Model Driven Architecture for Decentralized Software Defined VANETs

Afza Kazmi[1(✉)], Muazzam A. Khan[1], Faisal Bashir[2], Nazar A. Saqib[1], Muhammad Alam[3], and Masoom Alam[4]

[1] NUST College of EME, National University of Sciences and Technology (NUST), Islamabad, Pakistan
afzakazmi@gmail.com, {muazzamak,nazar.abbas}@ce.ceme.edu.pk
[2] Department of Computer Science, Bahria University, Islamabad, Pakistan
faisalwn@yahoo.com
[3] Instituto de Telecomunicaciones, University of Aveiro, Águeda, Portugal
[4] Department of Computer Science, COMSATS Institute of Information Technology, Islamabad, Pakistan

Abstract. Vehicular Adhoc Networks (VANETs) are considered as a breakthrough technology in providing robust Vehicular communication that guarantees Road safety and offer interactive set of applications. VANET faces certain orchestration and management challenges. Recently, the notion of Centralization of VANET is gaining importance. In this regard SDN (Software defined Network) is integrated with VANET to achieve management goals. SDN renders existing VANET architecture to provide centralized control. Yet for Large-scale VANET, SDN approach does not outperform due to SPOF (single point of failure) in SDN. We have proposed decentralized architectural solution that scales out overall network intelligence into respective local controllers that result in unprecedented elasticity and resource availability. To make this approach operational no standard Architectural approach yet exists. An emerging Model Driven Architecture approach is used that simplifies VANET development using abstract models. It only subdues VANET rigidity but also comply with SDN principals. This system is implemented in VEINS framework. Results demonstrate its efficacy for large scale VANET.

Keywords: Vehicular adhoc network · Software defined network · Model driven architecture · VEINS · Highway

1 Introduction

In the recent era of technological uplift, vehicular networking is considered a promising and enabling technology that helps in realization of a diversification of vehicle related applications. Intelligent Transportation System (ITS) streamline such Networks and makes it viable for road traffic safety, emergency management, and infotainment [1]. Self-organization of vehicles in conventional VANET

J. Ferreira and M. Alam (Eds.): Future 5V 2016, LNICST 185, pp. 46–56, 2017.
DOI: 10.1007/978-3-319-51207-5_5

confronts several Management challenges such as dynamic network topology, high mobility and unbalanced vehicular traffic. Apart from VANET communication challenges, standard VANET design and architectural model is always overlooked. Lack of centralization of control in VANETs has always remained an intractable issue. Centralization of control enables the network manager to view the network in global context. The concept of centralized VANET is realized by emerging SDN technology [7]. SDN segregates network intelligence from infrastructure. SDN adds flexibility in network through programming, which allows the network stakeholders to adapt VANET according to their needs. Existing VANET architectures provide state-of- the-art approaches to build proficient VANETs, but when network size increases, it becomes difficult to manage the extensive data plane requests. One of such significant factor is Scalability in SD-VANET (Software defined VANET). The term textitScalability is used to describe vehicle density for large road networks [6]. Secondly, there is absence of standard VANET architecture. Both experts handle different level of abstractions that clearly signifies a communication gap between them. Ultimately, this gap causes incomplete and inconsistent requirement engineering. These concerns are better addressed in MDA approach in which the system is broken down into its abstraction levels [9]. The organization of the paper is summarized as; Sect. 2 discusses state-of-the-art VANET architectures and provide critical analysis. Section 3 include design requirements and operational mode of distributed control plane structure of DSDVANET (Decentralized software defined VANET) and building models based on these requirements, a comprehensive process of MDA development in DSDVANET are discussed. Section 4 provides the simulation environment where proposed model is deployed. Results are made by taking parameters from existing VANET models. Section 5 concludes the research by validating the existing model and its scope for future VANET systems.

2 Literature Work

Initial efforts reveal the advent of various wireless communication standards in VANET [2]. Different VANET protocol standards and architectures were developed in European countries. US DOT [10] are taken as initial standardization effort in ITS. A top-down Information centric architecture is proposed [11] which provides information enriched VANET applications, using the design concept of three key features space, time and users. An Integration of Cloud Computing and Information Centric Networking has been used that deploy mobile cloud model to VANET, and optimizes data routing and dissemination [12]. Similarly WiMAX and DSRC capabilities are combined in VANET to provide internet access to the vehicles in [13]. Cellular network technologies are also leveraging VANETs capabilities. LTE4V2X, offers centralized VANET schemes using LTE [3]. In [7] SDN is adapted for VANET environments. In this architecture OpenFlow enabled switch is contained inside vehicle. This architecture provides different operational modes which are based on degree of control of the Controller. In [14] SDN and Fog Computing are integrated that provides location

oriented and delay-sensitive services for next generation VANETs. A Vehicular Cloud [15] consists of traditional and specialized RSUs, using SDN it helps network in handling multiple data plane requests and minimizes control plane (CP) overhead. A novel IEA (information exchange architecture) [16] handles several data e.g. the road traffic flows, vehicle tracking, based on the data forwarding scheme. In [5] characteristics of the inter-contact time and duration of contacts for vehicular networks is determined for SDN based VANET. A centralized scheduler is installed at RSU by using SDN concepts. In MDSE paradigm a nominal research work has been done in developing WSN applications [17]. In [4] MDA principals are used to construct wireless sensor and actuator application, here MDA partitions network into respective abstraction layers, and each layer is developed using specific DSML (Domain specific modeling language). We evaluate these architectures on the basis of selective parameters to assess their performance using Network QoS as well as Software Quality Attributes. Combined WiMAX/DSRC VANET and SDN based VANET have inherent. From modeling perspective most of the Vanet system are built using networking models only two VANET architectures support software oriented models namely InVanet and combined WiMAX/DSRC. Combined WiMax/DSRC based VANET can perform model/code transformations. LTE4V2X and FSDN VANET provides services for large-evolvable VANETs. From this analysis we deduced that the aspect of software oriented models are less applied. SDN provides the opportunity to leverage MDSE (Model driven Software Engineering) in VANET context. It enables in constructing flexible and interpretable models for building Scalable VANET. A Large-scale VANET system comprises of multiple domains (cities or regions). A major bottleneck of such network is Scalability, where a single SDN controller has to manage multiple domains that results in substantial performance degradation [18]. We have introduced an approach to scale-out the intelligence of such network where each domain has its own local intelligence. Next section enlightens MDSE role in SDN and VANET domains.

3 Proposed Methodology

Recent research on distributed SDN is conducted in Data centers where network is scaled out to control billions of data plane requests from various sites across the globe [22]. Our proposed architecture not only believes on fully decentralized intelligence but carefully managing distributed controllers by top RC (Root Controller) that have a global network view and supremacy to perform centralized management. The proposed model is novel from different aspects.

- In existing SDVANETs the RSU are solely assigned to perform Forwarding tasks. In our proposed architecture RSUs acts as OpenFlow-enabled switches to implement forwarding rules enforced by the controller.
- MDA principles are combined with SDVANET that add abstraction and flexibility to the network.
- The CP (Control Plane) distribution is made in hierarchical fashion that delegates the roles from RC to Domain/Local controller(s) (Fig. 1).

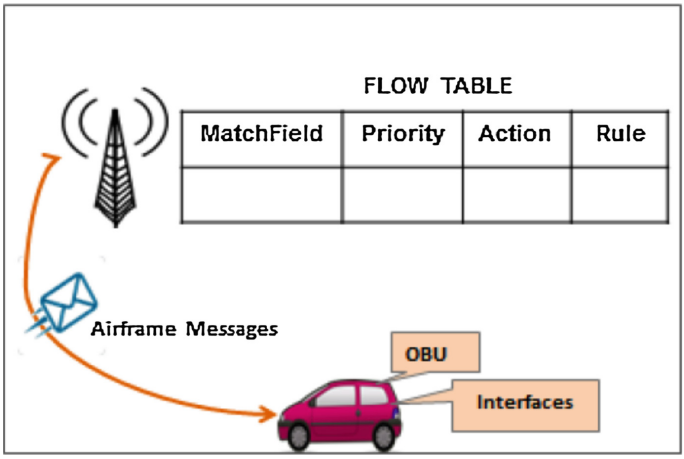

Fig. 1. Generic structure of forwarding plane.

3.1 Design Requirements

Due to distant positions of both Controller and RSUs there is high probability of delayed response. For safety related VANET applications time of response is the key constraint. Hence our design approach is based on Scalability requirement. We are going to exploit the middle layer that is the CP of the model. Our aim is to distribute the centralize intelligence into respective local territories termed as domains. We are examining our proposed approach on a Highway, connecting several regions (Domains). Each domain has its own centralized controller called DC (domain controller). The functional distribution is made in such a way that RC performs the most specialized functions while the generalized functions are handled by DCs.

DSDVANET Forwarding Plane Requirements: The core element of this layer is RSU that acts as an intermediary between moving vehicles and DC. RSU is connected via WAVE interface with the vehicles to provide fast communication in a single domain. It acts as Wireless OpenFlow-enabled switch (for Data Plane Communication) as well as Ethernet switches (for CP Communication). A primary Design requirement of Forwarding/Data plane is to enable Scalability that allows multiple distant nodes and RSUs to communicate with each other inside a domain. RSU maintains the Flow entries in the Flow Table (FT). The structure of OpenFlow enabled RSU is given below. Each flow entry of FT has some properties such as priority, MatchField, Action, Counters, timeouts etc. When a vehicle sends packet to the RSU it traverses the Flow Table to lookup for the Match of the respective flow entry [7]. If a match takes place the packet is processed otherwise the packet is send to CP that decides respective actions.

DSDVANET Control Plane Requirements: The CP of the Decentralized SDVANET is extensible as well as the crucial part to design. We partition this

plane, in order to achieve network wide orchestration goals. CP-Layer1 holds RC and the subsequent layer holds DC. From the concepts of distributed computing [8] the CP can be partitioned using two approaches:-

Vertical Partitioning: In this approach there exists a single physical machine (controller), which is distributed hierarchically [8]. The tasks of the CP are assigned to different controllers. The lower level controllers(DCs) perform frequent events' taking place in Data Layer while the upper layer handle global events.

Horizontal Partitioning: It is a physical distribution setting in which CP is divided into domains. Each domain has a DC which is responsible for its own territory. DC manages a disjoint set of *OpenFlow-Enabled RSUs* (Switches). DCs communicate with its adjacent to enforce global policies [19]. In our proposed model the distributed CP takes advantage of both distribution techniquesby relating their strengths. CP is partitioned into two layers, both hierarchically and physically. The top layer holds RC. The successive layer holds DCs, which supervises their own domain/territories. Each DC is connected to RC via dedicated high speed broadband connections. DC access RC, when unexpected events occur. e.g. when DC fails, network state changes, faulty links/flow tables discovered or DC receives out of domain information.

3.2 Operational Mode of Control Plane

Control Plane: Layer-1: It contains most specialized modules e.g. Flow Rules Manager and SDN Repository. DC acknowledges RC about its domain level activates by sending information such as link State, Vehicle/Nodes Availability, DC status etc. All this information is gathered for a scheduled time period inside SDN repository. Moreover it also keeps the network topology patterns as they kept on changing in VANET. In Forwarding Engine RC configures its submissive controllers (DC) based on Forwarding rules. These rules are developed from Domain level information. Path Computation Module provides a mathematical model to generate multi path topology based on efficient VANET routing schemes. Error Handling troubleshoots SDVANET issues and respond quickly by providing an optimal solution. The typical Errors handled by this module are Broken Links, Connectivity deadlocks etc. [21].

Control Plane: Layer-2: This is an outcome of horizontal scaling of CP. It contains physically distributed DCs. DC holds their respective territories and also connected with neighboring DCs to maintain consistent network topology. DC performs the similar tasks of centralized SDN Controller in SDVANET. The main functionalities of DC are; Domain Network Services and Domain Request Handling. *Domain Network Services*: These services are delegated to the DP elements to augment the network performance by reducing traditional VANET bottlenecks such as dynamic mobility, connectivity loss and Flow Management. DC serves the frequent events coming from Data plane. The main focus of DC is to provide high responsiveness and scalable deployment of network devices (RSUs).

Domain Request Handling: handles the requests coming from Data plane (DP). The *Dispatcher* module captures all the DP requests and processes them accordingly. It serves as the entry point to the CP. It gathers and forwards the DP requests to the *Configurator*. *Configurator* is the local agent which acts as a CPU of the DC. It is a visualization tool that displays the DC load statistics and provides a summary of Dispatcher-Configurator sessions. It is helpful in estimating Network performance at domain level. It timely collects the information from the configurator and identifies the condition of configurator overload (Fig. 2).

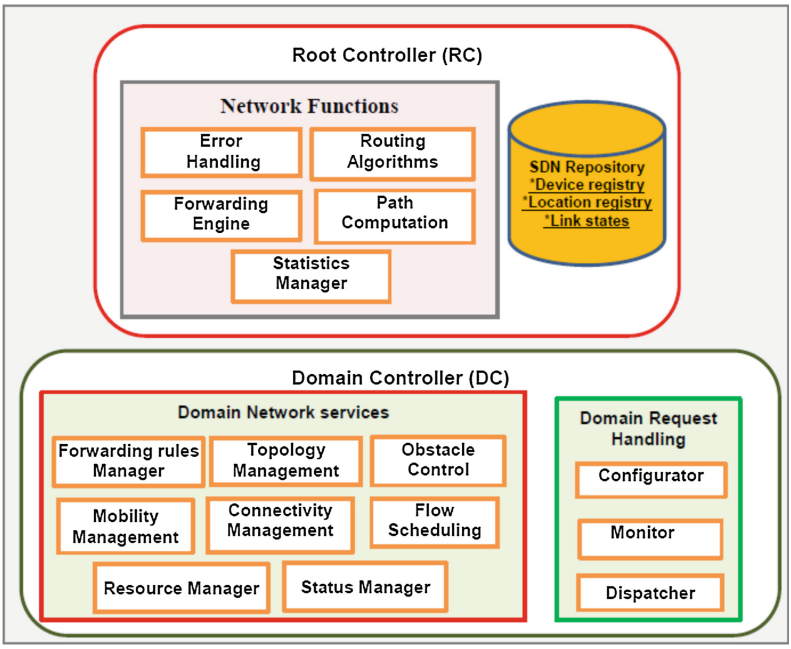

Fig. 2. Distributed control plane of Decentralized SDVANET

3.3 Mapping Network Requirements to Models

Preliminary activity of MDA is analyzing the requirements gathered from stakeholders. *Scalability* is Mission Statement of this proposed methodology. Next targeted users are identified. Domain experts are the application developers that develop application of SDVANET. *Network experts* perform administrative tasks e.g. topology configuration, protocol selection, and forwarding rules enforcement. Communication with CP is possible through an interface layer that holds protocols such as *OpenFlow* [23]. *PIM Meta Models:* is a visual representation of structural and behavioral details of Decentralized SDVANET modules. A generic DSML is used (UML) for describing internal and external system behaviors. Logical view of DSDVANET: Model diagram is used for modeling this SDN layered

system. The stereotypes inside each model (data, control, application) show the interactions and dependencies between the components. Application Plane model holds Applications package that handles generic application functionalities which are used in derived application packages in RC (as*SimpleServerApp*), and DCs (as *CtrlApps*), RSU (as *WAVEApp*)and vehicle (*TraCI*) applications (Fig. 3).

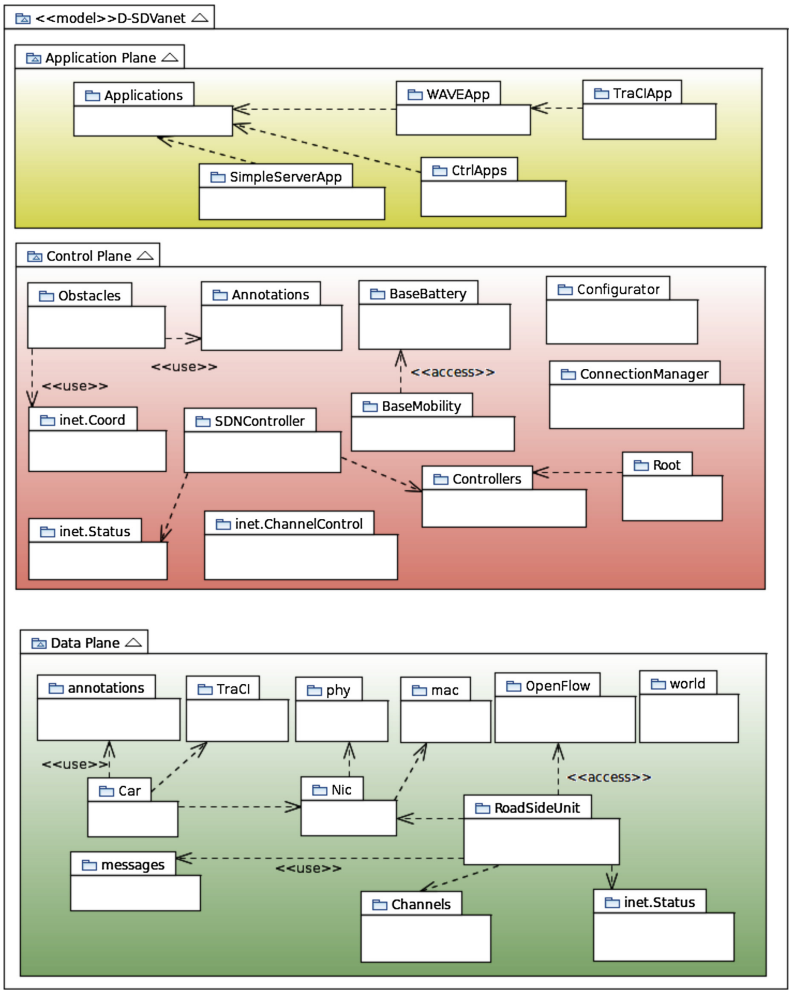

Fig. 3. Model diagram of Decentralized SDVANET

3.4 Defining Classes and Hierarchy

Due to extensive set of network modules, we have covered most significant artifacts. *Coord* class stores the dimensions of each object (vehicle, RSU or obstacle)

in an ITS Network. It plays a vital role in generating mobility patterns and computing distance in V2V and V2I for effective communication. *CtrlBehavior* is an interface which implements various controller modes of operations. Controller mode of operations are *Hub*, *Forwarding* and *Switch*. Hub class does not contain any intelligence, it instructs the RSU to flood all packets on all ports. *Switch* class learns the mapping of network elements to make easy re-configuration of RSUs if needed. *Forwarding* class contains prior knowledge of the whole network. *AnnotationManager* is composed of several geometrical classes (Line, Polygon) used to design virtual network for simulation. It manages the annotations on OMNET++ canvas. *Transformation Rules* ensure consistency between models at PIM with PSM. They are designed by considering the structural details of the system. In DSDVANET it is difficult to implement the transformation rules as we have to keep SDN policies consistent with MDA principals.

- *Level 1 (Network level)*: holds Network wide models which describes the overall data flow. These models are independent of Network. From PIM models of D-SDVANET such models are Car, RoadsideUnit, DC and RC.
- *Level 2 (Domain Level)*: are the domain VANETs. They depend on Network. It include SDN elements and Flow Processing models to enforce SDN rules.
- *Level 3 (Node Level)*: It is the lowest level and provides behavioral models of each vehicle. It holds the models that describe the contextual details of node. e.g. Channel Selection, Communication links, Protocol Selection and Interface Modeling. The transformation rule maps the similar elements of the Network and Domain-level models. It gives a fixed default value to the elements that newly appear in the Domain-level model. Same Step is repeated while transforming Domain-level to Node-Level. After developing PSM Models, we export these models to VEINS simulator [22], combined with OpenFlow Extension [23], to provide SDN facility in VANET.

4 Results and Discussion

After developing PIM Models, we exported the models to respective PSM tool (i.e. VEINS). In our scenario two cities are connected Peshawar and Islamabad. The network topology is partitioned into domains. Each domain holds a ping application that randomly send echo message to any of the 9 domains. On receiving, echo reply message calculates mean RTT. The total simulation time for this experiment is 125 s. We have utilized the following metrics to analyze the system performance.

Mean RTT (Round Trip Time): When a domain sends an echo request message to its linked domain and receive acknowledgment in form of echo reply message the time elapsed is taken as RTT. The simulation duration of each run is 30 s. We calculate RTT for 10 runs per domain and convert it into seconds. Mean RTT is described with respect to RC location indices. It can be seen that controller placement at some equidistant location like 4 and 5 provides best results in the form of least mean RTT for each domain of the network topology. However

moving towards the edge locations give worst results, that is high RTT. Green and Blue bars shows steady rise in RTT if RC is placed on the other extreme end of the network topology however red bars depict a stability of RTT among all domains (Fig. 4).

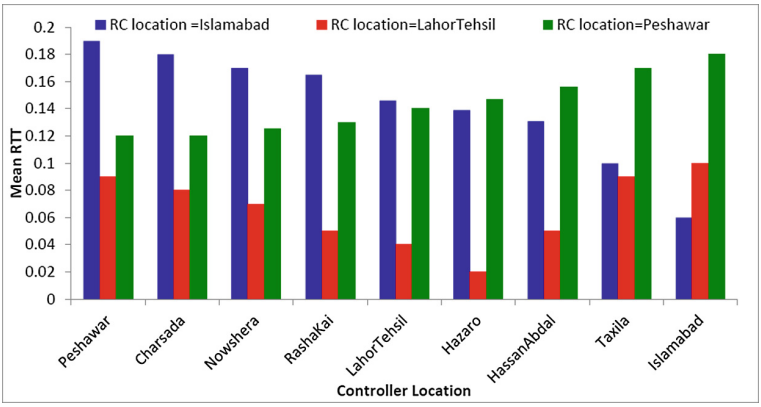

Fig. 4. Mean RTT Vs. RC location. (Color figure online)

Gain of Scalability (GoS): For a simulation time interval T, the number of Vehicles served either via V2I/V2V communication is termed as GoS [4]. We have defined 5 different traffic scenarios in Table 1 for 3 lanes (L1, L2 and L3) of Highway road. The sender node receives 64 RTTs per domain as each sender node waits for 2 s to get a reply and after that sends echo to another domain. From Fig. 5, in Traffic Scenario 1 node density is high as compared to the other roads. Certainly a reduction in vehicle speeds is observed. However in SDVANET and DSDVANET cases we observe a slight drop, due to handover between RSU and Controller. In 2nd and 3rd Scenario VANET faces a descent due to change in vehicle speed and road length. However SDVANET and DSDVANET maintained their levels as Topology management in controller keeps the network topology updated.

Table 1. Different traffic scenarios for M1 motorway

Sr	Distance source↔ Destination (km)	Mean vehicle speed (km/h)			Mean vehicle arrival rate (vehicle/h)			Mean vehicle density (vehicles/km)
		Lane1	Lane2	Lane3	Lane1	Lane2	Lane3	
1	Peshawar↔Charsada(32)	104	95	36	1163	38	17	24.3
2	Hazro↔Taxila(45)	95.5	96	50	554	42	29	20.6
3	Taxila↔Islamabad(40)	86.8	98	52	898	46	25	21.5
4	Charsada↔RashaKai(31)	70.59	100	62	740	24	20	16.3
5	LahorTehsil↔Hazro(63)	65	100	38	537	35	26	19.7

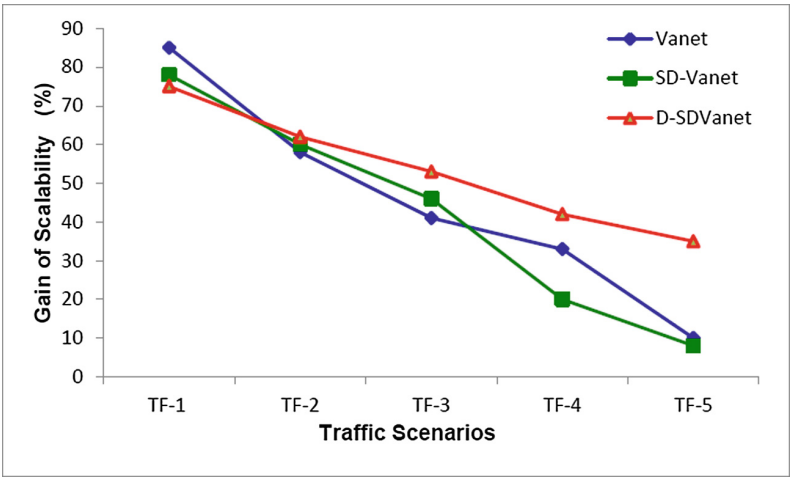

Fig. 5. GoS vs Traffic scenarios

5 Conclusion and Future Work

In this research concept of DSDVANET is introduced that reduces scalability concerns. Practicing MDA approach in VANET domain is found to be useful as it enable flexible and consistent model development. MDA and SDN both exhibit several similar features that help in generating consistent architectural models. MDA extensibility feature helps is building standard VANET architecture. This scheme is designed for large VANET systems, for limited VANET SDVANET is suitable. It is validated in real-world Motorway scenario. Results has been shown that the controller perform effective role in handling DP requests and RC resolves the issue of VANET failure. In future we will extend application plane of DSDVANET to deploy non-safety VANET applications.

References

1. Alam, M., Ferreira, J., Fonseca, J.: Introduction to intelligent transportation systems. In: Alam, M., Ferreira, J., Fonseca, J. (eds.) Intelligent Transportation Systems. SSDC, vol. 52, pp. 1–17. Springer, Heidelberg (2016). doi:10.1007/978-3-319-28183-4_1
2. Campolo, C., Molinaro, A., Scopigno, R.: From today VANETs to tomorrow planning and the bets for the day after. Veh. Commun. **2**, 158–171 (2015)
3. Remy, G., Senouci, S.-M., Jan, F., Gourhant, Y.: LTE4V2X: LTE for a centralized VANET organization. In: IEEE GLOBECOM (2011)
4. Rodrigues, T., Batista, T., Delicato, F.C., Pires, P.F., Zomaya, A.Y.: Model-driven approach for building efficient wireless sensor and actuator network applications. In: SESENA, USA (2013)
5. Xiao, X., Kui, X.: The characterizes of communication contacts between vehicles and intersections for software-defined vehicular networks. Mob. Netw. Appl. **20**(1), 98–104 (2015)

6. Kazmi, A., Khattak, M.A., Akram, M.U.: DeVANET: decentralized software-defined VANET architecture. In: 3rd IEEE Symposium (SDS-2016) (2016)
7. Ian, K., You, L., Gerla, M., Gomes, R.L., Ongaro, F., Cerqueira, E.: Towards software-defined VANET: architecture and services. In: 13th IEEE Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET) (2014)
8. Bhowmik, S., Tariq, M.A., Koldehofe, B., Kutzleb, A., Rothermel, K.: Distributed control plane for software-defined networks. In: 9th ACM Conference on Distributed Event-Based Systems (2011)
9. Araniti, G., Campolo, C., Condoluci, M., Iera, A., Molinaro, A.: LTE for vehicular networking: a survey. IEEE Commun. Mag. **51**, 148–157 (2013)
10. TERIS Website, Official web site of the (National ITS Architecture)
11. Bai, F., Krishnamachari, B.: Exploiting the wisdom of the crowd: localized, distributed information-centric VANETs. IEEE Commun. Mag. **48**, 138–146 (2010)
12. Lee, E., Lee, E.-K., Gerla, M., Oh, S.Y.: Vehicular cloud networking: architecture and design principles. IEEE Commun. Mag. **52**, 148–155 (2011)
13. Alam, M., Albano, M., Radwan, A., Rodriguez, J.: CANDi: context-aware node discovery for short-range cooperation. Trans. Emerg. Tel. Tech. **26**(5), 861–875 (2013)
14. Truong, N.B., Lee, G.M., Ghamri-Doudane, Y.: Software defined networking-based vehicular adhoc network with Fog computing. In: IFIP/IEEE International Symposium on Integrated Network Management (IM) (2015)
15. Salahuddin, M.A., Al-Fuqaha, A., Guizani, M.: Software-defined networking for RSU clouds in support of the internet of vehicles. IEEE Internet Things J. **2**(2), 133–144 (2015)
16. Ryoo, I., Na, W., Kim, S.: Information exchange architecture based on software defined networking for cooperative intelligent transportation systems. J. Cluster Comput. **18**(2), 771–782 (2015)
17. Rodrigues, T., Dantas, P., Delicato, F.C., Pires, P.F., Pirmez, L., Batista, T., Miceli, C., Zomaya, A.: Model-driven development of wireless sensor network applications. In: 9th IEEE/IFIP Conference on Embedded and Ubiquitous Computing (2011)
18. van Asten, B.J., van Adrichem, N.L.M., Kuipers, F.A.: Scalability and resilience of software-defined networking: an overview (2014)
19. Schmid, S., Suomela, J.: Exploiting locality in distributed SDN control. In: HotSDN 2013, 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking
20. Yeganeh, S.H., Ganjali, Y.: Kandoo: a framework for efficient and scalable offloading of control applications. In: (HotSDN), pp. 19–24 (2012)
21. Phemius, K., Bouet, M., Leguay J.: DISCO: distributed multi-domain SDN controllers. In: Network Operations and Management Symposium (NOMS) (2014)
22. VEINS. http://veins.car2x.org/documentation/
23. Klein, D., Jarschel, M., An OpenFlow extension for the OMNeT++ INET framework. In: Asia-Pacific Conference on Computer Aided System (2014)