \\ &= z_1\varphi + z_1\beta + z_2\varphi + k\beta - z_1\beta \\ &= (z_1 + z_2) \cdot \varphi + k \cdot \beta \\ &= z \cdot \varphi + k \cdot \beta \end{aligned} \tag{12}$$

This gives the same result as the baseline approaches (i.e., always migrate and migrate if needed).

B. Simulation Experiments

In this section, we present the evaluation of the FMeC system through computer simulation; we begin with a description of the used simulation tools and environment, then we present the simulation scenario and discuss the obtained results.

Like the theoretical analysis, in this section the system evaluation was also divided into three parts: (i) the street and mobile

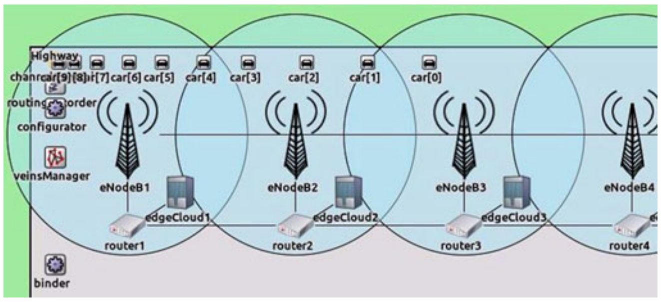


Fig. 8. The simulation experimental set up.

network environment part; (ii) the vehicle traffic flow simulation part; and (iii) the LTE-based network communication model part.

1) Street and Mobile Network Environment: The street environment is based on a linear highway scenario of 3.6 km long with three road segments; each segment is two-lane and 1 km long. For the mobile network environment, the eNodeBs are located along the highway and are separated by a distance of 0.35 km in such a way that each eNodeB covers a zone of 0.5 km of the highway. We also note the existing of an overlap area between each two successive eNodeBs.

Each eNodeB is connected through the *ppp interface* to the local edge network providing access to the federated edge cloud, and through the $X2$ *interface* to the eNodeBs pool in a full mesh manner. The envisioned network simulation environment is illustrated in Fig. 8.

2) Vehicle Traffic Simulation: For the traffic mobility simulation we use SUMO simulator. Moreover, the same mobility model as the theoretical analysis is considered (as discussed in part (V.a.2)).

3) Network Communication Model: In order to simulate LTE-based V2X communications, we chose OMNeT++ [27] as network simulator environment. OMNeT++ is a C++ based modular discrete event simulator that supports multiple frameworks for modeling complex and realistic communication networks. For the purpose of our simulation, the use of OMNeT++ is extended with the INET [28], SimuLTE [29] and Veins [30] well-known frameworks.

- **INET** framework is an open-source model library for the OMNeT++ simulation environment. It provides models for the Internet stack (TCP, UDP, SCTP, IPv4, IPv6, OSPF, BGP, etc.), routing protocols (ad-hoc and wired), wired and wireless link layer protocols (Ethernet, PPP, IEEE 802.11, etc), support for mobility, and many other protocols and components.
- **SimuLTE** is a framework extension based on OMNeT++ and INET for simulating LTE/LTE-A networks. SimuLTE implements RAN element (e.g., eNBs, UEs) and EPC elements (e.g., pgw) as compound modules that can be connected to other modules (e.g., routers, switches, servers) in order to compose networks.
- **Veins** is an open-source framework for running vehicular network simulations. It leverages the road traffic simulator SUMO in order to provide vehicular mobility to the network simulator OMNeT++. Veins integrates with OMNeT++ and uses the Traffic Control Interface (TraCI) module to connect OMNeT++ with SUMO to provide bidirectionally-coupled simulation of road traffic and network traffic.

TABLE VIII
COMMUNICATION SIMULATION PARAMETERS

Parameter	Value
eNB TxPower	100 mW
N^o of eNBs	10
N^o of VUEs	200, 3000
Frequency	2.1 GHz
eNB Antenna Gain	18 dBm
VUE Antenna Gain	0 dBm
eNB Noise Figure	5 dB
VUE Noise Figure	7 dB
Resource Blocks per Cell	25
LTE Scheduler	Round Robin
Duplex Mode	Frequency Division Duplex
Maximum Sending Power	10000 mW
Signal Attenuation Threshold	−110 dBm
Propagation Model	Free Space Model
VUE TxPower	26 mW
Handover Latency	0.05 s
Path Loss Scenario	URBAN MACROCELL
Simulation Time	80 s

Table VIII lists all OMNeT++ network parameters we used in the simulation.

The simulation scenario is run for 80 seconds, with the edge-Cloud QoS Evaluating Algorithm (eCQEA) runs every 0.5 seconds (T_{update} = 0.5 sec). The choice of T_{update} depends on the current state of the system (e.g., VUE's mobility model, E2E delay sensitivity, and edge-Cloud/cloud local load) and can be further tuned through a more detailed analysis.

Through this analysis, the VUE is assumed to follow a one-dimensional $(1-D)$ mobility model with one direction of traffic flow.

VI. RESULTS

In this section we descuss the results obtained through the two evaluation methodes: the theoretical analysis, and the simulation experiments.

A. The Theoretical Analysis Results

All of the above discussed performance factors are plotted to illustrate the improvement achieved by our Mobility Aware Migration approach. In our analysis, and without any loss of generality, we consider these parameter values: $\varphi = 1$ Gb, $\beta = 0.5$ Gb, $A = B = \alpha = 20$ ms, $ED\text{-}Thr =$ 60 ms, $CV - Thr = 1.5$ m/s, where some of the parameters are based on those used in [2] and [26].

Fig. 9 depicts the distance travelled by the VUE over the evaluation time (60 km, 21 min) and during the three phases (i.e., acceleration, constant and deceleration). It also compares the number of edge-Cloud service migration epochs needed to preserve the required VUE automotive driving e2e delay in each approach. We can clearly observe that our proposed Mobility Aware Migration approach performs better with 5 migrations comparing with Always Migrate approach (14 migrations) and Migrate If Needed (7 migrations).

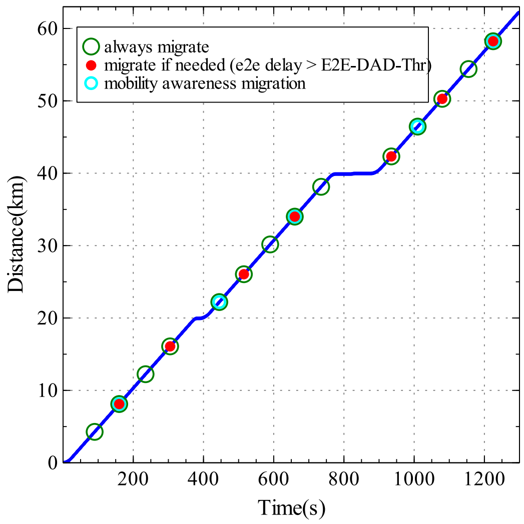


Fig. 9. The distance traveled by the VUE over time and migrations epochs for each approach.

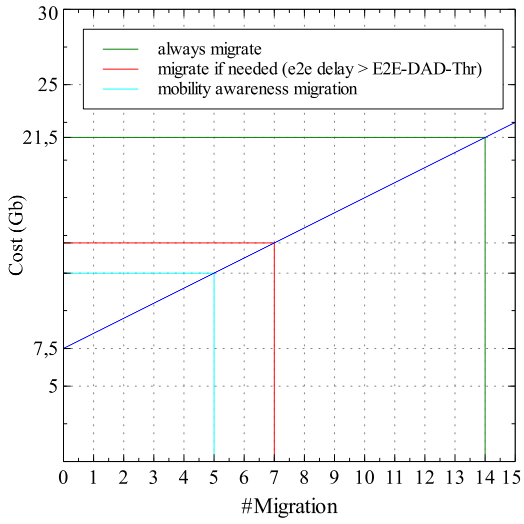


Fig. 10. The global cost as a function of the number of edge-Cloud service migrations for each approach.

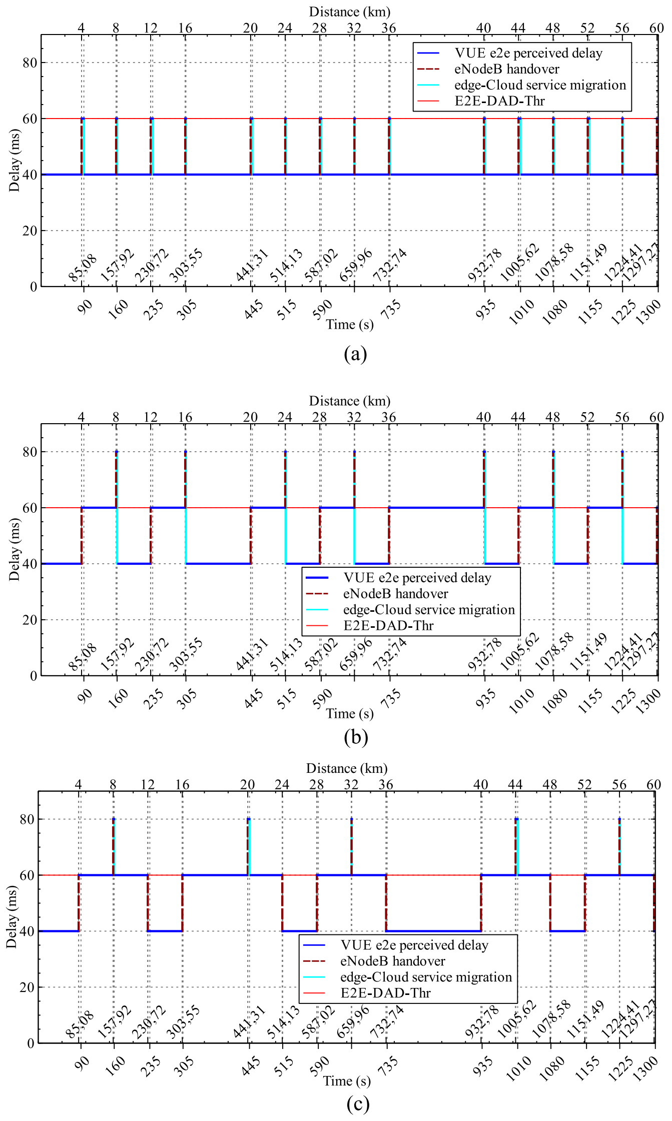


Fig. 11. Delay evolution over time for each approach along with VUE's eNodeB handover and edge-Cloud service migration epochs. (a) "Always migrate" approach. (b) "Migrate if needed" approach. (c) Our proposed approach.

In Fig. 10, we compare the global incurred cost as a function of the number of edge-Cloud service migrations under each approach. We can clearly observe that our proposed "Mobility Aware Migration" approach gives the best result with a global cost of 12.5 Gb when comparing it against the "Always Migrate" approach with a global cost of 21.5 Gb and the "Migrate If Needed" approach with a global cost of 14.5 Gb.

Fig. 11 illustrates the delay evolution over the simulation time under the application of the three studied approaches. We can observe that the delay variation is directly related to these two events: *(1)* the VUE eNodeB handover event and *(2)* the edge-Cloud service migration event. We can also observe that the three approaches permit to ensure the required VUE automated driving e2e delay when this latter exceeds the defined threshold by forcing an edge-Cloud service migration, aiming at placing the service in a nearest edge-Cloud with respect to the VUE location. Moreover, our proposed Mobility Aware Migration approach (Fig. 11(c)) stands out from the rest for its ability to reduce the number of edge-Cloud service migrations to a minimum (5 migrations in this case) in comparison with Always Migrate approach (Fig. 11(a), 14 migrations) and Migrate If Needed approach (Fig. 11(b), 7 migrations). This is mainly due to the fact that our proposed approach considers the VUE mobility pattern in the computation of the candidate edge-Cloud to place the migrated service.

Fig. 12 compares the evolution of the instantaneous cost over two axes; namely the simulation elapsed time and the VUE's traveled distance under the application of the three studied approaches. We can clearly see that our proposed Mobility Aware Migration approach (Fig. 12(c)) outperforms the other two approaches. Indeed, our approach permits optimizing the instantaneous incurred cost by minimizing the number of edge-Cloud service migrations. This is achieved through the application of our introduced algorithms, based on VUE's mobility patterns in the selection of the future edge-Cloud service. This permits, on one hand, ensuring the required VUE's automotive driving E2E

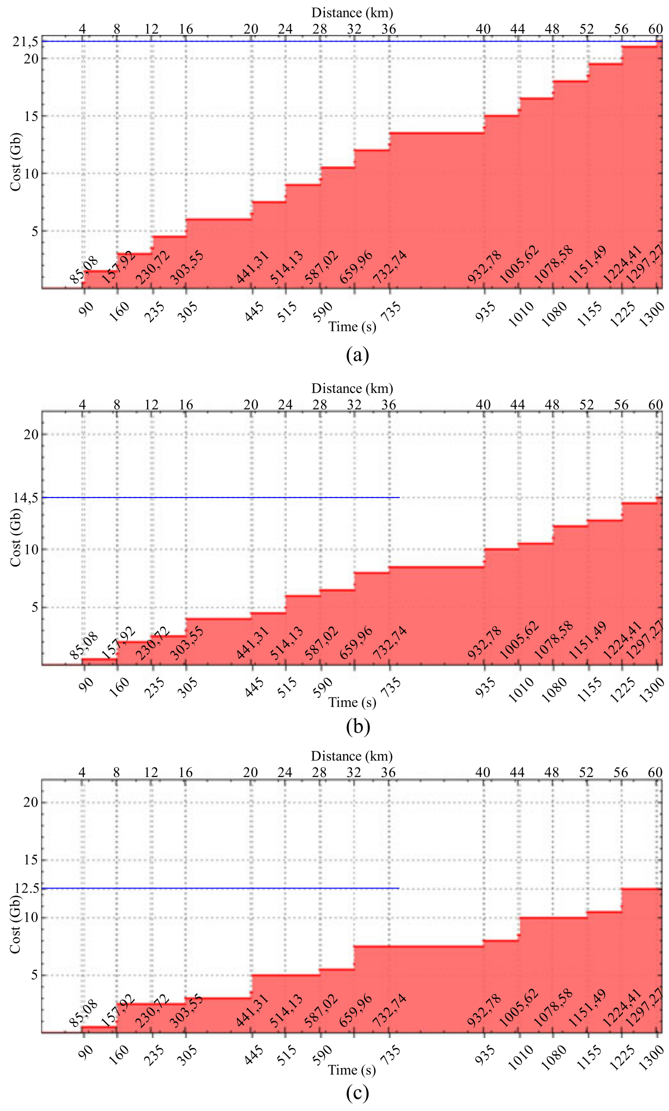


Fig. 12. The instantaneous cost evolution over time along with VUE's traveled distance. (a) Always migrate approach. (b) "Migrate if needed" approach. (c) Our proposed approach.

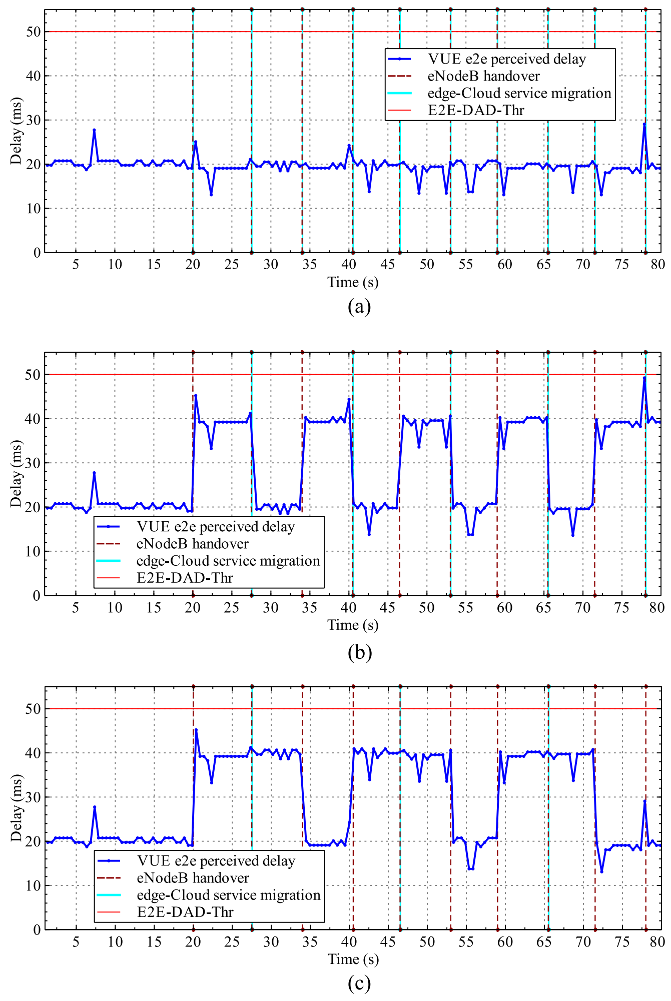


Fig. 13. The instantaneous cost evolution over time along with VUE's traveled distance. (a) Always migrate approach. (b) "Migrate if needed" approach. (c) Our proposed approach.

delay, and on the other hand, minimizing the incurred global sum cost.

B. The Simulation Experiments Results

Like in the analytical model, and without any loss of generality, we consider these parameter values: $\varphi = 1$ Gb, $\beta = 0.5$ Gb, $ED\text{-}Thr = 50$ ms, $CV - Thr = 1.5$ m/s, where some of the parameters are based on those used in [2] and [26].

Fig. 13 illustrates the delay evolution over the simulation time under the application of the three studied approaches. It is clear that the delay variation is directly related to these two events: (1) the VUE eNodeB handover event and (2) the edge-Cloud service migration event. We can observe that the required VUE automated driving e2e delay is ensured in the three approaches (i.e., delay under the ED-Thr bar). This is achieved by migrating the VUE automated driving instance to the nearest service point in the federated edge-Cloud, when the VUE perceived delay exceeds the defined threshold. However, and similar to the theoretical analysis results, our proposed Mobility Aware Migration approach (Fig. 13(c)) outperforms all others by its ability to reduce the number of edge-Cloud service migrations to a minimum (3 migrations in this case), compared to the Always Migrate (Fig. 13(a), 10 migrations) and the Migrate If Needed approaches (Fig. 13(b), 5 migrations). We argue this by the fact that our proposed approach considers the VUE mobility pattern in the computation of the candidate edge-Cloud to place the migrated service.

Fig. 14 compares the evolution of the instantaneous cost over the simulation elapsed time using the three studied approaches. Similar to the theoretical analysis results, we observe clearly that our proposed Mobility Aware Migration approach (Fig. 14(c)) outperforms all others in its ability to minimize the incurred global cost. Indeed, our approach permits optimizing the instantaneous incurred cost by minimizing the number of edge-Cloud service migrations. This is due to the application of our introduced mobility-aware algorithms, in the selection of the future VUE edge-Cloud service, thus ensuring the required VUE's

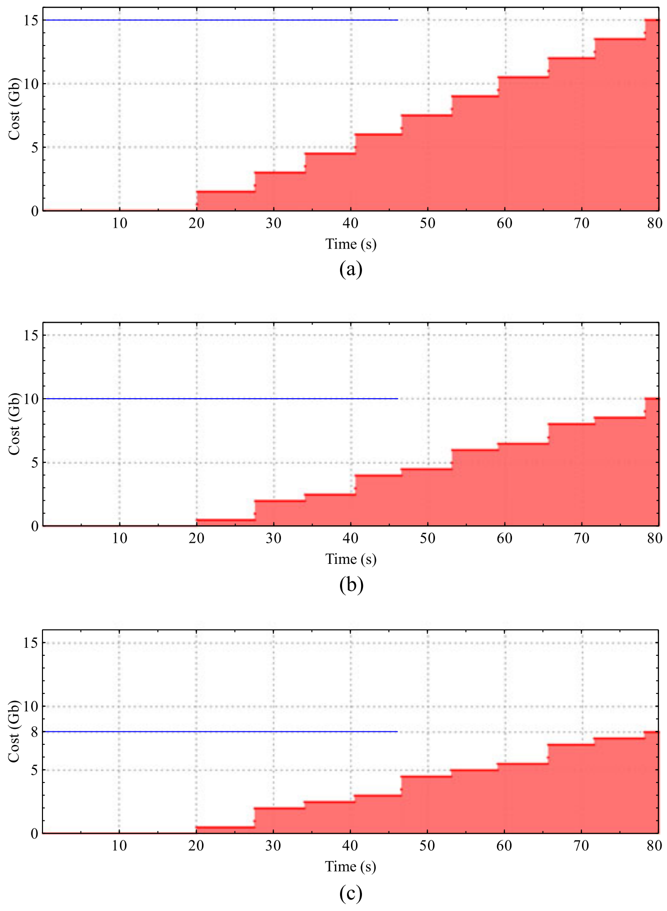


Fig. 14. The instantaneous cost evolution over time along with VUE's traveled distance. (a) Always migrate approach. (b) "Migrate if needed" approach. (c) Our proposed approach.

automotive driving E2E delay, and minimizing the incurred global cost.

VII. Conclusion

In this paper, we introduced the Follow Me edge Cloud concept, a promising key enabler of 5G future services and more specifically of 5G automotive systems, resulting from the application of the Mobile Edge Computing paradigm on the Follow Me Cloud concept. We first introduced the elements that allow integrating FMC within MEC and presented our FMeC solution. Then, we applied our FMeC solution in an automated driving use case dedicated for the future 5G automotive systems. Focusing on the LTE-based V2I communication type, we introduced our envisioned SDN/OpenFlow-based architecture and our mobility-aware framework based on a set of algorithms permitting to achieve the automated driving QoS requirements within 5G network. The evaluation results obtained conjointly via theoretical analysis and simulation experiments are consistent and showed that compared to baseline counterparts, our solution performs much better permitting, on the one hand, to ensure the required VUE's automated driving e2e delay, and, on the other hand, to minimize the incurred global cost.

Our future research direction will be to investigate the performance of the FMeC architecture for additional performance metrics (e.g., jitter, packet loss, workload, throughput, energy, cost, scale etc.). Further, we aim to extend the automotive mobility-aware framework to Vehicle-to-Vehicle (V2V)/Vehicle-to-Pedestrians (V2P) network communications and more generally to Vehicle-to-Everything (V2X) network communications within 5G network.

References

[1] T. Taleb and A. Ksentini, "Follow me cloud: Interworking federated clouds and distributed mobile networks," *IEEE Netw.*, vol. 27, no. 5, pp. 12–19, Sep./Oct. 2013.

[2] T. Taleb and A. Ksentini, "An analytical model for follow me cloud," in *Proc. 2013 IEEE Global Commun. Conf.*, Atlanta, GA, USA, Dec. 2013, pp. 1291–1296.

[3] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *Proc. 2014 IEEE Int. Conf. Commun.*, Sydney, NSW, Australia, Jun. 2014, pp. 1350–1354.

[4] T. Taleb, S. Dutta, A. Ksentini, M. Iqbal, and H. Flinck, "Mobile edge computing potential in making cities smarter," in *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 38–43, Mar. 2017.

[5] M. Satyanarayanan, G. Lewis, E. Morris, S. Simanta, J. Boleng, and K. Ha, "The role of cloudlets in hostile environments," *IEEE Pervasive Comput.*, vol. 12, no. 4, pp. 40–49, Oct. 2013.

[6] S. Davy *et al.*, "Challenges to support edge-as-a-service," *IEEE Commun. Mag.*, vol. 52, no. 1, pp. 132–139, Jan. 2014.

[7] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc MCC Workshop on Mobile Cloud Comput.*, 2012, pp. 13–16.

[8] NGMN, 5G White Paper, pp. 11–23. [Online]. Available: https://www.ngmn.org/5g-white-paper.html

[9] 3GPP RP-161894, "LTE-based V2X services," Sep. 2016.

[10] T. Taleb, P. Hasselmeyer, and F. G. Mir, "Follow-me cloud: An OpenFlow-based implementation," in *Proc. 2013 IEEE Int. Conf. Green Comput. Commun/IEEE Internet Things/IEEE Cyber, Phys. Social Comput.*, Beijing, China, Aug. 2013, pp. 240–245.

[11] R. Bifulco, M. Brunner, R. Canonico, P. Hasselmeyer, and F. Mir, "Scalability of a mobile cloud management system," in *Proc. SIGCOMM Workshop Mobile Cloud Comput.*, 2012, pp. 17–22.

[12] A. Aissioui, A. Ksentini, and A. Gueroui, "An efficient elastic distributed SDN controller for follow-me cloud," *Proc. 2015 IEEE 11th Int. Conf. Wireless Mobile Comput., Netw. Commun.*, Abu Dhabi, United Arab Emirates, 2015, pp. 876–881.

[13] A. Aissioui, A. Ksentini, A. M. Gueroui, and T. Taleb, "Toward elastic distributed SDN/NFV controller for 5G mobile cloud management systems," *IEEE Access*, vol. 3, pp. 2055–2064, 2015.

[14] T. Taleb, A. Ksentini, and R. Jantti, "Anything as a service for 5G mobile systems," *IEEE Netw.*, vol. 30, no. 6, pp. 84–91, Nov.–Dec. 2016.

[15] R. Bifulco and R. Canonico, "Analysis of the handover procedure in Follow-Me Cloud," in *Proc. 2012 IEEE 1st Int. Conf. Cloud Netw.*, Paris, France, Nov. 2012, pp. 185–187.

[16] A. Ksentini, T. Taleb, and F. Messaoudi, "A LISP-based implementation of Follow Me Cloud," in *IEEE Access*, vol. 2, pp. 1340–1347, 2014.

[17] T. Taleb, "Towards carrier cloud: Potential, challenges, & solutions," in *Proc. IEEE Wireless Commun. Mag.*, Jun. 2014. pp. 80–91.

[18] A. Aissioui, A. Ksentini, and A. Gueroui, "PMIPv6-based follow me cloud," in *Proc. 2015 IEEE Global Commun. Conf.*, San Diego, CA, USA, 2015, pp. 1–6.

[19] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," in *Proc. 2014 IEEE Mil. Commun. Conf.*, Oct. 2014, pp. 835–840.

[20] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *Proc. 2015 IFIP Netw. Conf.*, Toulouse, France, 2015, pp. 1–9.

[21] S. K. Datta, R. P. F. Da Costa, J. Härri, and C. Bonnet, "Integrating connected vehicles in Internet of Things ecosystems: Challenges and solutions," in *Proc. 2016 IEEE 17th Int. Symp. World Wireless, Mobile Multimedia Netw. Coimbra, Portugal*, Jun. 2016, pp. 1–6.

[22] M. Tanizaki and O. Wolfson, "Randomization in traffic information sharing systems," in *Proc. 15th Annu. ACM Int. Symposium Adv. Geographic Inform. Sys.* (ser. GIS 07), 2007, pp. 23:1–23:8.

[23] G. Rmy, S. M. Senouci, F. Jan, and Y. Gourhant, “LTE4V2X—Collection, dissemination and multi-hop forwarding,” in *Proc. 2012 IEEE Int. Conf. Commun.*, Ottawa, ON, Canada, 2012, pp. 120–125.

[24] J. Pillmann, B. Sliwa, J. Schmutzler, C. Ide, and C. Wietfeld, “Car-to-Cloud communication traffic analysis based on the common vehicle information model,” in *Proc. 2017 IEEE 85th Veh. Technol. Conf.*, Sydney, NSW, Australia, Jun. 2017, pp. 1–5.

[25] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, “Recent development and applications of SUMO—Simulation of urban mobility,” *Int. J. Adv. Syst. Meas.*, vol. 5, no. 3–4, pp. 128–138, Dec. 2012.

[26] NGMN, “NGMN Perspectives on Vertical Industries and Implications for 5G,” pp. 8–12. [Online]. Available: https://www.ngmn.org/uploads/media/160610_NGMN_Perspectives_on_Vertical_Industries_and_Implications_for_5G_v1_0.pdf

[27] OMNeT++, “Discrete Event Simulator.” [Online]. Available: https://omnetpp.org

[28] INET Framework, “An open-source OMNeT++ model suite for wired, wireless and mobile networks.” [Online]. Available: https://inet.omnetpp.org

[29] A. Virdis, G. Stea, and G. Nardini, “SimuLTE—A modular system-level simulator for LTE/LTE-A networks based on OMNeT++,” *Proc. 2014 4th Int. Conf. Simul. Model. Method. Technol. Appl.*, Vienna, Austria, 2014, pp. 59–70.

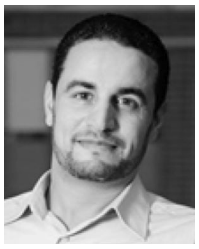
[30] C. Sommer, R. German, and F. Dressler, “Bidirectionally coupled network and road traffic simulation for improved IVC analysis,” in *IEEE Trans. Mobile Comput.*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

Abdelkader Aissioui received the Engineering degree in computer science from the University of Science and Technology Houari Boumediene, Algiers, Algeria, in 2004, and the M.Sc. degree in security of contents, networks, telecommunications, and systems from the University of Versailles Saint-Quentin-en-Yvelines (UVSQ), Versailles, France, in 2007. He is currently working toward the Ph.D. degree at the UVSQ.

His research interests include mobile edge computing, network function virtualization, software defined networking, 5G mobile communications, and vehicular communications.

Adlen Ksentini received the Ph.D. degree in computer science from the University of Cergy-Pontoise, Cergy-Pontoise, France, in 2005, with a dissertation on QoS provisioning in IEEE 802.11-based networks. From 2006 to 2016, he was an Assistant Professor with the University of Rennes 1. During this period, he was a Member of the Dionysos Team with INRIA, Rennes, France. Since March 2016, he has been an Assistant Professor with the Communication Systems Department, EURECOM, Biot, France. He has been involved in several national and European projects on QoS and QoE support in future wireless, network virtualization, cloud networking, mobile networks, and more recently on network slicing and 5G in the context of H2020 projects 5G!Pagoda and 5GTransformer. He is a COMSOC distinguish lecturer. He has coauthored more than 100 technical journal and international conference papers. He has been acting as the TPC Symposium Chair for the IEEE ICC 2016/2017, the IEEE GLOBECOM 2017, the IEEE Cloudnet 2017, and the IEEE 5G Forum 2018. He has been acting as a Guest Editor for the IEEE JOURNAL OF SELECTED AREA ON COMMUNICATION series on network softwerization, the IEEE WIRELESS COMMUNICATIONS, the IEEE COMMUNICATIONS MAGAZINE, and two issues of ComSoc MMTC Letters. He has been on the Technical Program Committees of major IEEE ComSoc, ICC/GLOBECOM, ICME, WCNC, and PIMRC conferences. He is currently the Vice-Chair of the IEEE COMSOC Technical Committee on Software, and the Secretary of the IEEE COMSOC France. He was the recipient of the best paper award from the IEEE IWCMC 2016, the IEEE ICC 2012, and ACM MSWiM 2005. He was also the recipient of the 2017 IEEE Comsoc Fred W. Ellersick (best IEEE communications Magazines paper). He has given several tutorials in the IEEE international conferences, the IEEE Globecom 2015, the IEEEE CCNC 2017, the IEEE ICC 2017, and the IEEE/IFIP IM 2017.

Abdelhak Mourad Gueroui received the M.Sc. degree from the Pierre-and-Marie-Curie University, Paris, France, and the Ph.D. degree in networking and computer engineering from the University of Versailles Saint-Quentin-en-Yvelines, Versailles, France. He is currently an Associate Professor in computer engineering with the University of Versailles, Saint-Quentin-en-Yvelines. His research interests include wireless networks (WATM, WLAN, and sensors), particularly performance evaluation and QoS provisioning.



Tarik Taleb received the B.E. degree (with distinction) in information engineering in 2001, and the M.Sc. and Ph.D. degrees in information sciences from Tohoku University, Sendai, Japan, in 2003, and 2005, respectively. He is currently a Professor with the School of Electrical Engineering, Aalto University, Aalto, Finland. He is the founder and the Director of the MOSA!C Lab. Prior to his current academic position, he was a Senior Researcher and a 3GPP Standards Expert at the NEC Europe Ltd., Heidelberg, Germany. He was then leading the NEC Europe Labs Team working on R&D projects on carrier cloud platforms, an important vision of 5G systems. Before joining NEC and till March 2009, he was an Assistant Professor with the Graduate School of Information Sciences, Tohoku University, in a lab fully funded by KDDI. From October 2005 till March 2006, he was a Research Fellow with the Intelligent Cosmos Research Institute, Sendai, Japan. His research interests include architectural enhancements to mobile core networks (particularly 3GPP’s), network softwarization and slicing, mobile cloud networking, network function virtualization, software defined networking, mobile multimedia streaming, intervehicular communications, and social media networking. He has been also directly engaged in the development and standardization of the evolved packet system as a Member of 3GPP’s System Architecture working group. He is a Member of the IEEE Communications Society Standardization Program Development Board. As an attempt to bridge the gap between academia and industry, he founded the IEEE Workshop on Telecommunications Standards: from Research to Standards, a successful event that got awarded best workshop award by the IEEE Communication Society (ComSoC). Based on the success of this workshop, he also founded and has been the Steering Committee Chair of the IEEE Conference on Standards for Communications and Networking. He is the General Chair for the 2019 edition of the IEEE Wireless Communications and Networking Conference to be held in Marrakech, Morocco. He is the Guest Editor-in-Chief for the IEEE JOURNAL OF SELECTED AREA ON COMMUNICATION series on network softwarization and enablers. He is/was on the editorial board of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE WIRELESS COMMUNICATIONS MAGAZINE, the IEEE JOURNAL ON INTERNET OF THINGS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE COMMUNICATIONS SURVEYS & TUTORIALS, and a number of Wiley journals. Till December 2016, he was the Chair for the Wireless Communications Technical Committee, the largest in the IEEE ComSoC. He also served as the Vice-Chair of the Satellite and Space Communications Technical Committee of the IEEE ComSoc (2006–2010). He has been on the Technical Program Committee of different IEEE conferences, including GLOBECOM, ICC, and WCNC, and chaired some of their symposia. He was the recipient of the 2017 IEEE ComSoc Communications Software Technical Achievement Award (December 2017) for his outstanding contributions to network softwarization. He is also the (co)recipient of the 2017 IEEE Communications Society Fred W. Ellersick Prize (May 2017), the 2009 IEEE ComSoc Asia-Pacific Best Young Researcher award (June 2009), the 2008 TELECOM System Technology Award from the Telecommunications Advancement Foundation (March 2008), the 2007 Funai Foundation Science Promotion Award (April 2007), the 2006 IEEE Computer Society Japan Chapter Young Author Award (December 2006), the NiwaYasujirou Memorial Award (February 2005), and the Young Researcher’s Encouragement Award from the Japan Chapter of the IEEE Vehicular Technology Society (October 2003). Some of his research works have been also awarded best paper awards at prestigious IEEE-flagged conferences.