IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 15, Issue 2 (Nov. - Dec. 2013), PP 01-16
www.iosrjournals.org

A Synopsis of Simulation and Mobility Modeling in Vehicular Ad-hoc Networks (VANETs)

Sabih ur Rehman, M. Arif Khan, Tanveer A. Zia and Rashid H. Khokhar
School of Computing and Mathematics, Charles Sturt University, Australia

Abstract : *Vehicular communication is considered to be a backbone for many critical safety applications. In order to achieve a better implementation of any vehicular communication scenario, an efficient, accurate and reliable simulator is essential. Various open source and commercial simulating tools are available for this purpose. One of the key issues in this regard is the selection of a reliable simulator which implements all standard algorithms and paradigms giving accurate results. In this paper, we first present IEEE standard and protocols for vehicular communication, IEEE 802.11p and IEEE 1609.x, also known as WAVE protocol stack. The paper then discusses the necessary requirements for a generic discrete event simulator which can be used to simulate Vehicular Ad-hoc Networks. Since not all the network simulators can be used in the scenario of vehicular communication, we highlight the key features of some network simulators in the context of vehicular ad-hoc networks. The paper also highlights some of the implementation limitations in these simulators. Furthermore, the paper presents a discussion on traffic simulators by emphasizing on the underlying mobility models used in order to generate the realistic traffic patterns. A comparative study of both network and traffic simulators show the pros and cons of these simulation tools. The paper suggests the appropriate choice of a network simulator to be used as a VANET simulator.*

Keywords: *VANET, IEEE 802.11p, WAVE-PHY, WAVE-MAC, Simulators, Modeling*

I. INTRODUCTION

According to World Health Organization (WHO), 1.3 million people lose their lives and 50 million get injured in road traffic accidents each year [1]. In March 2010, United Nations (UN) imparted on a mission to reduce this high number of casualties through road traffic accidents by unanimously voting for a global plan for *Decade of Actions for Road Safety 2011-2020* [2]. The key consideration is to drive *innovative solution* in tackling areas like road safety management, road infrastructure, vehicle safety and the post-crash responses. This has led to a demand for more research in the field of road traffic and safety management and significant effort has been devoted in experimenting with various technological aspects of Vehicular Ad-hoc Networks (VANETs). *Intelligent Transport Systems (ITS)* which is a global organization working for road traffic engineering has also been playing a key role [3] in this by using state of the art vehicular networking technologies. ITS focuses on establishing the deployment of advanced technologies across the traffic network. Since the early introduction of VANET concept, it has been a major part of ITS infrastructure. The quick advancement in wireless technologies has provided opportunities to exploit vehicular communication for various applications.

Vehicular Ad-hoc Networks (VANETs) is a special type of wireless communication network that gets established among nodes, where all nodes are vehicles, generally moving with a high speed. It is considered as a decentralised, infrastructure-less and self-organising network differing from other wireless combination networks such as Mobile Ad-hoc Network (MANET), Wi-Fi, Wi-Max etc. In VANET there is a wireless communication between Vehicle-to-Vehicle (V2V) and Vehicle-to-infrastructure (V2I) where an infrastructure is commonly referred as Road Side Units (RSU) similar to base station in mobile wireless communication. In VANET scenario, wireless communication among vehicles is a complex phenomenon because of several reasons. Some of the main reasons are that environment in which this communication takes place has very dynamic nature which includes several obstacle, frequency disconnections, high mobility of vehicles and interference from many other in-car and out of car wireless applications. In general, VANET architecture can be classified into three broad categories. In first category, vehicles communicate with each other through RSU is referred as WLAN architecture of VANET. In second category, vehicles communicate with each other directly without the assistance of any RSU, referred as Ad-hoc architecture of VANET. In third category, vehicles communicate with each other either through an RSU or directly depending on the communication scenario, referred as hybrid architecture of VANET [4].

Since last decade, VANET technology has gained a lot of attention from the research community and a number of critical life saving and comfort related applications have been designed using this technology [5]. Among many interesting applications of VANET critical safety applications are the most important ones. In an emergency situation either in urban areas or on highways, a vehicle should be able to promptly communicate

with other vehicles or any other infrastructure to disseminate the incident information. A timely dissemination of the information to emergency services may save lives. In other safety related applications, a driver may get alerts about city safety, blind spot warning, pre-crash warning, and highway collision warning and intersection collision avoidance. There are a number of vehicle manufacturing giants who are providing these applications in their commercial products [6]. This highlights the importance of investing in VANET related technologies for the betterment of society.

Along with all the benefits, physical implementation of VANET brings lots of challenges such as cost, advancement in technology and testing the application to provide error free communication. Therefore, it is important that before the physical implementation of any VANET scenario, a complete computer simulation or emulation must take place to get the desired results. One of the main objectives of these simulations is to generate a realistic scenario of VANET communication. For this purpose, a number of commercial and open source simulation tools have been developed and are available to the academic research community and commercial organisations. In the context of VANET, the simulation tools are called *VANET simulators*. An ample amount of research on the performance of different VANET simulators is still going on [7, 8, 9]. A good VANET simulator is one with the characteristics that can simulate a scenario as close to the realistic values as possible. However, not all current VANET simulators can perform this task.

In order to have consistent implementation of different VANET scenarios, Institute of Electrical and Electronic Engineers (IEEE) has formed a working group to establish the standards for vehicular communication [10]. As a result, the working group has given a combination of IEEE802.11p and IEEE1609.x standards to cater most of the implementation requirements of VANET. One of the main objectives of this paper is to highlight and discuss the IEEE WAVE architecture for vehicular communication. The emphasis in this article is on the fact that many implementations of VANET scenario do not properly use this standard. For realistic simulation results it is critical to properly follow these standards and architectures in the respective layers of VANET technology.

This article discusses a number of VANET simulation tools used in the research community. An overview of the performance of network simulators alone as well as when integrated with the traffic simulator is one of the topics of this paper. The main purpose of discussing these tools is to give the reader an insight to help him/her to choose a particular simulator for the communication scenario. The paper also presents the architecture and models used in the network simulator. A generic design requirement for an efficient VANET simulator is also presented. It is established that a VANET simulator gets various input parameters including the important mobility modeling patterns from the traffic simulator. Therefore it is critical that a proper mobility modeling algorithm must be used in the traffic simulator to get the realistic results. In addition to this, the network simulator needs to have implementation of IEEE WAVE protocol at the physical layer of network model so that the transceiver parameters can be obtained in the realistic way to include in the simulation. All the network and traffic simulators discussed in this article are in the context of VANET scenarios. Some of the main limiting factors of these simulators to generate the realistic result in VANET environment are also presented.

The rest of the paper is organized as follows. Section II presents an insight into the IEEE WAVE protocol for physical and medium access control layers of VANET. Section III discusses the requirements of general network simulation architecture in relation to VANET. This section also presents the architecture, advantages and limitations of these simulation tools. In Section IV, we discuss some well known simulation tools for vehicular communication and we investigate the advantages and limitation of both network and traffic simulators. Finally, Section V concludes the paper.

II. IEEE PROTOCOL FOR VANET

One of the key factors that affect the performance of any VANET simulator is the implementation of physical layer parameters. This layer provides the information on how the actual communication takes place. Therefore, it is important to properly use the standard architecture of wireless communication. This section provides readers the introductory information about standard physical layer implementation for VANET. The main purpose of this section which relates it with other sections is that any VANET scenario implementation with any network simulator has to be implemented with these protocols. For this purpose, IEEE has introduced a proper standard for VANET physical layer implementations which will be discussed in the later parts of this section.

IEEE 802.11 is a family of standards for Wireless Local Area Networks (WLAN) [11]. The main purpose of 802.11 family is to specify the implementation of physical (PHY) and medium Access Control (MAC) layers. Some of the main design parameters given in 802.11 such as operating frequency, typical data rate, range of communication, modulation techniques, link establishment, quality of service and data frame structure are a few to name. Different variants of 802.11 such as, 802.11, 802.11a, 802.11b, 802.11g, 802.11n and 802.11p were introduced to cater the needs of different wireless communication topologies.

In a wireless communication scenario where short to medium range communication is required, Federal Communication Commission (FCC) in 1999, developed a Dedicated Short Range Communications (DSRC) protocol [12]. This protocol later became the basis for vehicular communication standard 802.11p. DSRC is a key technology for next generation Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) wireless communication. The main structure of DSRC resembles with Wi-Fi architecture. After one decade of initial launch of DSRC and its evolution, the DSRC community decided to merge DSRC with IEEE 802.11 standard. As a result, a new standard emerged for vehicular ad-hoc networks which are a combination of DSRC and IEEE 1609.x standards [13, 14]. In Australia, DSRC has shown its presence by establishing industry driven consortium called Aus-DSRC [15]. Currently, Aus-DSRC is working in collaboration with local automotive industry giants to look into providing a real test-bed for wireless vehicular communication.

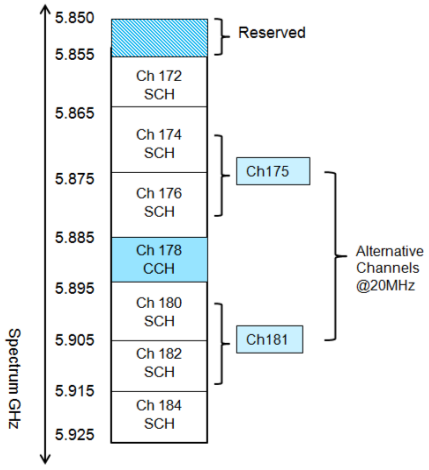

Figure 1: A DSRC Spectrum Format

IEEE 802.11p is based on Wireless Access for Vehicular Environments (WAVE) and multiple variants of IEEE 1609 family. WAVE is a modified version of original DSRC with less network overheads [16] and is responsible for communication between vehicles and road side unit (RSU) or among vehicles. It is important to understand from the implementation point of view that WAVE works at both PHY and MAC layers therefore its implementation on both layers needs to be considered. One can think of these architectures as WAVE-PHY and WAVE-MAC protocol.

WAVE-PHY consists of multiple communication channels which use frequency range of 5.850 GHz to 5.925 GHz of wireless spectrum. The modulation technique used for WAVE-PHY transceivers is Orthogonal Frequency Division Multiplexing (OFDM) with a band of 10 MHz allocated to every channel. In WAVE-PHY, Physical Medium Dependent (PMD) sub-layer is responsible for the communication with transceivers while the Physical Layer Convergence Procedure (PLCP) sub-layer is responsible for the interaction between PHY and MAC layers.

WAVE-MAC, on the other hand, is mainly responsible for the MAC layer operations that define the policies for resource allocation among competing nodes for the wireless medium. In order to avoid the congestion, nodes use IEEE 802.11 standard for collision avoidance which is Carrier Sense Multiple access/collision avoidance (CSMA/CA). In general, WAVE-MAC architecture exchanges information through multi-channel operations defined in IEEE 1609.4. The DSRC 75 MHz band is further subdivided into a single Control Channel (CCH) and six Service Channels (SCH) each having bandwidth of 10 MHz. The format of a DSRC spectrum is shown in Figure (1). The remaining 5 MHz is reserved for the guard interval in DSRC frame. The main purpose of the CCH is to carry high priority Short Message also known as Wave Short Message (WSM) or Management Data, while all other data is carried on the SCHs. These short messages in CCH are primarily used for the safety application and are defined as the Wave Short Messages (WSM) and their architecture is implemented with the help of Wave Short Message Protocol (WSMP) given in IEEE 1609.3. On the other hand, WAVE Service Advertisements (WSA) messages are used to announce the services offered in the area, where a *service* can be any information of interest such as restaurant, service station, hospital location etc. to the occupants of the vehicle.

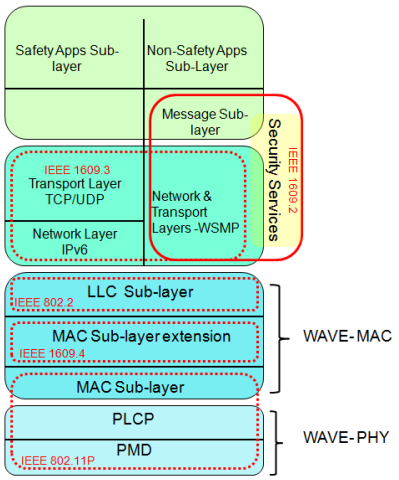

Figure 2: WAVE Protocol Stack for VANET

As far as the information from the layers above the MAC layer is concerned, it is dependent on the application of interest. The use of WSMP or typical IPv6 (TCP/UDP) depends on the data traffic applications. In the applications where limited bandwidth is a constraint, WSMP is a better choice for the efficient use of the limited resource whereas in the multi-hop routing scenarios, IPv6 is a better choice. A filtering mechanism must be provided between the MAC and upper layers which should cater that either WSMP or IPv6 Based packets need to be transferred from upper layer. In order to provide proper security mechanisms for these packets IEEE 1609.2 standard provides a detailed security architecture. However, the discussion of IEEE1609.2 is beyond the scope of this paper. A detailed layered architecture of WAVE protocol stack is given in Figure (2). In order to implement any application scenario in VANET where WAVE protocol is required, it is essential to understand the layered architecture of WAVE protocol stack for the proper implementation. In most of the network simulator the proper implementation of WAVE protocol is overlooked. This results in a non-realistic modeling of vehicular ad-hoc networks. In this paper, we have discussed the detailed implementation of WAVE protocol stack in a particular network simulator environment in order to get more realistic results.

III. NETWORK SIMULATIONS

A communication scenario where multiple devices exchange data with each other is known as a network. The fundamental building blocks of a communication network are data exchanging nodes which can be computers, mobile units or vehicles and the communication path between these devices known as communication channel. Depending on the nature of a communication channel, a network can be classified as fixed or mobile. Before the physical implementation of a communication network, the feasibility of implementation is verified using computer simulation tools known as network simulators.

A network simulator can be classified into two major types. (i) Continuous time Simulators (ii) Discrete Event Simulators. A continuous time simulator tracks the network dynamics continuously over a given period of time whereas discrete event simulators are event driven simulators which track network properties triggered by a particular event. Most of the network simulators these days are based on the discrete event approach which is more efficient in terms of resource usage compared to the continuous time. Discussion on discrete event simulators and their performance are the topic of this paper.

A generic network simulator creates a network topology consisting of nodes (either fixed or mobile) as well as the channels among the nodes. It also creates the data traffic (data packets) depending on the type of applications which are used to send/receive among transceivers. Various transmission mechanisms and protocols are implemented to achieve the desired performance metrics. Simulation results are analyzed with the help of proper visual and graphical tools. However, it must be noted that it may not be possible for a network simulator to simulate all the networks because of the different design parameters and the required results may vary from network to network.

A typical example of a network simulator can be a simulator which performs simulations for the vehicular ad-hoc networks. Figure (3) highlights the main architecture of such a simulator. In a very broad sense, a VANET simulator has three main components including input parameters, controlling sequences/algorithms and the desired output parameters. The input parameters may include network topology, data traffic models, mobility models and parameters from the physical layer of the network model. The controlling sequences may include the transmission algorithms, routing protocols, security mechanism and Quality of service requirements. The typical output requirements for a VANET simulator are end-to-end transmission delay, routing overheads and Packet Delivery Ratio. Provided all requirements in the simulator are

met and proper implementation of all standards and protocols (as defined in Section II, the network simulator should give realistic results. These results help in generating the proper feasibility for physical implementation of the vehicular network.

IV. SIMULATORS FOR VANET

A Vehicular Ad-hoc Network is a temporary network established on specific need basis among the moving vehicles. Due to the complex topology and a rigid environment, it is hard to implement real physical test-beds for VANET. In order to minimise the probability of failure in physical implementation, a network simulation close to the realistic scenario must be run prior to the physical implementation. To achieve this, a flexible, credible and accurate network simulator is required. Various commercial and non-commercial simulators are available to simulate a VANET scenario [17 - 24]. A typical VANET simulator is one that not only deals with demands of wireless communication among vehicles(such as data traffic generation and routing scheme) but also looks into other issues like node mobility and radio design parameters as specified in the WAVE standard. A VANET simulator can be subdivided in two following categories:

- Network Simulator
- Traffic Simulator

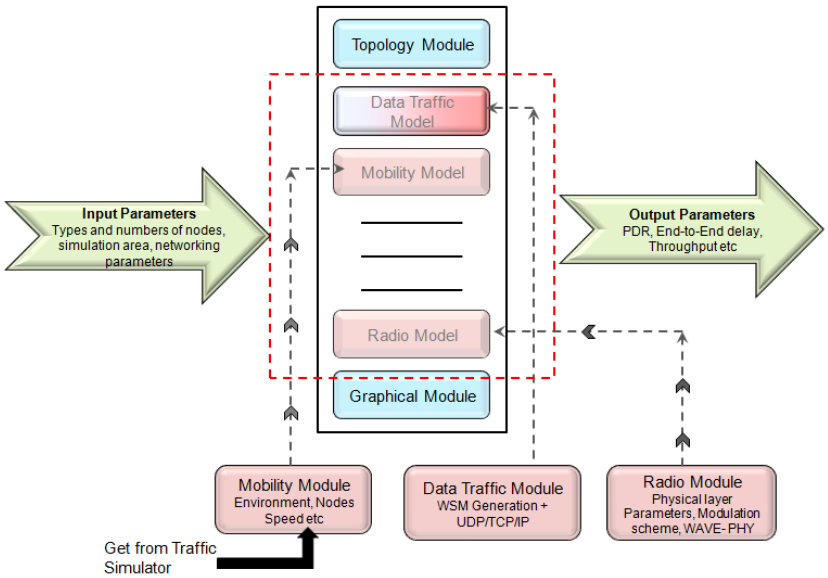

Figure 3: A Typical Design of a Network Simulator for VANET Environment

In the following subsections we discuss some well known network and traffic simulators in detail.

A. Network Simulators

A network simulator typically generates data traffic, gets traffic models from the traffic simulator, implements a proper routing scheme and transmits the data over the properly designed radio channel. Depending on the application of interest, data packets based on IPv6 or WSM need to be generated in the network simulator. After getting the proper mobility model/pattern from the traffic simulator, the network simulator implements the proper routing strategy for transmission of data packets. It is the responsibility of network simulator to make sure that data packets are properly sent and received over the radio channel. Therefore, it is crucial to choose a proper network simulator for the simulation of VANET environment. In this section, we discuss some well known network simulators focusing on the purpose that it will help reader to chose a proper simulator for his/her VANET scenario of interest. There are various open source and commercial network simulators available. Most of these simulators possess a unique feature of being discrete in event scheduling and this makes them highly reliable for rigid topological structure of vehicular ad-hoc networks. In following subsections, we provide an overview of few well known open source as well as commercial network simulators such as *ns-3* [25], *OMNET++* [21], *OPNET Modeler* [24], *GloMoSim* [19] and *JiST/SWANS* [20]. We also present an evaluation of these simulators in context of vehicular ad-hoc communication.

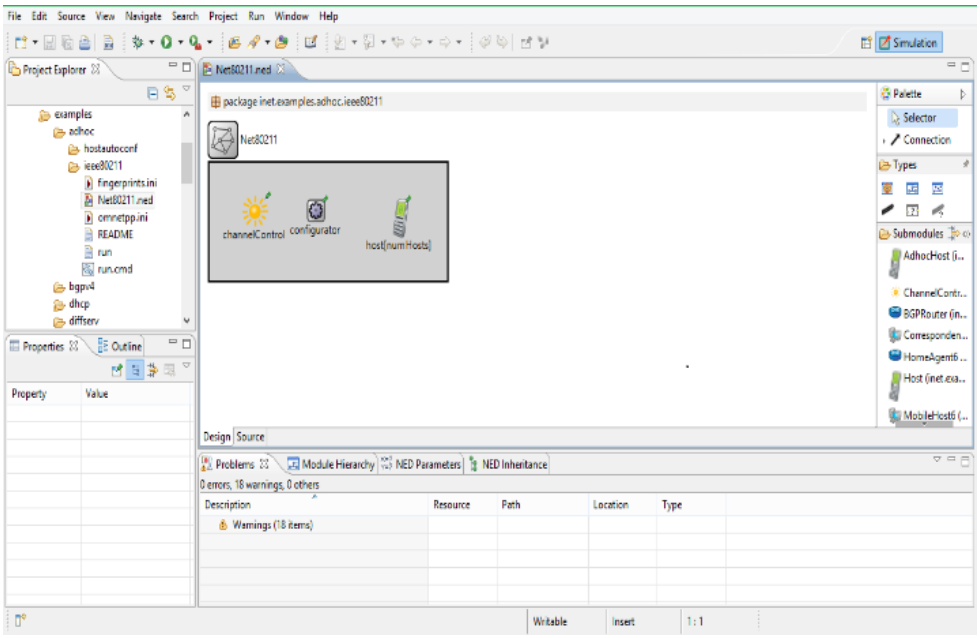

Figure 4: OMNeT++ NED Design View

1) Network Simulator-3 (ns-3)

For many years, ns-2 [18] has been a standard simulation tool for computer networks. It is widely accepted as a trusted simulator for network designing in the research community as well as for commercial applications. ns-2 is an open source network simulator and it is based on C++ programming language in order to better facilitate the inclusion of C-based routines from the academic, research and commercial code developers. One of the important features of ns-2 is its open source code availability due to which many researchers have extended ns-2 to support a number of communication scenarios which include modules such as covering mobility and wireless networking libraries.

As in many other network simulators, ns-2 has the capability to define and model most network elements in wired and wireless communication. In particular, one can define network nodes, wired and wireless communication channels, physical devices and routing protocols. In order to create a simulation scenario, we can define a network topology, create data on demand, execute the simulations and analyze the results effectively and efficiently in ns-2. IEEE 802.11 defines the PHY and MAC layers protocols and standards for most wireless communication systems. One of the short comings of ns-2 is that it does not fully implement 802.11 standard for many practical wireless communication scenarios. It also lacks in implementing the dynamic nature of many ad-hoc networking environments where realistic modeling is required. In order to overcome these limitations and creating more realism in the design of models, an improved version of ns-2 network simulator was introduced recently, namely ns-3 [25]. One of the major improvements in ns-3 from its predecessor is that it can implement most of IEEE 802.11 standards and can model the network scenarios closer to the real world.

Similar to ns-2, ns-3 is also based on C++ and OTcl language in order to be compatible with all the libraries and modules used in ns-2. This increases the overall implementation efficiency of the network simulator. OTcl is an object-oriented extension of the interpreted language Tcl (Tool Control Language) [26] which gives an easy to understand and dynamic simulation configuration environment to ns-3. OTcl is primarily used for control purpose whereas C++ programming structure is utilised for data manipulation in the simulations. ns-3 simulator has the capability to model network nodes, network physical devices, communication channels, communication protocols structured data with the information of protocol headers and network packets under the IEEE 802.11 standards and its all extensions. In writing any ns-3 code for a particular communication scenario, there are four basic steps to perform: creating the network topology, generating the network data, executing the simulation and analysing results.

2) OMNeT++

OMNeT [21] is classified as a modular discrete event network simulator. It is an open source software that provides a platform for modeling wired and wireless communication networks. One of the key features of OMNeT++ is that it gives the ability to combine small building blocks of a network by using the advantage of its modular structure. These blocks are referred to as *modules* and can be reused for the re-assembly of other

networks. Modules are connected by *gates* which resembles communication links between layers of communication stack. Two or more modules connected together create a bigger structure called *compound-modules*. Modules at the lowest level of hierarchy are called *simple modules* which are created using C++ via key components of simulation library provided within OMNeT++. Connections between modules are created using gates, which can be classified as *Channels*. Different link properties such as propagation delay, link capacity, data rate and bit error Rate are specified for these channels.

The network structure of modules is illustrated using a special language, known as *NEtwork Description (NED)* language. Simulation executions are easily configured via initialization files (*.ini files) that are configured prior to the simulation run. OMNeT++ interface provides graphical as well as animated structure to demonstrate real-time simulation environment. OMNeT++ also provides support for parallel distribution of simulations that helps to simulate two models concurrently. The latest versions of OMNeT++ are tested on various known operating platforms like Windows, Linux and Macintosh. In designing of computer networks, OMNeT++ uses a framework approach for specific application areas. Initially a framework, *Mobility Framework* [27] was proposed for implementation of mobility and connectivity management within ad-hoc networking. This framework was later extended and refined to include all of the resources for wired and wireless networking under well established frameworks such as *INET* [28] and *MiXiM* [29]. *MiXiM* primarily deals with design of lower layer protocol architecture and one of the key features of MiXiM is that it has a very detailed/realistic implementation of radio wave propagation for wireless environment.

Figure (4) shows a graphical user interface of OMNeT++ simulator NED design tab view. On the right hand side, a design palette with different sub-modules which can be used as network components is shown. These sub-modules can be dragged into the workspace. On left hand side, there is a project explorer which shows current working directory including all files. A particular file can be selected and opened from this palette. Figure (5) shows the source file view of OMNeT++ initialization file. The parameter being used such as data rate, packet size, routing protocol etc, in the simulation are also defined in this file.

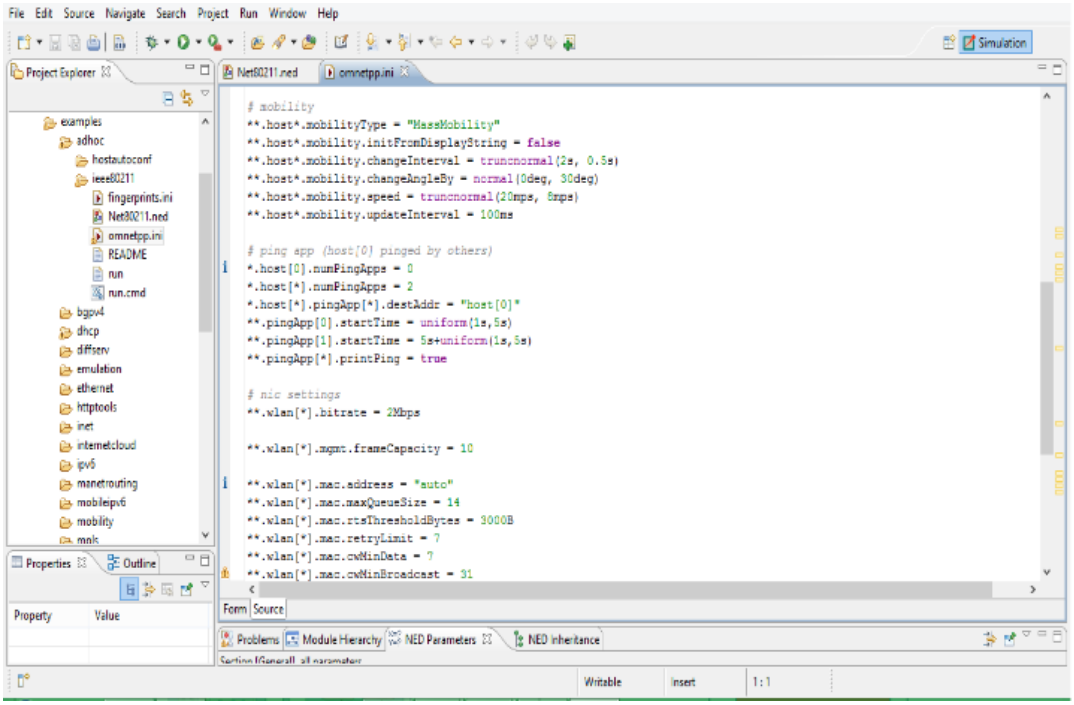

Figure 5: OMNeT++ Initialization (*.ini) Source File View

3) OPNET Modeler

OPtimised Network (OPNET) Modeler [24] is a well known commercial discrete event simulator which can be used for wireless communication networks. It can also be used as free license for academic purpose and can be obtained by researchers by applying to University Program of OPNET. OPNET is based on C++ programming language while it has different networking modules which can be used to create a network. OPNET Modeler is particularly used for designing the simulation and test-beds for different wireless communication networks. It defines a network model as a combination of a different sub models consisting of sub network for nodes by employing a hierarchical modeling. Different simulation topologies can be created by the users or can be imported and selected from the collection of different predefined libraries. A large number of

standard protocol models are available in OPNET Modeler and users can also create and implement their own models by writing C++ scripts.

In OPNET different models can be developed using a layer structure which is based on four levels. Network topologies are created at the network level where overall configuration and simulation parameters are set. The internal structure of different nodes such as transceiver design is done at node level and is modeled as finite state machine. In the process level, the functionalities of different nodes are defined. The lowest level, Proto-C, is where the programming of model behaviour is implemented.

OPNET Modeler has a good GUI which can be used for model creation, running simulation and analysing the results. It has built-in statistical analysis tools which can be used to analyse data. Based on features provided in OPNET Modeler, it can be used for a number of wireless networking technologies. Figure (6) shows a typical GUI view of OPNET Modeler with its various editors.

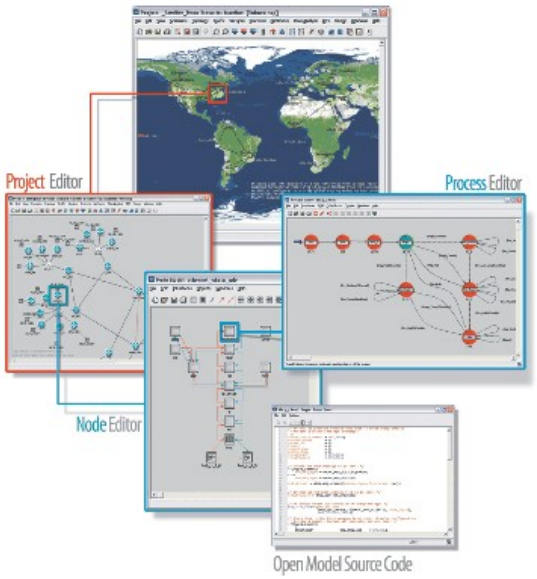

Figure 6: A Typical GUI View of OPNET Modeler [24]

4) GloMoSim

Global Mobile Information System Simulator (GloMoSim) [19] is used for scalable real-time network simulations for large wired and wireless communication networks. It is a discrete event simulator with the capability of *parallel event processing*. It uses a highly-optimized C-like simulation language called PARSEC (Parallel Communication Environment for Complex System) [30]. It uses node aggregation technique (to represent multiple nodes multiplexed within a single parsec node) to reduce the memory consumption. GloMoSim architecture is based on a multi layer stack model where a number of protocols have been designed for each layer from the physical propagation/mobility model, radio link, MAC, network layer and application layers. It gives an ease to users to implement different protocols at each layer and evaluate their performance separately. The major benefit of parallel processing is that a user can configure multiple protocols without effecting the performance of each other.

GloMoSim has extensible modular library for different network models. It has a customised GUI, built in statistic collection at each layer and open source code implementation for each of these layers. The parallel processing in GloMoSim creates a computational performance overhead and increases code complexity. A sequential version of GloMoSim is available as open source while the conservatively parallel version has been commercialized and is available under the name of QualNet [17].

5) JiST/SWANS

JiST/SWANS [20] is a simulator built by combining two general purpose simulation interfaces, JiST and SWANS (SWANS++ an extension of the JiST/SWAN framework) . JiST which stands for \textit{Java in Simulation Time} has been designed to transform the Java virtual machine into a scheduler for events. On top of this, a discrete event simulator engine called SWANS (Scalable Wireless Ad-Hoc Network Simulator), is structured to provide different environment models and communication protocols. In particular, it is used for simulation of ad-hoc networks. JiST/SWANS provides a built-in mechanism to support traffic generation as well as network simulation capabilities. It also gives a support to adopt various mobility models needed for realistic simulation such as Street Random Waypoint (STRAW) [31] specifically designed for vehicular ad-hoc

communication. One of the limitations of JiST/SWANS is that it generates the mobility of vehicles but does not provide any feedback between mobility and network modules.

B. A Comparative Study of Network Simulators

In this subsection, we will review the performance of different network simulator already discussed in the previous section. At the end of this section, we will present a specific VANET simulation framework and its requirements for the complete implementation of VANET scenarios. We will compare the above discussed network simulators based on their important features. Some of the key features under which a network simulator is evaluated as a reliable, efficient and scalable simulator are: availability of well known protocols in their library, mode of availability, graphical visualisation, ease of use in context of scripting/programming, memory requirement and available documentation.

Library of Known Protocols: In order to properly simulate a VANET scenario, an efficient routing is an important and critical part of implementation. There is plenty of research going on in the area of designing new routing protocols. However, there are some well known routing protocols such as AODV, OLSR, DSR, GPSR which are used as benchmarks in VANET simulations [32]. A good network simulator should have a bundle of rich libraries of existing routing protocol which can be used in simulations. In this context, all five network simulators discussed previously have rich libraries of well known protocols. GloMoSim has an extra feature of scalability which means that it can simulate large networks of nodes successfully.

Mode of Availability: There is a large number of network simulators available to simulate vehicular communication. Some of them are open source and freely available for use to anyone, while others are developed by commercial organisations and can be purchased. In the list of network simulator discussed in this paper, OPNET Modeler is the commercially available simulator. All other network simulators are open source and can be used for research purpose. However, it should be noted that OPNET provides free simulator to the academic research community under the University Program.

Graphical Visualisation: The success and popularity of a particular network simulator among the user community depends largely on the graphical user interface provided by the simulator. The network simulator with the capability of good GUI becomes famous for its use. In this regard, OMNeT++, OPNET Modeler and Jist/SWANS++ simulators have very friendly GUIs with good graphics. However ns-3 and GloMoSim do not provide such interfaces. It should be noted here that both ns-3 and GloMoSim with some external interface can provide a better graphical visualisation. In the latest versions of GloMoSim, the developers have provided a better Java based GUI for the simulator.

Ease of Use: Under this category, we review the network simulators whether their implementation requires complex scripting/programming in its implementation or it provides a palette based modules from where different network components can be imported into the workspace. In this regard, OMNeT++ and OPNET Modeler provide this kind of working environment that helps in designing the required network topologies. A lower level scripting/programming can also be done in these simulators if required. On the other hand ns-3, GloMoSim and JIST/SWANS++ do not provide such structure.

Computational Resource Usage: In order to simulate large networks, a simulator should take less simulation time and smaller memory usage. For this purpose ns-3 being C/C++ dependent simulator uses a large amount of computational resources. In fact in certain large scale networks (a network with 5000 or more nodes), ns-3 fails to achieve the simulation results in a reasonable amount of time with comparable size of memory. All other four network simulator being Java based, use less computational resources. Hence, we can classify them as resource efficient network simulators.

Documentation: A proper documentation including the installation guides, user manual and documents in regard to protocols and standards implementation helps the research community to select a network simulator for their research problems. A good and properly documented network simulator can provide more realistic results and analysing procedures. Having a large user community, ns-3 (extension of ns-2) has a good documentation available for the users. Being the commercial network simulator, OPNET Modeler enjoys the same features. The remaining three network simulators do have the sufficient documentation but they lack in the support for their user community.

In general, each simulator has its own advantages and limitations. For example considering the use of computational resources, the prime indication is that OMNeT++ executes at least an order of magnitude faster

than others and makes more efficient use of available memory [33]. Besides that ease of modifying the entire network and scalability, are two distinctive features of OMNeT++ that gives it an edge over others most widely used network simulators. One of the problems of ns-2 is its design structure that introduces much unnecessary inter-module dependency and thus makes the addition of new protocol models difficult to implement. On the other hand, NED (Network Description) language in OMNeT++ provides, design as well as code driven approach in defining network architecture and this gives advantage to OMNeT++ in comparison with other simulators in the market, where users need to rely on coding to define this. This gives an easy to understand structure to each module and also helps in debugging of code when required [34]. All these features and their comparison for the network simulators in context of VANET scenario is given in Table 1.

In context of VANET simulations, it is essential to create a simulator which includes all the necessary ingredients for the vehicular communication. In this regard, OMNeT++ existing framework such as INET [28] which is mostly designed for the wireless sensor network may not be appropriate for VANET. For the inclusion of various OSI layer protocols, such as TCP/IP, UDP as well as the implementation of WSMP architecture at the transport layer, it is necessary to properly implement these protocols in the simulator. WSMP protocol implementation is defined in Veins framework [35] which is primarily designed for the implementation of vehicular networks. Due to the module/framework structure of implementation in OMNeT++, a complete implementation of WAVE protocol stack as shown in Figure (2) can be visualized by integrating other frameworks together with the Veins framework. The most important task here is to understand which particular protocol (WSMP, TCP/IP or UDP) is needed for running a particular application of VANET. A better VANET simulator design must have inclusion of implementation of these mechanisms and depending on the need of applications.

C. Traffic Simulators

As shown in Figure (7), a real efficient VANET simulator takes an input from the Traffic Simulator. Without getting the input from traffic simulator, a VANET scenario may not result in a realistic model. The main purpose of a traffic simulator is to generate the mobility patterns of moving vehicles under certain pre-defined mobility models. Hence modeling the mobility of traffic pattern plays a crucial role in a traffic simulator and as a result in a VANET simulator. One of the main challenges in establishing a true simulator for VANET is to organise the functioning of network and traffic simulators in a concurrent manner [36]. Quite a few attempts [31, 35, 37, 38] were made to solve this important issue. In [31], authors provided a solution by first generating the traffic patterns and then saving them in a specified format. These saved patterns are then utilised as an input to the network simulator. In another approach [37], network simulator was developed with built-in capabilities of traffic simulator. This approach has been used in a well known simulator known as NCTUns (National Chiao Tung University Network Simulator - 4.0) [37]. In [35], authors presented an approach to *couple* the traffic simulator with a network simulator in order to create the realistic mobility patterns. This approach is utilised in the OMNET's framework called Veins which is primarily designed for vehicular networking.

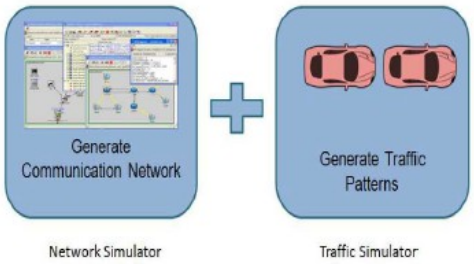

Figure 7: Design of a VANET Simulator

For a traffic simulator to generate an output pattern which can be used in a network simulator, a particular *realistic mobility pattern* of the vehicle movement is required which is the most important attribute of mobile ad-hoc wireless network. Therefore it is important to understand mobility issues in the designing of a true mobile ad-hoc network. With more and more reliance on simulation techniques for modeling networking requirements for MANET, one should look deeper into the key aspects of mobility in reference to the network design. Before discussing the traffic simulators in detail, we will get an insight into key mobility issues in VANET. Some of the common features in vehicular mobility are acceleration, deceleration, changing lanes and human driving patterns. In general, mobility models in VANET environment can be classified into two main categories as Macroscopic and Microscopic [39]. Macroscopic modeling considers road, building and streets in the mobility modeling where as the behaviour of vehicle such as acceleration/de-acceleration is considered as

microscopic mobility. Detailed discussion on mobility models in VANET is presented in [32], here for the sake of completeness of subject, we discuss a few commonly used mobility models in VANET.

One of the basic models to define mobility in VANET is Random Waypoint Model (RWM) [40]. In this model, each vehicle randomly selects the destination *d*, associated with speed *v* and moves towards that destination. In RWM, common examples of urban and highway mobility models are *Manhattan Model* and *Freeway Model* [41] as shown in Figures (8) and (9). In both models, movements of vehicles are restricted to pre-defined maps or graphs. These models also take into consideration minimum safety requirements for traffic engineering. Manhattan mobility model assumes the network to be divided into grids of square blocks having identical block size. So the network is basically composed of horizontal and vertical streets. Each street has two lanes: one for each direction (north and south direction for vertical streets, east and west direction for horizontal streets). A node is allowed to move only along the grids of horizontal and vertical streets.

Simulators	Library of known protocols	Computational Speed	Memory Usage	Ease of Use (Programming)	Realism in Result	Friendly GUI	Good Documentation	Type of Licence
ns-3	✓	x	x	x	x	x	✓	Open Source
OMNeT++	✓	✓	✓	✓	✓	✓	x	Open Source
OPNET Modeler	✓	✓	✓	✓	✓	✓	✓	Commercial
GloMoSim	✓	✓	✓	x	x	x	x	Open Source
Jist/SWANS++	✓	✓	✓	x	x	✓	x	Open Source

Table 1: A Comparison of Different Network Simulators in VANET Environment

Some other mobility models are based on flow theory named as flow based modeling in which traffic flow is the main ingredient for mobility modeling. *Car Following Models* (CFM) [42] is one of the examples of such category of modeling where vehicles follow the pattern of the vehicle at the front by maintaining a safe distance. This safe distance is computed using one of the core traffic engineering concepts presented in [43]. The advantage of using this model lies in the simplicity of the approach taken in the design of this model. One of the limitations of this model is that it does not consider other important parameters such as driving behaviour and traffic patterns in its modeling paradigm. In [44], authors proposed a modification for the improvement in CFM by integrating the main attributes of traffic engineering such as acceleration, deceleration and speed. This model is widely used for simulation in VANET environment. Another example of a flow based model is *Intelligent Driver Model* (IDM) [45], where the traffic state at a given time is considered by the positions and velocities of all vehicles. The decision of any vehicle to accelerate or to de-accelerate depends on its own velocity as well as the velocity of the vehicle in front. Lane changing has also been implemented in refined models of IDM like IDM-LC. Lane-changing decisions are taken depending on neighboring vehicles. In order to improve the realism in mobility models, authors presented another model called *Krauss Model* in [46]. Many efforts are being made in VANET simulators to introduce more realistic approaches in the modeling, leading to the trace based mobility modeling. In trace based mobility models accurate information about the mobility traces of vehicles and driver behaviour are captured and used in the mobility modeling. One major limitation for the creation of trace-based modeling is the limited availability of actual vehicular traces.

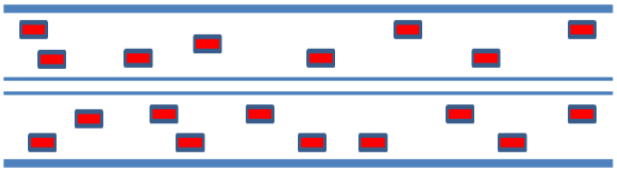

Figure 8: A Freeway Model

Besides this, a number of micro simulation models like as *Cellular Automaton* (CA) model [47] and *STRAW model* [31] have been created to satisfy the requirements of accurate and concise VANET modeling. In addition to that, learning from driver behaviour, or so called *Behaviour Modeling* has also been attempted. In [48], authors have proposed a traffic simulator named MATSim [49] based on this type of modeling. implementing complex models puts extra constraints like *simulation processing time* and *memory requirement* on traffic simulators and should be a deciding factor in selecting the right traffic simulator. In the following subsections, we provide an overview of a few well known traffic simulators such as *VanetMobiSim*} [22], *SUMO* (*Simulation of Urban MObility*) [23], *MObility model generator for VEhicular networks (MOVE)* [50] and *Traffic and Network Simulation Environment (TraNS)* [51].

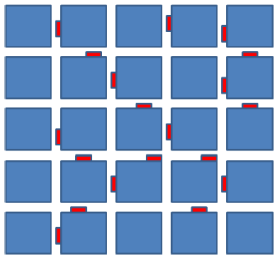

Figure 9: A Manhattan Model

1) VanetMobiSim

VanetMobiSim [22] is an open source traffic simulator derived from Communication in Ad-hoc Network for Ubiquitous Computing MOBility Simulator (CanuMobiSim) [52], which is a traffic simulator for user mobility. The main purpose of VanetMobiSim is to generate the traffic patterns of moving vehicles. VanetMobiSim is a JAVA based traffic simulator that aims to generate mobility traces close to a higher degree of realism which can be used in many network simulators, e.g. ns-3, GloMoSim, Qualnet etc. VanetMobiSim supports both macro-mobility as well as micro-mobility features by considering the road structure, road characteristics and the existence of traffic signs, as well as driver behaviour, vehicle speed and acceleration modeling. It also supports different mobility models like IDM and IDM-LC explained in Section IV. The basic format of simulation scenario in VanetMobiSim is defined in XML format which can be generated by specifying a simulation area, adding a global extension to the simulation, adding a node along with its characteristics to the simulation and grouping the nodes in the simulation.

VanetMobiSim includes an increased detail of macro-mobility and micro-mobility modeling features to produce realistic vehicular mobility models. However, it lacks in producing a user friendly environment and also does not include some of the well known map models such as Manhattan as well as well known mobility models such as STRAW. It also does not include different vehicle types and hierarchy of different road junctions. By inclusion of these parameters this simulator can produce more realistic traffic patterns.

2) Simulation of Urban MObility (SUMO)

Simulation in Urban mobility (SUMO) [23] is an open source traffic simulator which employs Krauss mobility model to generate traffic pattern. However, SUMO is not only a traffic simulator but is considered as a suite of applications which is used to prepare/perform the simulation of traffic patterns. Some of the key features of SUMO are *collision free vehicle movement*, *multi-lane street environment*, *junction-based right-of-way rules* and *inter-connectivity* with other application during the simulation execution. In SUMO, the network topology can be generated by using an XML file manually, or importing from any other source or using the built-in topology generation function. In order to further refine the network topology, SUMO can define a number of user specific modeling details. Figure (10) shows a GUI of SUMO simulator showing an intersection view of a road.

SUMO can take input from different file formats to generate road networks (maps) and traffic demands. An application provided within SUMO called *netgen* is used to generate internal format for the formation of road network. It can also take this information by importing a digital road map from any other source. Another utility called *netconvert* plays an important role in relation to this and allows to read networks topologies from other traffic simulators using common formats, as *shapefiles* or *Open Street Map* [53]. Each vehicle is defined by an identifier (name), the departure time and is assigned a route to follow within the network. In addition to this, parameters like lane to use, velocity or position of the vehicle can also be defined in SUMO. Few other built in variables allow in defining the physical appearance using powerful GUI. Each vehicle's position is described by the lane the vehicle is on and the distance from the start of this lane. The

distance between vehicles is controlled by using the mobility model adopted in the simulation run time. Large simulation environments can be simulated using SUMO without any constraint or limitations on execution speed [9]. In addition to this, SUMO gives the possibility to communicate directly with other network simulators to create a real-time simulation environment.

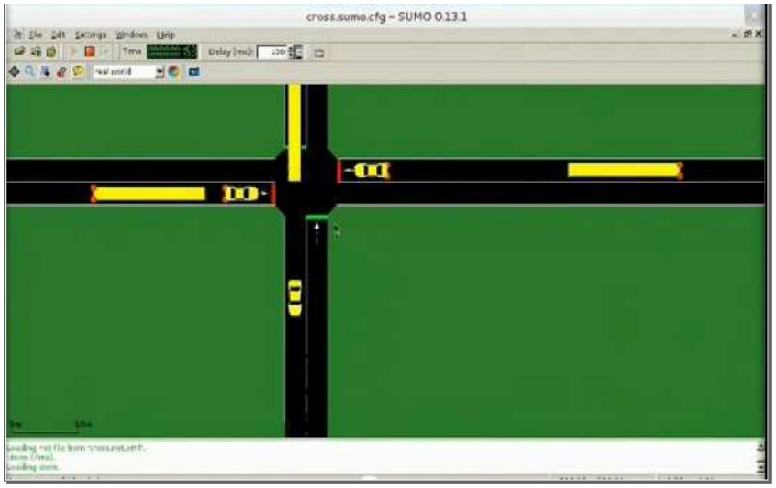

Figure 10: A GUI View of SUMO

3) MObility model generator for VEhicular networks (MOVE)

MObility model generator for VEhicular networks (MOVE) [50] is built on top of SUMO simulator in order to provide a better user interface for creating realistic traffic models. Figure (11) shows the GUI for MOVE. The output of the MOVE can be directly used by well known network simulators discussed in Section IV. MOVE is an open source traffic simulator allowing the users to quickly generate realistic simulation scenarios without going into the extensive coding and scripting details. MOVE, similar to other traffic simulators can use user defined, random, or map based graph models for generating traffic patterns.

MOVE generates the traffic patterns based on the mobility models used in SUMO. However, MOVE can also generate its own mobility model according to specifications. In large network scenarios, mobility models generated by MOVE do not provide satisfactory results because of the non-availability of realistic model such as lane changing and radio obstacles.

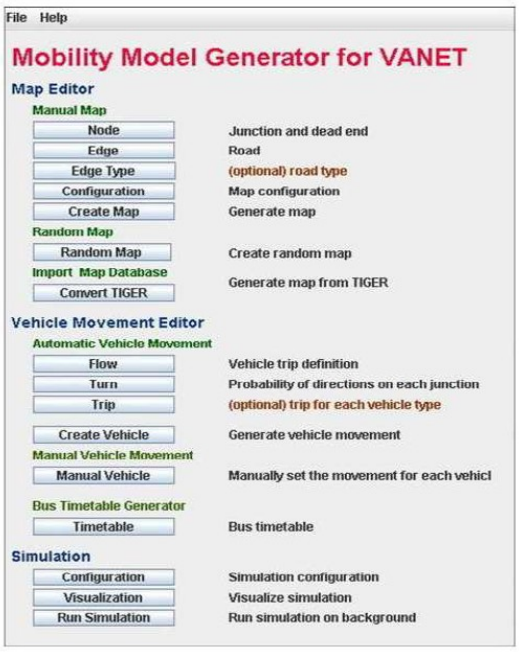

Figure 11: A User Interface View of MOVE [50]

4) Traffic and Network Simulation Environment (TraNS)

Traffic and Network Simulation Environment (TraNS) [51] is a Java-based traffic simulator designed by integrating the traffic simulator SUMO and the network simulator ns-2. It has good visualisation interface (GUI) in order to simulate vehicular networks. The main purpose of TraNS is to generate realistic traffic patterns by taking the input from the capability of network simulators. This improves the simulation scenarios of VANET by incorporating the characteristics and model used in the network simulators. The main attributes of TraNS are that it supports custom based graphs, grid based random graphs and graphs from different maps databases such TIGER database [54]. As far as the traffic level features are concerned, TraNS supports CFM mobility model and a good traffic intersection management. Even though the development of TraNS has stopped further, it is still available as a scalable tool that can simulate large number of nodes. One of the main limitations in TraNS architecture is its inability to have a mechanism to arrange communication from network simulator back to the traffic simulator [36].

D. A comparative study of Traffic Simulators

In order to conclude the discussion on traffic simulators for VANET simulations we present a qualitative performance evaluation of above mentioned traffic simulators by focusing on their key characteristics. Some of the characteristics under which the traffic simulators are evaluated are:

Software Attributes: In this subsection we compare the availability, ease of use, graphical visualisation and complexity of the traffic simulators. This will help the reader to choose the best traffic simulator for his/her simulation scenario. The above described traffic simulators are available as open source simulators. From the user utility point of view, MOVE is an easy to use traffic simulator due to its user friendly GUI and no lower level scripting involved in generating the traffic patterns. As far as computational speed and complexity is concerned, SUMO is preferred over its counterparts. This is because of the reasons that MOVE is built on top of SUMO and there is extra intercommunication involved between SUMO and MOVE before the actual information reaches the user. Similarly TraNS integrates SUMO and ns-2 before giving the final output. SUMO also out performs VanetMobiSim because of its ability of using XML scripting as compared to Java in VanetMobiSim.

Realistic Mobility Models: Implementation of up-to-date and realistic mobility models in traffic simulators is an important task. Most of the traffic simulators achieve this objective using the existing mobility models in their library. The commonly used mobility models in VANET are *Random Way Point*, *STRAW*, *Manhattan*, *Freeway* and *IDM*. VanetMobiSim supports *Random Way Poin*t and *IDM* mobility models but it does not support *STRAW* and *Manhattan* Mobility models. The second and third traffic simulators SUMO and MOVE support all four mobility models mentioned previously.

Geographic Information through Maps: In order to update the geographical location, traffic simulators rely on information from maps. There are different types of maps available around the globe free of cost and on commercial basis. All of the traffic simulators discussed in this paper, use real and user defined maps to generate traffic patterns.

V. CONCLUSIONS

In real world implementation of VANET scenarios, high cost, reliability and results close to reality are important aspects. The simulation of VANET mainly comprises of a network simulator working concurrently with a traffic simulator. In addition, an accurate implementation of IEEE standards architecture for VANET in network simulators is essential to achieve realistic results. Whereas in traffic simulators, a detailed mobility model including microscopic and macroscopic aspects of mobility must be implemented to get proper results from simulations. The paper introduced some of the most common and popular open source discrete event network simulators in the context of VANET. In this paper, we have discussed the importance of implementing IEEE standard architecture, WAVE protocol, in the network simulators. The paper emphasizes that a combination of SUMO and OMNeT++, known as Veins framework, is an efficient way of simulating a vehicular communication scenario that includes most of the standard implementation protocol in the network stack. A comparative analysis of different network simulators shows that OMNeT++ with the introduction of Veins framework is a reliable selection for simulating realistic VANET scenarios. Additionally, the paper has also discussed ns-3 which is the recent development of ns-2 by the inclusion of more realistic standard/protocol in the legacy ns-2. The paper has also discussed a number of well known and popular traffic simulators along with the mobility models. Among the traffic simulators discussed, it is concluded that SUMO provides most of

the realistic mobility features required for modeling traffic patterns in urban as well as highway scenarios. The paper points the future direction of extending Veins framework to include mechanism of supporting transport layer protocols TCP/UDP in OSI layer for various applications. This task can be achieved by the integration of INET framework with the Veins architecture resulting in a complete VANET simulation architecture.

Declaration: We, authors of the above article, declare that we do not have any conflict of interests in publishing this paper.

REFERENCES

[1] "World Health Organization - Global Plan for the Decade of Action for Road Safety," http:www.who.int/roadsafety/decade of action/, 2010.
[2] "World Health Organization - Global Status Report on Road Safety: Time for Action," http://www.who.int/violence injury prevention/road safety status/2009/, 2009.
[3] ITS - Intelligent Transport Systems, http://www.its-australia. com.au, 2004.
[4] M. Watfa, "*Advances in Vehicular Ad-Hoc Networks: Developments and Challenges"*, Intelligent Transport Systems. IGI Global, 2010.
[5] U. Lee, J. Lee, J. S. Park, and M. Gerla, "FleaNet: A Virtual Market Place on Vehicular Networks," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 1, pp. 344–355, Jan. 2010.
[6] "Ford Accelerates Intelligent Vehicle Research, Creating 'Talking' Vehicle to Make Roads Safer," http://media.ford.com/ , 2010.
[7] X. Xian, W. Shi, and H. Huang, "Comparison of OMNET++ and other simulator for WSN simulation," *2008 3rd IEEE Conference on Industrial Electronics and Applications*, pp. 1439–1443, Jun. 2008.
[8] K. Shafiee, J. B. Lee, V. C. M. Leung, and G. Chow, "Modeling and Simulation of Vehicular Networks," *Network*, pp. 77–85, 2011.
[9] F. J. Martinez, C. K. Toh, J.C. Cano, C. T. Calafate, and P. Manzoni, "A Survey and Comparative Study of Simulators for Vehicular ad hoc Networks (VANETs)," *Transportation*, 2009.
[10] "IEEE STANDARD - Wireless Access in Vehicular Environments," http://grouper.ieee.org/groups/802/11.
[11] "IEEE 802.11TM WIRELESS LOCAL AREA NETWORKS - The Working Group for WLAN Standards," http://grouper. ieee.org/groups/802/11.
[12] H. Hartenstein and K. Laberteaux, " *VANET Vehicular Applications and Inter-Networking Technologies"*, Intelligent Trans- port Systems. Wiley, 2009.
[13] Y. Morgan, "Notes on DSRC; Wave Standards Suite: Its architecture, design, and characteristics," *Communications Surveys Tutorials, IEEE*, vol. 12, no. 4, pp. 504–518, 2010.
[14] C. C. Lin, L. C. Hu, K. H. Hsu, and H. H. Li, "On the Performance of Vehicular Group Communications in IEEE 1609/802.11p Networks," vol. 11, 2011.
[15] AusDSRC, "Australian Dedicated Short Range Communications," http://www.ausdsrc.com.au/, 2008.
[16] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," Proceedings *of the IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.
[17] "Scalable Network Technologies - QualNet," http://web. scalable-networks.com/content/qualnet.
[18] "The Network Simulator- NS-2," http://www.isi.edu/nsnam/ns/.
[19] "GloMoSim - Global mobile Information System Simulation Librarym," http://pcl.cs.ucla.edu/projects/glomosim/.
[20] "JiST/SWANS Java in Simulation Time – Scalable Wireless Ad-hoc Simulator," http://jist.ece.cornell.edu.
[21] "OMNeT++ - Object-Oriented Discrete Event Simulation System," http://www.omnetpp.org/.
[22] "VanetMobiSim - The Vanet Mobility Simulator. 2.0," http://vanet.eurecom.fr/, 2009.
[23] "SUMO - Simulation in Urban Mobility," http://sumo. sourceforge.net/, 2009.
[24] "OPNET Technologies - Inc," http://www.opnet.com.
[25] "The Network Simulator- NS-3," http://www.nsnam.org.
[26] "Tcl (Tool Command Language)," http://www.tcl.tk.
[27] "Mobility Framework for OMNeT++," http://mobility-fw. sourceforge.net.
[28] "INET Framework -Package for the OMNeT++ Simulation Environment," http://inet.omnetpp.org.
[29] "MiXiM-(Mixed Simulator) a simulation framework for wireless and mobile networks using the OMNeT++ simulation engine," http://mixim.sourceforge.net.
[30] R. Bagrodia, M. Takai, Y. an Chen, X. Zeng, and J. Martin, "Parsec: A parallel simulation environment for complex systems," IEEE *Computer*, vol. 31, pp. 77–85, 1998.
[31] D. R. Choffnes and F. E. Bustamante, "An Integrated Mobility and Traffic Model for Vehicular Wireless Networks," *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks - VANET '05*, p. 69, 2005.
[32] S. U. Rehman, M. A. Khan, T. A. Zia, and L. Zheng, "Vehicular Ad-hoc Networks (VANETs) - An Overview and Challenges " *Journal of Wireless Networking and Communications*, vol. 3, no. 3, pp 29-38, 2013.
[33] X. Xian, W. Shi, and H. Huang, "Comparison of OMNeT++ and other simulator for WSN simulation," in *ICIEA 2008: 3rd IEEE Conference on Industrial Electronics and Applications*, pp. 1439–1443, 2008.
[34] A. Varga and R. Hornig, "An Overview of the OMNeT++ Simulation Environment," *SIMUTools*, no. March 07, 2008.
[35] "Veins - Vehicles in Network Simulations," http://veins.car2x. org/, 2010.
[36] S. Zeadally, R. Hunt, Y. S. Chen, A. Irwin, and A. Hassan, "Vehicular ad hoc networks (VANETS): status, results, and challenges," *Telecommunication Systems*, Dec. 2010.
[37] S. Y. Wang, C. L. Chou, Y. H. Chiu, Y. S. Tzeng, M. S. Hsu, Y. W. Cheng, W. L. Liu, and T. W. Ho, "NCTUns 4.0: An Integrated Simulation Platform for Vehicular Traffic, Communication, and Network Researches," *2007 IEEE 66th Vehicular Technology Conference*, pp. 2081–2085, Sep. 2007.
[38] S. Fischer, J.P. Hubaux, A. Wegener, M. Pi, M. Raya, and H. Hellbr, "TraCI : An Interface for Coupling Road Traffic and Network Simulators," pp. 155–163.
[39] J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: A Survey and Taxonomy," *IEEE*

Communications Surveys & Tutorials, vol. 11, no. 4, pp. 19–41, 2009.
[40] J. Broch, D. a. Maltz, D. B. Johnson, Y.C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking - MobiCom '98*, pp. 85–97, 1998.
[41] F. Bai, N. Sadagopan, and A. Helmy, "The IMPORTANT framework for analyzing the Impact of Mobility on Performance Of RouTing protocols for Adhoc NeTworks," *Ad Hoc Networks*, vol. 1, no. 4, pp. 383–403, Nov. 2003.
[42] D. C. Gazis, R. Herman, and R. W. & Rothrey, "Non Linear Follow the Leader models of Traffic Flow," *Operations Research*, pp. 545–567, 1961.
[43] L. A. Pipes, "An Operational Analysis of Traffic Dynamics," *Journal of Applied Physics*, no. 24, pp. 274–281, 1953.
[44] P. G. Gipps, "A Behavioural Car Following Model for Computer Simulation," *Transportation Research Part B*, pp. 105–111, 1981.
[45] M. Treiber, A. Hennecke, and D. & Helbing, "Congested Traffic states in Empirical Observations and Microscopic Simulations," *Physical Review E*, vol. 62, no. 2, pp 1805-1824, 2000.
[46] S. Krauss, P. Wagner, and C. Gawron, "Metastable States in a Microscopic Model of Traffic Flow," *Physical Review*, vol. 55, pp. 55–97, 1997.
[47] K. Nagel and M. Schreckenberg, "A Cellular Automaton Model for Freeway Traffic," *Journal de Pyysique I*, pp. 2221–2229, 1992.
[48] M. Balmer, M. Rieser, K. Meister, D. Charypar, N. Lefebvre, and K. Nagel, "MATSim-T : Architecture and Simulation Times," *A. L. C. Bazzan and F. Klugl , Information Science Reference*, pp. 57–78, 2009.
[49] "MATSim - The MATSim Framework," http://www.matsim. org/, 2009.
[50] "MOVE - MObility model generator for VEhicular networks ," http://lens.csie.ncku.edu.tw/Joomla version/index.php/, 2007.
[51] "TraNS - Traffic and Network Simulation Environment," http: //lca.epfl.ch/projects/trans/, 2007.
[52] "CANU Mobility Simulation Environment (CanuMobiSim)," http://canu.informatik.uni-stuttgart.de/mobisim.
[53] "Openstreetmap - A resource for Opensource Maps," http://www.openstreetmap.org/.
[54] "TIGER - Topologically Integrated Geographic Encoding and Referencing," http://www.census.gov/geo/ maps-data/data/tiger.html.