TECHNISCHE UNIVERSITÄT
CHEMNITZ

Traffic Simulation in a Model World

(Programming a Demonstrator for Semi-Autonomous Driving)

Masterarbeit

Chemnitz, 15 Mai 2015

zur Erlangung des akademischen Grades

Master of Science

Fakultät Informatik

Juniorprofessur Software Technologie für Eingebettete Systeme

eingereicht von: Aditya Gollapinni Manjunath

Geburtsdatum: 15 Oktober, 1989

Geburtsort : Mulund, Indien

Betreuer: Jun.-Prof. Dr. Alejandro Javier Masrur

Juniorprofessur Software Technologie für Eingebettete Systeme

Abstract

Research in Vehicular Ad-hoc Networks[1] (VANET) has developed technology that it is now possible to integrate this technology into real-world cases. But before this integration happens, vigorous testing has to be performed, which does not come without a cost. As field tests are limited to a small number of automobiles because of its infeasibility, it is practical to virtually test the development, and simulation happens to be the solution. Simulation provides two main advantages- Hardware-in-loop and Software-in-loop, where, in the former, technology developed will be integrated into the simulation scenario and tests performed on it, and the latter is a software model developed based on the hardware. While hardware is not available for everyone to use, software takes precedence, and Car-to-X has been implemented as software models and many simulation frameworks have been constructed in the past decade. The main components of Car-to-X communications, are the protocols and the environment in which they are implemented and simulation frameworks provide two distinct advantages: First, detailed network implementation of all the communication protocol layers. Second, a realistic road traffic simulation i.e. vehicle mobility, which gives very accurate estimates of vehicle speed and position in an environment. The intention of performing simulation is that it allows researchers and the automobile industry an insight into the behaviour and response of vehicles in VANET scenarios, such as time of response, braking, variation in speed, and change in duration and route of travel. Along with the preceding factors, the applications of VANET are evaluated in simulations- Safety Critical, traffic information, entertainment, etc. of which safety critical applications receive much attention. On the whole, simulations have proven to be very effective. One drawback of simulation of VANET is that, modelling human behaviour has not been very successful, but proceeding research in this field has furnished promising results. This study focusses on the evaluation of one simulation framework and a safety critical aspects of VANET application simulated using this framework. Further, free and openly available software have been used in this study, owing to the large number of applications and their results, developed by the community and involved in VANET research and the inexpensiveness of this software.

[1] VANET is developed with the application of mobile ad-hoc networks into automobiles, which allows vehicles to participate in wireless communications with a coverage of 100-300 m.

Contents

Figures

Listings

Equations

Chapter 1

Introduction

Road transportation is a fundamental part of society and the state of a country's economy is reflected by the state of the same. With the increase in traffic and safety measures, as a consequence of it, there is always room for development. Since, the early 1990s, road traffic has increased dramatically and there is little means of controlling this increase. One of the ways of making driving safer, is the concept of Car-to-X and Car-to-Car communication.

References of Car-to-X communication can be found as early as 1939, in New York World's Fair [1]. General Motors also had ideas about Car-to-X communication as displayed then, in its Futurama.

Since then, research projects have been undertaken on Intelligent Transportation Systems, but unfortunately many have failed, resulting in the non-implementation of the researched in reality.

One of the reasons for this was the inability of the then employed underdeveloped technology to achieve a very advance cause. Also, this research did not concentrate on non-infrastructure solutions.

Since 1990s, the world has seen the development of cellular mobile communication technology, which now has universal presence. Since the improvement in computing power, the pace of processing data, has led to the deployment of MANET[2] technologies. The U.S. FCC allocated the Dedicated Short Range Communications (DSRC) band in 1999, with 75MHz in the 5.9GHz region for the use of vehicular short-range wireless communication.

Although exciting, the application of short-range radios pose a challenge. It faces problems such as interference and ad-hoc networking, network mapping (topology) and security issues. These in turn pose troubling scenarios, with regards to dissemination of messages, urban related Car-to-Infrastructure/Car and highway related Car-to-Infrastructure/Car, concerning the speed of the cars driving. Along with this, cars must be able to update their positions, speed and targets periodically, and the road must be able to communicate restrictions and warnings to the cars.

Car-to-X involves exchange of messages, the messages need to be interpreted by the communication participants. The Intelligent Transportation Systems (ITS) [2] station updates the messages regarding the traffic situation and disseminates the messages periodically.

Cooperative Awareness Message (CAM) and Decentralized Environmental Notification Message (DENM) are message types whose standards have been specified by the European Institute for Telecommunication Standards (ETSI). This station is what is referred to in this project and the tool, as a Road-Side Unit (RSU). This informs the vehicles on traffic situation over a large area and to warn drivers if necessary.

[2] Mobile ad hoc Network

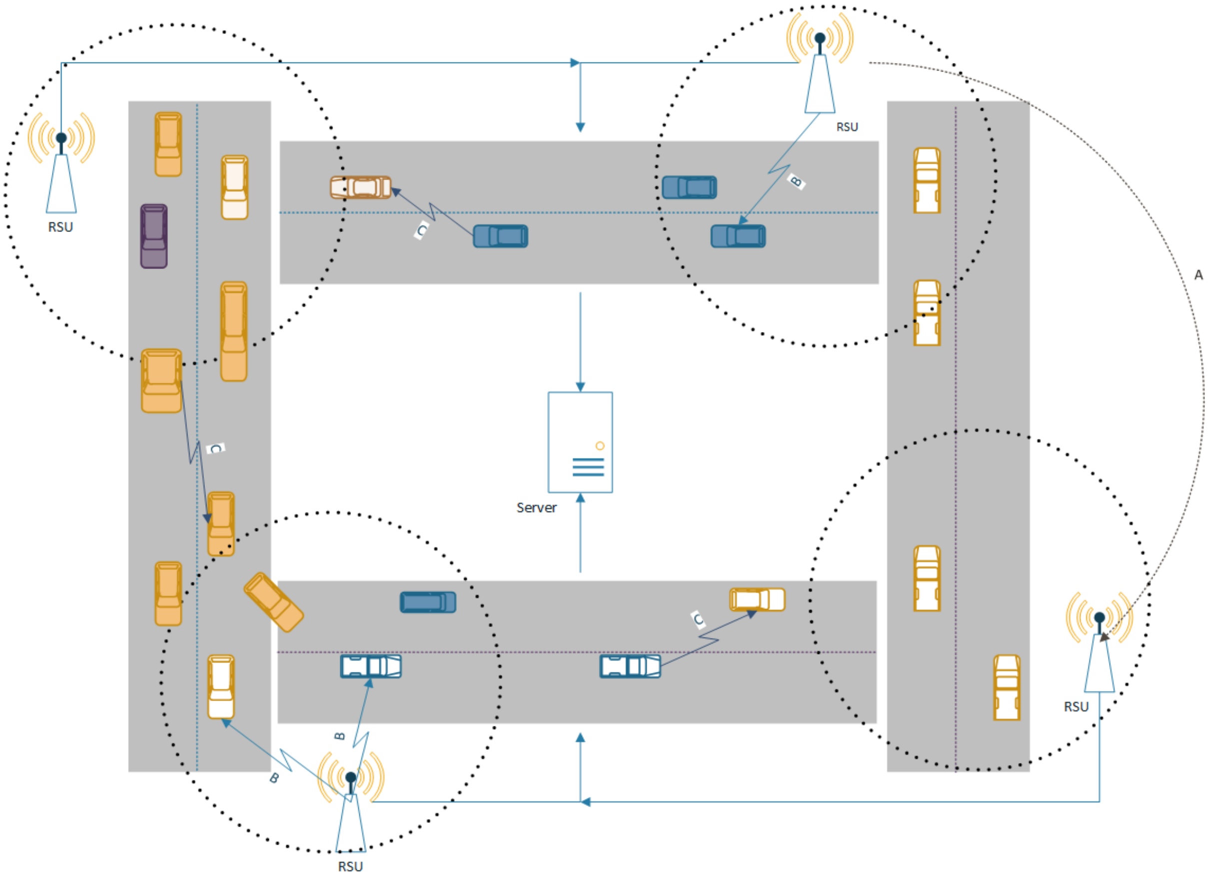


Figure 1.1 – Basic VANET Architecture (A) Infrastructure-to-Infrastructure, (B) Car-to-Infrastructure, (C) Car-to-Car

The Car-to-Car Communication Consortium, a non-profit, was setup by companies in the automobile manufacturing industry, supported by Original Equipment Manufacturers (OEMs), with an aim to set up open standards for ITS and also to develop business models to speedup market penetration.

As of November, 2014, the Central European Car-to-X Corridor [3], a drive lane linking Rotterdam, Frankfurt/Main and Vienna, is being laid. The aim, to develop application and harmonised vehicle interface. Initial plan is to provide construction work warnings, traffic information and restrictions by Wireless Local Area Network (WLAN) or radio, and vehicles are to transmit position and speed to the local stations via WLAN or radio. This project aims to provide these initial facilities by 2015 and work is underway.

Testing technologies for Car-to-X are increasing globally, and along with that the deployment of the technologies in the early stages. RSUs and other ITS technologies offer an advantage which should be used at this stage. They improve network performance and solve issues that Car-to-X architecture has failed to address or solve. A more hybrid network i.e. WAVE powered cars and RSUs to provide any service, is required. Protocols and mechanisms which enable communications on both sides, are key factors that affect Car-to-X. Deployment issues also need to be address in sense that sufficient numbers of these technologies have to be deployed so that effective services may be provided.

Chapter 2

Challenges with Car-to-X Simulations

There is no perfect technique to experiment with Car-to-X without incurring a high cost. Vehicles, communication equipment-both for the cars and the RSU, software are only the firsts that come to mind. But when the consequences of deviation and miss working are considered, the cost increase uncontrollably. Hence, the research teams and companies turn to simulations to provide the results until the desirable conditions are arrived at. Also, it is easier to design techniques by way of simulation, to obtain all the status information without much loss. It is important to know that, in the real world, collection of status messages is of paramount importance as they will be used to assist drivers to avoid dangerous road conditions. The transmission should be reliable and be delivered quickly to allow the driver to react in time.

The communication engineers develop protocols and algorithms based on the required parameters for the system to make a judgement. The simulation should be designed with a certain level of realism, i.e. consider all technological and driving environmental aspects.

One challenge in this field is the factor of mobility of the vehicles. Another challenge in Car-to-X simulation is the factor of including and estimating the features of a wireless communication channel. Manhattan grid[3] models are used to model mobility and address the former. But unfortunately models with high quality of realism is desired which is capable of simulating individual vehicle dynamics, along with the complex structure of drive lanes/roads, which will be able to handle traffic flows. Simulation of human driving behaviour is something that will be difficult to model into a simulation. Evaluation of simulation performance depends on the parameters considered during the simulation run.

Notably, just achieving simulation of exchange of messages, over a certain distance, is not enough. If the simulation needs to be designed to reflect the real world issues, then the simulation needs to consider obstacles, and the delay or even blocking as a consequence of obstacles. Alternative, strategies need to be designed to re-route transmission of messages. This would also constitute a major issue, as there is no concrete way of discerning if a message has reached the destination of not, as retransmission of message comes with overheads.

The direction of a vehicle's drive should decide in which direction the messages have to be transmitted. If the vehicle suddenly changes direction of drive, then the tools should be able to compute the direction and change the transmission characteristics. Also, acceleration and deceleration of the cars in motion should be taken into account to provide a realistic simulation.

König et al. [4], developed driver behaviour model using AI methods for driver's route planning.

There are many mobility and network simulation tools in the market, which differ from each other essentially based on the how the mentioned components are integrated and the level of interaction between them. They are typically loosely integrated, tightly or hybrid frameworks.

---

[3] Manhattan grid places streets and lanes adjacent to each other i.e. at 90°

Loosely, the mobility component functions independently from the network. Files with vehicular movements are reported to the network simulator, without any direct interaction. Tightly integrated frameworks interconnect the two components. Hybrid frameworks simulate both the network and vehicular components. Hybrid frameworks have years of validation to be done by the research community.

Intersection Management is another problem. There is no concrete method of deciding on how to manage intersections, whether to decide the message transmission based on vehicle density or traffic light.

Traffic density differs over a long period of time. Simulation has to differentiate between high traffic and low traffic and other occasions. This leads to change in route paths and trips of the vehicles.

The advantage of using tightly integrated frameworks is the realistic traffic simulation and well established, validated communication models. But as both the sides are separated, the interface can have communication and synchronisation overheads. But due to the lack of better simulation techniques, these are most favourable and this study uses them.

Chapter 3

State of the Art

Companies have developed tools to simulate communication. Vector Informatik GmbH has developed tools that simulate the communication for the parallel processing of WLAN 802.11p with CAN, FlexRay, Ethernet and others. They also have developed the libraries CANoe.Car2x, which allows direct analysis of the network and transport protocols pertaining to Car-2X.

This permits direct analysis of both the Car-to-X specific network and transport protocols and the application messages overlaid on them; in the Car-to-X field this might be the Co-operative Awareness Message (CAM) and the Decentralized Environmental Notification Message (DENM) [5]. Secured Packets are also supported here. As of now technology development of Car-2X seems bright and we may have a full fletched functioning ITS system globally soon.

In the research field, there are a number of network simulation tools that provide a number of highly validated protocols, but the drawback is that they are unable to simulate mobility. A practical approach would be to couple traffic simulators with the network simulators. Traffic simulators provide realistic traces of vehicle mobility and this is routed to the network simulators. In this approach, traffic traces are generated before the network traces and hence, the traffic has control over the network simulators and the other way around is not possible. Hence, not all applications can be tested. A better technique is the use of bi-directional simulation. This allows the network and traffic simulators to have a control on each other.

In their paper [6], the traffic simulator STRAW and network simulator SWANS are connected, SWANS being scalable wireless ad-hoc simulator. STRAW works on random waypoint mobility model. The mobility in STRAW is constrained by streets defined in road maps. But is provides an inaccurate simulation.

NCTUns [7] does not interconnect simulators, but tries to provide a new framework that simulates mobility and networking protocols. It uses the Linux TCP/IP protocol stack, being able to run real application program on nodes without any modification. However, large scale network simulation is difficult to perform, even as distributed emulation of a network over machines is possible.

TraNS [8] introduced a simulation framework which coupled ns-2 and SUMO, which was the first bi-directional simulation framework. TraNS runs a loop that continuously sends control commands to SUMO, allowing starting, stopping or changing of drive lane. But it is obsolete now and no longer maintained.

TraNS was succeeded by iTETRIS [9]. iTETRIS integrates SUMO and NS-3 via a control block iTetris Control System. It simplified the physical layer. However, communication and synchronisation overheads were not ruled out and hence reduced the efficiency of it on the whole. Along with this, it required installation and configuration of different components, which makes the setup complex.

While GrooveNet [10] supports a single framework, it supports interaction between nodes and also between real and simulated terminals. But it lacks validated communication modules.

Scientists and researchers of the Wireless Communication and Networks (WN) Department at Fraunhofer- Gesellschaft Heinrich Hertz Institute (HHI)[4] are working on the measurement data from the HHI Channel Sounder (Wireless), with a measurement bandwidth of up to 1GHz and 2x8 MIMO antenna system. It enables simultaneous measurement of multiple antenna position. Also, records directional channel measurement data from an antenna array. The wireless channel measurements are accompanied by extremely precise position location and a 360° video recording, which allows measurement, assessment and process the radio channel in real-world traffic. The data obtained in the Car-2X communication scenarios serve as basis for channel modelling which simulates radio channel. This data and channel scan serve other purposes too. When used in MATLAB®[5] simulations for evaluating the efficiency of communication systems according to IEEE 802.11p, the simulations developed are embedded in the NS-3 to take advantage of all the layers. Hence, applying real world measurements to realistically simulate the communication scenarios.

It goes without saying that OEMs and automobile manufacturing companies are investing in technologies to support Car-2X.

[4] http://www.hhi.fraunhofer.de/departments/wireless-communications-and-networks/research-areas/enabling-technologies-for-future-wireless-applications/car2x.html

[5] © 1994-2015 The MathWorks, Inc

Chapter 4

Thesis Foundation

The department of Junior-Professor Software Technology for Embedded Systems is developing a model world and car which will serve as demonstration and simulation platform for the testing of traffic control algorithms. This setup is essentially a semi-autonomous driving setting.

The model car is of scale 1:24, which is controlled by a 32Bit ARM Cortex M3 microcontroller, a motor driver, power management circuitry and other sensors. This car will be equipped to drive autonomously in the model world, follow streets, avoid crashes, etc. Also, this car will be controlled remotely by a control station. Apart from accelerating, flashing when turning, turning the lights on, etc. the car should be able to simulate complex behaviour such as mass inertia, driver reaction time and sensor failure. Communication with the control station should occur in real-time via a wireless communication port.

On the control station, a tool for simulating Car-to-X and traffic control algorithms will be running. The tool will connect to the model car to simulate traffic algorithms. The control station will provide an interface to connect to a database and simulation framework. The simulation will also implement the model world, which includes the demonstrator, traffic lights, pedestrians, etc. The simulation should be written in a way, such that parts of the simulation can be replaced with parts of the real world and vice versa.

This Thesis concentrates on implementing the control station simulation platform. As it is difficult and impractical to go from simulating Car-to-X technologies, directly to implementation, the model car in the model world provides good testing platform for such algorithms. It provides as a research platform for testing the network communication between the control station and the model car, such as delay, reaction time of the car, data packet size, how often transmissions need to occur (periodic), response to the received commands from the control station, etc.

This project will not cover the entirely of Car-to-X communication, rather concentrate on the following parts,

- Examining and opting simulators for the project, both network and mobility
- Setup simulation of traffic in a particular scenario(s) (mapping), clarify the parameters
- Implement the particular scenario in the chosen network simulator
- Analyse the implementation
- Analysis of Vehicle Dynamics

This project will simulate both Car-to-Car and Car-to-Infrastructure (RSU), which are addressed under the broader term of Car-to-X, along with mobility models.

Chapter 5

Wireless Communication

Car-2X can be implemented on any of the IEEE and other protocols depending on the intension. Quite surely these protocols have to be wireless, but can be based on either infrastructure-based or infrastructure-less. The below image provides a rough hierarchy of existing protocols.

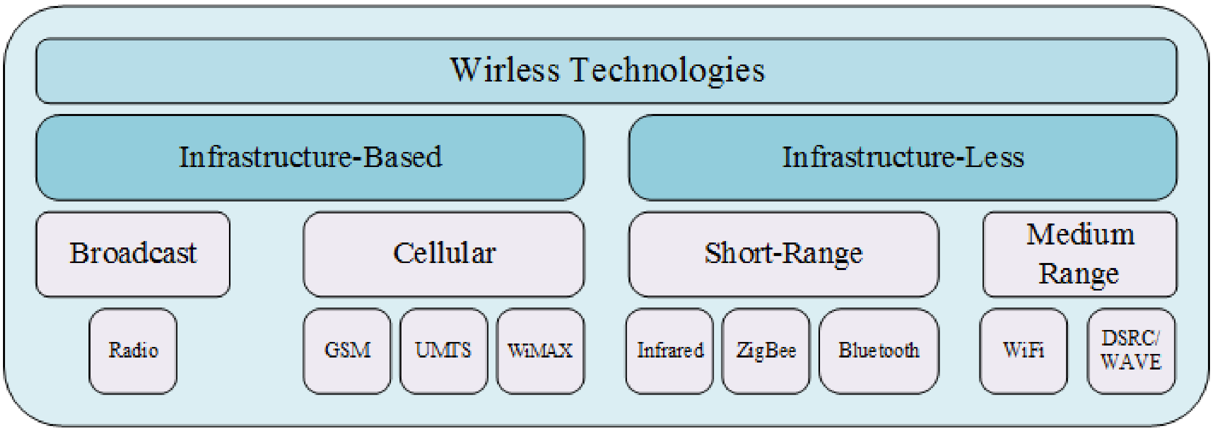


Figure 5.1- Hierarchy of Wireless Communication Technologies

Application of technologies depends on the deployment [11]. The CVIS [12] project has standardised in the ISO TC204 Working Group 16, an architecture- Communications Access for Land Mobiles (CALM) (retired name- continuous air interface, long and medium range). Its intension is to allow applications to communicate using any wireless communication protocols and be able to migrate to other technologies if required.

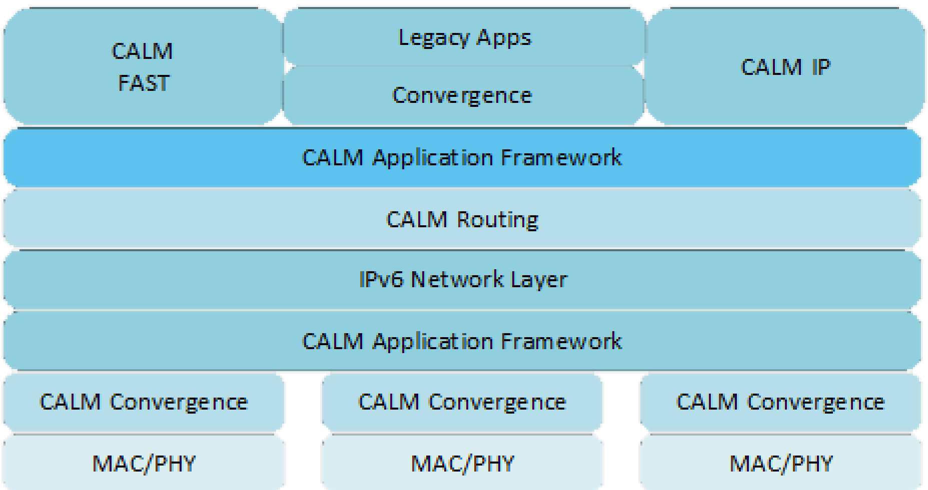


Figure 5.2- CALM architecture overview [12]

Wireless Access in Vehicular Environment (WAVE) based on IEEE 802.11p and cellular communication based on Universal Mobile Telecommunications Systems (UMTS), are two communication technologies that are based on CALM and recommended for both safety and comfort applications, respectively.

5.1 IEEE 802.11p and WAVE

Resulting protocols from the ASTM group's work on physical and MAC layer of the Dedicated Short Range Communication (DSRC) band, were drafted based on IEEE 802.11a standard. This later was amended to make the 802.11p standard.

This development is the basis of all vehicular communication technologies, which are specified in the IEEE 1609 protocol called Wireless Access in Vehicular Environments (WAVE) [13].

The WAVE defers many short-comings that previously hindered technology from working in the vehicular communication field:

- The problems faced in transmission range and speed from the multi-path propagation effects
- The problems faced in transmission range and speed from Doppler shift
- Parallel operation in infrastructure and ad-hoc modes
- Base station association takes long
- No concrete security mechanism for distributed system
- Quality of Service (QoS) mechanisms are not standardised

A WAVE system consists typically of RSUs and Onboard Units (OBUs). RSUs are generally installed street light poles, or traffic signals, or even be an independent unit. OBUs are mounted in automobiles. While OBUs function when the vehicle is in motion, the RSUs don't. WAVE units operate independently, and exchange information over a decided radio channel also known as control channel(s) (CCH). These WAVE units may also organise themselves into networks called WAVE Basic Service Sets (WBSS). WBSSs consist of either only OBUs or a mix of OBUs and RSUs. Members of a certain WBSS exchange data over one of the many radio channels, known as service channels (SCHs). WBSS connects to a wide-area network through portals.

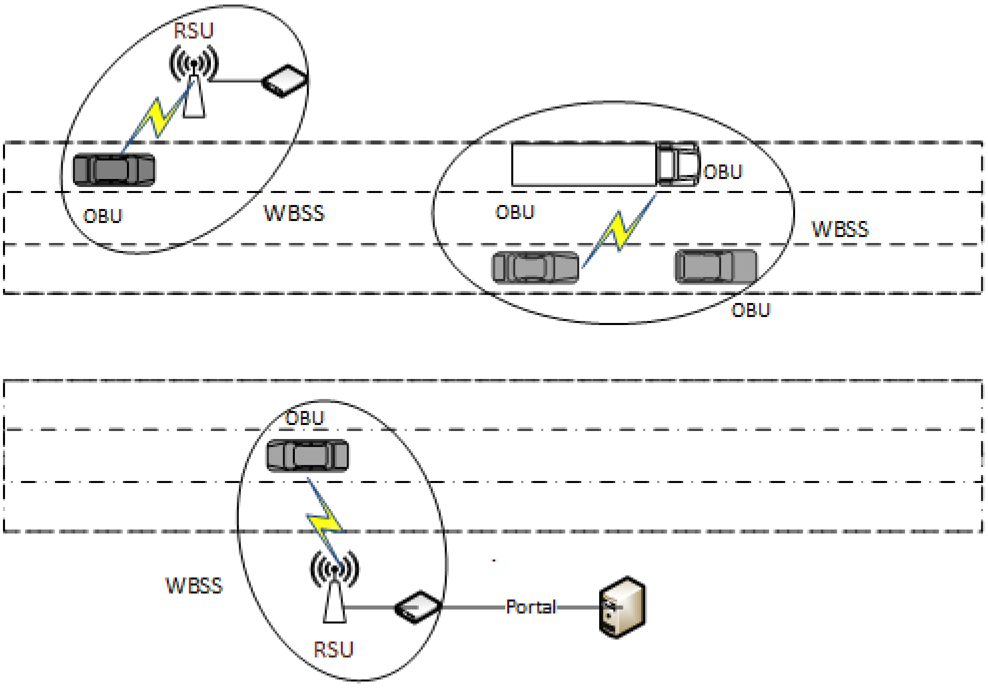


Figure 5.3- WAVE system with OBUs and RSUs and makeup of WBSSs

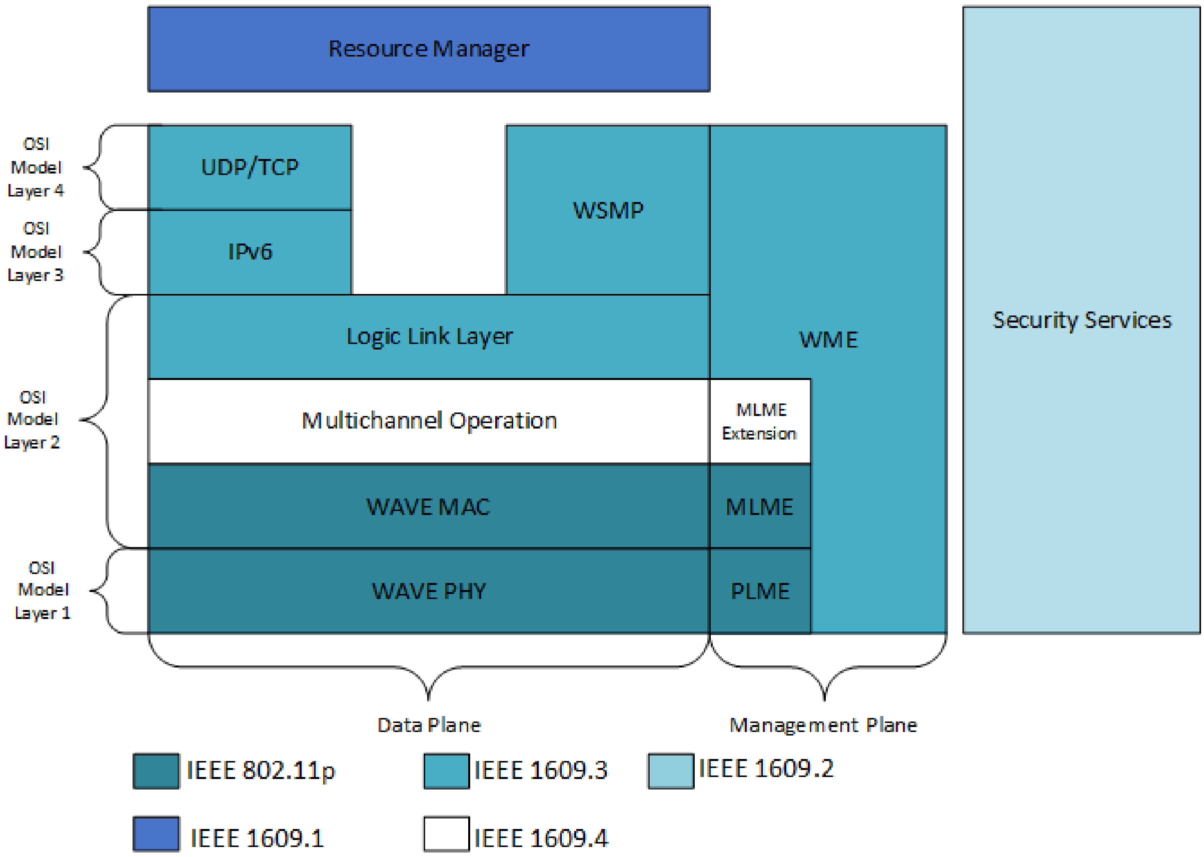


Figure 5.4- WAVE communication stack with standard of each layer. Security Services and Resource Manager do not fit into OSI model easily. [13]

The above problems are addressed in the WAVE in the following manner:

The physical layer operates on Orthogonal Frequency Division Multiplexing (OFDM), which is adapted to vehicular communication i.e. channel is 10 MHz instead of 20 MHz [14]. The DSRC band is planned to be used by allocating seven individual 10 MHz channels for communication.

The CCH, is restricted to the transmission of control and safety information. Four surrounding channels are SCHs. These channels may be used to communicate non-safety messages. But they are used only after verifying their use on the CCH. There are two extra channels at the top and bottom of the spectrum and are reserved for any special situation.

The link layer allows operation in WAVE mode [15], which allows the stations to address packets to Basic Service Set (BSS), allowing receiving stations to process packets regardless of their associated BSS.

The stations now may operate on WBSS instead of BSS. The WBSS does not require any data exchange to join it, as this decided locally and only communicated to the radio stack's lower layers. This is followed by periodic and optional broadcasting of On-demand Beacons advertising WBSS. A channel synchronisation ensures that members of that WBSS are utilizing the corresponding SCH during a common time interval (SCH interval).

Two mechanisms fulfil QoS conditions: first, data may be transmitted on multiple channels, the CCH beacons coordinate their use. These beacons may be sent only when stations are listening on the CCH, as there is no guarantee that stations will be equipped with many transceivers, otherwise safety or control messages may be missed. The CCH intervals may be defined, in sense that the station listens on the CCH for a certain designated period of time, for instance, the default setting is that stations tune to CCH for the first 50 ms of 100 ms slot.

Second, WAVE authorises the use of Enhanced Distributed Channel Access (EDCA) [13], used to coordinate channel access. The EDCA parameter must be set and used for all WAVE devices while in CCH mode. A high priority channel may access the EDCA by a parameter in the WAVE frame. Coordination of the channel is based on Coordinated Universal Time (UTC). This ensures the CCH monitoring by all the WAVE devices during common time interval (CCH interval). The sync interval is the sum of CCH and SCH interval.

WAVE supports two stacks: Internet Protocol 6 (IPv6) and the other WAVE Short-message Protocol (WSMP), this accommodates high-priority communications and the less demanding Transmission Control Protocol/User Datagram Protocol (TCP/UDP). WSMP allows the device to transmit short messages and directly control some parameters of radio to maximise the chances that all devices tuned will receive messages in time.

Based on the above operations of the WAVE stack, it is an integrated solution to wireless communications needs of Car-to-X systems. Although, it is much shorter ranged than the WiFi. Hence, a system that is purely operational on WAVE, is dependent on penetration rates of equipped vehicles or infrastructures.

5.2 UMTS

Cellular networks offer an alternative to the Car-to-X technologies. It avoids the installation of additional infrastructure. Universal Mobile Telecommunications Systems (UMTS), is a 3G mobile telecommunications technology. UMTS is developed and standardised by Third Generation Partnership Project (3GPP), a consortium of European Telecommunications Standards Institute (ETSI). It is a set of standards.

Data transmission is based on the concept of transport channels. They are mapped to physical channels. It is then, multiplexed into a data stream. Sommer et al. [16], have researched on UMTS and have presented here three channels that suit Car-to-X applications:

1. Dedicated Channel (DCH)

In a UMTS system, data will be transported via the DCH, which is present both in the uplink and downlink. This channel should be established for each mobile device prior use for transmission and takes resources to guarantee collision freeness.

Proper functioning of this wideband Code Division Multiple Access (CDMA), requires closed-loop power control to be performed between mobile equipment and base station. Devices keep adjusting the transmit power in order to balance the strength of received signal and the interference caused. This depends on the exchanged management data.

DCH needs to be quite well maintained.

2. Random Access Channel (RACH).

It is a common uplink transport channel, and is received from all devices. It may be used to establish connections to the base station and for short data packet transmission. This random access is modelled on slotted ALOHA with fast acquisition by Acquisition Indicator Channel (AICH). Here, an open-loop power control is required, as opposed to the DCH which used closed-loop and is therefore less accurate. This also increases collision risk.

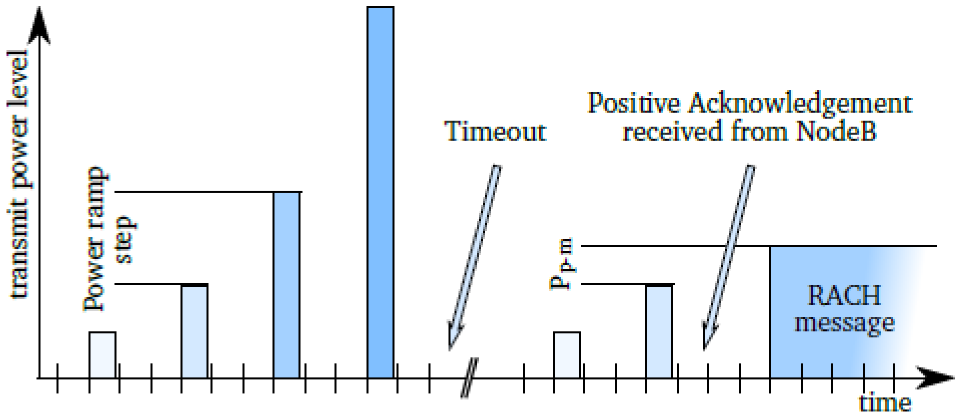


Figure 5.5- Access procedure on the UMTS RACH: one failed, one successful attempt. [16]

The above figure, shows the required process for coordinating access to RACH and determining the transmit power level to be used for data transmission. The equipment sends one of the 64 available RACH preambles. If it does not receive any response from the network/base station, it increases its power and sends the RACH preamble again, until it receives a positive acknowledgement or till the allowed number of repetitions is exhausted. If no positive acknowledgement form the base station is received by the maximum repetition, then the physical layer procedure terminates, and the process begins from the MAC layer.

As the slotted ALOHA scheme is used, preambles will be transmitted periodically, at fixed points or access slots. One radio frame ($10ms$) consists of 7.5 access slots, $20ms$ consists of 15 slots, about $^4/_3\ ms/slot$. A sub-channel is the concrete location of the access slots that a service can transmit on. A set of sub-channels is assigned for service to the success service class, and a service class may contain as many sub-channels. This reduces the waiting time between two access attempts [17].

Owing to its open-loop power control, no additional management overhead is incurred, but the access procedure is to be repeated after a burst of data to retain the transmit power level.

3. Forward Access Channel (FACH)

It is a transport channel, which forms the downlink. Used for downlink signalling and small quantities of data. Data transmitted by the base station on the FACH will be received by all the mobile nodes within the access area, and hence used for multicast message transmission on the Multimedia Broadcast Multicast Service (MBMS), but only if network supports this.

Downside of this channel is that the location of the mobile equipment needs to be known to the base station for it, to be able to transmit data to the device. The base station manages the time slots and hence no coordination of channel access needs to happen.

UMTS networks are suitable for exchange of data in a Car-to-X setting as the delay in transmission is independent of the distance between the nodes. There is unlimited coverage in most urban areas in the world of UMTS networks. Regardless of its efficiency, delays are not as low as that of short-range radio communication networks [18] (ex. WAVE). This requires considerable more scientific research. High data load, in terms of, the low delay incurs high network resource costs. Also, multiuser issues in the UMTS area of Car-to-X systems still needs research.

Chapter 6

Protocols

Many Car-to-X communication technologies have been designed, based on existing wireless technologies and coupled with hardware, a wide range of applications can be designed [19].

IVC technologies have two distinct advantages

- Direct Communication- cars communicate with each other directly, the communication delay is low. Also, this allows for a good communication in time critical situations. Also, communication is possible in areas where cellular networks don't reach, the network for communication is established by the vehicles on their own.
- No service cost- As the cellular network providers\network providers are uninvolved in IVC application, there is no cost involved.

IVC applications can be further classified [20]:

1) Comfort Applications make the drive for the travellers comfortable or improves the comfort level. Also, optimises the drive route, avoiding traffic or lanes that are congested. Some examples are, traffic information, weather information, locating the nearest fuel station, locating restaurants, providing access to internet, etc.
2) Safety Applications are developed with the intension of improving the safety and security of the passengers. Vehicles may exchange safety information such as emergency warning, lane-changing, road condition warning. The information exchanged will alert the driver and activate the safety systems. Delays are not preferred in these application areas, and hence the messages are preferably few and short.

With a diverse application area, two types network applications have been developed- single hop [21] and multi hop [22], with the former focussing on safety applications and the latter on traffic congestion avoidance and dedicated information systems. Comfort application are being developed swiftly with the improvement in 3G communication systems. Focus is now shifting to development of application for Traffic Information System (TIS).

This section discusses briefly three prevalent but different approaches to communication.

6.1 Ad Hoc Routing

Mobile Ad-hoc Network (MANET) routing techniques have been successful and now are being applied to Car-to-X called Vehicular Ad Hoc Network (VANET). Three classes of MANET protocols are: proactive, reactive and hybrid.

Proactive Routing Protocols define protocols for the exchange of topology data. These protocols maintain updated routing information and is used to forward data packets to their destinations. Depending on the protocol used network maintenance is required.

Reactive Routing Protocols define protocols that do not keep updated information on topology so as to avoid overheads, but instead search for routes only when data is required to be transmitted. Keeping updated information consumes network resources and is high maintenance.

Hybrid Routing Protocols were defined to take advantage of both proactive and reactive protocols. For instance, they use the ability of proactive protocols to deliver data quicker as routing information is always available, and also employ the techniques of reactive protocols in trying to keep the cost low.

Maintenance of full network topologies seems to be more of a cost factor as it causes overheads, as opposed to the delays caused. One of the protocols that has extensive focus on it is the reactive protocol Dynamic MANET on Demand (DYMO) [23]. Routes are computed on demand i.e. as and when required. It does not support unnecessary HELLO messages and operates on sequence numbers attached to the packets, so as to exploit the loop freedom. It operate as follows.

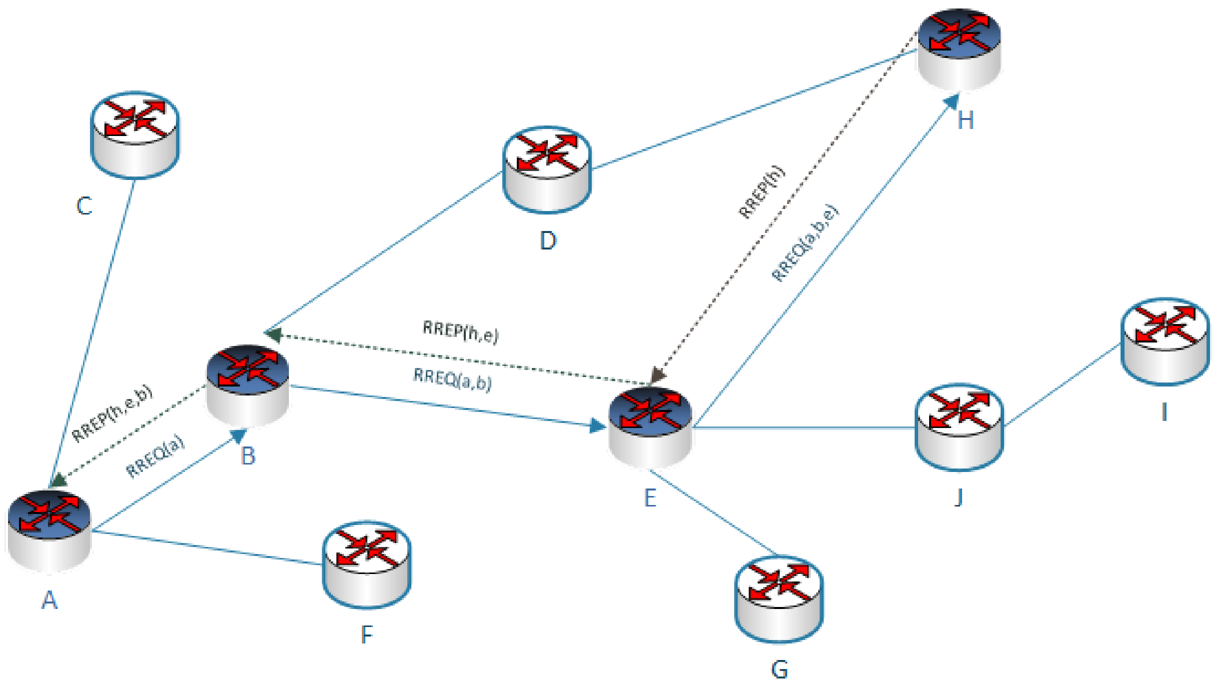


Figure 6.1- DYMO Route Discovery

DYMO [24] creates and enhances unicast routes in IPv4 and IPv6 network in the following manner:

1. Three messages are implemented, Route Request (RREQ), Route Reply (RREP) and Route Error (RERR). RREQ is used to discover route to destination. RREP establishes valid route to the destination by routing the intermediate nodes between them and RRER indicates an invalid route to the source node from any intermediate node. Each node is to maintain a sequence number, an unsigned unique number, which ensures delivery of packets and avoids loop free routes.
2. The source broadcasts an RREQ when it no route entry to the destination node. If an intermediate node has an entry, it responds with an RREP message to the source, or else it broadcasts the primary RREQ message, and attaches it address to this RREQ. The intermediate nodes that broadcast the RREQ make note of the backward path of

the route. Each intermediate node also add a sequence number to the RREQ. While the source awaits an RREP, the destination replies with an RREQ message and the same path accumulation process happens backwards to the source. Nodes discard any stale packets, if an incoming packet has the same or inferior sequence number, then the packet is discarded. Packets with superior sequence number are updated. Also, nodes avoid loops by dropping packets having the sequence number the same as the node sequence number. If a node is on low energy then it has the choice of not participating in the routing protocol, but updates itself with the incoming RREQ messages.

3. Nodes maintain a route by broadcasting RERR messages whenever there is a break in the link to other nodes. This message is broadcasted only to those nodes which are concerned with this disconnection of the link. The receiving node, then discards the entry and initiates the route discovery process to the source/destination

6.2 Flooding: Distributed Vehicular Broad-CAST (DV-CAST) Protocol

Multi-hop data broadcast, known widely as Flooding, disseminates messages over wide-areas in mobile networks [25]. The following is a very simple mechanism of flooding:

> An idle node receives a fresh packet. It broadcasts the packet almost immediately.
> If the node receives a duplicate packet, this packet is received but no action taken.

Flooding protocol broadcasts more packets than is required, to ensure complete dissemination.

1. The DV-Cast protocol was designed under the following assumptions:
   - Unavailability of infrastructure in the considered network
   - Every vehicle is equipped with Global Positioning System (GPS) and a wireless communication device
2. Use per-hop routing, i.e. use local connectivity to decide on routing
3. Each vehicle monitors its local connectivity, so as to determine its operational state at the time of packet arrival.

As there are more packets broadcasted, packet delay, loss and access delay, increase. Hence, better approaches need to be designed so as to not re-broadcast every received packet, but to go ahead selectively.

A mechanism was developed by Wisitpongphan et.al [26], using an estimation of distance between source node and sender node, i.e. no requirement of exchange of topology information or even the maintenance of the same.

The distance between the source and sender nodes is expressed as a relation to maximum distance $0 \leq \varrho_{ij} \leq 1$. To estimate ϱ_{ij}, GPS data on the calculated distance D_{ij} of the sending node, or on the measurement of the received signal strength (RSS) of incoming transmission,

$$\varrho_{ij} = \begin{cases} 0 \text{ if } D_{ij} < 0 \\ \frac{D_{ij}}{R} \text{ if } 0 \leq D_{ij} < R \\ 1 \text{ otherwise} \end{cases} \qquad [26]\ (\text{Eq } 1.1)$$

Where, R is the approximate transmission radius.

Based on the GPS distance data, and based on the RSS, we get the relation

$$\varrho_{ij} = \begin{cases} 0 \text{ if } RSS_x < RSS_{min} \\ \frac{RSS_{max} - RSS_x}{RSS_{max} - RSS_{min}} \text{ if } RSS_{min} < RSS_x < RSS_{max} \\ 1 \text{ otherwise} \end{cases} \quad [26]\ (\text{Eq } 1.2)$$

Based on the above techniques, the following method is devised. After receiving a fresh packet, the node waits for a short interval. After, this interval elapses, the node re-broadcasts the packet with a probability p. Were the node to choose to wait for a longer period, it does so, and puts off re-broadcasting. It does this, so as to confirm a minimum of one re-broadcast from another node, thus preventing a cease in message transmission. This works better over larger distances instead of shorter.

There is one drawback. If any part of the network is disconnected, even temporarily, then the approach will not reach those nodes. Thus flooding approaches are combined with delay. This is known as DV-CAST.

Here, the nodes periodically, broadcast Hello messages, which contain GPS data. Each node that receives this message, stores the data in three tables: one storing neighbour nodes driving in the same direction and in front of itself, the other stores the neighbouring nodes driving in the same direction and behind itself and the third stores the neighbour nodes driving in the opposite direction.

From the data from these tables, the node decides for or against flooding, or whether the message will be queued for re-broadcast.

DV-CAST also deals with reaching nodes on the opposite lanes i.e. sparsely connected network.
To assume, sparsely connected network, the DV-CAST applies the following algorithm:

- If no neighbour nodes can be reached, driving in either direction, the node assumes Totally Disconnected conditions
- If only neighbours driving in the opposite direction are found, then the node assumes sparsely connected conditions. The neighbour node then ensures delivery of the message to the destination of back to a vehicle driving in the original direction
- If only nodes driving behind the car are found, but the message is to be transmitted to the back or front, then the node assumes well connected conditions and broadcast suppression techniques are adopted.

Chapter 7

Tools

7.1 OMNet++

OMNet++ is an application that acts as a platform for simulation of networks. It is built on Eclipse and on C++. It provides libraries which can be used to create network simulators widespread in sense that it includes wired and wireless ad-hoc, Internet Protocols, queuing networks and performance modeling. Also, real time simulation and database integration is also possible.

OMNet++ provides three layers of abstraction. The modeling language Network Description (NED), a component based language, also provides interfaces, that can be used as an interim, which will have to 'implement' the actual module and can be chosen during run-time. There are two types of modules, simple and compound. Simple modules are active modules, which implement the behaviour of a single component. Compound modules are formed from 2 or more simple modules. Hence, they implement the behaviour of a system. Parameters for the declared variables may be assigned here or in the configuration file, often named omnet.ini.

Alongside the NED language, there is the second layer of abstraction, the topology. OMNet++ allows the developer to construct a graphical topology of the desired system. Connections between simple modules and compound modules may be established here also, but the assignment of the ports and the messages being exchanged may be specified by the NED language. The topology provides the advantage of having a blue print of the system conceptualised.

The third layer of abstraction is the object oriented C++ language. The developer may describe the behaviour of the simple modules using C++ language.

Communication between the simple modules takes place between interfaces called gates. The gates between the simple modules have to be compatible in order to establish a connection.

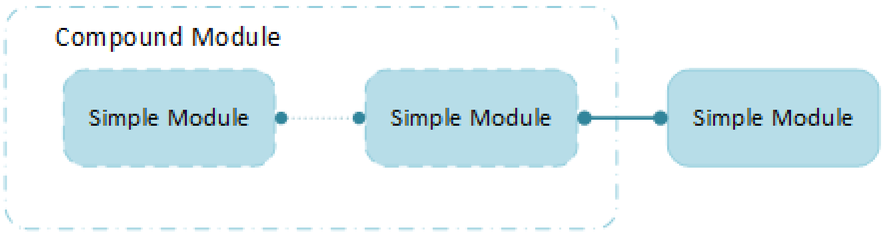


Figure 7.1- Basic topology hierarchy

7.2 Simulation of Urban Mobility (SUMO)

There are four classes of traffic flow models, macroscopic, microscopic, mesoscopic and sub-microscopic, mesoscopic and sub-microscopic which are differentiated from each other by the level of simulation detail.

Macroscopic is the basic entity. Simulation of vehicle movements, based on the assumption that the drive behaviour depends on both the vehicle's physical attributes and the driver's behaviour, is the microscopic. The mesoscopic simulates vehicle drive based on queuing techniques and between these queues. Sub-microscopic allows very detailed simulation, which dive into vehicle substructures, but require more computation time.

SUMO is a tool designed based on microscopic simulation techniques. It is an open source software, designed on C++, which enables simulation and testing of traffic algorithms. Not only does it implement a complex network of roads and walk ways, but also the signals and other traffic flow and control methods, which provides a detailed view and evaluation of traffic flow.

SUMO requires a set of software tools that generate a set of xml files which finally form the SUMO network. This network file provides information regarding traffic, roads and junctions, and vehicles. Not every map type is supported by SUMO, hence there are converters available, that assist in converting the map types into SUMO supported.

One of the main tools is NETCONVERT. The map on which traffic is to be simulated is first downloaded from the open source 'openstreetmap' which will be in an xml format. This xml file is then processed with NETGENERATE, which also produces an xml file with specific network description. It provides a cartographic projection valid for this network, edges, traffic signal logic, junctions-plain and then internal junctions. Connections and then circular junctions. All the measurements are in metric in this Cartesian projection. This SUMO compatible map can now be opened and viewed in the SUMO-GUI. To be able to have simulated vehicles move around on the map, a second tool DUAROUTER is required. DUAROUTER generates an xml document that details vehicles trips and routes, which is called traffic demand. Trips define the starting and the destination edge and the travel duration. Route is the expanded trip, includes all the in-between travel edges. DUAROUTER computes vehicle route using shortest path method, assigns using dynamic user assignment when called iteratively. It also rebuilds routes when there exists connectivity problems in route files.

Polyconvert, is a tool that is required to convert geometrical shapes into SUMO-GUI compatible representations. It is capable of applying different attributes to the shapes in dependence of their "type". One of its major uses is to apply a projection on the given shapes, and also reduce the sizes of the shapes if they are too big for the SUMO-GUI projection. If any of the read shapes do not correspond to those of the tool, then default parameters are applied to the shapes, so that the projection can take place.

Traffic Control Interface, abbreviated as TraCI, is a protocol designed to give access to SUMO to external applications. Access to SUMO for external application is gained by connected to it via the Transmission Control Protocol (TCP) through a particular designated port.

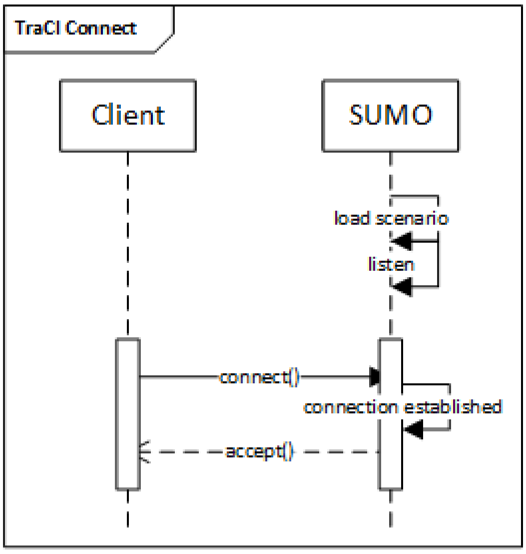


Figure 7.2- TraCI Connect signal exchange

The client sends commands to SUMO in order to control the simulation, to influence single behaviour or to ask for environmental details. SUMO responds with an answer to each command and additional details depending on the command given.

The client has to trigger each simulation step in SUMO using the TraCI commands Simulation Step command. If any subscriptions, the subscribed values are returned.

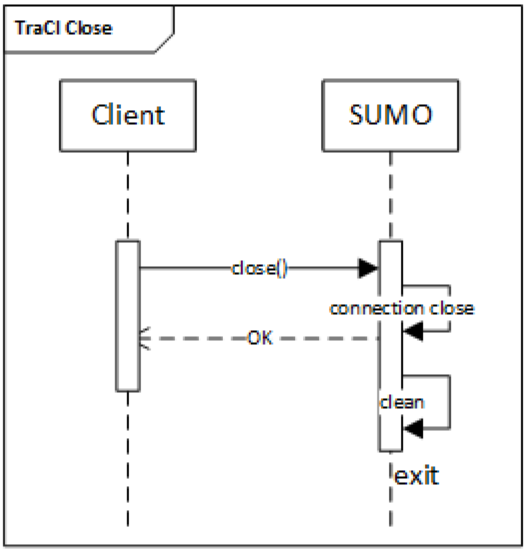


Figure 7.3- TraCI Close signal exchange

The client is responsible for disconnecting the session using the close() command.

7.3 MiXiM

Though OMNet++ serves as a good base for the simulation of many protocols, it falls short of support for wireless communication. This is provided by MiXiM [27]. It brings together and extends many simulation frameworks for wireless and mobile simulation. Along with models of wireless channels, connectivity, mobility models and models for obstacles, it also provides implementation of these protocols at the Medium Access Control (MAC) level. The infrastructure provided my MiXiM can be divided into five categories,

Environmental Models only those portions of the real world that are relevant to the simulation are to be projected. Obstacles, that hinder communication, are good examples.

Connectivity and Mobility The distance and the position of nodes communicating needs to be considered. When moving, the increase or decrease in the distance between the communicating nodes and its influence on the communication itself needs to be recorded and simulated accurately.

Reception and Collision Due to the movement of nodes in a wireless communication, there may be an impact on the reception of communication. The handling of reception needs to be considered.

Support of Experimentation This is necessary for evaluation of the results in comparison with the real state or ideal state of communication.

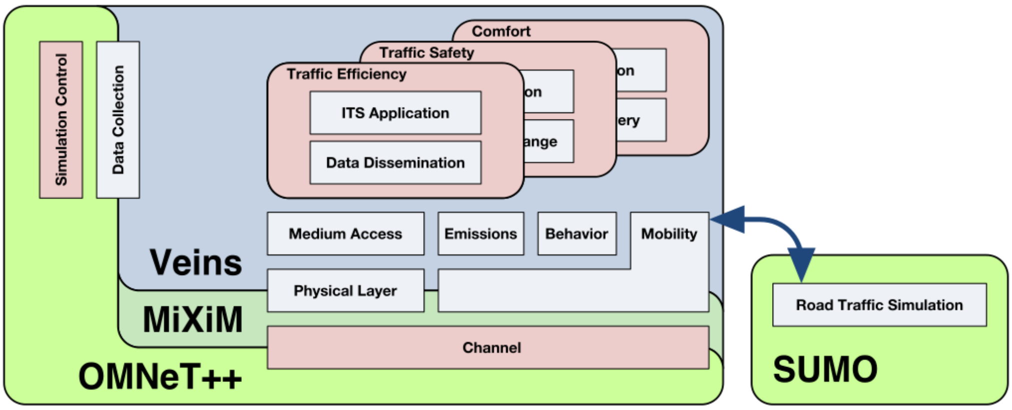
Library A protocol library which implement a variety of protocols, so that new communication techniques may be tested and compared with those already implemented.

MiXiM also provides the advantage in that, it already has implemented models for radio wave propagation delays and loss, due to the interference of free space or obstacles along with MAC protocols.

7.4 Vehicles in Network Simulation (Veins)

VEINS, is an open-source Inter-Vehicular Communication simulation framework, designed in OMNet++. VEINS framework connects OMNet++ with SUMO bi-directionally, which allows a dynamic simulation.

The Veins framework includes a Mixim version for its own use, and this project makes use of this availability.



6

Figure 7.4- Modular framework of Veins [7]

As described, VEINS consists of two simulators, OMNet++ and SUMO. OMNet++ for network communication simulation and SUMO for traffic or vehicle roads traffic simulation. OMNet++ communicates with SUMO via a python script, while the scenarios that need to be simulated communicate with SUMO via the TraCI interface. This is the bi-directionally coupled simulation.

As the road traffic is simulated in VEINS, it updates OMNet++ with this information, where the TraCIScenarioManager updates the position of the nodes using the TraCIMobility module. The update occurs periodically. .

[6] Taken from the website http://veins.car2x.org/documentation/modules/
[7] Taken from the website http://veins.car2x.org/documentation/modules/

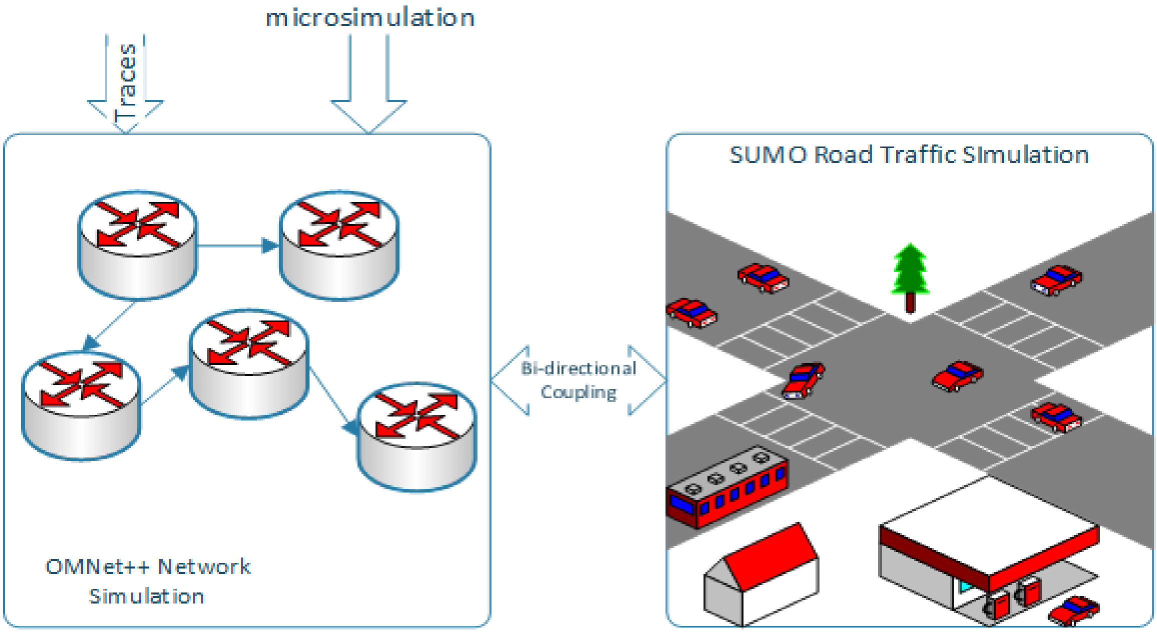


Figure 8.1- Setup for the simulation of Inter-vehicular Communication

Chapter 8

Approach

1. Wireless Access in Vehicular Environment\DSRC IEEE 1609 AND IEEE 802.11p

Standards have been developed for Dedicated Short Range Communication (DSRC), which enable the exchange of messages between vehicles and Road Side Units (RSUs). This project focusses on the communication of messages between both the vehicles and RSUs, and inter-vehicular communication.

The Veins framework is further extended by a model for IEEE 1609.4/802.11p, which allows simulation to evaluate the properties of radio technology. This also overcomes the problems of the MAC/PHY, which shows abnormal results which reduce the reliability of the results.

Hence, a favourable model is one that also features alternating access – change frequencies at regular time intervals, various channel access systems, with both virtual and physical contention, and bit rates along with bit error probability.

Veins framework has an extension to the existing IEEE 802.11 based on the changes to it as per the IEEE 802.11p.

2. WAVE Implementation

WAVE operates on various wide non-overlapping channels of 10 MHz. There exists just one Control Channel which is tuned into for 50 ms out of 100 ms, but there are many Service Channels which can be tuned into for transmission during the rest of the 50 ms. An additional 4 ms guard interval is present at the beginning of every slot to tackle the synchronization error along with frequency switching of the radios.

One of the four possible Access Categories (ACs) is assigned to each packet and this affects the internal virtual and the external packet as per [28] [29].

Parameter	Value
CCH/SCH slot length − Guard Interval	50 ms − 4 ms
Contention Window aCW_{min} and aCW_{max}	15 and 1023
SlotTime	13 µs
SIFS	32 µs
Bandwidth	3 Mbit/s . . . 27 Mbit/s

Listing 8.1- WAVE Settings [28] [29] [30]

The MAC layer employs the Enhanced Distributed Channel Access (EDCA) scheduling, it has one queue per AC and channel type, totalling to eight queues, each being controlled by one EDCA Function (EDCAF).

Each queue has a back-off counter which is controlled by EDCAF. EDCAF controls the transmission initiation of packets as well. When the back-off counter for a queue is 0, a packet is dispatched, given that the physical channel was idle for a minimum of one Arbitration Interframe Space (AIFS), the length of which is derived from the AC of a packet, hence, higher the AC, higher is the priority of channel access. Contention Window (CW) is the period in which a node may transmit data at any time.

Parameter	AC_BK	AC_BE	AC_VI	AC_VO
CW_{min}	aCW_{min}	aCW_{min}	$\frac{aCW_{min}+1}{2} - 1$	$\frac{aCW_{min}+1}{4} - 1$
CW_{max}	aCW_{max}	aCW_{max}	aCW_{min}	$\frac{aCW_{min}+1}{2} - 1$
AIFSN	9	6	3	2

Listing 8.2- Contention Window Parameters for ACs, AC_BK-Background, AC_BE-Best-effort, AC_VI-Video, AC_VO-Voice

The back-off mode [31] is activated when,

a) the channel turns busy when the back-off counter is 0,
b) a packet of a queue was transmitted successfully,
c) a packet with higher AC was ready to be dispatched at the same time,
d) a different channel was active or a packet was ready to be transmitted in the guard interval,
e) No ACK is received.

To deduce the back-off time, a random number between the interval [0, CW] and the slot length of IEEE 802.11p are multiplied [31]. The following cases apply to CW changes:

a) CW remains unchanged when the back-off counter is 0
b) CW remains unchanged when a packet was transmitted successfully
c) CW is doubled or set to CW_{max} if a packet with higher AC was ready or dispatched at the same time
d) CW is doubled or set to CW_{max} if a different channel was active or a packet was ready to be transmitted in the guard interval
e) CW remains unchanged when no ACK is received

The back-off counter is reduced by the EDCAF at the slot boundaries. This happens after one AIFS, as the channel turns idle and at every passed slot afterwards.

The OFDM PHY layer is also modelled here, and to be accurate with the bit error probabilities, the bit error models that fit in the real world measurements are adopted.
To compute the decision of whether a packet can be decoded or not, Signal to Noise + Interference Ratio (SNIR) from MiXiM interference computation is applied. Hence, the bit error probabilities for the header and payload are calculated separately from the Protocol Data Unit (PDU).

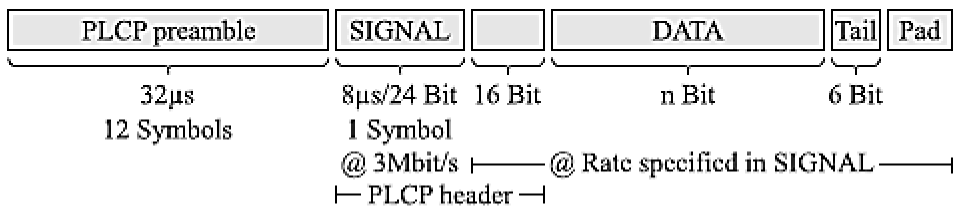


Figure 8.2- PDU Packet format in IEEE 802.11p [31]

Various data rates of the PLCP signalling are taken into account and data rates of the payload as well. This leads to a practical design of the physical layer, which allows for a simulation of the IVC applications on the MAC layer.

Chapter 9

SUMO and GeoData

9.1 Pre-Analysis

One of the objectives of this study, is to evaluate the possibilities of replicating the events and communication, designed in the simulation, in a model world, using model cars with communication modules attached to them. Hence, a scenario that would be suitable for a model study is designed. Along with the above, this project also evaluates various traffic scenarios and studies this simulation focussing on the vehicle-to-vehicle and vehicle-to-infrastructure communications. There are two main parts to this designed.

Firstly, a map with the required model scenario is to be developed in SUMO, using its accompanying tools. The number of vehicles running on the designed network is decided in advance, and deployed in the network files. The map is compiled based on the model scenario which is to replicate those implemented in SUMO, hence, a simple scenario is developed.

Secondly, a suitable algorithm for vehicle-to-vehicle, and vehicle-to-infrastructure algorithm is designed. This is developed in OMNet++. Once, deployed, the algorithm developed is to communicate this message to the model cars with the communication modules attached to them.

Veins framework consists three elements. A concise version of MiXiM framework, modified to suit Veins. A python script that starts SUMO, that connects SUMO to Veins framework in OMNet++ and proxies TraCI messages, and OMNet++ modules that implement the TraCI mobility messages and protocols characterised by node mobility[8].

In order to transmit the messages between the vehicles and the vehicles and infrastructure, selective algorithms have been developed and analysed. The considered algorithms are to be optimised for the intended model world. Messages will be exchanged when the vehicles are in range of each other or that the vehicles are in range of the infrastructure. Obstacles are considered in one environment and not in the other, which accounts for the feasibility of the algorithms.

[8] TraCI is the interface between SUMO and OMNet++.

9.2 Generation of the SUMO compatible maps

As explained earlier, SUMO is equipped with many tools attached to it. All the network files that need to be generated will be in the XML format. These XML files are not be handled by the user alone, but instead to be generated using the tools specified. But on the other hand, it would be easier to edit the network files with the desired roads and the connections between them.

9.2.1 OpenStreetMap Geo Database

The quality of the simulation, depends on the quality of the data. Many sources of data for the simulation, cost very expensive, hence SUMO support was extended to OpenStreetMap.

To generate a SUMO compatible map, first an OSM map of desired city and dimensions are downloaded from the open-source platform[9]. The open-source platform is built using maps from aerial imagery, GPS devices and low-tech field maps. The database of building the maps uses local knowledge. There are five map types available on this platform, Standard, Cycle Map, Transport Map, MapQuest open and Humanitarian. For this project, a Standard map is used, as it contains the direction of traffic flow on each road and also the junctions. Also from this map, using the various SUMO tools, the traffic density and the type vehicle may also be derived. A map of desired dimensions may be downloaded. For project, 3 different maps of the city Chemnitz, of varied dimensions are studied.

The downloaded map is not compatible with SUMO applications. Hence, the maps need to be configured and compiled to SUMO compatible files. It is essential to know that the maps, themselves will not be converted, but only information extracted from them to XML files, to finally build the network of roads and infrastructure in SUMO. The following tools come into play when extracting information of the network from the downloaded maps,

NETCONVERT reads the map files from the source and generates and XML document road networks which may be later imported into SUMO to build the required map. It is possible to create the network using simple projection, and also load traffic lights from other formats,

NETGENERATE creates road networks when may be read by other SUMO tools. It also allows to build Grid, Spider and Random networks.

DUAROUTER computes vehicle routes using shortest path computation.

OD2TRIPS accepts Origin/Destination files and generates single vehicle trips.

JTRROUTER uses junction turning percentages and generates vehicle routes from demand definitions.

DFROUTER generates vehicle routes from the number and definitions of induction loops.

MAROUTER computes vehicle flows using origin destination matrices.

POLYCONVERT accepts a file with definitions of geometrical shapes and exports definitions that may be read by SUMO.

[9] www.OpenStreeMap.org

```xml
<?xml version="1.0" encoding="UTF-8"?>
<osm version="0.6" generator="CGImap 0.3.3 (31041 thorn-01.openstreetmap.org)">

        <node id="20466004" lat="50.8322616" lon="12.9469679"/>
        <node id="20466014" lat="50.8373432" lon="12.9357416"/>
        <node id="20466025" lat="50.8361769" lon="12.9422415">
                <tag k="highway" v="turning_circle"/>
        </node>
        <way id="3961141">
                <nd ref="20466004"/>
                <nd ref="20466014"/>
                <nd ref="20466025"/>
                <tag k="bicycle" v="yes"/>
                <tag k="foot" v="yes"/>
                <tag k="highway" v="residential"/>
                <tag k="motorcar" v="yes"/>
                <tag k="name" v="Uhlandstraße"/>
         </way>
```

Listing 9.1- Example code of XML OpenStreetMap data dump encoding information of three interconnected nodes by a road i.e., a way.

The semantics of this file are based on tags assigned to objects. This is represented as key=value pairs. There is a technical restriction on this denotation that two tags which are assigned to a single data primitive shall have the same key. Most text is arbitrary except for the preceding restriction.
This make OpenStreetMap data representation format very liberal, and expresses most data as relations, nodes and ways.

The representation of relations is thus very efficient and capable of producing a detailed 2D representation map from rightly tagged geodata. The frontend displays roads, concrete structures, public transport network and public properties i.e., parks, etc.

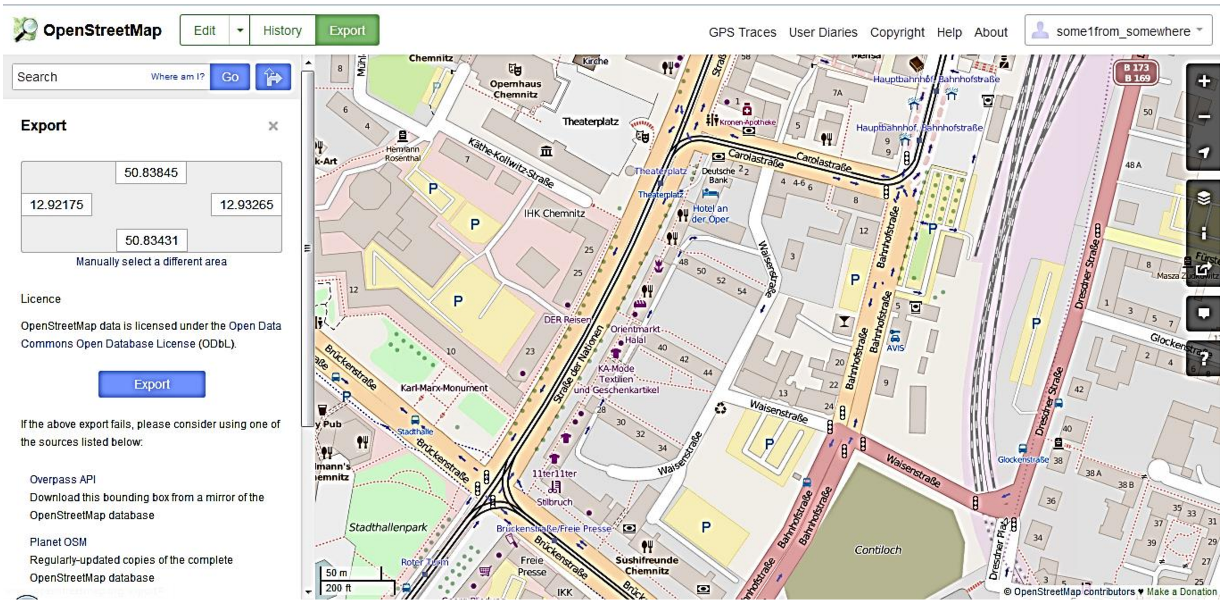


Figure 9.1- A shot of the city Chemnitz, Germany on OpenStreetMap web frontend

1. NETCONVERT

With the NETCONVERT tool we generate the following network file of roads. This example is supplementary to the listing above,

```
<edge id="-182248506#2" from="20466004" to="20466014" priority="4" type="highway.residential">
        <lane id="-182248506#2_0" index="0" speed="8.33" length="110.96" shape="2443.81,1944.02
        2549.48,1910.16"/>
</edge>
<edge id="-182248506#3" from="20466014" to="20466025" priority="4" type="highway.residential">
        <lane id="-182248506#3_0" index="0" speed="8.33" length="97.32" shape="2341.98,1976.95
        2434.57,1947.00"/>
</edge>
```

Listing 9.2- Sample data dump from a file generated using NETCONVERT

The above listing shows a network listing between the streets 20466004, 20466014 and 20466025. The streets are of type highway. The connections between the streets forms edges.

SUMO map projections are based on planar maps. Hence, a projection has to be chosen, and SUMO uses the UTM projection, which is a zone based variant, as we can see in the above example listing. It is based on the WGS 84 reference ellipsoid and data. The default values are not explicitly set, for instance, the example above has the maximum speed.

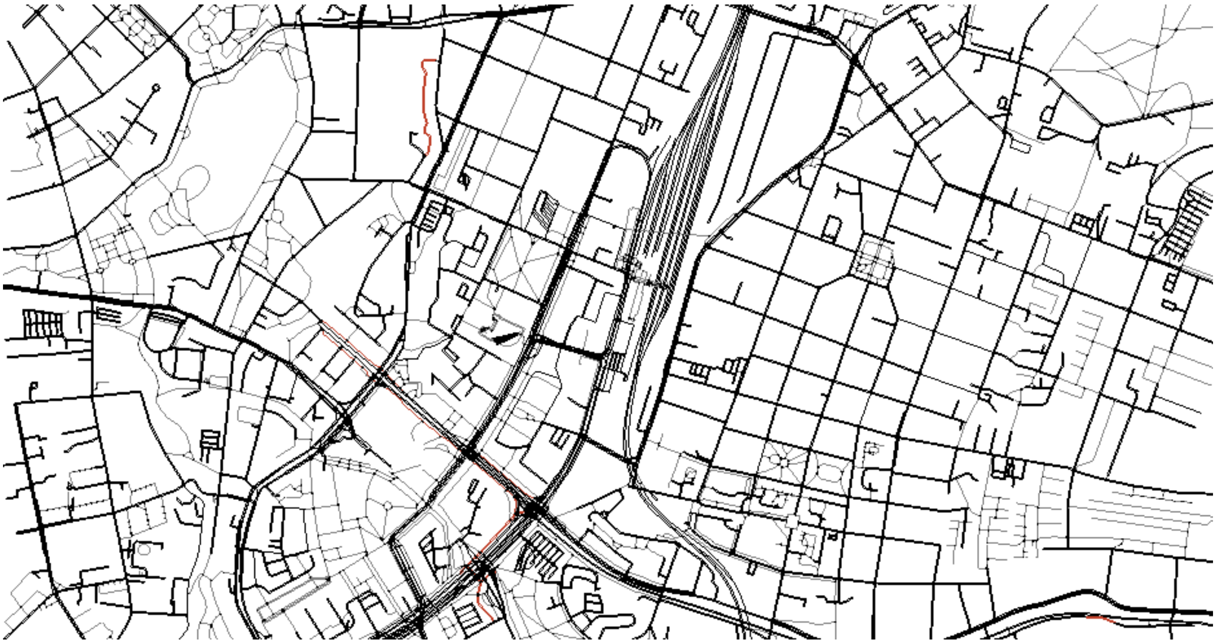


Figure 9.2- Road network created from the NETCONVERT file imported into SUMO

The listing also contains lane attributes such as speed and length. Every road segment will be represented by one edge element per direction. Junctions form nodes connected by these edges. NETCONVERT will only generate road networks, hence, to extract details of the road infrastructures, other tools need to be used to generate them for the simulation.

After generating the road network topology, it is possible to open the file in the SUMO GUI as shown in **Figure 9.2**, but no vehicles driving would be simulated as the routes, the vehicles have to drive on, have not been generated.

2. DUAROUTER

To generate the specification of the travel route of the vehicles, the tool DUAROUTER or randomTrips.py may be used.


```
<routes>

<vType id="vtype0" accel="2.6" decel="4.5" sigma="0.5" length="2.5" minGap="2.5" maxSpeed="14"
color="1,1,0"/>

<route id="route0" edges="20466004 20466014 20466025"/>

<flow id="flow0" type="vtype0" route="route0" begin="0" period="3" number="195"/>

</routes>
```


Listing 9.3- Sample data dump generated using DUAROUTER

The above XML notation denotes the vehicle type, the acceleration and the deceleration of the cars flowing on the mentioned edges. The maxSpeed denotes the maximum speed of each car. From the above specification 195 cars will be generated to flow on the route route0.

3. POLYCONVERT

Polygons are modelled in SUMO using the tool polyconvert XML element. This uses the configuration in one the linked files. Polygons are modelled based on the poly XML element linked in the configuration file. This file is then available to SUMO via the TraCI. A file called typemap is used to specify the types that are to be projected. It specifies additional information on the representation in SUMO.

The resulting file contains the concrete structures specified in the OpenStreetMap data dump.


```
<poly id="99511270" type="building" color="255,230,230" fill="1" layer="2.00" shape="544.54,1270.46
551.89,1258.62 539.83,1250.51 532.32,1262.62 544.54,1270.46"/>

<poly id="99511278" type="parking" color="184,184,179" fill="1" layer="-2.00" shape="584.72,1405.34
605.87,1408.90 605.17,1412.69 619.42,1415.52 620.28,1412.40 643.05,1416.33 645.83,1404.68
666.48,1409.74 660.82,1438.44 642.63,1434.91 643.42,1424.53 632.26,1425.79 583.16,1416.95
584.72,1405.34"/>
```


Listing 9.4- Data dump generated using POLYCONVERT

The following example shows a typefile.xml,

```
        <polygonType id="shop" name="shop" color=".93,.78,1.0" layer="2"/>
        <polygonType id="landuse.industrial" name="industrial" color=".82,.82,.80" layer="-11"/>
        <polygonType id="man_made" name="building" color="1.0,.90,.90" layer="2"/>
        <polygonType id="building" name="building" color="1.0,.90,.90" layer="2"/>

<polygonType id="highway" name="highway" color=".10,.10,.10" layer="-1" discard="true"/>
<polygonType id="boundary" name="boundary" color="1.0,.33,.33" layer="0" fill="false" discard="true"/>
```

Listing 9.5- XML specification file with specification of obstacles

The above example shows a generated polygon settings. The below file shows the setting when opened in SUMO, and the another example that shows the polygons numbered.

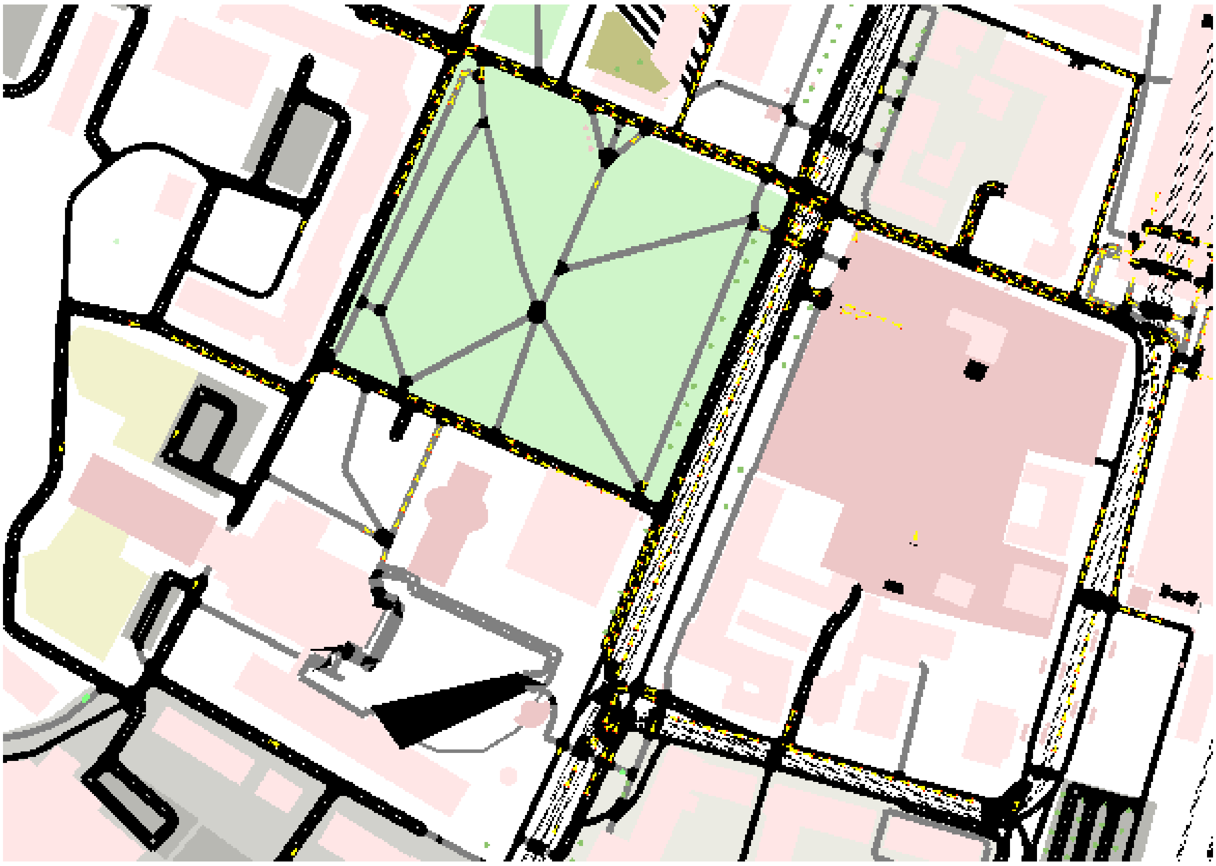


Figure 9.3- Data file generated using DUAROUTER and POLYCONVERT when imported into SUMO

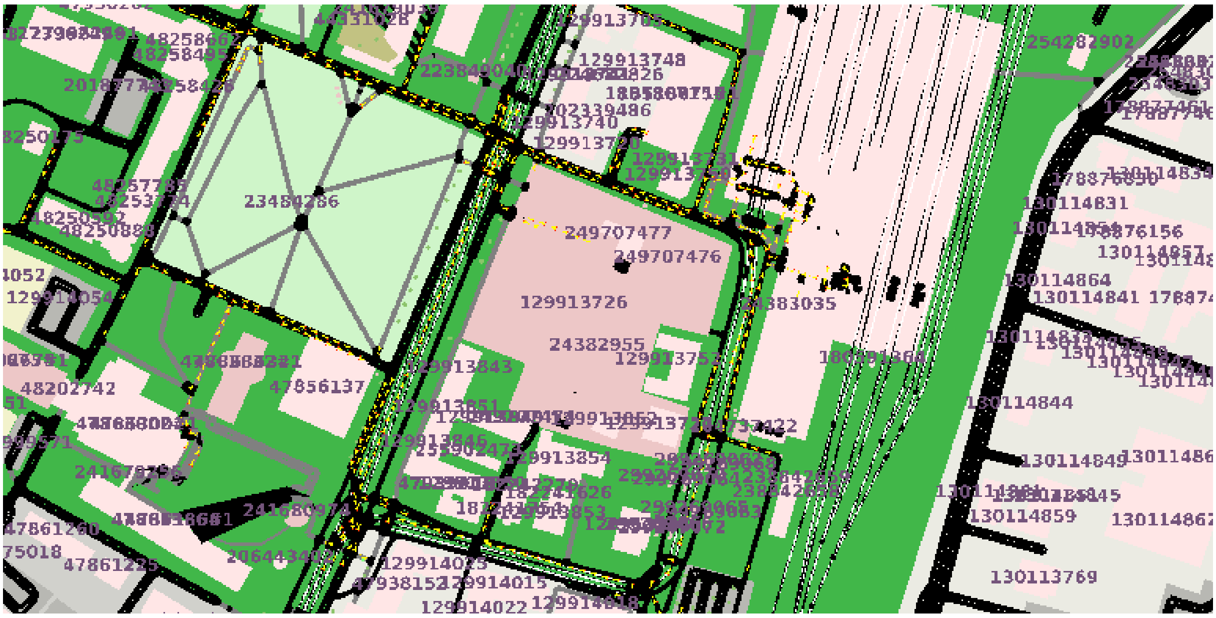


Figure 9.4- The same figure as above, but with the IDs of the polygons.

Once the above files have been generated, the final file that includes the names of all the above files should be generated in order that SUMO, combines all the details needed for simulation from the files at the same time.

```
<?xml version="1.0" encoding="iso-8859-1"?>

<configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo.sf.net/xsd/sumoConfiguration.xsd">

  <input>
    <net-file value="netgenerate.net.xml"/>
    <route-files value="randomTrips.rou.xml"/>
    <additional-files value="polygenerate.poly.xml"/>
  </input>

  <time>
    <begin value="0"/>
    <end value="1000"/>
    <step-length value="0.1"/>
  </time>
</configuration>
```

Listing 9.6- Final configuration XML file which may be imported into SUMO-GUI.

```xml
<configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://sumo.sf.net/xsd/sumoConfiguration.xsd">

    <input>
        <net-file value="chemnitz2.net.xml"/>
        <route-files value="chemnitz2.rou.xml"/>
        <additional-files value="chemnitz2.poly.xml"/>
    </input>

    <time>
        <begin value="0"/>
        <end value="1000"/>
        <step-length value="0.1"/>
    </time>

    <gui_only>
        <start value="true"/>
    </gui_only>

</configuration>
```

Listing 9.7- The configuration file, used in one of the scenario

9.2.2 Results based on Geodata Quality

The following results were obtained by comparing simulations based on the quality of geo-data [30].

Neghbour Count / GeoData	0	20	40	60	80
140m					
RT	0.0-0.1	~0.5	0.5-0.6	1	1
C-S	~0.1	1	1	1	1
F-F	~0.1	1	1	1	1
180m					
RT	0.0-0.1	~0.5	~0.5	~1	1
C-S	~0.1	1	1	1	1
F-F	~0.1	1	1	1	1
240m					
RT	~0.0	0.4-0.5	~0.5	~0.9	1
C-S	0-0.1	1	1	1	1
F-F	~0.1	1	1	1	1
1200m					
RT	~0.0	0	0-0.1	~0.1	~0.1
C-S	0	~0.1	0.6-0.7	1	1
F-F	~0.1	1	1	1	1

Listing 9.8- RT-Road Topology, C-S-Crowd Sourced Geodata, F-F-Full Featured Simulation

The above table shows the distribution displays the number of nodes in four approx. communication ranges, 140 m, 180 m, 240 m and 1200 m, which denote the power of radios [31]. The above study used only the road geometry and class information, depending only on extrapolation features of SUMO.

Traffic for the scenarios were developed based on Dynamic User Assignment. The focus was on an ROI of 1.5km X 1km. This study by Christoph Sommer et al. in the paper "Improving the Accuracy of IVC Simulation using Crowd-sourced Geodata" [32], evaluate the dependency of IVC simulations on more than only network topology. The metric used in the above table is the number of nodes in communication range. The readings were plotted using the neighbour count on the empirical cumulative frequency distribution, focussing on individual measurements. The results of the data indicate a wide variation of vehicle density on

the networks, which indicated a discrepancy between traffic demand and road network capacity. It also showed a variation with some roads with high lane count and other not being able to handle the traffic leading to jams. Crowd sourced geodata solved this.
In Full featured simulation, with low range of distance, with the inclusion of buildings, the impact on low-power radios was minute. However, the impact of inclusion of buildings was high with high-power radios. In conclusion, the full-featured simulation was more pragmatic, than mere simulation on Road Topology.

One of the features this study is the focus on the ability to simulate vehicle dynamics and incidents such as accidents in SUMO. During the Pope's visit in 2005, SUMO was used by the traffic department to forecast the flow of traffic in Köln and in 2006, for the Football World Cup. Also, used during in-vehicle telephony behaviour for performance evaluation of GSM traffic surveillance. This means that SUMO needs to be very versatile and indeed is.

Currently, accidents in SUMO are simulated in the following methods,

a. Allow the vehicle in focus to stand/stop on a road for a period of time.

b. Placement of variable speed sign of the road where the accident is planned and then reduce the speed of the vehicle.

c. Combine the above mentioned methods. Stop the car on one lane and reduce the speeds of the other cars on the particular lane and other lanes.

1. Stopping Cars on the lane

Roads can be stopped for a period of time or 'wait' for persons on the 'side of the road'.

This is edited in the routes file from generated from DUAROUTER or randomTrips.py.

```
<routes>
  <route id="rtype_0" edges="beg middle end rand">
    <stop lane="middle_1" endPos="50" duration="20"/>
  </route>
  <vehicle id="vtype_0" route="rtype_0" depart="0">
    <stop lane="end_1" endPos="10" until="30"/>
  </vehicle>
</routes>
```

Listing 9.9- Data specification for stopping a vehicle during simulation

In the above description, the vehicle stops twice. First, at road middle_0, as is specified in the route lasting 20 seconds. The second time, as the stop in the vehicle itself is described in simulation, when the vehicle stops driving, lasting until simulation second 30.

As in the above example, the duration and until parameters are specified, the vehicle will stop for the specified period. If duration is set to 0, then the vehicle will decelerate to velocity 0, but will not make a full stop, but will accelerate from there. When, until is specified and duration isn't, then the vehicle will reach the point but will not decelerate. There is a parameter triggered, which will be called or set to **true** if neither until nor duration, are specified. If triggered, sets to false, the vehicle will stop for the rest of the simulation. However, if the parameter parking is set to true, then the vehicle stops at the 'pavement' and does not block the traffic flow.

2. **Placement of variable speed sign of the road where the accident is planned and then reduce the speed of the vehicle.**

Using an additional-file specification,

```
<vss>
  <step time="<TIME>" speed="<SPEED>"/>
  <step time="<TIME>" speed="<SPEED>"/>
  ……n-number…….
  <step time="<TIME>" speed="<SPEED>"/>
</vss>
```

Listing 9.10- Declaration file, defining the time of the simulation and the Corresponding timing of their speeds.

The above instance shows, the specification of the time at which the speed of the vehicles should be.

The above file can be used to simulate stops and accidents.

Chapter 10

Implementation and Analysis of Test Scenarios

The previous section describes how a high quality map was developed map which adds a level of realism to the simulation. This section describes how the network simulator is configured. VEINS framework has the 802.11p protocol implemented in OMNet++. It simulates, almost, the whole WAVE interface. This study has been done with no change to the default implementation of the WAVE interface..

This study used the already implemented communication scenario from the veins example, but the maps, the communication protocol and the flow of vehicle traffic are implemented differently. The scenario consists of modules, obstacle control, that simulates transmission blocking due to obstacles. Annotation manager draws annotations on the scenario. Connection manager enables the communication between nodes, and finally, the TraCIScenarioManagerLaunchd which launches SUMO during the beginning and to kill it at the end of the simulation.

10.1Scenario One

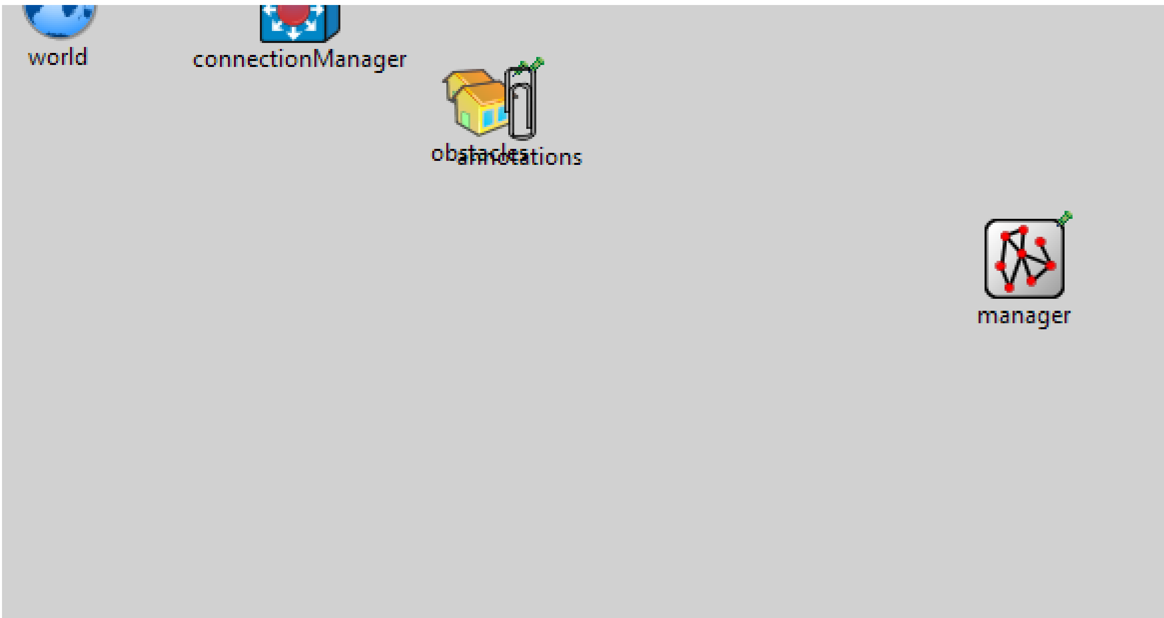


Figure 10.1- The above scenario is under the module scenario.ned.

In the above scenario, it is possible to simulate any protocol and events required for the simulation.

This projects simulates two different simulations. One with a Road Side Unit with the already available TraCIDemoRSU11p and few cars to simulate an accident. The second, with an implementation of a flooding protocol.

Figure 10.2- Above scenario with the RSU is present. As the ned topology and the ned code for the same generate simultaneously.

The scenario above is a simple network, where the RSU is located at some position (x,y,z) on the scenario. The module of RSU is designed as below

```
import org.car2x.veins.nodes.RSU;
import org.car2x.veins.nodes.Scenario;

network RSUScenario extends Scenario
{
    submodules:
        rsu[1]: RSU {
            @display("p=150,140;b=10,10,oval");
        }
}
```

Listing 10.1- Description of the scenario of Figure 10.2.

.

1. Road Side Unit

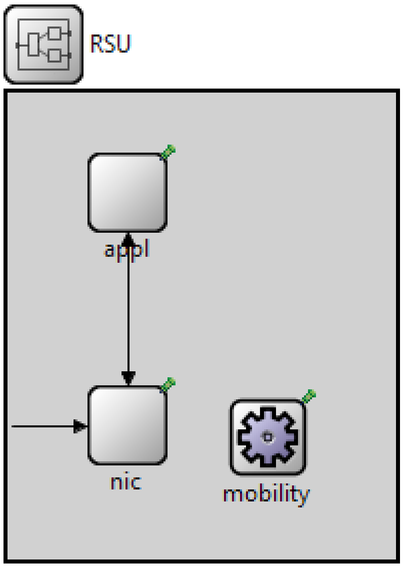
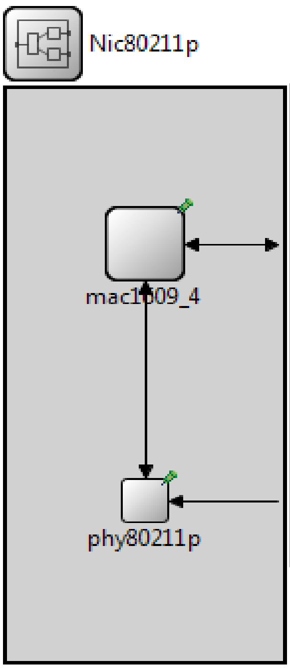


Figure 10.3- ned topology of the RSU module.

The application layer of the RSU module, is the module interface IBaseApplLayer, which itself does not implement the application layer but only extends the BaseWaveApplLayer, the WAVE application layer base class. The above is the default setting of the RSU module. The nic (network interface card) module implements the 802.11p physical layer and IEEE 802.11p Mac layer 1609.4.




```
radioIn --> phy80211p.radioIn;

        mac1609_4.lowerControlOut --> phy80211p.upperCon-
trolIn;
        mac1609_4.lowerLayerOut --> phy80211p.upperLayerIn;
        phy80211p.upperLayerOut --> mac1609_4.lowerLayerIn;
        phy80211p.upperControlOut --> mac1609_4.lowerCon-
trolIn;

        mac1609_4.upperControlIn <-- upperControlIn;
        mac1609_4.upperLayerIn <-- upperLayerIn;

        mac1609_4.upperLayerOut --> upperLayerOut;
        mac1609_4.upperControlOut --> upperControlOut;
```

Listing 10.2- Connection specification between the PHY and MAC layers

Figure 10.4

The above image **Figure 10.4**, shows the connection topology of the Nic 802.11p on the left, and the connection specification on the right. The default parameters are used in the execution.

The mobility module implements the BaseMobility, which in turn implements the IMobility interface. As the RSU needs to be stationary and not mobile, the BaseMobility defines a static mobility for it. Hence, the position of the RSU is fixed in any scenario (500,500) in this scenario.

2. Car

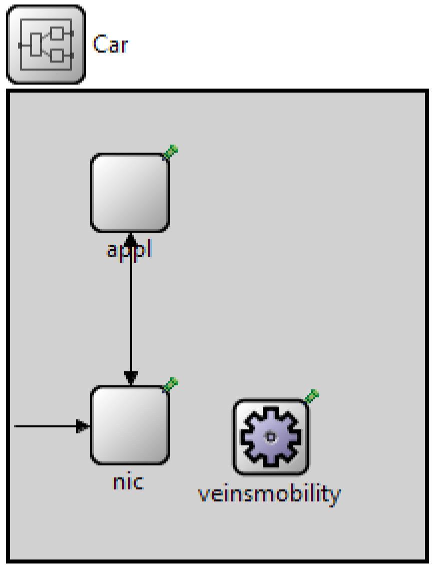


Figure 10.5- Topology of the Car module

The above graphic shows the topology and the connections of the Car module. The nic module is the same as that described above. The appl modules describes the type of application layer, which this case IBaseApplLayer, which extends the application type **TraCIDemo11p**, in this scenario. The veinsmobility module extends the mobility class **TraCIMobility,** which provides the car nodes the ability to move in the scenario and remain static as in the case of RSU.

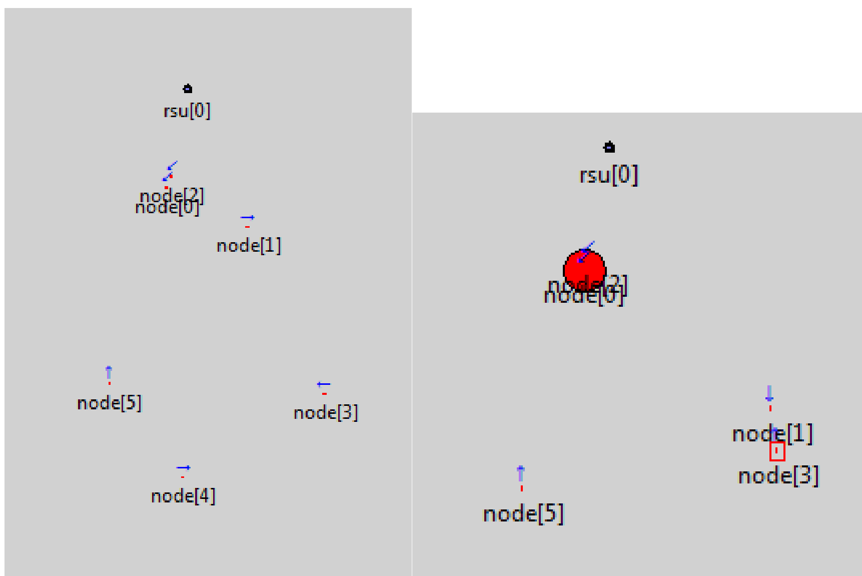


Figure 10.6 (a) (left) and Figure 10.6 (b) (right) - Beginning node generation in the scenario and accident of node [0] at time 20s

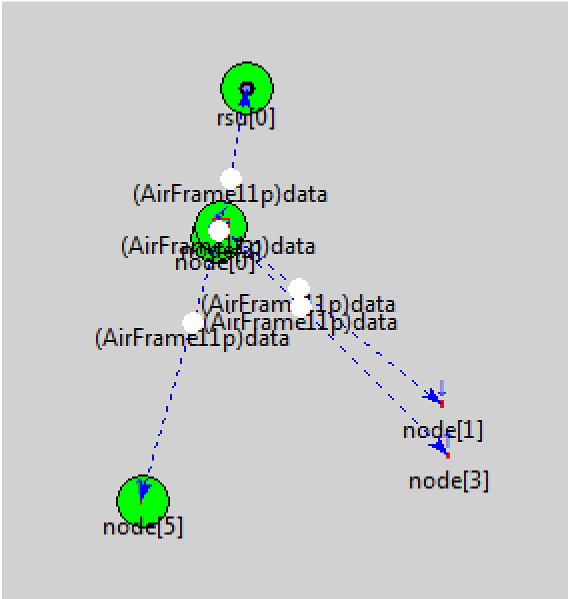
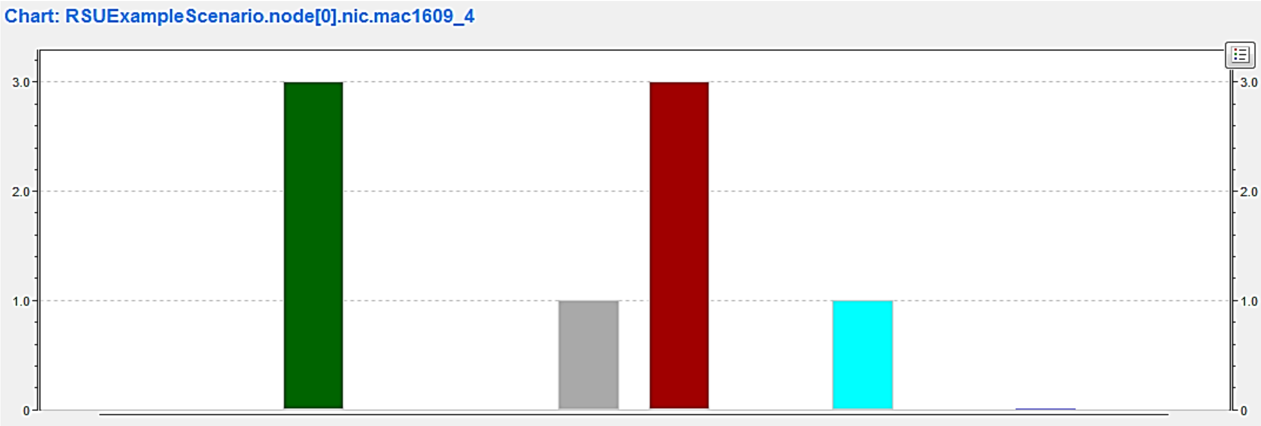


Figure 10.7- Dissemination of data packets from node [2] to the other nodes within the network

From, the above scenarios, **Figure 10.6** and **Fig 10.7** show the initial setting i.e. generation of Car nodes [0-5] with one RSU. A random node was chose to simulate an accident, in this scenario node [0] meets with an accident at time20s, i.e. 20s into beginning of the simulation and the incident is configured to continue till the next 30s.

As node [0] meets with an accident, it transmits an **AirFrame11p** to the **rsu [0]** located at the end corner of the street and node [2] that has stopped behind node [0] because of the accident. As the rsu [0] receives the message, it disseminates the message to all the available nodes in the vicinity. If any node is currently unreachable, a node that has received the message would transmit that same to that node or the rsu re-broadcasts the message until the node receives the message. The stretch of road analysed here is approximately 2000 meters long.

As time elapses, the nodes begin driving again and reach their destination or continue until end of the simulation. For this scenario, the results are as follows:



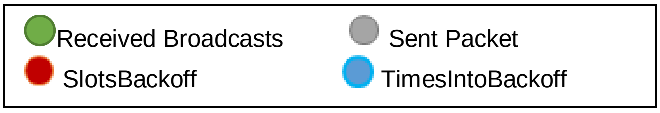


Figure 10.8- The grey column shows the number of packets sent i.e. 1. The dark green colum represents the number of received packets. We can see from the above graph that there has been no packet loss.

The following graph provides the variation in drive speed of the node [0]:

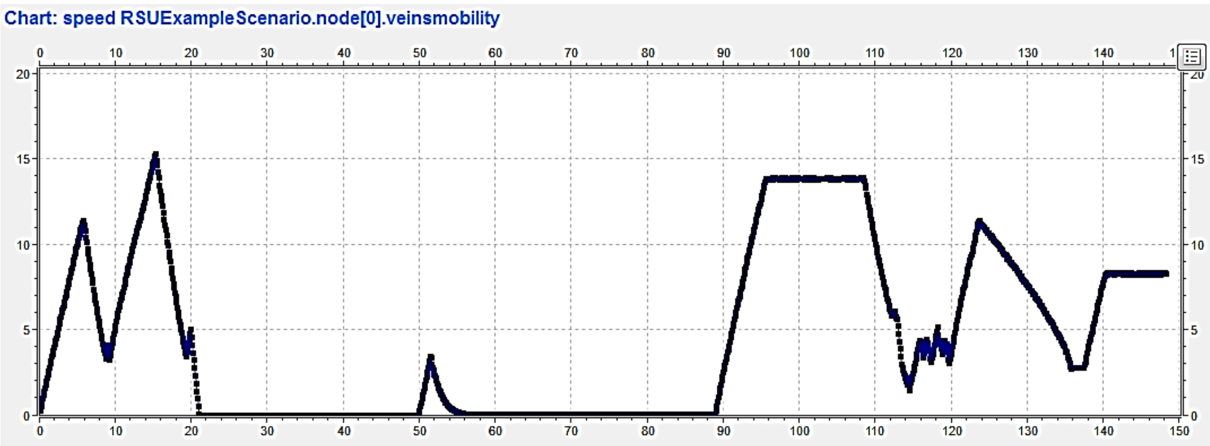


Figure 10.9- The speed of node [0] during the simulation run

From 20s onwards, the car (node [0]) remains without mobility for the next 30s and slowly gains mobility from 50s onwards. The node [0] stays without movement till 90s, as the traffic signal was not yet given for the vehicles to drive again.

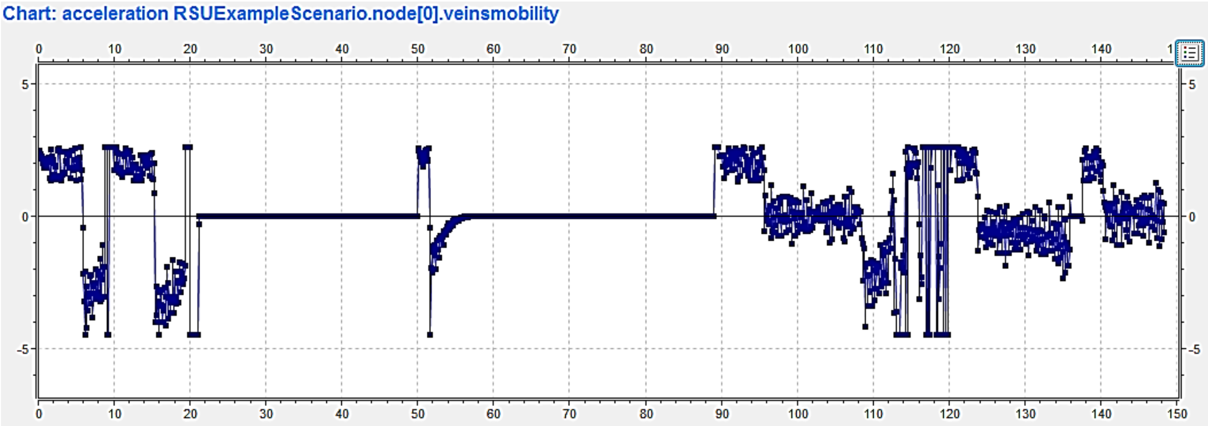


Figure 10.10- Acceleration changes of node [0]

The variations in the acceleration of node [0] from the above graph corresponds well with the simulation and velocity.

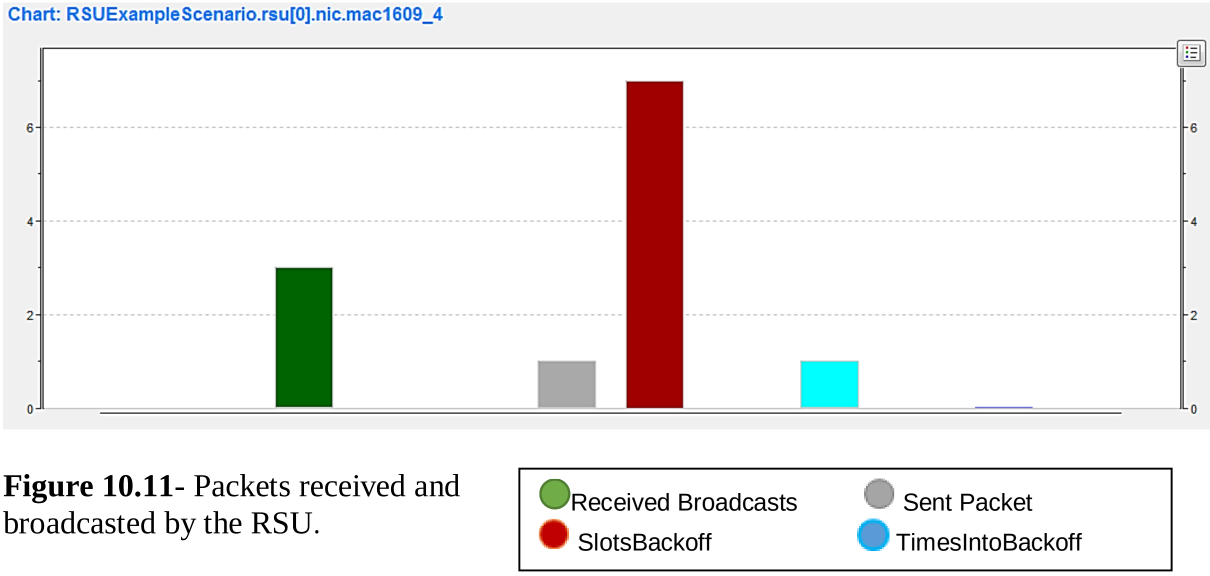


Figure 10.11- Packets received and broadcasted by the RSU.

10.2 Scenario Two

This scenario will demonstrate the flooding protocol. The chosen map for this scenario the map of the city Chemnitz, on which vehicles will be connected to each other and flooding of data between them occurs periodically.

The scenario from Figure 10.1 will be used here. The following sections will explain the construction of the network topology, the connections between the modules and the working.

1. The Flooding Protocol

As discussed in Chapter 6, the protocol generated and tested here is the basic Flooding protocol. Every node that receives a packet will broadcast the same to the rest of the nodes in the network. Although in an actual Flooding, the number of messages generated and flooded is much more than the required number. But the protocol implemented here floods the packet to only those nodes that it is connected to.

2. Flooding and Periodic Data Generation

Depending on the requirements of the scenario, a technique was modelled and an application that simulates periodic data generation. In order to implement this application layer was suited to the design.

```
import org.car2x.veins.base.modules.IBaseApplLayer;

simple Flooding like IBaseApplLayer
{
    parameters:
        bool debug = default(false);
        bool sendData = default(false);
        int headerLength = default(32bit) @unit(bit);

          @display("i=block/app2");

    gates:
        input lowerLayerIn;
        output lowerLayerOut;
        input lowerControlIn;
        output lowerControlOut;
}
```

Listing 10.3- The protocol is implemented in the lower layers of the Application Layer.

```
WaveShortMessage* wsm = new WaveShortMessage("packet");
    wsm->addBitLength(headerLength);
    wsm->addBitLength(128);

    wsm->setChannelNumber(Channels::CCH);
    wsm->setTimestamp(simTime());
    wsm->setSenderPos(Coord(0,0));

    sendDown(wsm);
```

Listing 10.4- WAVE Specification

The data packet dispersed is a WaveShortMessage, of headerLength 128bits. The channel number is as specified by the IEEE 802.11p standards, and hence the CCH number will be 178, as reserved by the FCC. Also, the simulation time is retuned here by a call placed to **simTime.**

Since this scenario, concentrates on generating a packet once and then that packet is broadcasted to the network and the nodes in it, the packet is generated once by every node periodically.

```
if (sendData) {
            scheduleAt(simTime(), new cMessage("send initial packet"));
        }
```

Listing 10.5- Generation of a data packet

If bool variable **sendData** is set to true, at some simulation time, a message will be generated and transmitted. The above code is implemented over every Car in this scenario, hence this is the protocol that is associated with the OBU. Under the assumption that all the nodes in this network are known to each other, the number of messages received is not kept track off. Irrespective of the position of the sender node, the position of the node at the time of data generation and dissemination is always (0,0).

This scenario will also demonstrate the different speeds at which vehicles travel, but regardless of their respective speeds, the vehicles still remain largely connected to each other and will exchange data periodically. It is obvious that RSUs will not be present every kilometre (theoretical maximum range of IEEE 802.11p) or at every junction, hence, this scenario was designed to demonstrate plain car-2-car techniques, in the absence of RSUs.

Connection between the application layer and the MAC layer is achieved by using the WAVE Short Message Protocol (WSMP), defined by the IEEE P1609.3 substandard. In this case, the WAVE short message can be transmitted through two different channels, the Control Channel (CCH) and the Service Channel (SCH), with CCH being used for broadcast communication, and for short and high-priority communication, such as safety-critical ones, and the Service Channel for 2-way communications or some particular applications such internet access or tolling. This scenario has rightfully used CCH, as the concern is communication between on-board units. There is no queue implemented here, as it is dangerous to put safety critical messages in a position that causes delay.

3. Car

Below is the topology and specification for the Car module for this scenario.

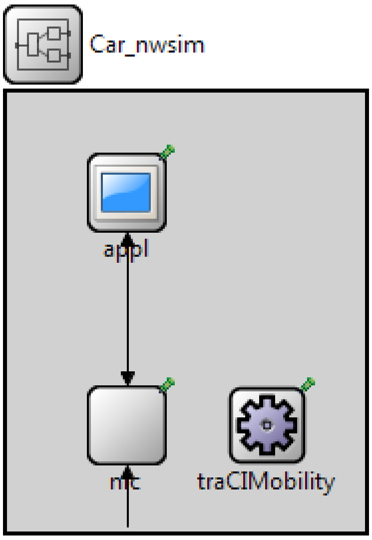


Figure 10.12- Car topology

```
        submodules:
        appl: Flooding {
            parameters:
                @display("p=60,50");
        }

        traCIMobility: TraCIMobility {
            @display("p=130,166");
        }
    connections:
        nic.upperLayerOut --> appl.lowerLayerIn;
        nic.upperLayerIn <-- appl.lowerLayerOut;
        nic.upperControlOut --> appl.lowerCon-
trolIn;
        nic.upperControlIn <-- appl.lowerCon-
trolOut;

        veinsradioIn --> nic.radioIn;
```

Listing 10.6- Connections of the Car module

In this definition of the Car or OBU, the application type is specified as Flooding and the mobility of this module is directly connected to TraCIMobility, eliminating any interfaces between the vehicle and the mobility class. Here too the Network Interface Card is the input point for the Car. The following diagram shows the inheritance of the Car module.

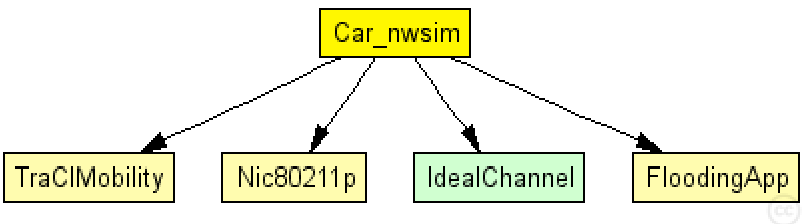


Figure 10.13- Inheritance of the Car module

One of the tasks of this project is to evaluate vehicle dynamics, and hence, this scenario implements a function that alerts a vehicle, if another vehicle is overtaking. This functionality is especially helpful, when vehicles are in the following scenarios,

- When vehicles are at high speeds
- Driving on narrow lanes
- When there is considerable traffic, which makes driving a difficult task as the driver needs to concentrate on many events with little or no room for any negligence
- On highways, where traffic density is high and includes vehicles of all sorts- small cars to heavy transport vehicles, and many times includes construction vehicles
- In the absence of illumination

Every vehicle has a certain area of coverage. When in motion, a vehicle broadcasts beacons to the network, and in effect the nodes within the network, and any node (s) under the coverage area receives this beacon and updates its data base as explained previously (Chapter 6) . When in range of an RSU, a vehicle receives data on traffic congestion, road condition, etc. This data is maintained offline i.e. information within its network range. To collect speed data of the vehicles in its range, the RSU transmits queries periodically. This is called query period [32] and differs from protocol to protocol. To update this speed data, the query period should be longer than the transmission delay from the vehicle and RSU. This delay increases with the number of vehicles present in the range of the RSU, and this communication would be inexistent if the RSU were absent from the scenario. It would be simpler if a vehicle, V1 overtakes another vehicle, V2, informs V2 of its speed and that it will overtake. In this project, this is accomplished by using the command

```
c*mobility=parent->getSubmodule("TraCIMobility");
double myspeed=mobility->getSpeed();
```

Listing 10.7(a) - Specification to query the speed of the Car

A module to update the speed of vehicles in the network is defined, with the above command. This connects to the TraCIMobility class. This is then included in the Car module.

The speed of vehicles is collected during the run of the simulation. The speeds of the vehicles in the scenario are compared to speed limit set on each lane. The number of vehicles on the lane is known from the TraCI classes. Whenever, a vehicle exceeds the speed limit set on the lane, a beacon is transmitted to all other vehicles in the network, warning them that a vehicle might overtake.

A boolean variable **overtaking** is defined, which is transmitted to the neighbouring vehicle, in the following message:

```
vehicle_ID = {vehicle_ID, overtaking};
```

Listing 10.7(b) – Passing of parameters to alert the vehicles of an overtake

With this, message beaconed to the vehicles, the other vehicles are warned about the oncoming high-speed vehicle.

4. Socket to External Port

Using the existing sockets library in OMNet++, a TCP/UDP socket has been included with the program. This socket is configurable to connect to the existing BaseApplLayer, so that it may draw messages out of the running simulation and transmit the command to the model car via the external communication/wireless device.

For this project the external device, was not yet available, hence, the complete demonstration was not carried out.

Chapter 11

Simulation Results

1. Data acquisition and analysis

OMNet++ provides built in tools that record the results of the simulation-output vectors and output scalars and Histograms.

Output Vectors

Data recorded over time and from simple modules and channels, can interpret all the action that took place during the simulation. These vectors help analyse the model

Output Scalars

Summary of the results, it is computed during the run of the simulation and is penned on completion. Mainly statistical data.

Histograms compute basic statistics from the input and plot the result. Uses cLongHistogram class.

Two ways of recording results from a simulation exist:

1. According to signal mechanism, applying pre-declared statistics.
2. Direct from the programmed code.

It is more important to extract data from a running simulation that has the waiting time, delay, number of packets lost, etc. This allows the analysis of protocols so that the researchers know the efficiency of the technique and how it can be bettered. The data in this simulation is represented by messages generated while the car is on the road. The data was collected by signalling which records the measurements required from the simulation.

Simulation outline

This project studies two scenarios to analyse the behaviour of the protocols and vehicle dynamics that affect packet dissemination and to see what would happen during the communication.

For the simulation, an approximately 2000 meters long, at the corner of which was an RSU present in the first simulation, to simulate and accident.

In the second scenario, a flooding protocol was introduced to and programmed such that the car remain in contact in the absence of an RSU and exchange messages between themselves.

Data Extraction Algorithm

To check for the exact delay and event log, extract time stamps a command

```
*.Node[*].DelayStats.result-recording-modes = +histogram
```

Was introduced in the oment.ini file. This command is the index of static property. The dispersed data message information is recorded in the vector stats.

Also, a command is set in the omnet.ini file

record-eventlog = true

2. Results according to Simulation Scenario

Scenario 1

In the first scenario, an accident was simulated with one car behind the car that meets with an accident and 3 other cars in the vicinity. An RSU was present at the other end of the road to analyse the message scheduling. The accident was set to take place at 20s into the simulation and lasting for 30s. Following is the sequence chart of message exchange between the RSU and the cars.

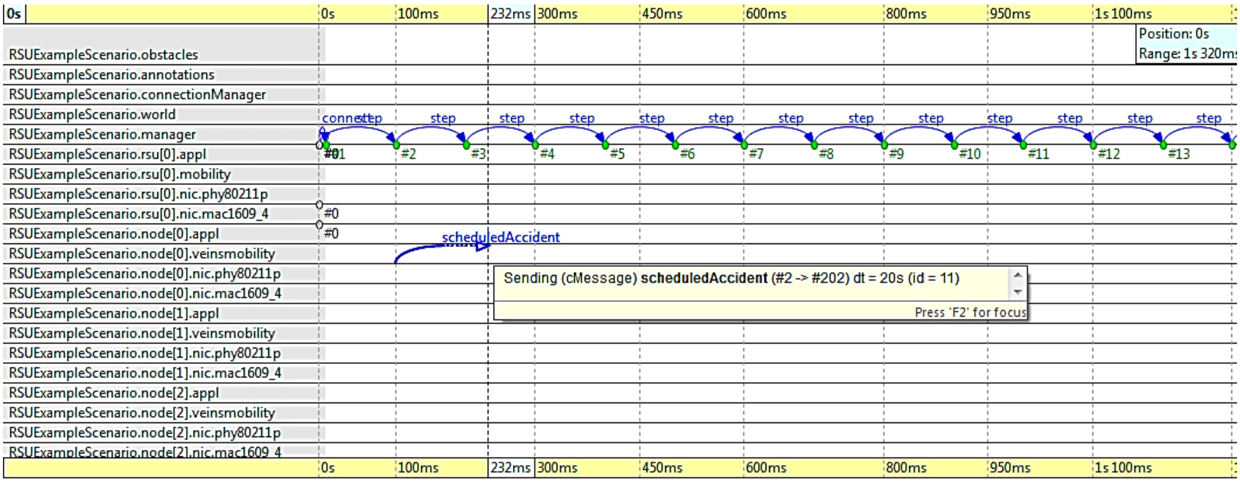


Figure 11.1- Log file containing the time trace of the occurred event and the messages exchanged. In the above graphic only the accident being scheduled is seen.

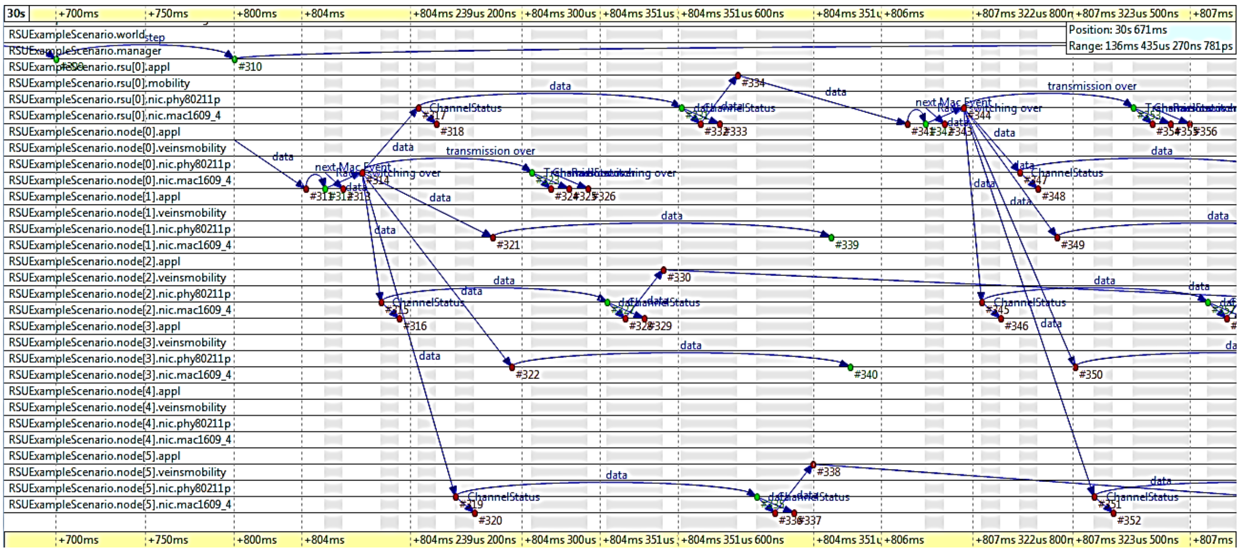


Figure 11.2- The above graphic plot show the exchange of data between the node [0] and the rsu [0] after the scheduled accident occurred. Data from node [0] then transmits data to all the other nodes in the vicinity. This happens two more times, once from the rsu [0] to the other nodes and then between all the other nodes.

As this is a safety critical situation, the delay in transmission between the nodes is negligible, and there is no loss of packets, which is always a good result.

Scenario 2

This is a more complex scenario as there is an absence of RSU. Here, the flooding protocol is demonstrated and the results recorded. As the RSU cannot be present at all the driving lanes, the car need to communicate among themselves. Here, irrespective of the speed of the vehicle, the cars remain connected.

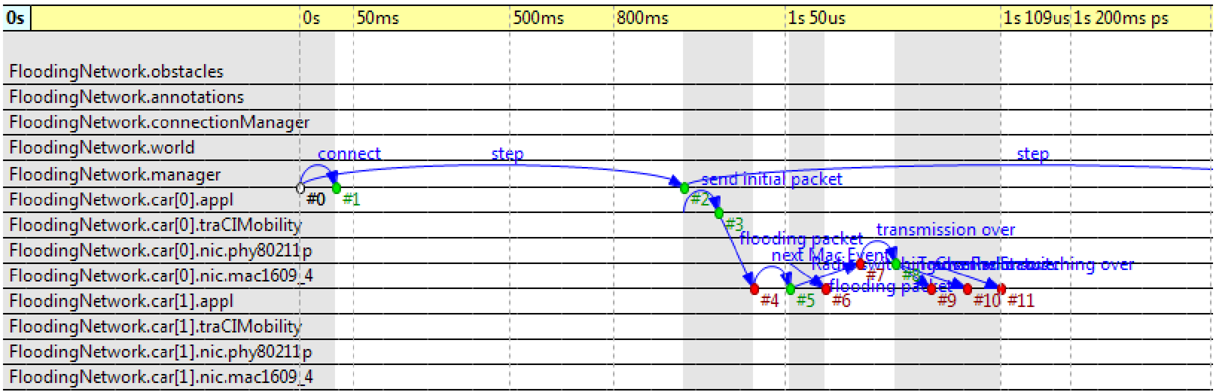


Figure 11.3- Generation and Transmission of Initial Packet by node [0]

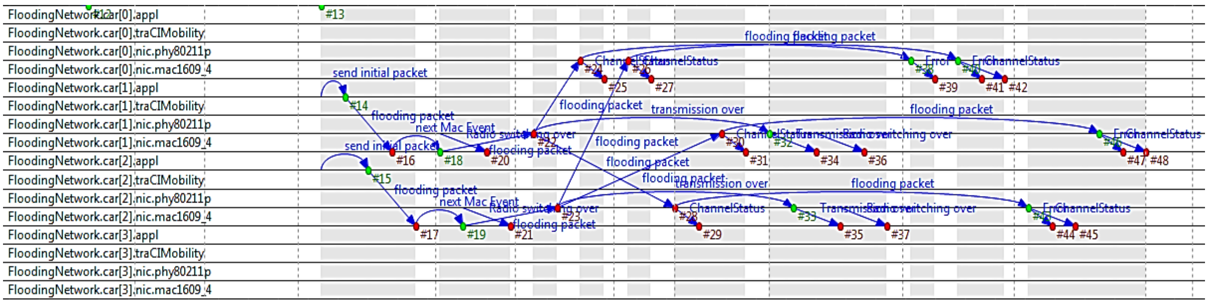


Figure 11.4- Receiving of the Initial Packet and re-broadcasting of the same packet by node [1] to node [2]

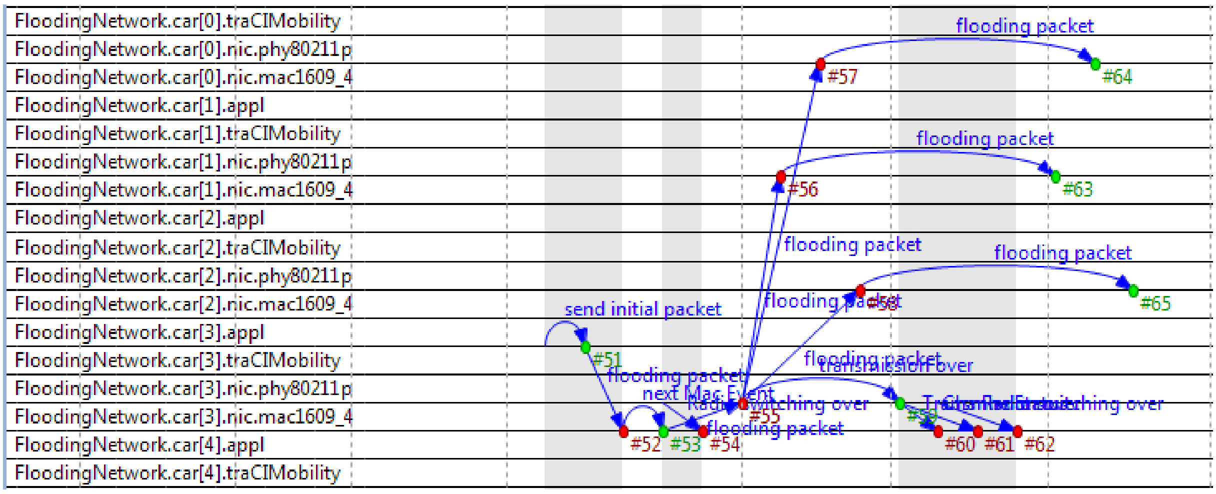


Figure 11.5- Initial Packet Sending by node [2] to the nodes [3] and [4]

This flow repeats until the packets have reached all the destination nodes and finally, a random node initiates this process again.

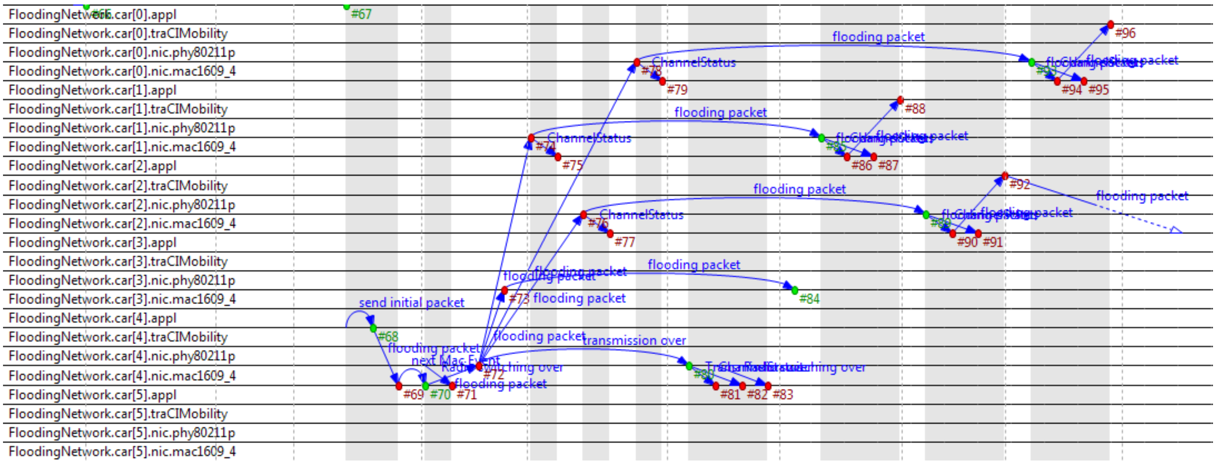


Figure 11.6- Packets arriving at nodes [4] and [5]

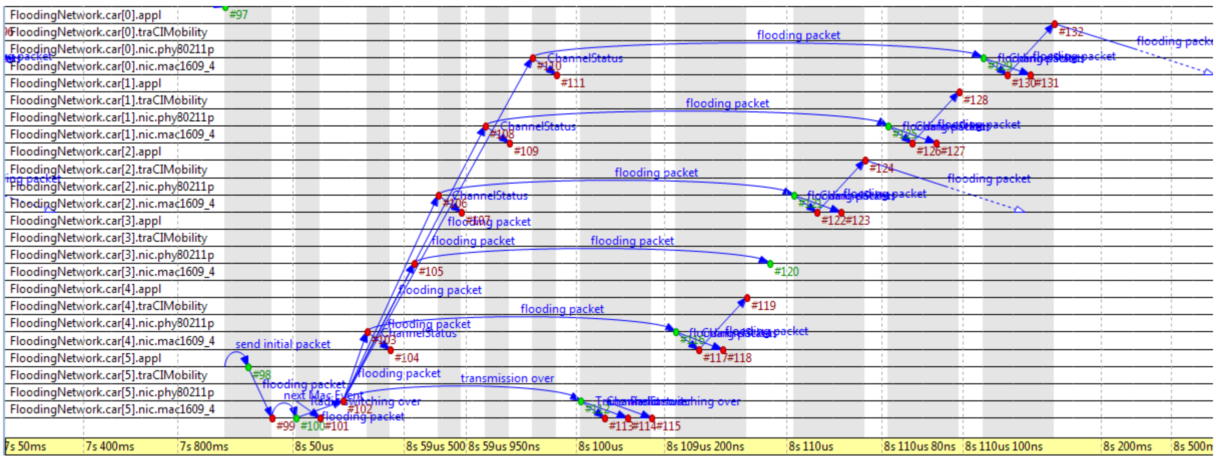


Figure 11.7- Finally all the packets have reached their destinations and the flooding of packets continues.

The flooding packet, if duplicate, is suppressed by the receiver node in order not to cause any kind of looping to occur.

It was interesting that, when the packets reach their destination, the next process is that, the receiver node initialises the packet, and send it across to the nodes. But if any duplication, as explained above, the packets are supressed and not transmitted anymore. Even, if they are transmitted, the other receiver packet supresses this phenomenon.

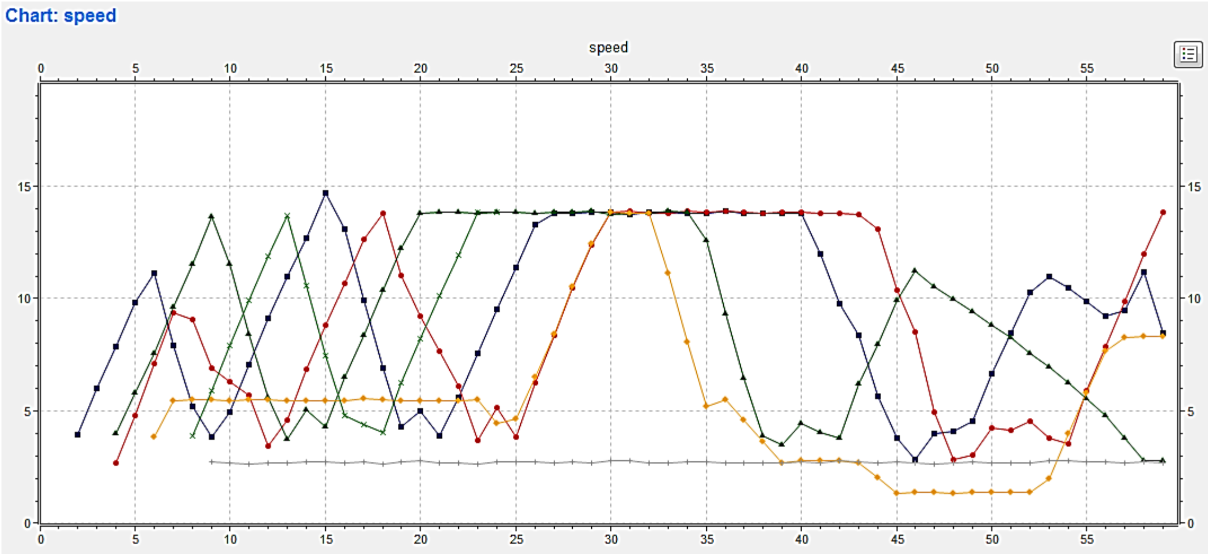


Figure 11.8- Plot of different speeds is shown here. Even, while travelling at different speeds, the vehicles remain connected, and transmit data.

The speed limit remains with the 15m/s range for all the cars. As the simulation is a small scale representation of the larger picture, the speed of the nodes will remain rather slow.

The flooding protocol is a very useful strategy of disseminating data packets. But at the same time this protocol causes overheads, because of the very nature of it. Any node that receives a packet broadcasts it regardless of the uniqueness or the duplicity of the packet. Therefore, strategies are being adopted, so as to avoid this sort of overheads. Some of the other problems attached to flooding are interference on systems, collision with users and contention on applications.

An approach that is more efficient that the family of flooding protocols is Geocast. It is Topologically Assisted Geo-Opportunistic Routing. Here, the nodes send Hello beacons periodically to its neighbours. Thus, informing nodes about the number of neighbours, their IDs and the IDs of the furthest nodes.

Conclusion

VANET came about with the intension of increasing security on the roads. Which is now "smart" drive/networks. Attracting interests from both academia and industry, it is now one of the most researched and invested fields of technology. Although, continuously researched, it still has a particular drawbacks.

The current standards are only amendments to the earlier standards. Along with them came an additional family that provides a range of excellent protocols but there are still many unfinished protocols remaining. The WAVE interface is by-far the most versatile family of standards, but it still has many unfinished protocols.

This project focussed on the possibility to setup a physical model of Car-to-X instead of only the available simulation models. Firstly, the existing protocols in the simulation framework were analysed and some scenarios implemented so that they make be physically replicated. For this, a high level simulation framework and toll is required that connects to an external device and is capable of working according to the time in the real world.

Much of the technologies developed in this field are tested only by simulating their performances. One the troubling issue of VANETs, is privacy. Privacy regarding the location of the target, its location at a certain point in time and its location in the future. As it is also difficult to discern any break in privacy which may target the identity of the vehicle, MAC address, IP address, login data and most existing protocols do not consider making the car anonymous. Some ways of ensuring anonymity, is by using pseudo identity. The connecting node will first acquire an authorisation from a Certification Authority, and then connect to the node. This way both the nodes do not reveal their identities. But this is not as simple as it seems. There needs to be ways of identifying an attacker(s).

1. Maximum tracking time – keep a track of all the cars around/following and use techniques of predicting the next position of the car(s), using dead reckoning. Cars that change positions too many time can be discarded. Here, the technique is trying to track the time period for which the attacker followed the car. But again predicting is tricky and yield a varied result set

2. Anonymity Set Size - An anonymity set is a collection of all nodes that are indistinguishable from the target node and that a target node is located within this anonymity set. The problem is that not every vehicle is probable, thus probability overestimations.

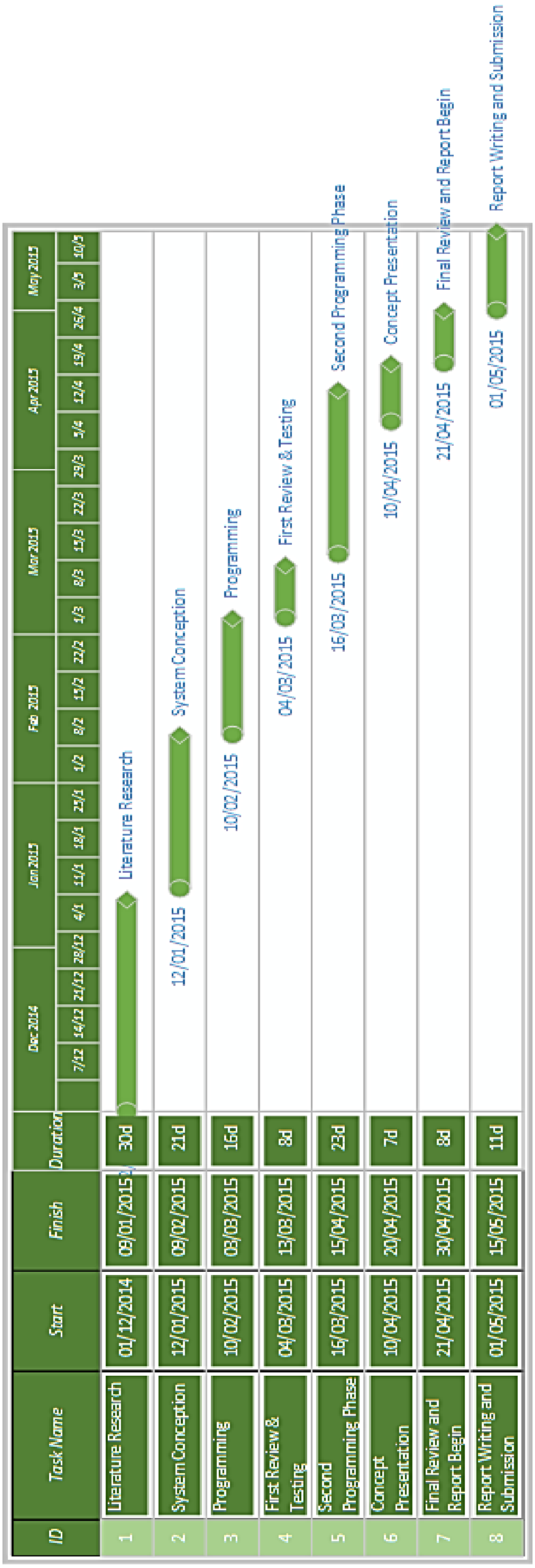
3. Entropy – Consider a set and that the target is located within this set. All the probabilities in this set is 1, then entropy H

$$H_p = \sum_{i=1}^{|A|} p_i * \log_2 p_i$$

Eq[1.3]

Hence, by attaching a small probability to everyone, the nodes may remain safe.

While a decade does not seem much, but a great lot of research has been accomplished and we look forward to a complete implementation of VANETs soon.



Gantt Milestone Chart

Acknowledgements

My gratitude Jun.-Prof. Dr.-Ing. Alejandro Javier Masrur, my Academic Advisor, for the opportunity to pursue this Master Thesis study and for his guidance that contributed immensely to the successful completion of this Thesis. I thank Him for his patience.

To Dipl.-Ing. Philip Parsch, for his help and contribution.

I'd like to acknowledge Prof. Dr. Wolfram Hardt, for this chance to pursue Master studies, here in Technische Universität Chemnitz, in the Professorship of Computer Engineering, and for contributing to the improvement of my academic and industrial perspectives.

.

To my family, my Parents and my Sister- I respect and appreciate the moral support, motivation and for providing me with the best thoughts every moment.

Many thanks to all my friends that have believed in me and helped me throughout, culminating in the completion of my Master Studies.

References

[1] T. A. LASKY and B. RAVANI, "A Review of Research Related to Automated Highway Systems (AHS)," Davis, October 1993.

[2] H.-W. Schaal and T. Löffler, "Car2x – From Research to Product Development, How automotive OEMs and suppliers are successfully completing production Car2x projects," Vector Informatik GmbH, December 2012.

[3] "Car2X conquers the road," Siemens References, Munich, November 5, 2014.

[4] R. König, A. Saffran and H. Breckle , "Modelling of Drivers'-Behaviour," in *Vehicle Navigation & Information Systems Conference Proceedings*, 1994.

[5] "CANoe.Car2X-Product Information," Vector Informatik GmbH.

[6] D. R. Choffnes and F. E. Bustamante, "An integrated mobility and traffic model for vehicular wireless networks," in *2nd ACM International workshop on Vehicular ad hoc networks*, 2005.

[7] S. Wang, C. Chou, C. Huang, C. Hwang, Z. Yang, C. Chiou and C. Lin, "The design and implementation of the NCTUns 1.0 network simulator," *Computer Networks,* Vols. vol. 42, no. 2, p. 175–197, 2003.

[8] M. Pi´orkowski, M. Raya, A. L. Lugo, P. Papadimitratos, M. Grossglauser and J.-P. Hubaux, "TraNS: realistic joint traffic and network simulator for VANETs," *SIGMOBILE Mob. Comput. Commun. Rev.*, Vols. vol. 12, no. 1, p. 31–33, Jan. 2008.

[9] V. Kumar, L. Lin, D. Krajzewicz, F. Hrizi, O. Martinez, J. Gozalvez and R. Bauza, "iTETRIS: Adaptation of ITS Technologies for Large Scale Integrated Simulation," in *Vehicular Technology Conference (VTC 2010-Spring),* May 2010.

[10] R. Mangharam, D. Weller, R. Rajkumar, P. Mudalige and F. Bai, "Groovenet: A hybrid simulator for vehicle-to-vehicle networks," in *Third Annual International Conference on Mobile and Ubiquitous Systems : Networking & Services*, 2006.

[11] K. DAR, M. BAKHOUYA, J. GABER, M. WACK and P. LORENZ, "Wireless Communication Technologies for ITS Applications," *IEEE Communications Magazine,* Vols. vol. 48, no. 5, p. 156–162, May 2010.

[12] "Intelligent transport systems – Communications access for land mobiles (CALM) – Architecture," ISO, 2010.

[13] R. A. Uzcátegui and G. Acosta-Marum, "WAVE: A Tutorial," *IEEE Communications Magazine, TOPICS IN AUTOMOTIVE NETWORKING,* Vols. 0163-6804/09, pp. 126-133, May 2009.

[14] D. JIANG and L. DELGROSSI, "IEEE 802.11p: Towards an international standard for wireless access in vehicular environments," in *67th Vehicular Technology Conference (VTC2008-Spring),* Marina Bay, Singapore, May 2008.

[15] “802.11p/D10.0 - Amendment 7: Wireless Access in Vehicular Environments,” IEEE 802.11 WORKING GROUP OF THE IEEE 802 COMMITTEE, February 2010.

[16] C. SOMMER, A. SCHMIDT, R. GERMAN, W. KOCH and F. DRESSLER, “Simulative Evaluation of a UMTS-based Car-to-Infrastructure Traffic Information System,” in *IEEE Global Telecommunications Conference (GLOBECOM 2008), 3rd IEEE Workshop on Automotive Networking and Applications (AutoNet 2008)*, New Orleans, LA, December 2008.

[17] “Physical channels and mapping of transport channels onto physical channels (FDD),” 3rd Generation Partnership Project (3GPP), 2008.

[18] C. WEWETZER, M. CALISKAN, K. MEIER and A. LUEBKE, “Experimental evaluation of UMTS and wireless LAN for inter-vehicle communication,” in *7th International Conference on ITS Telecommunications (ITST 2007)*, Sophia Antipolis, France, June 2007.

[19] T. L. WILLKE, P. TIENTRAKOOL and N. F. MAXEMCHUK, “A Survey of Inter-Vehicle Communication Protocols and Their Applications,” in *IEEE Communications Surveys and Tutorials*, 2009.

[20] L. WISCHHOF, A. EBNER and H. ROHLING, “Information dissemination in self-organizing intervehicle networks,” in *IEEE Transactions on Intelligent Transportation Systems*, March 2005.

[21] T. K. MAK, K. P. LABERTEAUX and R. SENGUPTA, “A multi-channel VANET providing concurrent safety and commercial services,” in *2nd ACM International Workshop on Vehicular Ad Hoc Networks(VANET 2005)*, Cologne, Germany, 2005.

[22] N. WISITPONGPHAN, F. BAI, P. MUDALIGE, V. SADEKAR and O. TONGUZ, “Routing in Sparse Vehicular Ad Hoc Wireless Networks,” in *IEEE Journal on Selected Areas in Communications (JSAC)*, October 2007.

[23] I. CHAKERES and C. PERKINS, “Dynamic MANET On-Demand (DYMO) Routing,” IETF, November 2007.

[24] A. K. Guptal, H. Sadawarti and A. K. Verma, “IMPLEMENTATION OF DYMO ROUTING PROTOCOL,” *International Journal of Information Technology, Modeling and Computing (IJITMC)*, Vols. Vol.1, No.2, May 2013.

[25] O. TONGUZ, N. WISITPONGPHAN, J. PARIKH, F. BAI, P. MUDALIGE and V. SADEKAR, “On the Broadcast Storm Problem in Ad hoc Wireless Networks,” in *3rd International Conference on Broadband Communications, Networks, and Systems (BROADNETS)*, San Jose, CA, October 2006.

[26] N. WISITPONGPHAN, O. K. TONGUZ, J. S. PARIKH, P. MUDALIGE, F. BAI and V. SADEKAR, “Broadcast Storm Mitigation Techniques in Vehicular Ad Hoc Networks,” *IEEE Wireless Communications*, Vols. vol. 14, no. 6, p. pp. 84–94, December 2007.

[27] K. WESSEL, M. SWIGULSKI, A. KÖPKE and D. WILLKOMM, “MiXiM – The Physical Layer: An Architecture Overview,” in *2nd ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems, (SIMUTools 2009); 2nd ACM/ICST International Workshop on OMNeT++(OMNeT++ 2009)*, Rome, Italy, March 2009.

[28] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments," IEEE, 2010.

[29] "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," IEEE, 2007.

[30] "IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) - Multi-channel Operation," Intelligent Transportation Systems Committee, IEEE, February 2011.

[31] D. Eckhoff and C. Sommer, "A Multi-Channel IEEE 1609.4 and 802.11p EDCA Model for the Veins Framework," in *5th ACM/ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2012): 5th ACM/ICST International Workshop on OMNeT++ (OMNeT++ 2012) Poster Session*, Desenzano, Italy, March 2012.

[32] C. Sommer, E. David and D. Falko, "Improving the Accuracy of IVC Simulation using Crowd-sourced Geodata," *Praxis der Informationsverarbeitung und Kommunikation (PIK),* vol. 33, no. 4, pp. 278-283, December 2010.

[33] C. Gawron, "Simulation-Based Traffic Assignment – Computing User Equilibria in Large Street Networks," University of Cologne, Cologne, 1999.

[34] Q. C. Doan, T. Berradia and J. Mouzna, "Vehicle speed and volume measurement using vehicle-to-infrastructure communication," WSEAS Transactions on Information Science and Applications, 2009.

[35] F. Dressler, C. Sommer, D. Eckhoffy and O. K. Tonguz, "Toward Realistic Simulation of Intervehicle Communication: Models, Techniques and Pitfalls," *Vehicular Technology Magazine,* vol. vol. 6 (3), pp. 43-51, September 2011.

[36] I. T. WG16, "Intelligent transport systems – Communications access for land mobiles (CALM) – Architecture,"," IEEE, 2010..

Nomenclature

AC	Access Categories
AICH	Acquisition Indicator Channel
AIFS	Arbitration Interframe Space
BSS	Basic Service Set
CALM	Communications Access for Land Mobiles
CAM	Cooperative Awareness Message
CCH	Control Channel
CDMA	Code Division Multiple Access
CW	Contention Window
DCH	Dedicated Channel
DENM	Decentralized Environmental Notification Message
DSRC	Dedicated Short Range Communications
DV-CAST	Distributed Vehicular Broad-CAST
DYMO	Dynamic MANET on Demand
EDCA	Enhanced Distributed Channel Access
EDCAF	EDCA Function
ETSI	European Institute for Telecommunication Standards
FACH	Forward Access Channel
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronic Engineers
IPv6	Internet Protocol 6
ITS	Intelligent Transportation Systems
MAC	Medium Access Channel
MANET	Mobile ad hoc Network
MBMS	Multimedia Broadcast Multicast Service
NED	Network Description
OBU	Onboard Units
OEM	Original Equipment Manufacturers
OFDM	Orthogonal Frequency Division Multiplexing
OMNet++	Objective Modular Network Testbed in C++

PHY	Physical Layer
QoS	Quality of Service
RACH	Random Access Channel
RERR	Route Error
RREP	Route Reply
RREQ	Route Request
RSS	Received Signal Strength
RSU	Road-Side Unit
SCH	Service Channels
SNIR	Signal to Noise + Interference Ratio
SUMO	Simulation of Urban Mobility
TCI	Traffic Control Interface
TCP/UDP	Transmission Control Protocol/User Datagram Protocol
TIS	Traffic Information System
UMTS	Universal Mobile Telecommunications Systems
VANET	Vehicular Ad-hoc Networks
Veins	Vehicles in Network Simulation
WAVE	Wireless Access in Vehicular Environment
WBSS	WAVE Basic Service Sets
WLAN	Wireless Local Area Network
WSMP	WAVE Short-message Protocol