

# Control and Management of a Connected Car Using SDN/NFV, Fog Computing and YANG data models

Ricard Vilalta, Selva Vía, Fermín Mira,  
Ramon Casellas, Raul Muñoz,  
and Jesus Alonso-Zarate  
Centre Tecnologic de Telecomunicacions de Catalunya  
(CTTC/CERCA), Spain  
ricard.vilalta@cttc.es

Apostolos Kousaridas and Markus Dillinger  
Huawei Technologies  
Munich Research Center  
Germany  
apostolos.kousaridas@huawei.com

**Abstract**—There are several use cases that claim the need for a connected car. Among them there is the need for connectivity between vehicles and information sources, or V2V and V2X exchanges for accident prevention. In order to cope with the need for novel applications running on top of an interconnected network, the concept of fog computing appears as a realistic solution for both intra-car and inter-car data processing and decision making.

This paper describes the proposed architecture and experimental evaluation of an innovative proof-of-concept (PoC) for a connected car, modeled with YANG, which can be remotely controlled using SDN/NFV and fog computing technologies. As an example, the remote control of the car might be based on a service application running on a fog node, which can be located close to a road side unit (RSU). We also propose a fog architecture in order to enable cooperative perception between connected cars.

Finally, the performance evaluation uses a RESTCONF server installed in a Raspberry Pi aboard of a small car. This server is responsible for the sensors and actuators of the car and allows for its remote control from a user terminal (e.g., a smartphone, tablet, or laptop) and through the fog node, running a control application as a service.

**Index Terms**—Connected Car, 5G, SDN, NFV, YANG

## I. INTRODUCTION

It is expected that in the near future all vehicles will have connectivity. Vehicles might be connected to the Internet, to their peers (vehicle to vehicle - V2V), pedestrians (vehicle to pedestrian - V2P), the road infrastructure (vehicle to infrastructure - V2I), or the communication network (vehicle to network - V2N). The communication between two entities where at least one of them is a vehicle is typically referred to as Vehicle-to-Anything (V2X) communications.

V2X is expected to improve road safety, increase traffic efficiency, and enhance user comfort. New services are appearing on top of this connectivity, thus extending the automotive ecosystem to new players entering into this business domain [1]. However, many challenges are still unsolved. One of the key open questions to be solved is how communication networks should be tailored for V2X communications. In some sense, the problems inherited from Internet of Things (IoT) also apply to V2X: i) need for interoperability, ii) coexistence of a wide variety of sometimes opposite requirements defined

by different applications, and iii) need to support high scalability [2]. In order to cope with these challenges, 5G technologies are being introduced.

As exposed, and understanding the connected car as a complex cyber-physical system (CPS), many of the communication techniques which have been designed for the IoT can be applied and adapted (i.e., redesigned) to those networks which will handle the connected vehicle. This includes, among others, optimized wireless communication protocols, data formatting protocols, cloud computing, Software-Defined Networking (SDN), Network Function Virtualization (NFV), etc.

Fog computing [3] is defined as a system-level architecture to extend compute, network, and storage capabilities of the cloud to the edge of the network. Fog and cloud computing can help processing the data gathered by billions of smart things interacting with each other. In the context of fog computing, the introduction of the SDN paradigm enables global orchestration of all network resources. This also includes the management of distributed fog and cloud domains and the coexistence of heterogeneous networks combining different types of communication technologies. Multi-access Edge Computing (MEC) technology has been proposed by [4] with its on-premises feature to support data transmission time decrease and to enhance quality of user experience in latency-sensitive applications. MEC and fog computing bring together a cooperative solution [3] in order to support the needs for integration of different applications.

In its turn, NFV [5] has introduced a novel paradigm where services can be deployed on demand in order to fulfill the end user needs. These three techniques (fog computing, SDN, NFV) are intertwined: in future communication networks, services will be deployed over a cloud computing infrastructure, where the necessary connectivity is provided by an SDN controller. The authors have previously proposed in [6], the usage of a service orchestrator for IoT applications.

The modern automobile is an electronic system with computing capabilities on wheels, with more than 100 on-board processors per vehicle [7]. This trend is motivated by a number of converging requirements and proposed applications, such as the described in [8]: i) the need for connectivity of automobiles

to sources of travel information and entertainment, ii) the need for V2V and V2I exchanges for accident prevention, iii) the move toward more dynamic and modern vehicle maintenance, iv) the need to rationalize the electronic vehicle control architecture by reducing control system weight and cost of software development, v) the evolution toward electric vehicles, and, vi) the need to assist or even replace drivers.

Beyond connectivity, the ultimate key element here is the data, from which real value can be obtained. In the end, connectivity is just the means to gather and obtain the data. So, when it comes to processing and operating the data, formatting this data becomes a key design decision. Indeed, the adoption of a common, flexible, and powerful data and information modeling language to define all sensors, actuators, gateway facilities and services is a first important step towards the standardisation of IoT frameworks across multiple vendors beyond the existing ones. The automotive sector is not an exception to this.

Among other options, over the last years, YANG has been steadily growing in the IT and networking communities as a data modeling language suitable for the IoT [9]. For such purpose, YANG data models need to be complemented with NETCONF/RESTCONF protocols [10]. These protocols enable the control and management of YANG data models.

The common modeling of sensor and actuator data will enable and simplify the architecture of cooperative sensing and perception applications [11], which support use cases for the exchange of sensor data (e.g., raw sensor data), object information and real-video streaming that increase the environmental perception of vehicles, to help the driver or automated car to perform critical manoeuvres (e.g., overtaking) and navigate safely through dangerous areas (e.g., city intersections, highways merge-in). Low latency in the order of 10 ms and high data rates (above 20 Mbps) are characteristic requirements of this category.

The authors have previously presented [12] a demonstration of a cloud control application based on YANG/RESTCONF. Motivated by all this context, this paper describes the first proof-of-concept of an architecture supporting a remotely-controlled car through a fog node and SDN/NFV-enabled communications network where YANG and NETCONF/RESTCONF have been used to model, manage, and control the data associated to the car. The remote control is based on an application running on a proposed fog node architecture.

The remainder of the paper is organized as follows. The proposed architecture is described in Section II. Section III describes the proposed fog node architecture. Section IV is dedicated to cooperative perception application. Section V is devoted to describe the experimental evaluation set-up and discuss the key demonstrated results. Finally, Section IV provides conclusions.

## II. PROPOSED SYSTEM ARCHITECTURE

The authors have previously proposed in [6] the usage of a service orchestrator for IoT applications. Under this context,

the SDN orchestrator must carry out the following three key functions: i) facilitate the transport of the huge amount of data generated at the terminals, sensors, machines, nodes, etc., to any distributed computing node, edge, or core data center; ii) allocate computing and storage resources in distributed fog nodes and data centers; and iii) process the collected data to make proper decisions, leading to the concept of cognition.

Figure 1 describes the proposed network architecture for a connected car. From the network perspective it provides the definition of specific QoS policies, traffic management and network slicing schemes in multi-RAT, multi-link operation. Moreover, it shows both a network infrastructure-based and sidelink communication based solutions, including positioning insights.

Figure 1 shows proposed locations for fog computing in a connected car environment:

- A fog node could be inserted inside the car, in order to offer the various third party OAM services and applications on top of the same infrastructure (e.g., lane merge, see through, ftp client, video client). This approach would simplify the vehicle control architecture, reduce control system weight and cost of software development.
- Following Multi-access Edge Computing (MEC) architecture, a fog node could be located on the BTS, where RAN information can be accessed in real time. Moreover, this location could allow the allocation of ITS services and applications near the edge of the network in order to provide low-latency.

## III. FOG NODE ARCHITECTURE

The fog node (Figure 2) is the fog/network infrastructure which is responsible for provisioning the necessary computation, storage and networking resources. Physically, a fog node, might consist of a set of (mission critical) servers, hard disks and network interface cards (including WiFi, Ethernet or 5G connectivity). Sensor and actuators might be connected to the fog node. Logically, the fog node incorporates both a node manager in order to handle the offered virtual resources and a security manager, which is the responsible for handling the device authorization, authentication and encryption.

The Node Manager orchestrates the optimal allocation of the required virtual resources onto the node physical resources, which are the pool of existing resources within the node. The virtual allocated resources might include containers or virtual machines (depending on the capabilities of the node), allocated virtual memory, virtual disk space or virtual network interface cards and virtual switches.

Possible fog services abstract the access to the different smart things and devices. Example of these applications are lane merge, see through, FTP client, trajectory check, or video client.

Finally, in order to allow the usage of all these functionalities, the fog node offers a common NorthBound Interface (NBI) that can be split into three different sections: i) Network API, ii) Service API, and iii) Station API, which allows to

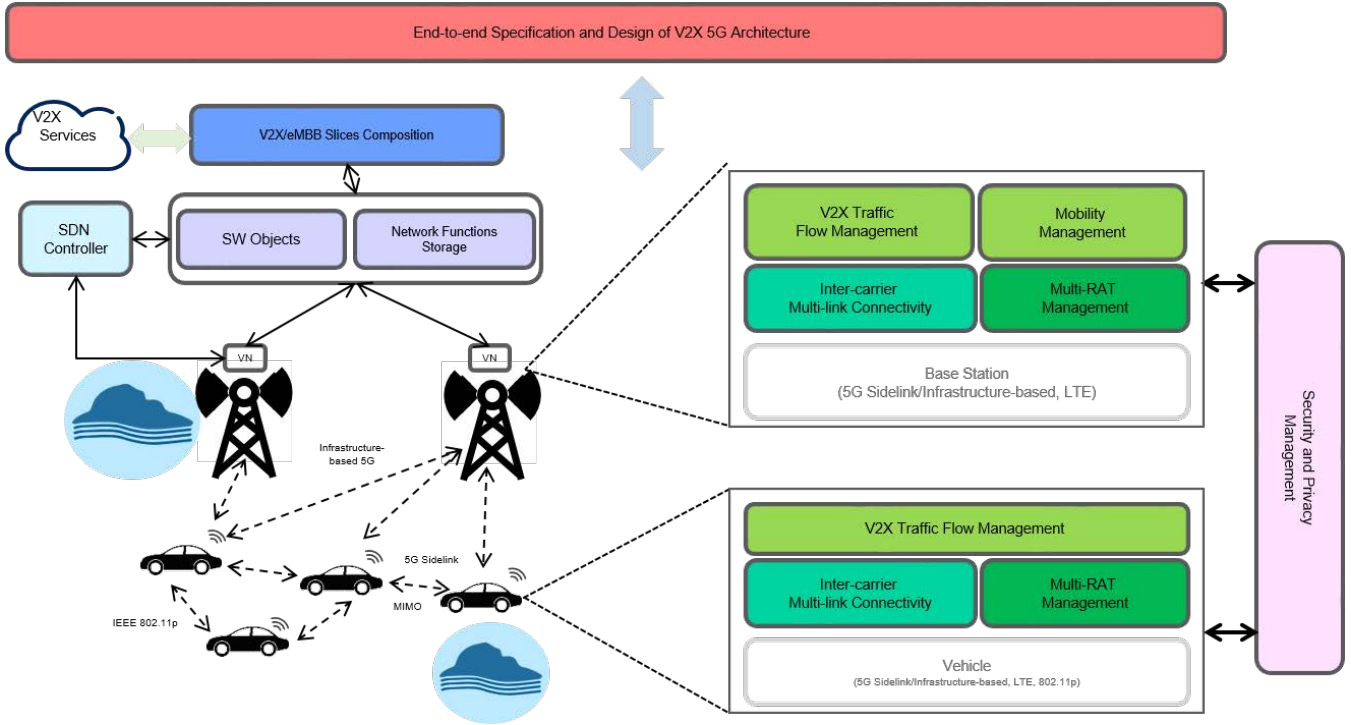


Fig. 1. Connected Car demonstration architecture

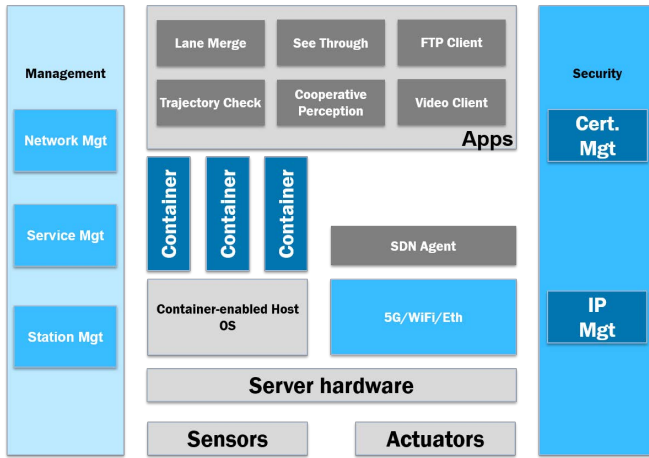


Fig. 2. Fog node architecture

control and manage both the fog node and the connected sensors and actuators.

#### IV. COOPERATIVE PERCEPTION

Future vehicle can be perceived as a powerful mobile device that is equipped with various sensors (camera, radar, ultra-sound, etc.), having adequate computational resources to support a wide range of automotive services. With the increasing availability of vehicles that are capable of supporting higher automation levels, the need for coordination among vehicles and their capability to do so becomes increasingly more relevant. All automated vehicles rely on the premise

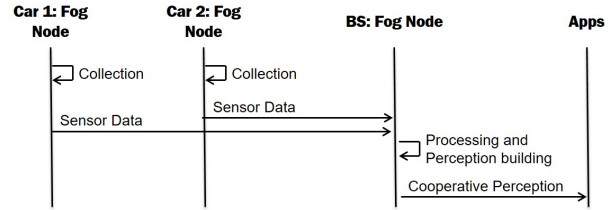


Fig. 3. Cooperative perception workflow

that they continuously plan their trajectories and, based on the observed environment, select the driving trajectory. A list of operations that will be combined are required by a cooperative and automated vehicle (cooperative manoeuvres, cooperative perception, traffic efficiency, platoon etc).

Cooperative perception describes the increase in situational awareness of vehicles with automated driving capabilities to provide longer perception range. The sensing region of a specific vehicle can be extended beyond line-of-sight by combining the sensing regions of different connected vehicles in the vicinity. Cooperative perception is beneficial for different levels of automated driving from partial automation up to full automation, especially for the cases where connected and normal vehicles co-exist. The cooperative extension of the perception range could be built based on exchanging and combining data from different sources, e.g., radars, lidars, stereo-vision sensors or video streaming from on board cameras.

The cooperative perception (on-demand or proactive) facilitates better driving decisions, provides assistance to perform

critical manoeuvres (e.g., overtaking, cooperative collision avoidance or early lane change) and improves traffic safety and efficiency by detecting and avoiding hidden or sudden obstacles, help the driver or automated car to navigate safely through dangerous areas (e.g., city intersections, highways merge-in).

The fog architecture could be used to support the building of cooperative perception of connected cars, as it is visualized in Figure 3. Each connected car via its fog node provides the (processed and/or unprocessed) sensor data (e.g., obstacles, vehicles pedestrians) to the fog node located at the BS. The later by collecting the various reports by the cars is able to process, correlate and build in a cooperative manner the perception of the corresponding cars about their surroundings. The fog node at the BS allows the building of cars awareness in low latency that could be provided to their local dynamic maps or used by other Apps available at the fog node architecture (see Figure 2).

## V. EXPERIMENTAL EVALUATION

Figure 4 shows the proposed architecture for the control and management of the connected car. On it, the Service Orchestrator acts as a generalized NFV Manager and Orchestrator (NFV MANO), and is responsible for: 1) Triggering the necessary flows in the SDN controller in order to interconnect the OpenVSwitch (OVS) bridge, 2) requesting the necessary resources and deploying the requested services on top of them, and 3) deploying services, which are independent from each other, using the Fog Controller.

In our particular case, two applications are run: the car control application and the car register. is a service that the Service Orchestrator deploys. The car control application consists of an Apache PHP application, which is able to act as a RESTCONF client of the connected car. The car register is implemented using a MYSQL database, which logs all the received commands from the car control application.

The fog node implementation is properly described in [3]. It is based on Dockers and OpenVSwitch, in order to deploy containers per each necessary service. In order to offer each Fog node as a Virtualized Infrastructure Manager (VIM) in the proposed NFV architecture, we have created an agent, which exposes Fog node capabilities [13].

The VIM is responsible for the creation/migration/deletion of container instances (computing service), disk images storage (image service), and the management of the network interfaces (networking service). An SDN controller (e.g., ONOS) is the responsible for the control of the network resources (such as the OVS bridge). The SDN controller translates high level connectivity intent requests into OpenFlow (OF) protocol commands, in order to configure the necessary underlying network resources for the establishment of the requested connection.

The connected car data model is described in YANG modeling language. RESTCONF is used as transport protocol which uses JavaScript Object Notation (JSON) encoding for data transmission. RESTCONF is an HTTP-based protocol for

configuring data defined in YANG. It uses HTTP methods (i.e., POST, PUT, PATCH, and DELETE) to provide Create Read Update and Delete (CRUD) operations on a conceptual datastore containing YANG-defined data.

In the implementation described in this paper, the connected car data model is composed of three main parameters (shown in Fig. 5): 1) robotId, 2) speed, and 3) command. The allowed commands are: FORWARD, STOP, BACKWARD, LEFT, and RIGHT, indicating the direction of motion of the car.

The demo toy-car is composed of a metal chassis and 4 wheels each driven by 4 DC motors powered with dedicated 5 AA batteries in series, providing 7.5V. The car is equipped with a Raspberry Pi Zero W (RasPiZW) [14]. RasPiZW is a basic computer consisting of a single-core 1GHz ARM11 processor (Broadcom SoC BCM2835), with 512MB of RAM. This version offers 802.11n wireless LAN and Bluetooth 4.1 and BLE wireless connectivity with a Cypress CYW43438 chip and a 2.4GHz PCB antenna. It needs constant 5V power supply via a micro USB connector. Power consumption in Idle state is about 120mA. The RasPiZW in the car is supplied with a battery pack. The operative system, based on Debian Jessie, is Raspbian Jessie Lite.

The OVS bridge is implemented using a Raspberry Pi 3, including various Ethernet to USB adapters. OpenVSwitch is setup in order to control the data plane Ethernet connections.

The RESTCONF server has been autogenerated with Open-SourceSDN.Org project EAGLE YANG to code tools [15]. These tools allow for the rapid prototyping of RESTCONF servers using the feed from YANG data models.

Figure 6 presents the suggested workflow for the control and management of the connected car using YANG/RESTCONF. The Control Application is a PHP application, which shows the Control UI, and translates the pressed commands into the necessary HTTP RESTCONF commands for the control of the connected car. At least three scenarios are envisioned.

Scenario A consists on the movement of the connected car. A Forward command is sent from the client to the Control Application. The Control Application issues the RESTCONF HTTP PUT operation, including the received command, the selected robotId, as well as the previously configured speed for the motors. The OVS bridge forwards the command, and the Connected Car RESTCONF server processes the command and activates the necessary motors. Once activated, the HTTP 200 OK command is answered to the Control Application which, in its turn, notifies the client.

Scenario B focuses on speed selection. In this scenario, the client selects the speed for a certain robotId. This information is received by the Control Application and it is stored in a local MySQL database. No interaction with the Connected Car is performed in this case.

Finally, Scenario C provides a stop mechanism for the connected car. Finally, the client requests the stop command to the Control Application. As in Scenario A, the Control Application issues the RESTCONF HTTP PUT operation, including the received command and the selected robotId.

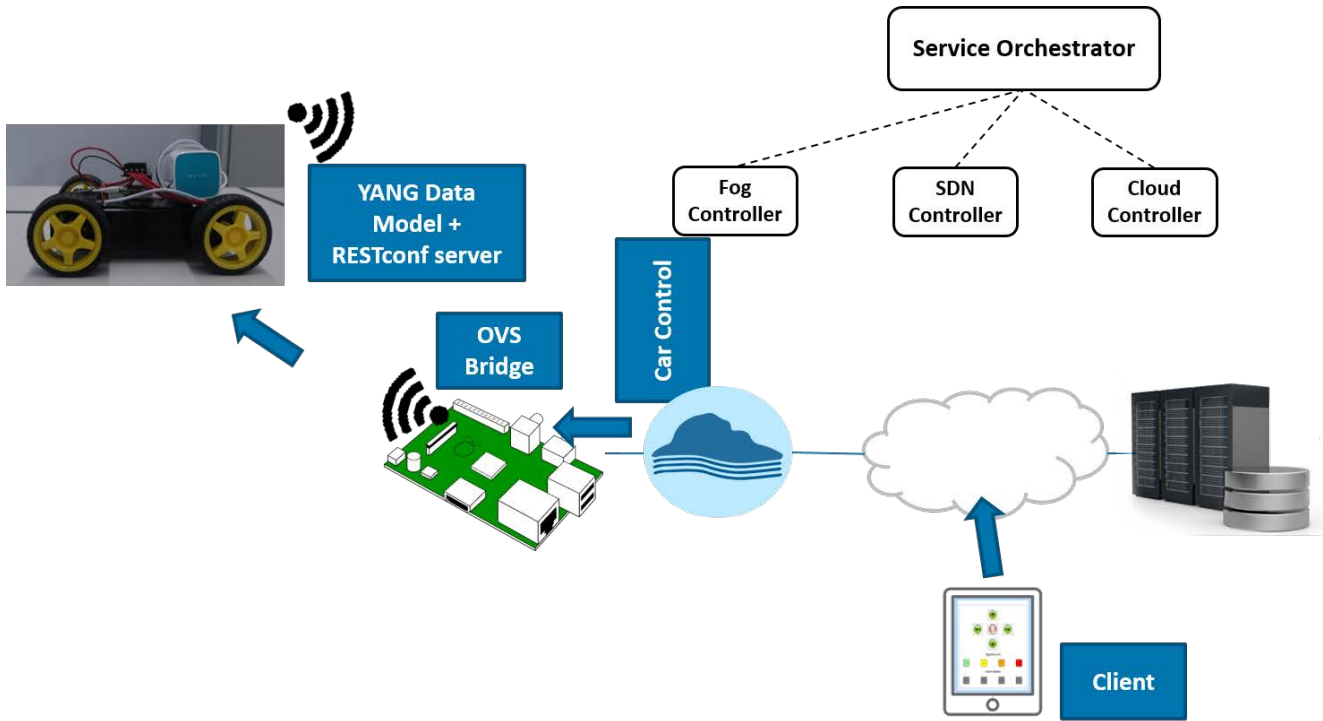


Fig. 4. Example of Connected Car UI and JSON command

```

JavaScript Object Notation: application/json
Object
  Member Key: command
    String value: STOP
    Key: command
  Member Key: robotId
    String value: robot1
    Key: robotId
  Member Key: speed
    Number value: 8
    Key: speed

```

Fig. 5. Captured message exchange with wireshark

As a key performance indicator (KPI), Figure 7 shows the round-trip delay time between the connected car and the car control application running in the fog node during time. This delay includes the processing delay, and the engine delay. It can be observed that the mean is around 260ms. This delay can be easily reduced by optimizing the server code and increasing the computation capacity at the connected car. It can be noted that RasPiZW does not provide enough resources for the RESTCONF server.

## VI. CONCLUSION

In this paper, we have described the first implementation of a proof-of-concept for a remotely-controlled car using YANG modeling for the data associated to the car, and operating an SDN/NFV network with dynamic service provisioning at the network edge using fog computing. A user can interact with

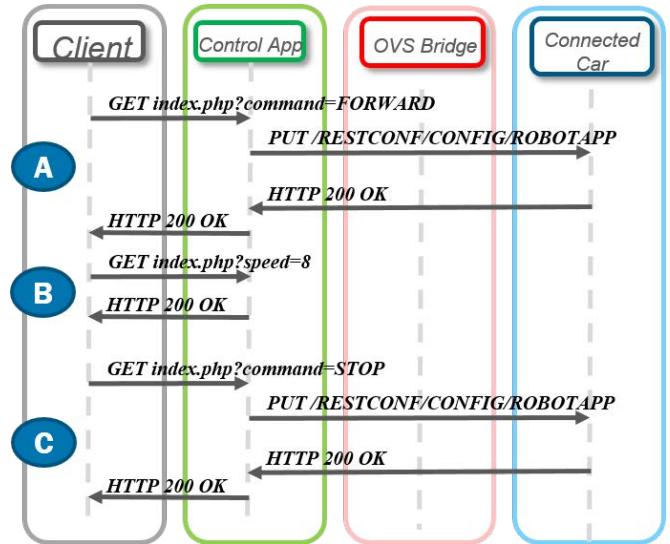


Fig. 6. Proposed Message exchange workflow

the car through an application with a user interface which is served by a network orchestrator and can be operated from any handheld or desktop device. This proof-of-concept has demonstrated: first, the feasibility of the proposed approach of using YANG modeling language for the description of IoT services, and second, the suitability of fog computing for unifying the NFV paradigm together with IoT services.

In a more complex scenario the vehicles will have sensors and their sensor data are reported from the fog node of the

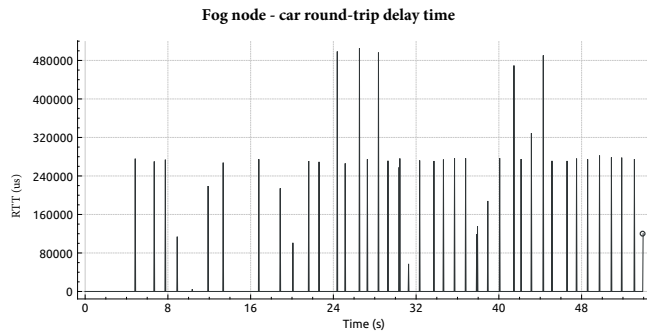


Fig. 7. Round-trip delay time between fog node and connected car

vehicle to the Fog node of the BS. At the BS fog node the sensor data are processed (in low latency) and the derived environment awareness (e.g., obstacles, road conditions) is provided to the screen of the client to control the connected car.

In future work, we plan to consider the use of the gRPC protocol, instead of general purpose RESTCONF protocol, to achieve lower latencies and higher reliability, with the ultimate goal of meeting the requirements posed by 5G technologies. Moreover, sensors are being installed in the toy car platform in order to produce more data to be processed in the fog, and also in the cloud. This will allow the introduction of cooperative algorithms in the experimental evaluation.

Novel radio interfaces such as 802.11p can be incorporated in order to reduce latency measurements.

Mobility support for the architecture will also be considered in further research work, including discovering and rendezvous between V2X actors, handover between V2X and fog nodes, and trust aspects.

#### ACKNOWLEDGMENT

Part of this work has been performed in the framework of the H2020 project 5GCAR co-funded by the EU. Authors would like to acknowledge the contributions of their colleagues from 5GCAR although the views expressed are those of the authors and do not necessarily represent the views of the 5GCAR project. This research also has been partly funded by Spanish MINECO projects DESTELLO (TEC2015-69256-R) and CELLFIVE (TEC2014-60130-P).

#### REFERENCES

- [1] B. Martinez de Aragon, J. Alonso-Zarate, A. Laya, "How Connectivity is Transforming the Automotive Ecosystem," *Wiley Internet Letters*, in press 2017.
- [2] NGMN Alliance, "NGMN 5G White Paper," NGMN Alliance, February 2015, Tech. Rep.
- [3] R. Vilalta, V. Lopez, A. Giorgetti, S. Peng, V. Orsini, L. Velasco, R. Serral-Gracia, D. Morris, S. De Fina, F. Cugini *et al.*, "Telcofog: A unified flexible fog and cloud computing architecture for 5g networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 36–43, 2017.
- [4] G. Secinti, B. Canberk, T. Q. Duong, and L. Shu, "Software defined architecture for vanet: a testbed implementation with wireless access management," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 135–141, 2017.
- [5] ETSI ISG, "Network Functions Virtualisation (NFV); Architectural Framework," ETSI, February 2014, Tech. Rep.
- [6] R. Vilalta, I. Popescu, A. Mayoral, X. Cao, R. Casellas, N. Yoshikane, R. Martínez, T. Tsuritani, I. Morita, and R. Muñoz, "End-to-end sdn/nfv orchestration of video analytics using edge and cloud computing over programmable optical networks," in *Optical Fiber Communications Conference and Exhibition (OFC)*, 2017. IEEE, 2017, pp. 1–3.
- [7] F. Bonomi, S. Poledna, and W. Steiner, "8 the role of fog computing in the future of the automobile," *Fog for 5G and IoT*, p. 191, 2017.
- [8] 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 Communications Magazine*, vol. 55, no. 7, pp. 94–100, 2017.
- [9] M. Yannuzzi, F. van Lingen, A. Jain, O. L. Parellada, M. M. Flores, D. Carrera, J. L. Pérez, D. Montero, P. Chacin, A. Corsaro *et al.*, "A new era for cities with fog computing," *IEEE Internet Computing*, vol. 21, no. 2, pp. 54–67, 2017.
- [10] A. Bierman, M. Bjorklund, and K. Watsen, "Restconf protocol, rfc 8040," 2017.
- [11] A. Kousaridas, D. Medina, S. Ayaz, and C. Zhou, "Recent advances in 3gpp networks for vehicular communications," in *Standards for Communications and Networking (CSCN), 2017 IEEE Conference on*. IEEE, 2017, pp. 91–97.
- [12] R. Vilalta, S. Via, F. Mira, L. Sanabria, R. Martinez, R. Casellas, R. Muñoz, and J. Alonso-Zarate, "Control and management of a connected car using yang/restconf and cloud computing," in *Network of the Future 2017*. IEEE, 2017, pp. 1–3.
- [13] "Docker-agent," <https://github.com/rvilalta/docker-agent>, accessed: 2018-03-26.
- [14] E. Upton and G. Halfacree, *Raspberry Pi user guide*. John Wiley & Sons, 2014.
- [15] OpenSourceSDN.org, "EAGLE Project," <https://github.com/OpenNetworkingFoundation/EAGLE-Open-Model-Profile-and-Tools/>, 2017.