IoV Simulation Architecture for Software-Defined Vehicular Fog Network Orchestration

Leonel D. C. Alvarenga[1,2(✉)], Pedro Sousa[2], and António Costa[2]

[1] Instituto Federal Goiano, Rio Verde - GO, Brazil
leonel.carvalhaes@ifgoiano.edu.br
[2] Centro Algoritmi, LASI, Department of Informatics, University of Minho, Braga, Portugal
{pns,costa}@di.uminho.pt
https://www.ifgoiano.edu.br/rioverde , https://algoritmi.uminho.pt/

Abstract. Internet of Everything (IoE) extends Internet of Things (IoT) by going beyond device connections and creating a novel interconnected society with innovative business models supported by intelligent applications that integrate people, things and data. As a specific case of IoT, Internet of Vehicle (IoV) paradigm is directly influenced by IoE-based applications, bringing even more intelligence to vehicular networks. In IoV context, many applications require lower latency, making Fog Computing a promising model to orchestrate cloud-based IoE services consumed by vehicle nodes. Recent research suggests that the Software-Defined Networking (SDN) paradigm can optimize Fog Computing in resource and service management, leading to the concept of Software-Defined Vehicular Fog Computing Networks (SDVFN). However, testing novel research solutions in real SDVFN environments is a challenge due to the cost of physically build these environments or the disruptive model they represent. In addiction, the lack of a comprehensive simulation tool for the SDVFNs scenarios represents a barrier to the advancement of research in this domain. This article introduces a simulation framework designed to facilitate the prototyping of Intelligent Transportation System Applications within a SDVFN. The framework enables the investigation of various fog node selection strategies and the implementation of adaptable datapaths using the SDN paradigm. Furthermore, it contributes to the evaluation and implementation of efficient computing services orchestration in SDVFN environments. Among the key features of the environment are its handling of vehicular mobility through service-oriented packet routing and its capability to anticipate OpenFlow forwarding rules by predicting handovers.

Keywords: Vehicular Fog Networks · Software-Defined Vehicular Fog Networks · Internet of Vehicles · Internet of Everything for Intelligent Transportation

T. Pereira et al. (Eds.): IoECon 2024, LNICST 607, pp. 40–54, 2025.
https://doi.org/10.1007/978-3-031-84426-3_4

1 Introduction

With the expansion of urban areas and the increase in the number of vehicles, applications of Intelligent Transportation System (ITS) are revolutionizing the management and interaction within vehicular networks. Typically, ITS refers to the application of information, communication, and sensing technology to transportation and transit systems. It is likely to be an integral component of the smart cities of the future [1]. These applications fully fulfill the concepts of Internet of Everything (IoE), using advanced technologies such as artificial intelligence and machine learning to improve the efficiency, safety and sustainability of transportation networks [2].

Internet of Everything (IoE) paradigm goes beyond device connections, extending the Internet of Things (IoT) concept and creating a new interconnected society with innovative business models supported by intelligent applications that integrate people, things and data [3].

From this IoE perspective, by facilitating real-time communication between vehicles, infrastructure and central management systems, ITS applications can optimize traffic flow, mitigate congestion, and improve the overall driving experience.

In this context, the Internet of Vehicles (IoV), under the umbrella of IoT, emerges as a promising paradigm to enable a wide variety of ITS applications, taking advantage of connectivity and data exchange capabilities between vehicles and infrastructure, real-time sharing of information such as traffic conditions, road hazards, and weather updates. In addiction, it can improve safety and efficiency on the roads. [4,5].

Vehicular nodes in IoV can benefit from various services provided by Cloud and Fog Computing Networks. While Cloud Computing offers a centralized platform for processing and storing data, Fog Computing brings computational resources closer to the edge of the network, enabling low-latency processing and real-time decision-making. Being the junction of these two network paradigms, Vehicular Fog Network (VFN) is an environment in which vehicular nodes can access many services that are hosted on a Fog Computing Network. This is particularly beneficial for applications that require immediate responses. By handling data locally, VFN reduces the amount of data to be transmitted to the cloud, conserving bandwidth, and reducing costs. It also improves service availability, allowing operations to continue even when the connection to the cloud is lost, and increases data privacy and security by minimizing data exposed to network transmission [6].

Within an IoV environment, fog platforms need to face additional challenges as vehicles move between Road-Side Units (RSUs) while preserving their roles in computing, communication, and end-user functionalities. Furthermore, the stringent Quality of Service (QoS) requirements of numerous ITS applications impose significant challenges for the deployment of VFN [4].

Modern networking paradigms, specifically the Software-Defined Networking (SDN), can be used to improve Fog Computing orchestration to manage resources and services across edge devices [7]. It can solve problems and limita-

tions that occur in traditional environments by using an architecture designed to simplify and improve network management. In the SDN architecture, the control plane and the data plane are decoupled, so the control of the network is separated from the packet forwarding mechanism. Thus, the network control function can be programmed directly, adapting to the reality of each scenario, as well as promoting improvements in existing mechanisms [8].

Software-Defined Vehicular Networks apply the SDN approach to get better results on typical tasks in IoVs and vehicular ad hoc networks. This paradigm has become an excellent alternative to promoting centralized network control, improving flexibility and programmability, simplifying network management, optimizing wireless network resources, improving heterogeneity management and enabling even more intelligent and scalable services in Vehicular Network environments [9,10].

The authors in [7] present research on SDN and Fog Computing in vehicular networks. They envision an architecture that uses SDN to support Fog Computing in Vehicular Ad Hoc Networks (VANETs). The authors do not deal with ITS applications/services and with Fog Computing node orchestration. In addiction, they do have not implemented the architecture, but they pointed important future works:

- The creation of a Fog orchestration model that takes into account the capabilities of SDN to support Fog Computing;
- The implementation of an SDN protocol to optimize the reorganization of datapaths, responding to the mobility nature of vehicular networks;
- The improvement of load balancing techniques;
- The need to implement and evaluate the proposed architecture in a simulation environment or on a real environment.

In their work, the authors of [9] present a comprehensive review on Software-Defined Vehicular Networks. They envision research challenges for Software-Defined Vehicular Ad hoc Networks (SDVN), among which are the management of changes in SDVN and the adaptive reconfiguration of the network.

The authors in [11] wrote a review of VANET simulators. They compare available simulators in many dimensions, including support for new technologies such as SDN and Edge Computing. However, the study does not mention that existing simulators could address service orchestration in Vehicular Fog Computing Networks using SDN.

There is still a lack of work in the literature that comprehensively evaluates Software-Defined Vehicular Fog Network (SDVFN) scenarios, with the ability to orchestrate ITS services/applications in a Fog Computing paradigm to be consumed by connected vehicles, as well as promoting the network's ability to adapt to the mobility nature of vehicular networks.

Experimental environments are necessary to put IoV, SDN, and Vehicular Fog Networks models into practice. However, it is not always possible to implement real test-beds, either because of the cost of the physical construction of such environments or because of the disruption model of the experiment when compared to usual models.

The simulation of an SDVFN scenario, including the orchestration of ITS applications in Vehicular Fog Computing environments and the exploitation of SDN resources, is therefore essential for carrying out research in this field. Simulations can allow the evaluation of various metrics, including vehicle mobility and network communication, as well as other resources, depending on the focus of the research.

However, reviews in the literature, such as [7,9,11], do not point to a network simulator that can deal with an SDVFN scenario directly and comprehensively.

This paper presents a simulation framework designed to directly support the implementation and the evaluation of service orchestration in Software-Defined Vehicular Fog Networks (SDVFN) with the following capabilities:

- Implement different strategies for selecting Fog Computing nodes to compare load balancing algorithms;
- Use the SDN paradigm to provide adaptability to datapaths, given the mobility of vehicles;
- Employ a service-oriented routing rule strategy, as proposed in our previous work [12];
- Predict vehicular handover, and anticipate the sending of openFlow routing rules for the correct reorganization of datapaths.

In addition, an ITS application use case is also implemented to evaluate the proposed framework.

The rest of this document is structured as follows. Section 2 presents the proposed SDVFN simulation approach to test and evaluate the deployment of ITS applications, the Eclipse MOSAIC adaptation, and details of the implementation. Section 3 describes the experimental use case, analyzing its effectiveness and the obtained results. Section 4 presents the conclusions about the work.

2 Proposed Simulation Approach for SDVFN

This work proposes a simulation framework for SDVFN environments to test and evaluate the deployment of ITS applications. We test our approach by using a fog application that processes computing tasks from connected vehicle nodes. We used Eclipse MOSAIC Framework [13] to implement various SDVFN-related functionalities, resulting in a simulation environment that enables:

- Prototyping of ITS applications in SDVFN.
- Orchestration of fog nodes to perform processing tasks from vehicles, using a load-balance strategy;
- Employ a service-oriented routing rule strategy using an specific [vehicle,service] tuple;
- Distribution of OpenFlow based forwarding rules to switches, and reorganization the forwarding datapaths;
- The implementation of vehicular handover prediction algorithms.

The proposed SDVFN is made up of a VANET on the wireless side and a Software-Defined Fog Computing Network on the infrastructure side, both interconnected by Road-Side Units (RSUs) distributed along urban transportation roads, as can be seen in Fig. 1.

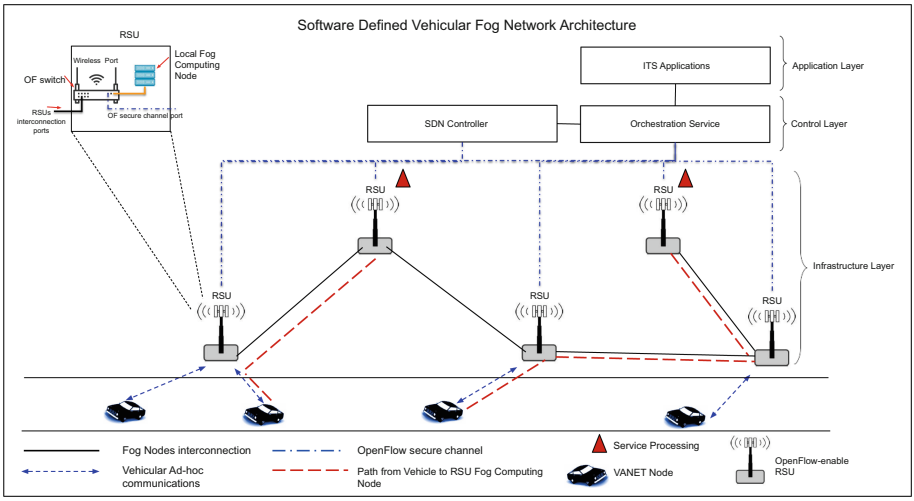

Fig. 1. SDVFN simulation approach.

The SDVFN architecture is composed of three layers. The Infrastructure Layer, the Control Layer, and the Application Layer.

In the infrastructure layer are included all physical and virtual hardware elements that facilitate data processing, storage, and communication within the network. This comprises vehicles with computational and communication features, RSUs, and fog nodes that can be used for data processing and storage tasks near the network edge.

Each RSU depicted in Fig. 1 provides a fog node dedicated to computation tasks, connected to an SDN OpenFlow switch. This switch acts as an access point, interconnecting the wireless VANET nodes with the SDVFN, and is also responsible for forwarding packets along the communication datapath. Furthermore, each switch is directly interfaced with an SDN controller through a dedicated channel, as can be seen in Fig. 1. The details of the SDN OpenFlow switch are described in Sect. 2.2.

In the control layer, the SDN principles are applied to orchestrate and manage the underlying infrastructure. It has a comprehensive view of the network, facilitating centralized governance and dynamic configuration of network resources. This layer encompasses SDN controllers and management systems that oversee network performance, allocate resources, optimize data flow, and enforce policies to meet network requirements. The implementation of the control layer is described in Sect. 2.3.

The application layer comprises multiple services and applications that make use of the network infrastructure and control systems to provide functionalities to end users. It takes advantage of the distributed computing power of the fog nodes to handle data locally, thus decreasing latency and enhancing service delivery.

2.1 Simulation Framework

To build our proposed Simulation framework, we use Eclipse MOSAIC [13]. It is a multi-domain and multiscale simulation framework for testing and developing connected and automated mobility solutions. Eclipse MOSAIC uses High Level Architecture (HLA) to standardize interfaces and employs the concept of federates and ambassadors, coupling individual simulators to the whole environment, so acting as a co-simulation framework. Federates can be wrapped into a Federate object, which is linked to an Ambassador for direct connection with the MOSAIC runtime infrastructure (RTI), as can be seen in Fig. 2. All the management tasks of the simulation life cycle are in charge of the RTI. It includes Federation Management, Time Management, and Interaction Management.

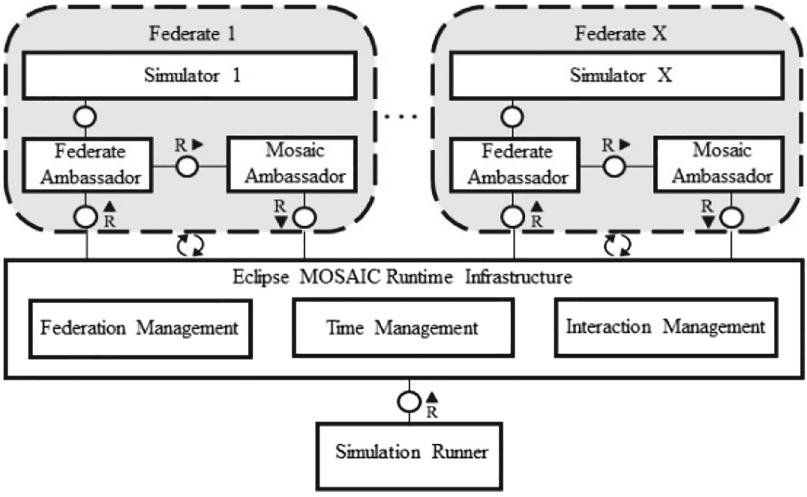

Fig. 2. HLA architecture of Eclipse MOSAIC, Available at [14]

Traffic simulation is carried out using the Eclipse SUMO simulator [15]. All interactions between SUMO and Eclipse MOSAIC are performed through the Traffic Control Interface (TraCI). It is from this interface that it is possible to retrieve information about simulated entities, as well as manipulate their behavior in real-time. On the Eclipse MOSAIC side, the *SumoAmbassador* class interacts with Sumo TraCI and is responsible for implementing the object that represents this Federated Simulator.

The simulation of applications is in charge of the *ApplicationAmbassador* class provided by the built-in MOSAIC application simulator. This simulator prototypes applications in all simulation units, which can be Vehicles, RSU, Traffic Lights, Traffic Manager Center (TMC), and Servers.

Applications must be deployed into each Unit. They are compiled JAVA classes, which extend the abstract class *AbstractApplication* and implements the *CommunicationApplication* interface.

Each abstract application interacts with the unit's operating system application to gain access to the simulated unit's parameters and perform specific actions of these units, such as navigation and communication between them, using their respective communication modules.

Eclipse MOSAIC offers a domain-specific architecture to couple with a range of simulators: Application simulators, Traffic and Vehicle Behavior Simulators, Network and Communications Simulators, Environment Simulators, and E-Mobility Simulators.

In the field of Network and Communication Simulations, the framework enables the simulation of Cellular Communications and ITS-G5 Ad Hoc Communications. These simulations can be performed using external simulators such as OMNeT++ and NS-3, or using built-in Eclipse MOSAIC simulators. The built-in MOSAIC Simple Network Simulator (SNS) is used to simulate ITS-G5 Ad Hoc networks, while the built-in MOSAIC Cell Simulator is used to simulate cell-network communications. Eclipse MOSAIC does not offer directly wired communications, but we adapt the Cell communications to behave like wired ones. In our work, we use native SNS simulator because the Ambassadors for OMNeT++ and NS-3 do not support wired networks, too, and they impose extra delay because of the integration.

2.2 SDVFN Infrastructure Layer

The built-in Eclipse MOSAIC network simulators offer many communication modes. Cellular communications in MOSAIC Cell are available to entities that have a cellular interface enabled. The communication modes are as follows:

- **GeoBroadcast**: Broadcast cellular communication in a specific area.
- **Geocast**: Unicast Cellular Communication in a specific area.
- **Topocast**: Unicast Cellular Communication regardless of geographical area restrictions.

Ad hoc communications simulate the IEEE 802.11p standard in the 5.9 GHz frequency band (5.85–5.925 GHz), which is intended for ITS across seven possible channels. The available modes of ad hoc communications are:

- **Geobroadcast**: In this mode, messages are broadcasted, and therefore all simulated entities with ad hoc enabled within a specific range will receive the messages.

- **GeoCast**: Ad hoc geocast mode enables unicast communication between two entities on the same ad hoc channel if the receiver's IP address is known. This type of simulated transmission does not impose geographical restrictions on communication between entities.
- **Topobroadcast**: This mode is designed to disseminate information to all nodes within the signal range. This mode can operate in single-hop or multi-hop configurations, and the entities involved in the communication must be operating on the same ad hoc channel. In multi-hop, message dissemination is accomplished through flooding.
- **Topocast**: This mode enables unicast communication in ad hoc networks, subject without restrictions of signal range and the number of hops.

However, the Eclipse MOSAIC framework does not offer a native method to simulate SDN and Fog Computing Networks and, consequently, SDVFN. Consequently, we engineered a suite of applications for Eclipse MOSAIC to address this deficiency.

Although the coupled OMNET++ and NS-3 simulators can simulate wired networks, Ambassadors do not have this functionality. Furthermore, there is no switch entity available to build the infrastructure network, leading us to implement software switches within the RSU Unit.

OpenFlow Switch Abstraction. The proposed SDVFN architecture simulates a subset of the OpenFlow protocol for communications between the Control Layer and the Infrastructure Layer. In addition, we implemented a novel service-driven packet forwarding strategy that specifies the actions to be taken for each incoming packet.

To establish an infrastructure network which simulates a fog network, we created a packet-switching network interconnected by OpenFlow software switches implemented as a Mosaic Application. Switches are equipped with cellular network modules and ad hoc networks to communicate with other elements of the network. Each switch abstracts four communication ports:

INTRAUNIT_PORT: This port abstracts an interface to provide communication between the OpenFlow switch and the Fog Computing Node of the local RSU where the switch is attached.

ADHOC_PORT: This port uses an ad hoc communication module abstraction to provide communication between the OpenFlow switch and the vehicular nodes of the SDVFN.

RSUS_PORT: This port uses a cellular communication module to send and receive packets from adjacent switches in the fog side of the SDVFN. It is configured to use Cellular Topocast communication with a constant delay of 10ms, without loss and no range restriction.

SERVER_PORT: This port abstracts communication with fog server to exchange SDN OpenFlow messages. Communications are carried out via cellular module using topocast mode and with a constant delay of 20ms without loss and without range restriction.

By reducing communication delays, eliminating range restrictions, and utilizing topocast communications, we enable *RSUS_PORT* and *SERVER_PORT* to function as seamlessly as wired networks.

Each switch contains a flow table comprising various flow rules designed to match incoming data packets and determine the corresponding action. The SDN OpenFlow controller can add, remove, or modify these rules via the connection, using the *SERVER_PORT*.

The infrastructure component of the SDVFN was set up using RSU OpenFlow switches, which provide standard OpenFlow packet forwarding rules. Communications occur through cellular topocast transmissions via the *RSUS_PORT* of each switch.

2.3 SDVFN Control Layer

The SDVFN control layer is composed by the network Orchestration Service and the SDN OpenFlow Controller as can bee seen in Fig. 1. The Orchestrator coordinates the allocation of services on Fog Computing nodes and optimizes communication paths, to ensure low latency and high throughput.

By integrating SDN with Fog Computing, the Orchestrator supports services that are sensitive to location and require a shorter response time. The Orchestrator also communicates with the OpenFlow Controller to manage the routing paths between vehicles and fog nodes.

The Orchestrator maintains a comprehensive data set of vehicular metrics, including positional coordinates, service utilization, velocity, directional heading, and RSU access point of the vehicle.

The SDN controller constitutes a critical component of the SDVFN. It encompasses a subset of conventional OpenFlow Controller Functions tailored to address the specific requirements of the SDVFN context. The controller processes packet-in messages transmitted from the switches via the secure OpenFlow channel, thereby allowing the addition, removal, or modification of matching rules on the switches.

Additionally, the SDN Controller maintains a network topology model to retain an abstract perspective of the whole Fog Network, encompassing the flow table of every switch. This abstraction is crucial for determining the best paths for packet forwarding and subsequently applying the forwarding rules within the switches. The abstraction of the topology includes only the infrastructure side of the SDVFN, where there are the OpenFlow switches.

OpenFlow Service-Driven Forwarding Rules on SDVFN. In the developed framework, we built personalized OpenFlow matching rules for a given service to a given vehicle. In this logic, if the same vehicle consumes different services in the SDVFN, each service can be processed on a different Fog Computing node. In the same way, different vehicle service packets could be forwarded to a different Fog Computing node. Each vehicle has a unique OpenFlow matching rule [vehicle,service] to a specific service.

The Service Orchestration Process: The SDVFN Orchestrator receives vehicle data periodically. When a vehicle sends a message to an RSU, if the vehicle is not present in the Local Dynamic Vehicle Map[1] of the RSU, it sends the vehicle's data to the Orchestrator and awaits a decision on which Fog Computing node will process the requested service. The decision will be used while the vehicle is connected to this RSU.

When the Orchestrator receives vehicle information, the data are recorded in the Global Dynamic Vehicle Map. If it is the first message, the Orchestrator chooses the appropriate Fog Computing node for the [vehicle,service] pair and requests to the SDN Controller to calculate the best path between the vehicle and the Fog Computing node. It also sends the forwarding rules to all switches along the routing path. In the developed framework, the orchestrator includes an RSU Prediction module that forecasts the next RSU access point for the vehicle.

The [vehicle,service] forwarding rule pair allows the prediction module to focus on identifying the next RSU Access Point to vehicle communications without requiring knowledge of the exact time it will happen. It compares the current path with the predicted next path to the Fog Computing node and sends OpenFlow rules in advance to the switches that are not in the current path but are in the predicted path.

3 Experimental Use Case Analysis and Results

To validate our work, we developed a use case scenario using Eclipse MOSAIC. The objective of the use case is to evaluate the suitability of the SDVFN simulation framework for the deployment of a task processing service for vehicular nodes within an urban context.

The ability of the developed SDVFN simulation scenario to orchestrate processing tasks was tested with the following criteria:

1. The simulation environment should allow the selection of specific fog nodes for processing tasks assigned to each vehicle, thus evaluating the environment's ability to employ intelligent load balancing techniques between fog nodes.
2. The simulation environment must be capable of establishing a correct datapath for each vehicle from its RSU access point to its respective processing node within the SDVFN. This is achieved by using individualized OpenFlow rules for each [vehicle,service] pair. Additionally, it should dynamically adjust the datapath if the vehicle moves to another access point. The objective is to analyze how effectively the proposed environment uses the SDN paradigm to adapt to vehicular mobility and rearrange data forwarding paths.
3. The simulation environment needs to support the execution of handover prediction algorithms and the subsequent dispatch of related OpenFlow routing directives to the switches affected in the predicted path. The main goal is to

[1] Dynamic Vehicle Map is used to maintain a list of information about the connected vehicles.

assess how effectively the environment can facilitate a comparative analysis of various handover prediction methodologies.

The mobility scenario was created for use with the Eclipse SUMO simulator [15]. This simulator interfaces with Eclipse MOSAIC through their coupling. Our setup involves a scenario with 10 RSUs interconnected by switches, forming a network topology defined by the orchestration application's JSON configuration file. Figure 3 illustrates this topology.

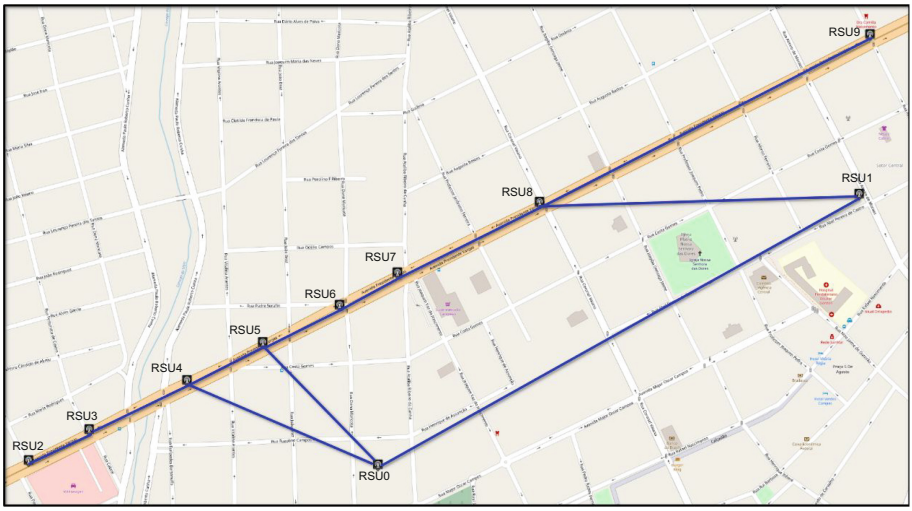

Fig. 3. Use case: RSUs Interconnection Topology

The vehicles travel along the avenue depicted in Fig. 3. They initially move through the range of RSU 2 and proceed to perform handovers between various RSUs, excluding RSU 0 and RSU 1 due to their out-of-range status. The vehicles continue until they reach the range of RSU 9, after which they return along the same route. This scenario effectively validates the simulation environment by making it possible to test our objectives.

3.1 Fog Computing Node Selection

When a vehicle connects to the SDVFN for the first time, the orchestrator applies the resource allocation strategy and defines which fog node will process the vehicle's requests. To test the allocation mechanism, only the two fog nodes associated, respectively, with RSU 0 and RSU 1 were configured as processing nodes for the requested service. The orchestrator applies the Round Robin algorithm to alternate between the two fog nodes when processing new vehicle requests as they connect to the SDVFN for the first time.

The use of the round-robin balance algorithm to select individual fog nodes for processing tasks has been shown to be both efficient and effective. Figure 4 presents a snippet of the log file from the fog nodes of RSU 0 and RSU 1. This figure illustrates the most recent processing requests from vehicles, each serviced by the fog nodes within either RSU 0 or RSU 1.

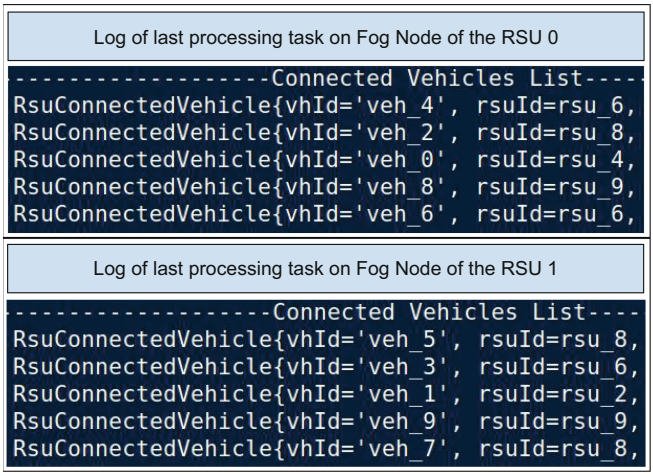

Fig. 4. Allocation of processing tasks in RSU 0 and RSU 1 using the Round Robin algorithm

The use case demonstrates that the Orchestrator, together with the SDN Controller, can make the decision regarding the allocation of processing tasks following a defined strategy, as well as implement the construction of the datapaths needed to forward data from the access point of the SDVFN to the processing Fog Node. The environment is specifically designed to enable detailed comparative analysis of different methodologies for selecting Fog Computing nodes.

3.2 Datapath Adaptability

Figure 4 illustrates each vehicle using a specified RSU as its access point to the SDVFN, denoted by *rsuId=rsu_x*. The Orchestrator has accurately determined the datapaths for each vehicle in accordance with the decisions executed by the node selection algorithm, in this case the Round Robin method.

The path taken by the data is established by a decision of the SDN Controller and distributed to all the switches that make up this path.

In Fig. 5, two distinct moments in the simulation can be visualized through the logs of two switches. The criteria for forwarding messages from different vehicles are shown.

In the first simulation moment in Fig. 5, we can observe in the rule directed to vehicle 7 that the switch of RSU 5 composes the datapath for vehicle 7 to

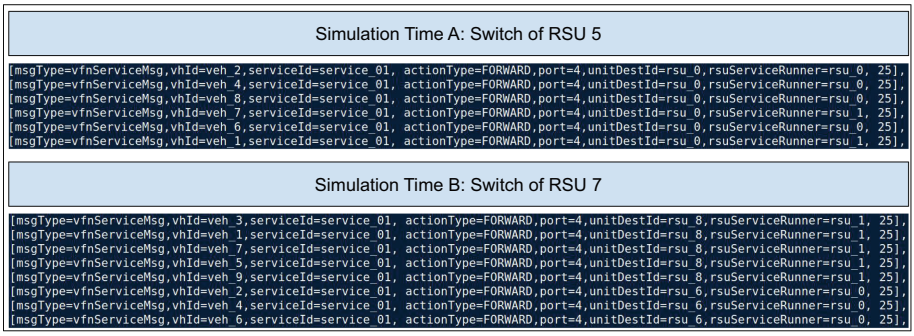

Fig. 5. Matching rules in different simulation times, when near vehicles uses the switches of RSU 5 and 7 as part of the path.

communicate with its processing fog node in RSU 1. To do so, the switch of RSU 5 forwards messages originating from vehicle 7 to the switch of RSU 0. It can be seen that at that moment the datapath goes through RSU 5 → RSU 0 → RSU 1, therefore, consistent with the topology in Fig. 3.

The next moment described in Fig. 5 occurs when the same vehicle 7 is now near RSU 7. You can see that the SDN controller has already made a change to the datapaths, as expected. Note that the data from vehicle 7 arrives at the same RSU 1, but now it passes through RSU 8, taking the path through RSU 7 → RSU 8 → RSU 1, also consistent with the topology shown in Fig. 3.

The proposed framework within Eclipse MOSAIC met expectations in creating datapaths that adapt to vehicle movements using the SDN paradigm, allowing investigations that take into account vehicle mobility to maintain optimized load balancing between fog nodes, reducing traffic on the infrastructure network, and minimizing communication latency.

3.3 Prediction of the Next RSU Access Point

In our case study, our aim is to have a platform capable of comparing handover prediction algorithms. We applied a simple prediction strategy based on the position of the RSUs and the vehicles, as well as the heading and speed of the vehicle. After the prediction, the SDN Controller is triggered by the Orchestrator to anticipate the sending of forwarding rules to the switches that are on the new datapath and were not on the previous one.

The Orchestrator performs the prediction of the next RSU when the vehicle is moving away from the current RSU Access Point. The simulation environment met the requirements, being capable of using a next RSU Access Point prediction algorithm which generates a decision-making process. Therefore, the platform is able to work with other algorithms by comparing their use in the same scenario. The anticipation of the forwarding rules also proved to be effective, especially by using OpenFlow rules based on the [vehicle,service] pair, as these rules are specific to each vehicle and do not affect other packet forwarding.

4 Conclusion

SDN networks have shown great potential in the implementation of ITS applications in IoV scenarios, especially in the implementation of SDVFN. The proposed SDVFN simulation framework makes possible the investigation of various strategies for selecting Fog Computing nodes, including load balance algorithms.

It also correctly implements the adaptability of datapaths through the SDN paradigm, even with vehicular mobility, and proves to be efficient in anticipating route sending, based on decision-making generated by algorithms that predict the next RSU Access Point for the vehicle.

In future work, we aim to improve the orchestration of Fog Computing nodes so that processing tasks can be migrated to other processing nodes. The aim is to test methodologies that not only optimize the allocation of processing tasks within the Fog Network, but also migrate processing tasks between nodes, with the aim of increasing efficiency in the consumption of services in the SDVFN, throughout the movement of vehicles.

The provision of a SDVFN simulation framework for SDVFN to the scientific community with the characteristics described in this work proves to be of great value, as it enables advances in scientific research in this promising paradigm.

In the near future, when more intended functionalities will be developed, the simulation framework can be used in more diverse IoE scenarios using emerging technologies and provide a broad range of metrics to evaluate the system's performance, including security features. In addition, the framework will be publicly available for free use by the scientific community.

Acknowledgements. This work has been supported by FCT - Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020.

References

1. Yuan, T., Da Rocha Neto, W., Rothenberg, C.E., Obraczka, K., Barakat, C., Turletti, T.: Machine learning for next-generation intelligent transportation systems: a survey. Trans. Emerg. Telecommun. Technol. **33**(4) (2022). https://doi.org/10.1002/ett.4427
2. da Costa, V.C.F., Oliveira, L., de Souza, J.: Internet of everything (IoE) taxonomies: a survey and a novel knowledge-based taxonomy. Sensors **21**(2), 1–35 (2021). https://doi.org/10.3390/s21020568
3. Langley, D.J., van Doorn, J., Ng, I.C., Stieglitz, S., Lazovik, A., Boonstra, A.: The internet of everything: smart things and their impact on business models. J. Bus. Res. **122**(2020), 853–863 (2021). https://doi.org/10.1016/j.jbusres.2019.12.035
4. Darwish, T.S.J., Abu Bakar, K.: Fog based intelligent transportation big data analytics in the internet of vehicles environment: motivations, architecture, challenges, and critical issues. IEEE Access **6**, 15679–15701 (2018). https://doi.org/10.1109/ACCESS.2018.2815989
5. Sharma, S., Kaushik, B.: A survey on internet of vehicles: applications, security issues & solutions. Veh. Commun. **20**, 100182 (2019). https://doi.org/10.1016/j.vehcom.2019.100182

6. Ji, B., et al.: Survey on the internet of vehicles: network architectures and applications. IEEE Commun. Stand. Mag. **4**(1), 34–41 (2020). https://doi.org/10.1109/MCOMSTD.001.1900053
7. Truong, N.B., Lee, G.M., Ghamri-Doudane, Y.: Software defined networking-based vehicular Adhoc network with Fog computing. In: 2015 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 1202–1207. IEEE (2015). https://doi.org/10.1109/INM.2015.7140467
8. Liu, K., Xu, X., Chen, M., Liu, B., Wu, L., Lee, V.C.S.: A hierarchical architecture for the future internet of vehicles. IEEE Commun. Mag. **57**(7), 41–47 (2019). https://doi.org/10.1109/MCOM.2019.1800772
9. Bhatia, J., Modi, Y., Tanwar, S., Bhavsar, M.: Software defined vehicular networks: a comprehensive review. Int. J. Commun. Syst. **32**(12), e4005 (2019). https://doi.org/10.1002/dac.4005
10. Ku, I., Lu, Y., Gerla, M., Gomes, R.L., Ongaro, F., Cerqueira, E.: Towards software-defined VANET: architecture and services. In: 2014 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET), pp. 103–110. IEEE (2014). https://doi.org/10.1109/MedHocNet.2014.6849111
11. Weber, J.S., Neves, M., Ferreto, T.: VANET simulators: an updated review. J. Braz. Comput. Soc. **27**(1), 8 (2021). https://doi.org/10.1186/s13173-021-00113-x
12. Alvarenga, L.D.C., Sousa, P., Costa, A.: Allocation and migration of microservices in SDN-based vehicular fog networks. In: 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–4. IEEE (2022). https://doi.org/10.23919/CISTI54924.2022.9820608
13. Schrab, K., et al.: Modeling an ITS management solution for mixed highway traffic with eclipse MOSAIC. IEEE Trans. Intell. Transp. Syst. **24**(6), 6575–6585 (2023). https://doi.org/10.1109/TITS.2022.3204174
14. Eclipse Mosaic Website: Eclipse mosaic: a multi-domain and multi-scale simulation framework for connected and automated mobility (2024). https://eclipse.dev/mosaic/
15. Lopez, P.A., et al.: Microscopic traffic simulation using SUMO. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 2575–2582. IEEE (2018). https://doi.org/10.1109/ITSC.2018.8569938