

VANET FLEET MANAGEMENT



By

Muhammad UmairHadi

FarhanNazir

WardaNaim

Haleemah Zia

Project Advisor

Maj. AsimRasheed

Submitted to the Faculty of Electrical Engineering
National University of Sciences and Technology, Rawalpindi in partial fulfillment for the
requirement of a EE Degree In Telecommunication Engineering

June 2012

ABSTRACT

VANET FLEET MANAGEMENT

The project aims to assist long convoy / Fleet holders, such as Defense organizations, Bus Systems and Logistics Transport holders who do not have any on-road / centralized road management system. An application is developed which would assist for location update and internal messaging on dynamic maps. The application shall run on a processor small enough to be installed in a car. It shall use the car's battery for power and be connected to a touch screen graphicLCD on which a friendly GUI shall be displayed for the driver.

An application is designed for a three car environment, but one which is scalable enough to be extended for a number of cars as ever desired. The moving vehicles can send and receive messages to each other via text or by call. They may send private messages to a single car in their vicinity or broadcast the message to a certain number of cars. Most importantly, every vehicle would be able to view a local dynamic map, displaying it's own location and the locations of neighboring vehicles with respect to it's own. Through the map, the driver can not only be aware of the traffic conditions in his vicinity but also determine the best routes. The processor that has to be fitted in the car may vary with the demand and requirements of every company. Currently, the application was tested on Intel ATOM processor.

CERTIFICATE OF CORRECTNESS AND APPROVAL

Certified that the contents and form of project report entitled "**VANET FLEET MANAGEMENT SYSTEM**" submitted by 1) Haleemah Zia, 2) WardaNaim, 3) FarhanNazir and 4) UmairHadi have been found satisfactory for the requirement of the degree.

SUPERVISOR: -----

MAJASIMRASHEED

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose support and cooperation, a work of this magnitude would not have been possible.

ACKNOWLEDGEMENTS

We offer our utmost gratitude to our Supervisor Maj. Asim Rasheed Chaudhary for his constant guidance and encouragement. The project wouldn't have been possible without his invaluable efforts.

We also thank Mr. Yasir from Mohammad Ali Jinnah University ,our teacher Mr. Intisar Rizwan and our colleague Rabia Sultan from CS Dept for taking out time through their busy schedules and helping and supporting us throughout the development of our project.

Lastly, we acknowledge the Lab attendants of Broadband Lab and Image Processing Centre who rendered their help during the period of our project work.

TABLE OF CONTENTS

1.	Introduction	01
1.1	Overview.....	01
1.2	Organization of the Document.....	02
1.3	Current VANET Systems Worldwide.....	03
2.	Literature Review	06
2.1	VANETS.....	06
2.1.1	Concept of VANET.....	06
2.1.2	VANET Architecture.....	06
2.1.3	VANET Applications.....	08
2.1.3.1	Safety Application.....	09
2.1.3.2	Traffic Management/Control.....	09
2.1.3.3	User Applications/Infotainment.....	10
2.2	DSRC.....	11
2.2.1	DSRC Concept.....	11
2.2.2	DSRC Features.....	14
2.2.3	DSRC Architecture.....	15
2.2.4	DSRC Applications.....	16
2.3	GeoNet.....	16
2.3.1	Scope.....	17
2.3.2	GeoNet Domain.....	18
2.4	Local Dynamic Map.....	18
2.4.1	Advantages.....	19
2.4.2	Implementation of LDM in VANET.....	19
2.5	Controller Board.....	20
3.	Project Design and Environment Selection	22
3.1	Project Scope.....	22
3.2	Advantages of Proposed System.....	22
3.3	Environment Selection.....	23
3.4	Design of the Project.....	24
4.	Specifications	27
4.1	Hardware specification.....	27
4.1.1	GPS Dongle.....	27
4.1.2	GPS with Serial Port.....	28
4.1.3	Controller Board.....	29
4.2	Software Specifications.....	30
4.2.1	Operating System.....	30
4.2.2	Programming Language.....	30
4.2.3	Routing Protocol.....	31
4.2.4	GPS Inspector.....	31
4.2.5	Asterisk and Twinkle.....	33

4.2.5.1 Asterisk.....	33
4.2.5.1 Twinkle.....	33
5. SYSTEM MODULE.....	34
5.1 Application Part.....	34
5.1.1 GUI.....	34
5.1.2 LDM.....	35
5.1.3 Logging.....	36
5.1.4 Voice Chat.....	37
5.1.5 Configuration of Asterisk.....	37
5.1.6 Configuration of Twinkle.....	38
5.2 Networking Part.....	40
5.2.1 OLSR.....	41
5.2.2 Configuration of OLSR.....	42
6. Installation of Equipment.....	43
6.1 Installation of Equipment.....	43
7. Future Enhancements and Conclusion.....	49
7.1 Future Enhancements.....	49
7.2 Conclusion.....	51
Appendix A.....	54
Bibliography.....	71

LIST OF FIGURES

Figures	Page No
2.1 VANET Architecture	07
2.2 DSRC Architecture.....	15
2.3 C2C-CC Architecture and Scope of GeoNet	17
2.4 Geo Net Domain.....	18
2.5 LDM.....	20
2.6 Controller board.....	21
3.1 Design Over View.....	25
3.2 Detailed Design.....	26
4.1 GPS Dongle	27
4.2 GPS Dongle Serial Port.....	28
4.3 GPS Inspector.....	32
5.1 GUI.....	35
5.2 Asterisk Connecting andRunning.....	38
5.3 Twinkle Connecting and Running.....	39
5.4 Twinkle Connecting and Running.....	40
5.5 Routing Protocol.....	41
5.6 Configuration of OLSRD.....	42
6.1 Installation of GPS Dongle	44
6.2 Installation of OBU.....	45
6.3 Installation of LCD.....	46
6.4 220-12V Power Inverter.....	47
6.5 Car Lighter	47
6.6 Close View of OBU.....	48
7.1 The Front End of GUI at each Node.....	51

LIST OF TABLES

Tables	Page No
2.1 DSRC Features.....	14
4.1 Controller Specifications.....	29

KEY TO ABBREVIATIONS

VANET	Vehicular Adhoc Network
LDM	Local Dynamic Map
ISO	International Organization for Standardization
CALM	Communication Architecture for Land Mobile
WAVE	Wireless Access in Vehicular Networks
ITS	Intelligent Transportation system
DSRC	Dedicated Short Range Communications
OBU	On Board Unit
RSU	Road Side Unit
V2V	Vehicle to Vehicle
V2I	Vehicle to Infrastructure
V2V2I	Vehicle to Vehicle to Infrastructure
GPS	Global Positioning System
WiMAX	Worldwide Interoperability for Microwave Access
WiFi	Wireless Fidelity
FCC	Federal Communications Commission

CHAPTER 1

INTRODUCTION

1.1 Overview

Fleet management is the management of a company's vehicle fleet. With the number of vehicles / transport business increasing day by day, fleet management has become a need of time.

VANET (Vehicular Ad-Hoc Network) is a technology that uses moving vehicles as a node, thus allowing vehicles to communicate with each other and with Road Side Units. As cars fall out of the signal range and drop out of the network, other cars can join in. Back end connectivity is provided by public network (WIMAX, 3G).

VANET is claimed to be next big thing for road safety and infotainment on move. The moving vehicles can send and receive messages to each other via text or by call and have access to infotainment services as well. Most importantly, every vehicle would be able to view a local dynamic map, displaying the locations of neighboring vehicles with respect to his own vehicle. Through the map, the driver can not only be aware of the traffic conditions in his vicinity but also determine the best routes. In the project, an application is developed which depicts these functionalities and tested it in a 3-vehicle scenario.

1.2 Organization of the Document

Chapter 1 tells us about the concept of VANETS and meaning of Fleet Management and the reasons for its importance. It then proceeds towards explaining about the existing VANET architectures worldwide, listing some of the current projects which are currently being worked upon and implemented in certain parts of the world. The next chapter, Chapter 2 deals with the literature review carried out thoroughly at the beginning of project. VANETS architecture and applications, the Communication Protocols and the Hardware selection is also discussed in this chapter. In the next, Chapter 3 explains the design of the project is explained in detail. Illustrations explain the scenarios developed, Project scope and environment selection are also explained in this chapter. The environment selection is basically that of peculiar fleet/environment. Chapter 4 then deals with the hardware and software requirements of the project. It clarifies the need for the selected hardware components. It also explains the choice of the software environment and comparison with similar softwares. Chapter 5 deals with the explanation of system modules. The modules describe the working of the project clearly. Working and configuration of all modules and sub modules are explained in detail. Chapter 6 illustrates and explains the installation of equipment in the specific test environment i.e. cars as nodes. Chapter 7 explains future enhancements possible for this project. Results and analysis are presented and discussed in this chapter. A conclusion is presented in the end.

At the end of the document a list of any reference material used while preparing this document is provided and the application codes in Java are provided in an Appendix A.

1.3 Current VANET Systems Worldwide

VANET is an upcoming technology, widely being researched upon all over the world. It aims to revolutionize traveling at a very large scale through the implementation of road safety and management architectures. Underlying concept is to convert each and every vehicle into a wirelessly communicating entity, hence increasing driver's perception of horizon beyond human eye's range. Vehicles acting as communication nodes will make a network from small geographical region to virtually at global scale. The most targeted and ultimate goal is to provide a safer travel by creating early warning and timely response to the situations. However, to increase the market penetration, other classes of applications such as traffic control and provision of infotainment are also being considered. [1]These goals require backend Infrastructure connectivity to all nodes. The use of infrastructure may vary from architecture to architecture and from service to service.

Currently large number of countries is working on VANET architectures, either individually or in collaboration with regional regulatory authorities and car manufacturers. However most of the development is still in research phase with very limited practical deployment. Research standardization agencies, such as Institute of Electrical and Electronic Engineering (IEEE), International Standard Organization (ISO) and European Telecom Standardization Institute (ETSI) have

proposed different VANET architectures. Countries like Japan, USA and Brazil are working on the standards according to their own need and infrastructure layout.

Researchers are in process to find more intelligent and new services for VANET, such as involvement of pedestrians and amalgamation of trains in the communication architecture etc.

Fleet and convoy management systems can be categorized under traffic control, load management and timely response to eventualities. These services, if applied carefully, can not only enhance the overall efficiency of the network from vehicle management perspective, but can also improve the business model of these operators. A framework was developed for building a VANET based Fleet/Convoy Management system and, as a proof of concept, discuss and provide results of a small scale implementation. A test bed is developed for the real-time implementation of the architecture according to available communication resources in Pakistan.

Some of the projects (but not all) that are being worked upon regarding VANETS and DSRC., worldwide.[2].ISO is working on the development of DSRC and DSRC-like communications under two working groups named 'Working Group 15' and 'Working Group 16'. CALM Presentation given at ITS America Annual Meeting, April 2002.

The European CEN organization has developed its own set of DSRC standards, which includes standards for the Physical Layer (L1), Data Link Layer (L2), and

Application Layer (L7).The Japanese have developed their own set of DSRC standards published as ARIB T55

Korea Highway corporation adopted the Italian system) whereby 17,000 OBU are distributed, which are operating for a trial service. The active DSRC system is being tested.

The Electronic Toll Collection Application of VANETS is being tested in Brazil.

CHAPTER 2

LITERATURE REVIEW

2.1 VANETS

VANET is an upcoming technology widely being researched upon. It aims to revolutionize traveling at a very large scale through the implementation of road safety and management architectures. Underlying concept is to increase each and every vehicle into a wirelessly communicating entity hence increasing driver's perception of horizon beyond human eye.

2.1.1 Concept of VANETS

VANET's aim is the development of an Intelligent Transport System so as to provide a safer and easier travelling on road.[1] Vehicles shall be equipped with wireless communication technologies which shall enable them to act like computers and hence revolutionize the concept of traveling.

VANETs bring lots of possibilities for new range of applications which will not only make traveling safer but fun as well. The basic concept is to create an inter-communication network among vehicles, as well as between vehicles and the supporting infrastructure. The system pretends to offer drivers data concerning other nearby vehicles, especially those within sight.[1]

2.1.2 VANET Architecture

Architecture of VANETS is discussed widely in literature. The figure 2.1 illustrates it.

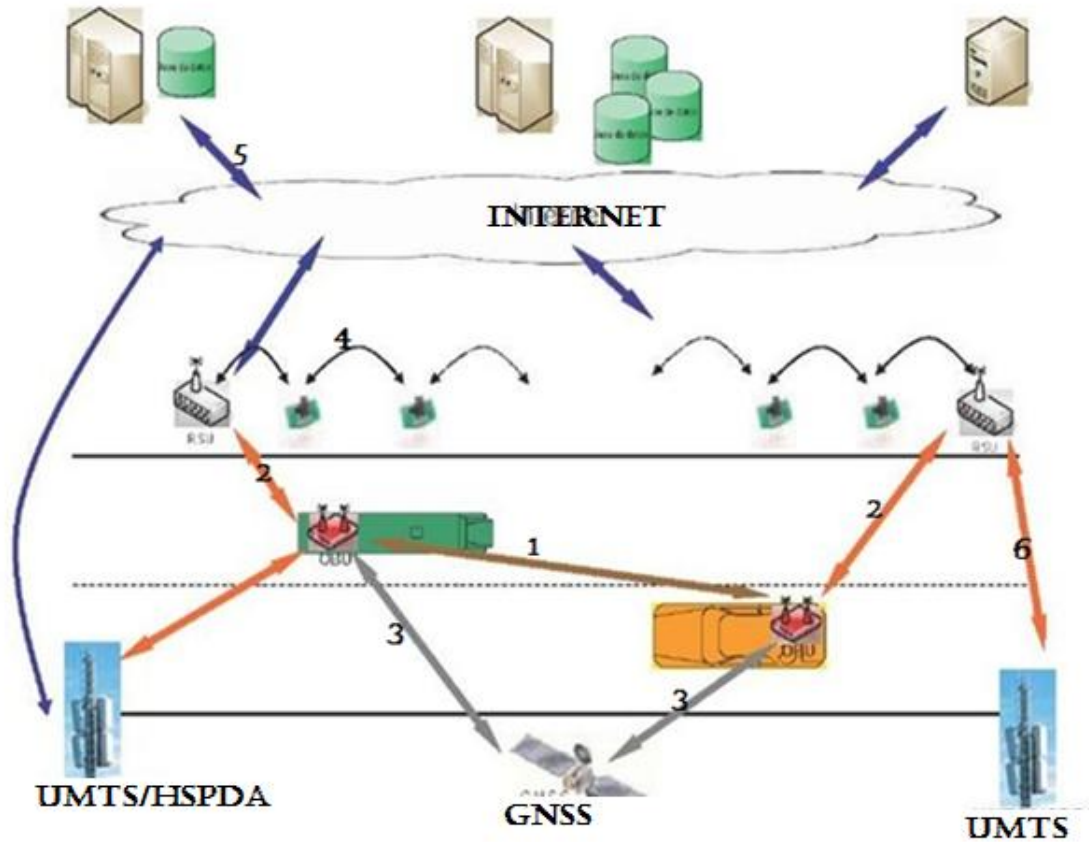


Figure 2.1 VANET Architecture

The architecture is based upon nodes called RSU's and OBU's and these are labeled in the figure 2.1.

On-BoardUnit (OBU): Computing device located in a vehicle that establishes the connection with the RSU and the other OBUs on nearby vehicles.

Roadside Unit (RSU): Computing device located on the roadside that provide connectivity support to passing vehicles.

The categories of possible communication are also illustrated.

'1' illustrates Vehicle to Vehicle (V2V) communication that relates all the data communication that takes place between various vehicles on the road.

'2' illustrates Vehicle to Infrastructure (V2I) communication involves all the communication aspects between vehicles and the supporting infrastructure.

'1' to '2' illustrates V2V2I communication whereby A vehicle communicating with an RSU but one that is out of range. It uses an in-range node to relay the messages with the help of a routing protocol. Such multi-hop communication is also possible between vehicles i.e V2V2V.

Label '3' illustrates the vehicles use GPS Receivers to communicate with GPS Satellites and hence obtain their geographic coordinates. Label '4' illustrates the RSU's are all connected to each other, hence forming an intranet. Label '5' illustrates these RSU's are further connected to the intranet of management servers (such as Daewoo terminal servers for the case of this project). The interconnection of these two networks is labeled as "Internet" in the figure 2.1.

Label '6' RSU's may also be connected to 3G stations.

The basic design of this project is based upon this very architecture , but with a few modifications owing to cost and environmental limitations. This will be discussed in greater detail in the design section.

2.1.3 VANET Applications

For the purpose of illustration, VANET applications may be divided into the Categories of safety, traffic management (also referred to as control applications in literature and Infotainment. The above are mentioned in order of priority. Infotainment application shall only be deployed once all other VANET applications are fully functional according to standardization organizations.

2.1.3.1 Safety Applications

Road safety applications can play an important role in avoiding accidents or at least minimizing the impact of accidents, if accident is unavoidable. This may be achieved using several mechanisms.[6] Keeping the driver informed of the prevailing road scenario at all times would alert him on potential risks for example, an accident, thus giving him enough time to apply brakes well before hitting the place. Other mechanisms proposed for the purpose of safety include work zone warning, stopped vehicle warning, low bridge warning for trucks, etc.

Timing is an important aspect when dealing with safety applications. Even a fraction of a second is important in decision making. For safety applications, lower layers are required to handle the imposed deadlines very strictly, while, for networking layer, routing is not big a issue. This is because safety messages are usually destined for neighboring vehicles only. Therefore multi hopping is seldom required.[5]

2.1.3.2 Traffic Management/Control

Another application for VANETs is to tackle road congestions and provide the best route to a driver with updated road conditions. This may be achieved with the use of certain road side infrastructure such as intelligent traffic signals, e-sign boards etc. Informing all nodes about road congestions ahead should definitely

assist them to reduce it and hence improve the capacity of roads aswell. Some other applications analyzed in literature include automated call to emergency services, en-route and pre- trip traffic assistance etc.

An interesting application , which is much focused upon in scholarly articles is e-Toll plaza. The concept here is that for toll payment, the need for vehicles to stop at plazas be eliminated. Vehicles can communicate with the roadside infrastructure, where it can be recognized and a fee can be charged against its account.

Another aspect is using intelligent traffic signals and hence reducing congestion at road intersections as well. These traffic signals would work and adjust themselves according to the traffic conditions at intersection and would communicate the status to other neighboring intersections as well. Neighboring intersections would then display this information on the 'e-sign boards' and also adjust their own signals accordingly. [7]

Traffic management applications involve extensive use of the road side infrastructure.

2.1.3.3 User Applications/ Infotainment

Apart from safety and control applications, infotainment (information and entertainment) applications have also been proposed for VANETs. The passengers, while travelling on road, may enjoy the facility of Internet connectivity where other conventional wireless internet connectivity options (Wi-

Fi, Wi-MAX etc.) are not available. When these technologies are present, a vehicle using these itself, can share its connectivity with other vehicles through the developed adhoc network. Peer-to-peer applications are also applicable in VANETs and discussed in literature as well, e.g., gaming, chatting, file sharing, etc.

The messages sent by such type of applications usually need to be delivered over multiple hops, hence routing will be involved.[5]In this project, the fleet management part which comes under 'Control. Management' category of the above discussed application is focused upon. Some basic safety applications like accident alert were integrated in the project as this was a main feature for any fleet management process.

2.2 DSRC

A short range (1000 m to 3000ft) ^[3,4] wireless interface designed specifically for vehicular communication(supports efficient delivery of data that is time and location dependent) . Federal Communications Commission (FCC) has allocated and licensed a frequency band ,based upon the WiFi standard IEEE 802.11 forDSRC Applications. The range of this band is 5850-5925MHz. Another unlicensed band about which literature quotes for DSRC is 902-928MHz.The communication is rapid enough for the ranges required in s typical VANET scenario

2.2.1 DSRC Concept

DSRC stands for dedicated short range communications. It supports both inter-vehicle and vehicle to infrastructure communications. Data rates range from 6 to 27Mbps^[3.4] (depending on distance) and operates on seven channels , one of these being control channel while others, service channels. Control channel announces all application requests and broadcasts safety messages as well. OBUs monitors the control channels periodically every 100 msec. An OBU wanting to avail a certain application will first register for that at the control channel. Service channel is where the actual services are configured.

According to the DSRC Implementation guide ^[3.4], each of the service channels is intended for use for differing power levels. For example, one channel may be allocated a higher power level and be used by fire trucks to request signal priority, whereas another channel may be used for payment and need low power to prevent adjacent lane reads.

Vehicles are allocated certain frequencies as Service Channels based on FDM. Lower Priority messages such as GPS correction information are sent at these.

The communication process handled by the DSRC technology is defined under the following steps in literature.

First step is Registration whereby Applications must be registered in both the RSU and OBU.

Registration parameters include Application ID (AID) and other application information.

Second step is Control Channel Monitoring whereby OBUs monitor the control channel to listen for application announcements and broadcast.

For Safety messages the control channel must be monitored every 100 msec for a minimum required amount of time (the minimum time may be variable based on load).

The third step is about Application Announcements. Applications may be announced periodically or on demand within a provider service table (PST) .PST contains AID among other application parameters .Periodic announcements are used on applications such as toll collection .On demand announcements are used to invite DSRC devices to participate in an application. These are also based on events (e.g., collision avoidance).

Next step is Application Initialization and Execution .Applications are initialized by matching the locally registered AID with an advertised AID (application announcement) received on the radio link .Application data exchange is usually executed on a service channel 6 service channels are available .In total, there are two types of wireless DSRC data communication:

First one is WSM – Wave Short Message: The WSM is useful for broadcast applications and applications that do not require infrastructure or extensive routing and addressing mechanisms. These particularly include short safety messages between vehicle and roadside devices. Example applications include intersection collision avoidance, traffic signal warning, and emergency electronic brake lights.

Other type is of Internet Protocol version 6 (IPv6) Datagrams which support safety applications (unicast, multicast or broadcast) and other applications include traveler information, parking payment, toll payment, fleet vehicle monitoring, and remote vehicle diagnostics. Extensive routing is required for such protocols and certain protocols are henceforth analyzed in literature for that. A brief description of these shall be covered in upcoming chapters.

2.2.2 DSRC Features

The main features for the two DSRC Bands are summarized in the table 2.1. These include frequency ranges, data rates, channel capacity etc.

Table 2.1 DSRC Features

PARAMETERS	902 - 928 MHz Band	5850 - 5925 MHz Band
SPECTRUM USED	12 MHz (909.75 to 921.75 MHz)	75 MHz
DATA RATE	0.5 Mbps	6 Mbps - 27 Mbps
COVERAGE	Large distances between communication zones	Overlapping communication zones needed and allowed
MAXIMUM RANGE	300 ft	1000 m (~ 3000 ft)
CHANNEL CAPACITY	1 to 2 channels	7 channels
POWER (Downlink)	Nominally less than 40 dBm (10 W)	Nominally less than 33 dBm (2 W)*
POWER (Uplink)	Nominally less than 6 dBm (< 4mW)	Nominally less than 33 dBm (2 W)*

2.2.3 DSRC ARCHITECTURE

Figure 2.2 illustrate the DSRC architecture. Three main layers are shown namely, Application layer, Logic link control, Medium Access control and physical layer.

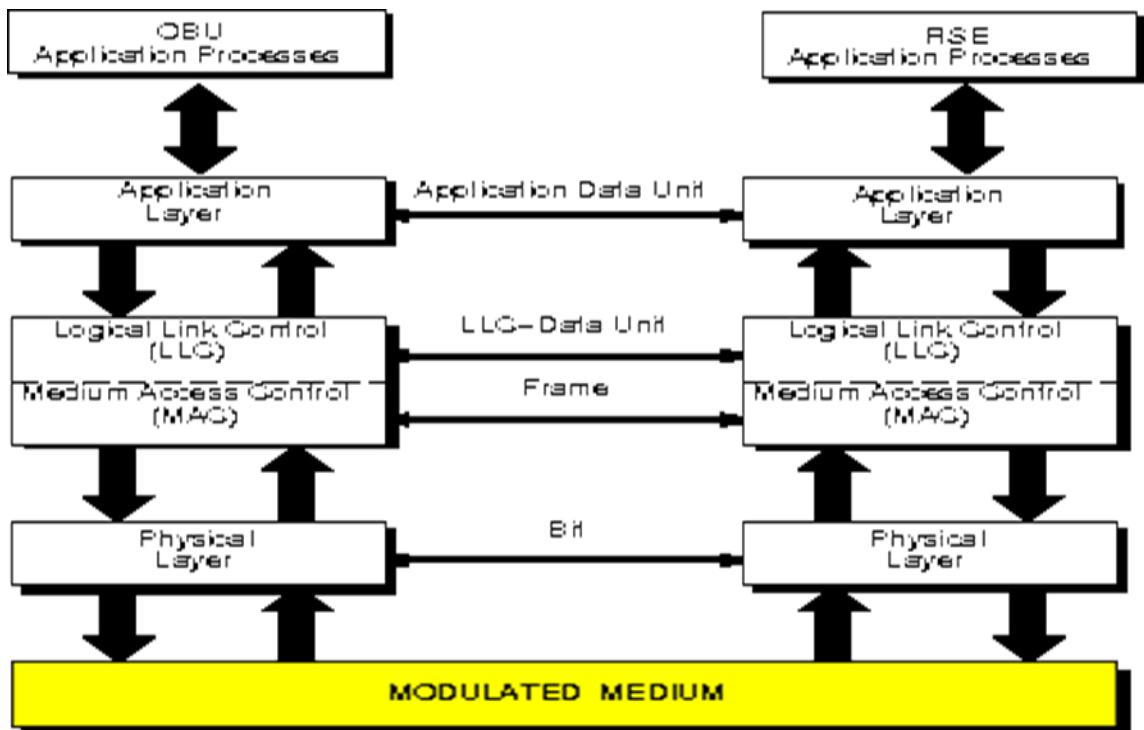


Figure 2.2 DSRC Architecture

The DSRC Physical Layer standard proposals provide specifications for two media. Microwave at 5.8 GHz, in particular the bands 5.795-5.805 GHz according to CEPT recommendation for pan-European usage, and 5.805-5.815

GHz available for national usage, Infrared at 850 nm .The DSRC Data Link Layer consists of the Logical Link Control (LLC) and Medium Access Control (MAC) sub-layers. The DSRC Application Layer provides a service interface for the applications and specifies “fragmentation, application multiplexing and a common initialization mechanism” for each communication process between vehicular equipment (OBU_ and the roadside equipment (RSU) [3.4]

2.2.4 DSRC APPLICATIONS

The DSRC Applications are broadly categorized into two , “PUBLIC SAFETY APPLICATIONS “ and “PRIVATE APPLICATIONS”. Public safety include those such as vehicle safety inspection. Driver’s daily log,tool collection and warnings such as low bridge, work zone etc. The Private category refers to applications such as fleet management, rental car processing etc[3.4]

The idea and applications of IPv6 are implemented in this project.However the actual DSRC chip in this project could not be used as it was very costly and hence not feasible at student level. In fact in most projects worldwide, similar approach is being implemented for projects till yet.

2.3 GeoNet:

GeoNet is European standard of wireless communication in vehicular networks developed by European Telecom Standard Institute (ETSI). The concept behind GeoNet is to combine IPv6 and geonetworking for Intelligent Transport Systems

(ITS).ITS incorporates Internet Protocol (IP) standard for uniform exchange of information in vehicular systems and Internet for ensuring interoperability, portability and wider deployment.

With the increasing number of vehicles day by day IPv4 address will get exhausted by 2011/2012, hence IPv6 is deployed in place of IPv4. Also IPv6 comprises new features like NEMO (network mobility) to support the mobility of complete networks; a feature very much essential for ITS architecture.

2.3.1 Scope:

C2C-CC (Car-to-Car communication consortium) is a European organization for setting up standards to ensure road safety and providing information services. Its architecture is shown in the figure below. GeoNet basically involves working on the network layer, being transparent to the Application and the Transport Layer above it. It integrates the working of IPv6 layer and C2C-CC geonetworking layer. Figure 2.3 displays the C2C-CC Architecture and Scope of Geonet.

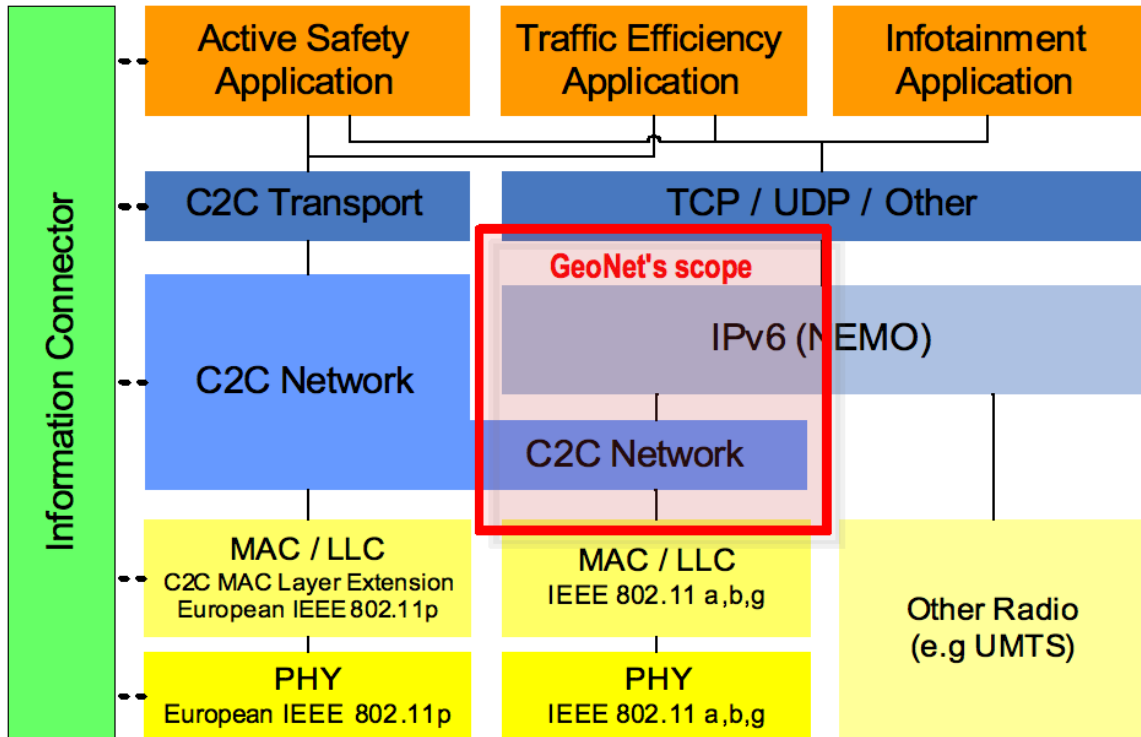


Figure 2.3 C2C-CC Architecture and Scope of GeoNet

2.3.2 GeoNet domain:

All the nodes (OBUs and RSUs) together form a GeoNet domain in a vehicular adhoc network known as VANET. In this domain routing is performed using GeoNet addressing and routing.

The figure 2.4 illustrates such a decentralized network where communication between application unit (AU) functioning as IPv6 node and it's attached OBU takes place over the IPv6 layer, while between two OBUs takes place over the C2C network layer. [3.15].Figure 2.4 shows the Geonet domain and main features in a compact but explicable format.

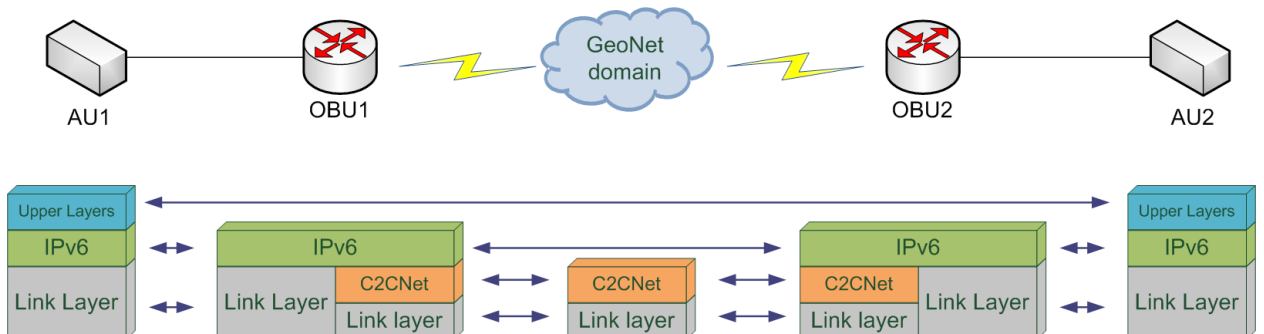


Figure 2.4 GeoNet Domain

2.4 Local Dynamic Map:

Local dynamic map refers to the road map geometry representing static information along with the relative positions of the neighboring vehicles/nodes in real time. Thus the road geometry from a standard digital map is integrated with the information collected by the vehicles and the infrastructure.

An LDM for each vehicle/node in this system was developed, in order to display its surrounding environment with reference to other vehicles/nodes such that the if node that maintains the LDM is moving , the map window will be moving as well ,with the node at its center point.

2.4.1 Advantages:

The system is able to provide more information than a standard digital map. It is also able to represent highly dynamic objects like fast moving vehicles. It also Provides a database that gathers the static information and attributes of the neighboring vehicles. Also, it enhances the traffic efficiency by locally storing and handling huge amount of information received from vehicles like different sensors

installed and the road side equipment. It is very essential for the safety of the vehicles preventing collisions and traffic hazards.

2.4.2 Implementation of LDM in VANET:

In order to have a complete perception of the environment around the vehicle, LDM is an essential feature for any VANET architecture. Each node maintains a LDM in client-server architecture. LDM holds all the information gathered from the sensors installed in the vehicle or from the surrounding infrastructure (like other vehicles or road side units) after it is fused on to a single platform. This platform is referred to as data fusion module. After the fusion, an enhanced and elaborated scenario of the surrounding environment and traffic conditions is generated that is stored and displayed in LDM. Figure 2.5 illustrates LDM features and architecture, some basic components.

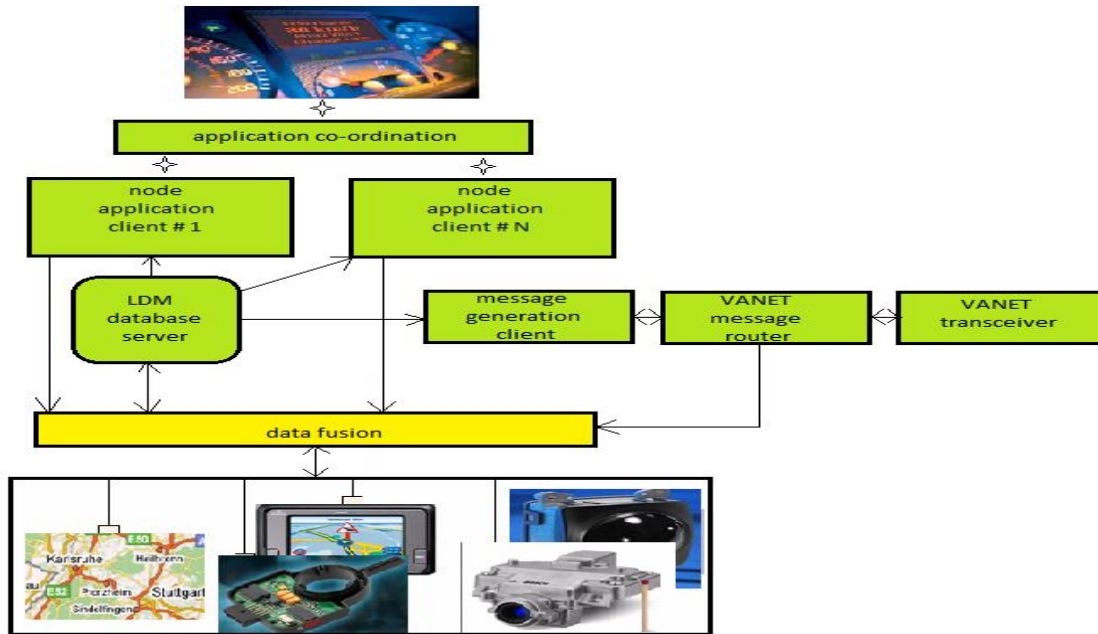


Figure 2.5 LDM

2.5 Controller Board

The requirement was a controller board with integrated hardware components illustrated above along with a GPS Receiver. Moreover the processor required was to be fast enough to run the application and the memory large enough to store the GPS coordinates' database. Also, Linux operating system was a requirement of the chosen routing protocol, so a controller board was required which had the capability to run Linux over it.

Keeping these all features in mind and the availability in college, the choice of Intel ATOM processor was made for testing the application. Figure 2.6 is a screen shot of the controller board used in the project.



FIGURE 2.6 Controller board

Chapter 3

Project Design and Environment Selection

3.1 Project Scope:

The project proposes a system which will assist and manage long convoy and fleet. The project involves the understanding and implementation of VANET architecture along with the integration of different applications.

Each vehicle is equipped with an OBU. The moving vehicles/nodes are connected into an adhoc network .Applications such as voice chat, text chat and viewing own location on a local dynamic map (which in addition to its own location plots the coordinates of other vehicles) are supported on each OBU. The adhoc network in this project carries out relay operation for out-of-range vehicles.

The system also consists of a static roadside unit that will provide the drivers with information about their current area, can deliver text or voice data to remote OBUs or can even access the internet.

3.2 Advantages of Proposed System:

The system provides an intelligent solution with certain advantages to the increasing demand for information on the current location and specifically for data on the surrounding traffic, by converting the vehicles directly to nodes/router. There is no need for a preexisting infrastructure since it is a type of ad-hoc network. Relatively good availability of resources (esp. energy) is provided

compared to small mobile devices. Size of the cars/nodes provides enough space to embed/carry infrastructures/ PC based systems: something which is not possible when small devices such as cell phones are used as nodes. Law enforcement will be very easy to implement since each vehicle shall be registered with the authorities. Emergency vehicles often find it difficult to travel through the heavy traffic flow. The system presents a solution by informing the vehicles ahead for clearing the road.

Traffic flow will be more effective and road safety will be ensured as the driver will be alerted about the upcoming driving hazard. The driver of vehicles will be alerted if they are approaching a vehicle at speed, thus preventing collision.

3.3 Environment Selection:

To accomplish this task three units were setup; two laptops and one board(Intel ATOM) on which the application is running. Out of these three units, one is serving as RSU and two as OBU's. A management server providing the function of monitoring each OBU and on which the voice chat server is running was also setup and configured.

The Management Server is a Public IP and hence can be accessed remotely by any node having Internet Access/connected to WiTribe(The RSU Network).

RSU was placed in the SDR lab while the OBUs were to be installed within a vehicle moving on roads within MCS. Since the system was setup to work in MCS ,onlythe road map of MCS on which the current locations of each OBU was downloaded and were displayed.

3.4 Design of the Project

Considering the VANET architecture purely from fleet and convoy management perspective, GeoNet was selected as base line architecture. It dove tailed the overall design to make it simpler and generic according to ground needs. The concept of multi interface being used by CALM and GeoNet was considered as compulsion to provide backhaul link in the absence of large scale deployment of VANET. Resultantly, the nodes could communicate directly to management server over the internet using long range communication. V2V communication was made available using short range Wi-Fi links.

After detailed analysis of geometry and design requirements of fleet and convoy operators, different services were considered to be provided to the nodes. These services were considered over and above to the road safety applications and services defined in the standard VANET designs, such as alarm for emergency brakes, accident alert etc. In the absence of deployed architecture, need was felt to provide route alerts, road and weather condition warnings etc through central server basing on the concept of geographical tagging.

Figure 3.1 and 3.2 illustrate the design of the project. Figure 3.1 shows an overview while the second one displays greater details.

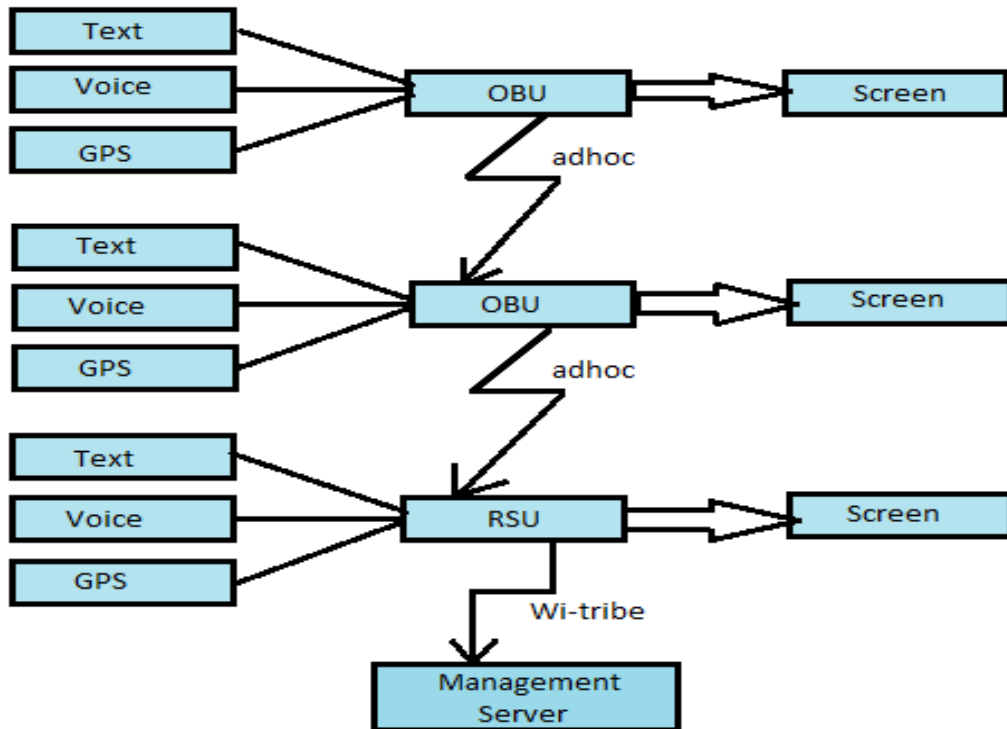


Figure 3.1 Design Overview

An OBU is designed so that it can be conveniently installed in vehicles. OBU is basically a controller board (Intel ATOM Processor) which is equipped with certain modules required for the successful running of the application.

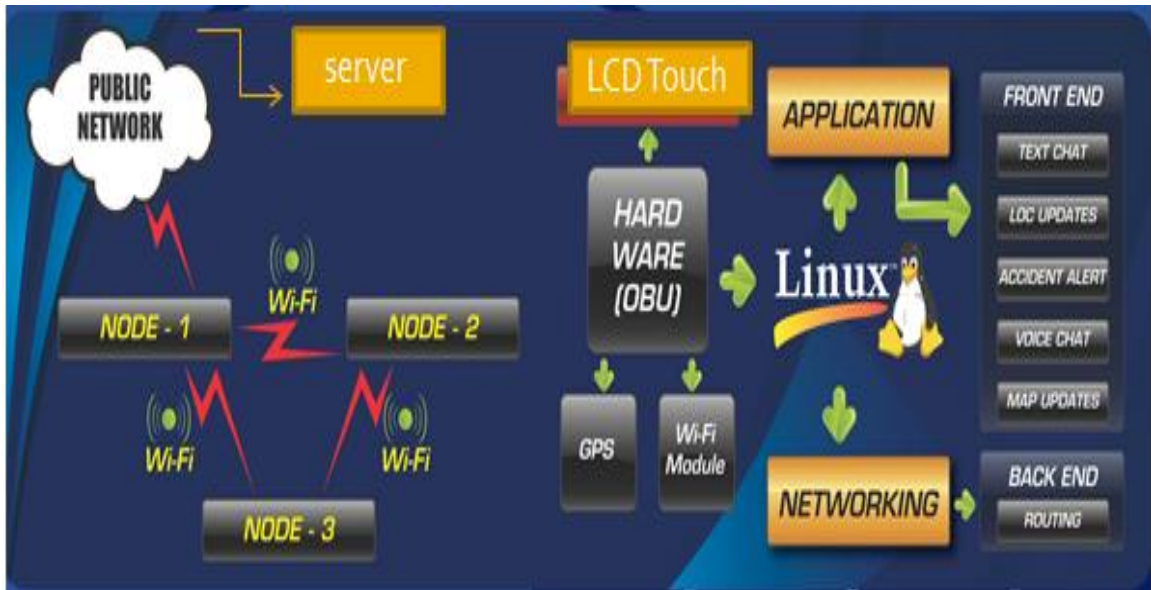


Figure 3.2 Detailed Design

Each OBU is equipped with a microphone for the voice input, USB GPS dongle for determining the current location of each vehicle, Wi-Fi module (for providing ad-hoc access interface in place of the very expensive DSRC module) and touch screen graphic LCD for displaying a user friendly GUI. Text input can be provided with the on screen LCD keyboard; however a separate USB keyboard can also be attached. All the OBUs are connected in an ad-hoc network.

The designing of RSU is similar to the OBU with a modification as can be seen in the first figure above. RSU is connected to the WiTribe Network and through it, to the Management Server. In the second figure 'Node 1' is basically shown as the RSU.

CHAPTER 4

SPECIFICATIONS

4.1 Hardware specifications

Specifications of Hardware components are discussed in this section. Hardware specifications include those of the components used such as GPS Dongle , controller board etc are explained. The main features which were essential for the proper functioning and efficiency of the system were kept in view and are henceforth discussed here.

4.1.1 GPS Dongle:

Figure 4.1 is a screenshot of GPS dongle used. The Dongle Sirf GM1-86 was chosen, because of its availability in Pakistan. It is compatible with most of the



Figure4.1GPS Dongle

Softwareand works both in Linux and Windows.

Features of the GPS dongle include maximum update rate of 1Hz and baud rate of 4800 bps. It is a non-DGPS (Differential GPS) having USB interface port with position of 5-25 meter CEP without SA, velocity of 0.1 meters/second without SA and time of 1 microsecond synchronized GPS time.

It has position accuracy of less than 2.2m, horizontal (95% of the time) and less than 5m, vertical (95% of the time).

4.1.2 GPS with Serial Port

Figure 4.2 is a screenshot of the GPS Module with serial port. It has very high sensitivity (Tracking Sensitivity: -159 dBm). It is a HOLLX M89 with baud rate of 4800 bps and supports NMEA 0183 data protocol.

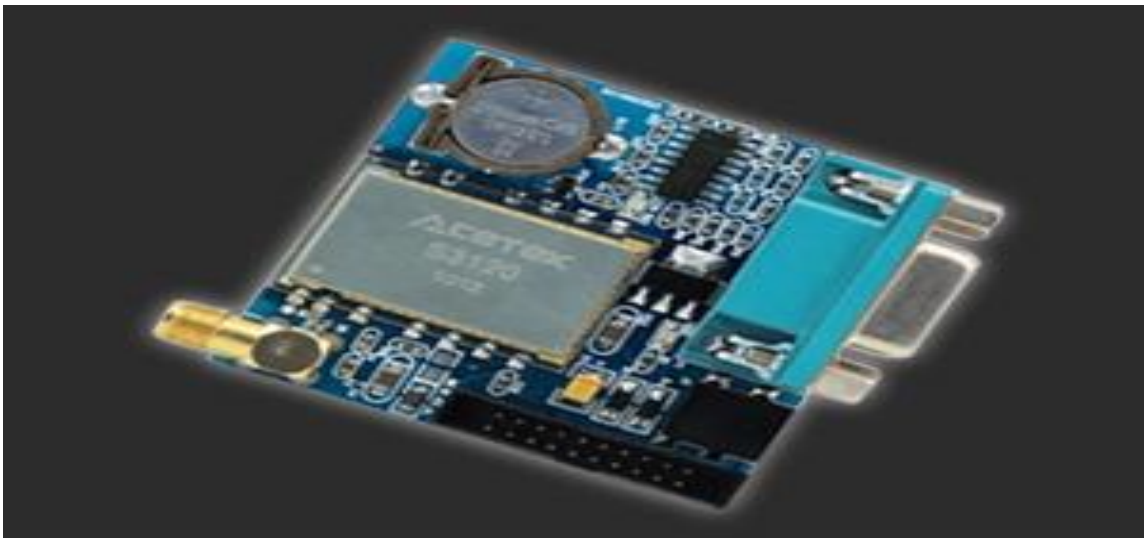


Figure 4.2 GPS Dongle Serial Port

4.1.3 Controller Board

Intel Board has specifications as shown in table 4.1

Table 4.1 Controller Specifications

Processor	1.6 GHz Intel Atom 330 (dual core) 533 MHz FSB
System Memory	Up to 2 GB of memory
VGA	Built-in Intel GMA 950 Graphics
Expansion Slots	PCI
Onboard IDE	1 ATA 100 40-pin
Onboard Serial ATA	2 SATA (3 Gb/sec.) connectors
Onboard USB	8 USB 2.0
Onboard LAN	Realtek RTL8111C GbE 10/100/1000
Onboard Audio	Realtek ALC662 5.1 channel HD codec with multi-streaming support
Back Panel I/O	1 Parallel port 4 USB 2.0 ports 1 VGA port 1 LAN port 1 RS-232 COM port 1 PS2 keyboard port 1 PS2 mouse port 3 Audio jacks: line-out, line-in, mic-in
Onboard I/O Connectors	1 IDE slot 2 SATA connectors

	4 USB 2.0 via 2 pin headers 1 P4 power connector Front panel audio pin header 1 Fan pin header 1 Back panel pin header 24-pin ATX connector
BIOS	Intel BIOS 4 Mb flash memory
System Monitoring & Management	Power management, Wake on USB, Wake on PCI, Wake on keyboard, Wake on mouse, Wake on LAN, Fan speed control
Operating Temperature	0°C ~ 55°C
Form Factor	Mini-ITX (17 x 17 cm)

4.2 Software Specifications

Specifications of Software modules are discussed in this section. Specifications include those of the separate modules such as LDM, OS, Programming language etc are explained. The main features which were essential for the proper functioning and efficiency of the system were kept in view and are henceforth discussed here.

4.2.1 Operating System

Linux is a strong support for network functionality; client and server systems can be easily set up on any computer running Linux. It can perform tasks such as network backups faster and more reliably than alternative systems. The routing

protocol, OLSRD specifically requires Linux as the OS. Choice of the routing protocol is discussed in section 5.2. [3]

4.2.2 Programming Language

Java was chosen as the programming language for the complete project. It is a platform independent language hence can easily run on the board without installation of any major operating system. Java is much simpler than C++ is because Java uses automatic memory allocation and garbage collection where else C++ requires the programmer to allocate memory and to collect garbage.

Java is multithreaded: Multithreading was a necessity in network programming as well as LDM part of the project.

4.2.3 Routing Protocol

Wikipedia defines Optimized Link State Routing Protocol (OLSR) as an IP routing protocol optimized for mobile ad-hoc network. It is a proactive link-state routing protocol, which uses *hello* and *topology control* (TC) messages to discover and then disseminate link state information throughout the mobile ad-hoc network. Individual nodes use this topology information to compute next hop destinations for all nodes in the network using shortest hop forwarding paths. OLSRD stands for OLSR daemon. A daemon is a computer program that runs as a background process, rather than being under the direct control of an interactive user.”

OLSRD was chosen as the best routing protocol after thorough literature review. Throughout it has been proven as one of the most suitable existing protocols for VANETS. References are given in Bibliography [4]

4.2.4 GPS Inspector

For the project, the source code similar to that of GPSInspector software was used that was available on the web, with some required modifications. The Choice of GPSInspector was made after thorough consideration and study, testing of many other software. It is an Open source software with source code available in java .Moreover, It has the ability to cache the map tiles and then display locations offline, a major requirement of the VANET architecture. Figure 4.3 is a screenshot of the GPSInspector.

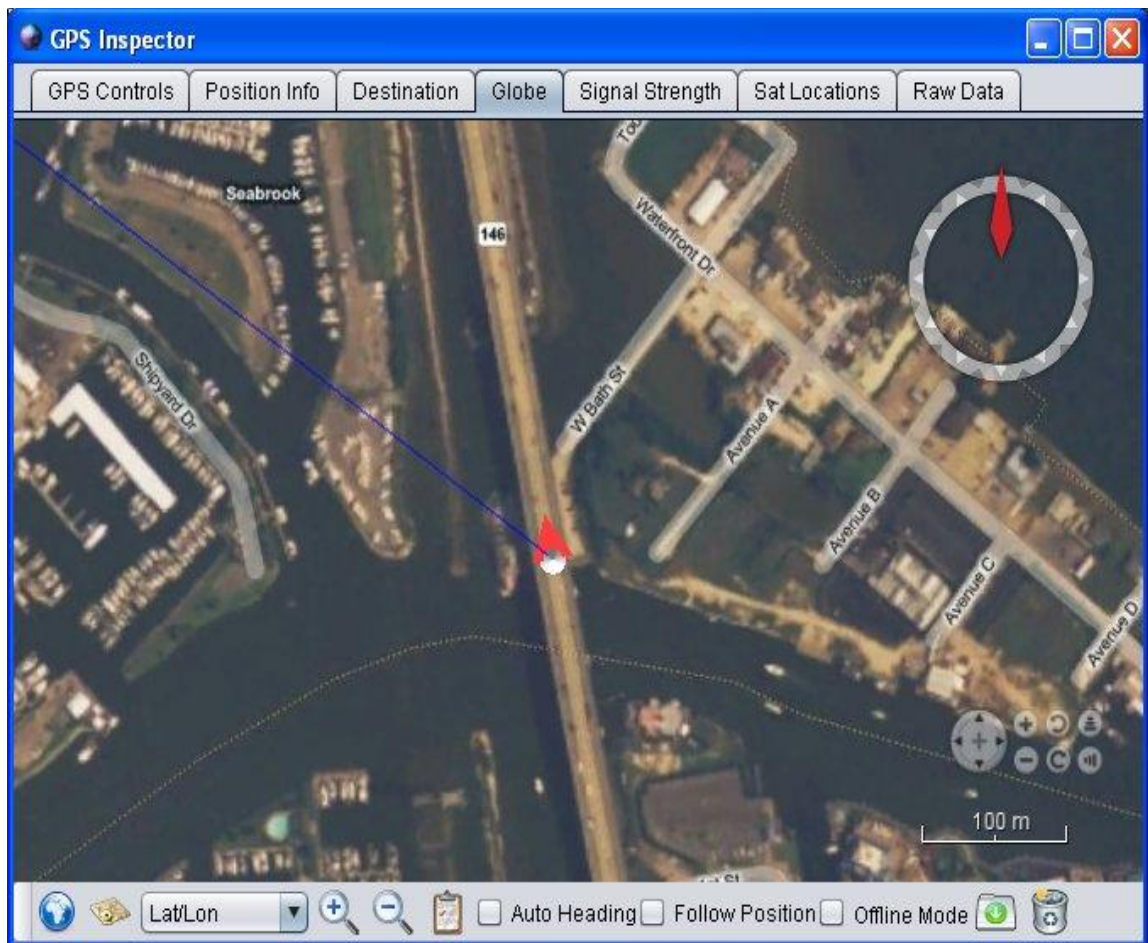


Figure 4.3 GPS Inspector

Major features of GPS Inspector include serial port scan to make it easier to find GPS devices, ability to capture raw NMEA data to a file and to replay/simulate the data, built in simulated NMEA data (mostly used for debugging), destination direction and distance pointing, with the ability to search for destinations using plain text searches, 3D/2D plotting of location (and optionally path history) with streaming global satellite imagery layers (via World Wind Java), bulk imagery downloader for caching imagery for offline use.[4] It also displays bar plots of signal to noise ratios of satellites (tracked and in the GPS Almanac), polar plot showing the azimuth/elevation locations of the GPS satellites, raw NMEA data and lots of GPS output data: position, fix info, fix quality, speed, heading and fix time.

4.2.5 Asterisk and Twinkle

In the application, Asterisk is used as the voice chat server and Twinkle as the softphone. Both the softwares' features that were essential for the efficiency of the system were kept under consideration and are listed here accordingly. Further details are also available in appendices and reader may also refer to the respective websites.

4.2.5. Asterisk

It is a software implementation of a telephone private branch exchange (PBX). Like any PBX, it allows attached telephones to make calls to one another, and to connect to other telephone services including the public switched telephone

network(PSTN) andVoice over Internet Protocol(VoIP) services. The Asterisk software includes many features available in proprietary PBX systems:voice,conference calling, interactive voice response(phone menus), andautomatic call distribution. It was originally designed for Linux.

4.2.5.2 Twinkle

Twinkle was chosen because of its easy and widely tested configuration with Asterisk. [5]It runs specifically on Linux. It has a number of attractive features some of which include two call appearances (lines), multiple active call identities, mute, call reject, repeat last call, do not disturb and auto answer. It supports multi party calls and short messages as well.

CHAPTER 5

SYSTEM MODULE

The system is divided into various modules, each module works independently taking the input from the other module and giving the desired result to the next level. There are mainly two parts of the entire system, Application part and Networking part. Then there are further distributions of the Application part that perform various functions that are elaborated upon in the coming pages.

5.1 Application Part

The application part is divided into following sub parts. We shortlisted four major applications as bare minimum to provide optimum efficiency to the fleet/convoy management system. The significance of four targeted services namely, text chat, voice chat, location updates and situational alerts are also discussed. But first the softwares that were used:

5.1.1 GUI:

Figure 5.1 shows the screenshot of the graphical user interface of the application. It enables the user to perform all the basic operations on a single tap onto the touch screen panel of the OBU. The GUI has been developed in Java and open source codes have been taken as an aid. Map shows the position of own vehicle plus the position of all the vehicles in range, the map would always be centered

at own position, the concept of local dynamic map will be described in detail in coming pages. The user can check the serving satellites, their relative positions on a polar plot and their signal strengths. One can nevertheless log his/her GPS data into a file for further use.

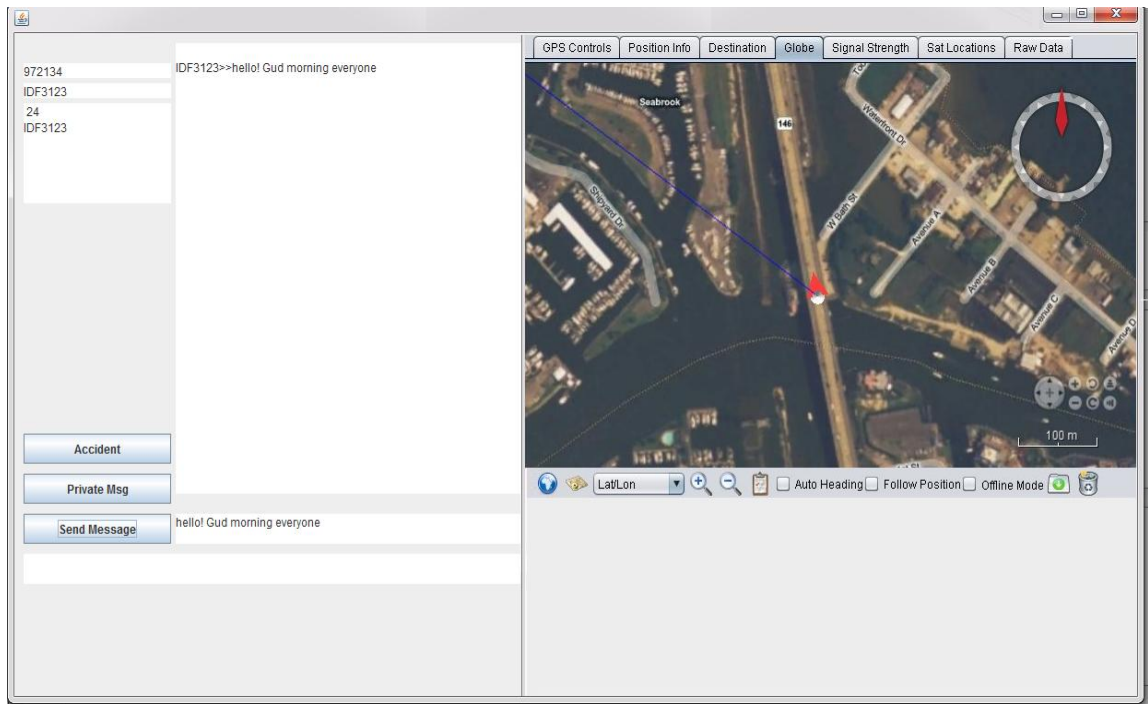


Figure5.1GUI

5.1.2 LDM:

Setting up a local dynamic map at on each OBU required a GPS and an application supporting it and giving the output in desired format and shape. The maps are cached and hence displayed in offline mode.

Local Dynamic Map has three modules; Front End, Database and Command/Control module which handles all the queries and uploads chunks of images according to the length of the screen and relative position of vehicles.

There exists a backend Static Map and a Front End Dynamic Map with multiple layers. For querying the data there are two options available: Centralized Server and Localized Server. In the Centralized Server all vehicles send their updates/own locations to one single server and query this server for all other vehicles' location and hence maintain their database .In Localized Server all vehicles directly communicate with each other updating one other about their locations and hence maintaining their database.

Both approaches shall be employed in the project as Centralized Server is required for fleet management whereby vehicles are not in close vicinity to each other, while Localized Server is employed for Convoy management as reverse is true here. The Public IP server in MIS serves as the Centralized server. The network part updates the LDM with latest coordinates all the vehicles in the local network, the LDM in turn plots the coordinates. This cycle continues after a fixed interval that is set by the system designer depending upon the available network and system requirements.

5.1.3 Logging:

The distance covered by vehicles per day and their status during the run can be logged into a file for addressing the queries of the remote fleet manager if he wants to check the distance covered by his vehicles and the time wise positions of his vehicles moving at a remote location. This feature is added into the application by simply saving the time and position along with the vehicle id's in a text file, and using the AWK tool to get the required information and displaying it

on the screen of the fleet manager. The fleet manager just has to set the parameters of his query. This adds an enormous strength in the system as these positions and timings cannot be tempered by the drivers and thus resources can be saved from getting wasted and misuse of the fleet vehicles can be strictly controlled.

5.1.4 Voice chat:

Voice chat feature can be activated by a simple tap on the voice chat button, the user can then select the other party whom he/she wants to call from the list of vehicles in range. Open source code has been used for adding this feature. The server used is Asterisk and it works on softphone 'twinkle' running on every OBU. The server is only enabled on the Management Server and all vehicles registered to it and having twinkle can carry out voice chat.

5.1.5 Configuration of Asterisk:

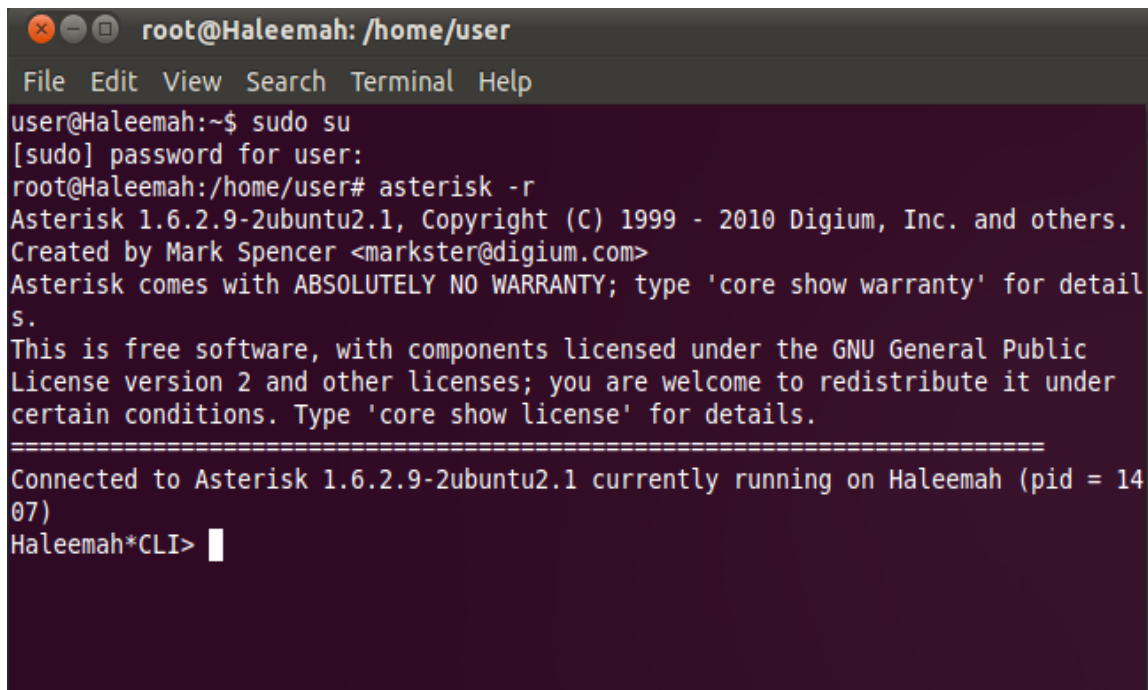
Asterisk needs to be configured according to the nodes that have to use it for voice chat. All nodes must be first registered to Asterisk. The sip.conf file in /etc/asterisk folder contains this information.

Three nodes were configured and hence provided registry information for these three. The user names are defined and type is defined for each user. Three types are provided in asterisk user (outgoing only), peer(incoming only) and friend(both).All three nodes were defined as "friend". The sip.conf file is provided in APPENDIX A and explanations are commented over there.

After all participating nodes are configured, a dialing plan must also be provided to asterisk. There is a file called extensions.conf in the /etc/asterisk folder. It contains the "dial plan" of Asterisk, the master plan of control or execution flow for all of its operations. It controls how incoming and outgoing calls are handled and routed.

Dialing plan for three nodes CAR_A, CAR_B and CAR_C was provided by assigning them sip numbers which are just like telephone numbers in out PSTN network. Fullextensions.conf file is provided in APPENDIX A.

Figure5.2 is the screen shot of connecting and running the Asterisk server.



```
root@Haleemah: /home/user
File Edit View Search Terminal Help
user@Haleemah:~$ sudo su
[sudo] password for user:
root@Haleemah:/home/user# asterisk -r
Asterisk 1.6.2.9-2ubuntu2.1, Copyright (C) 1999 - 2010 Digium, Inc. and others.
Created by Mark Spencer <markster@digium.com>
Asterisk comes with ABSOLUTELY NO WARRANTY; type 'core show warranty' for details.
This is free software, with components licensed under the GNU General Public
License version 2 and other licenses; you are welcome to redistribute it under
certain conditions. Type 'core show license' for details.
=====
Connected to Asterisk 1.6.2.9-2ubuntu2.1 currently running on Haleemah (pid = 14
07)
Haleemah*CLI> █
```

Figure5.2 Asterisk Connecting and Running

5.1.6 Configuration of Twinkle:

Profiles need to be configured on twinkle. User names and sip numbers are added here. In the project ,one profile each was made on every node 'CAR_A:SIP=1000','CAR_B:SIP=1002' and 'CAR_C:SIP=1004'.

After that, IP addressing of host/Asterisk Server is provided in each profile and any natting information.

Figure5.3 and 5.4 are the screenshot of a profile.

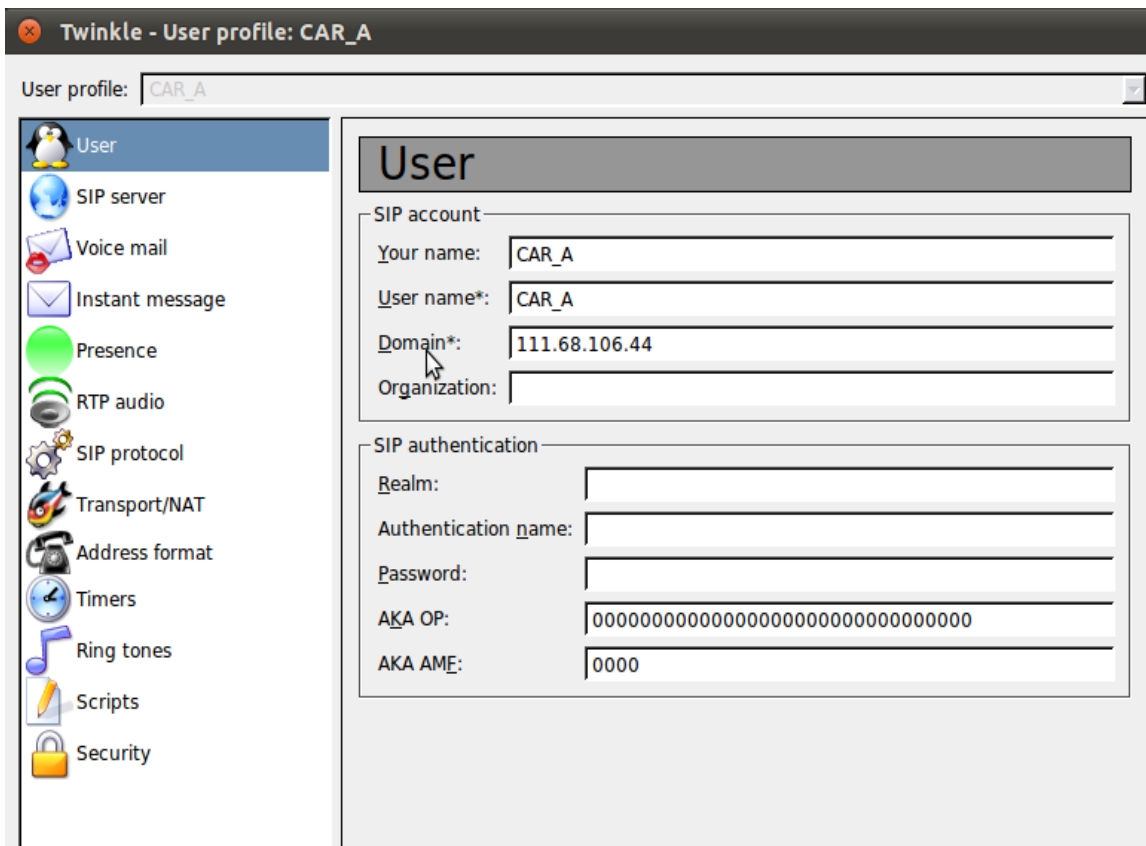


Figure5.3 Twinkle Connecting and Running

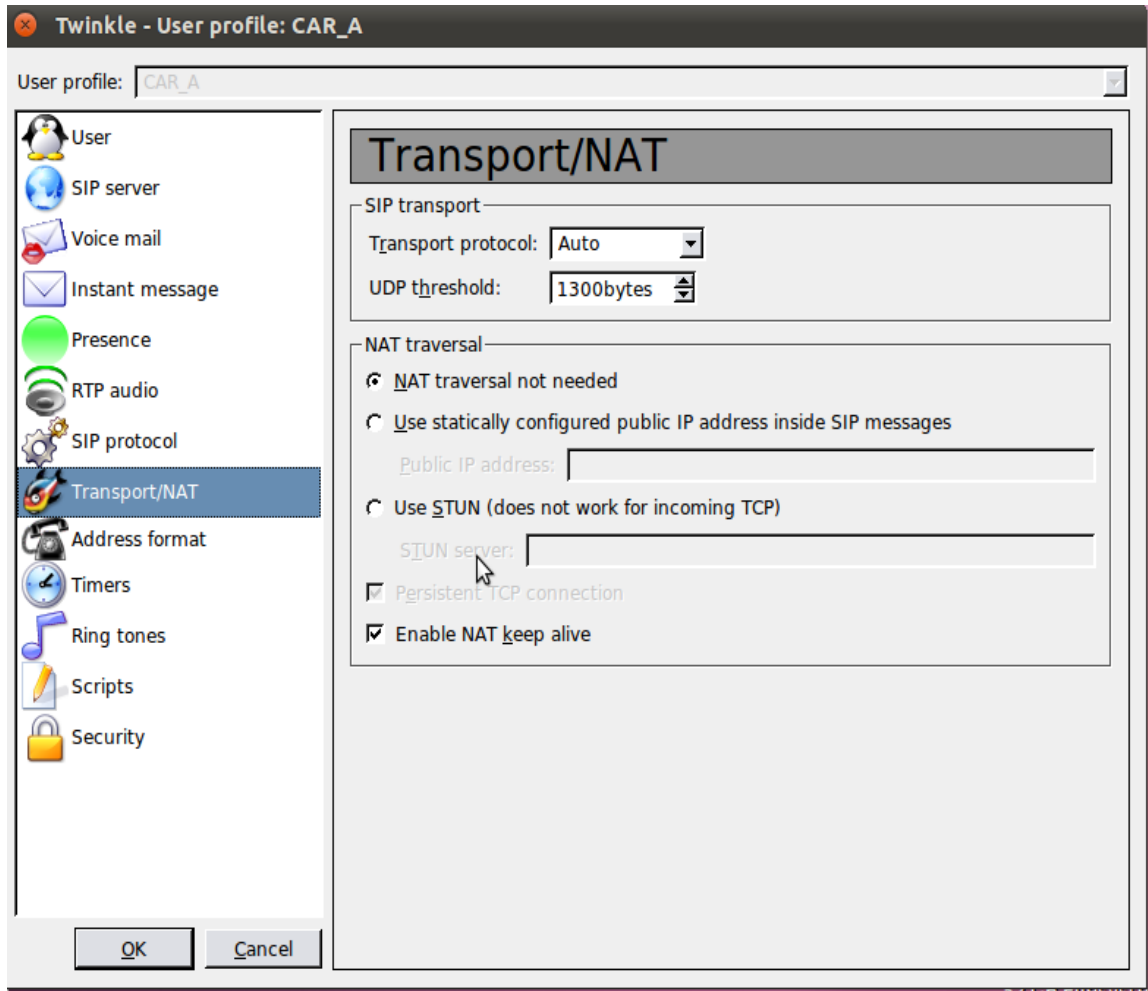


Figure 5.4 Twinkle Connecting and Running

5.2 Networking Part

WiFi is used for communication between various OBUs. First of all, the vehicle which enters the network broadcasts its ID and coordinates so that all the nodes in the network get to know about its position and ID and thus its position is plotted into the LDM of every OBU. To achieve that every vehicle transmits its coordinates every 3 seconds so that its position and status are updated in the entire network.

If any OBU wants pass a message to the entire network, the driver has to simply type a message and press the send button, the public chat window will display the message to all the nodes in the network. Same socket is used for all kinds of communication. Every node listens to the same socket and sends its messages on the same socket. The difference is in the identifiers of various kinds of messages. The identifier defines the kind of handling of the input stream by the OBU.

Incase of private chat, the sender has to select the recipient from the list of available vehicles and send the message; again this message would be received by all but parsed and displayed to the recipient only.

Each OBU generates various streams of data that are broadcasted in the network. And in turn all the other OBUs including the sender itself make use of the information in that stream by parsing the data of every stream.

5.2.1 OLSR:

OLSR has been used as the routing protocol that performs the function of traversing the information on the network until it reaches the destined node.



Figure 5.5 Routing Protocol

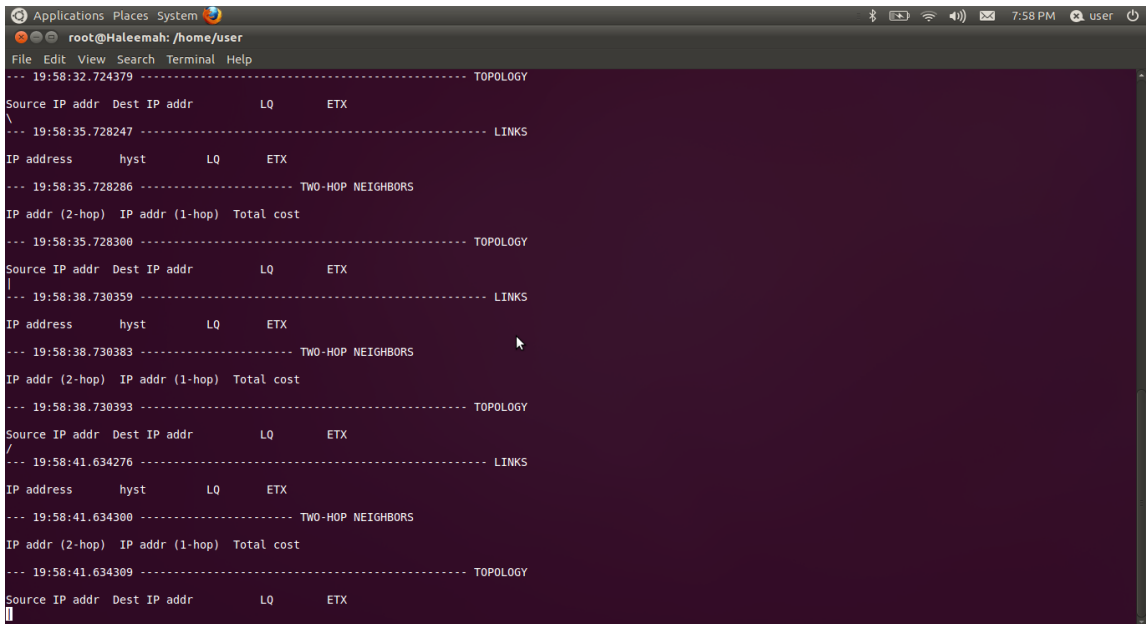
Routing protocol was tested by moving the three nodes so that node A and C must speak through node B to reach each other. Three nodes A, B and C were moved such that node A and C remain connected to node B, but loose connection which each other at a certain point. Node A was stationed in the RnDBlock, Node B

outside EC Lab 1, while Node C was moved till library and once the third node entered library, it lost connection to A . When node "A" was moved far enough away so that it can't hear node "C", all traffic was being redirected through node "B" to C. Figure5.5 is a scenario where Node A and C are communicating to each other through Node B.

5.2.2 Configuration of OLSRD:

OLSRD needs to be configured according to the wireless interfaces and amount / mechanisms of routing required etc.

A file lying in the root folder (/etc/olsrd.conf) was configured such that the protocol works well with wireless adapters of all devices. A copy of the file is provided in APPENDIX A. Explanations of various parameters are commented in that copy. Figure5.6 is a screen shot of OLSRD running on a single device.



```
root@Haleemah: /home/user
File Edit View Search Terminal Help
--- 19:58:32.724379 ----- TOPOLOGY
Source IP addr  Dest IP addr      LQ      ETX
\
--- 19:58:35.728247 ----- LINKS
IP address      hyst      LQ      ETX
--- 19:58:35.728286 ----- TWO-HOP NEIGHBORS
IP addr (2-hop) IP addr (1-hop) Total cost
--- 19:58:35.728300 ----- TOPOLOGY
Source IP addr  Dest IP addr      LQ      ETX
|
--- 19:58:38.730359 ----- LINKS
IP address      hyst      LQ      ETX
--- 19:58:38.730383 ----- TWO-HOP NEIGHBORS
IP addr (2-hop) IP addr (1-hop) Total cost
--- 19:58:38.730393 ----- TOPOLOGY
Source IP addr  Dest IP addr      LQ      ETX
/
--- 19:58:41.634276 ----- LINKS
IP address      hyst      LQ      ETX
--- 19:58:41.634300 ----- TWO-HOP NEIGHBORS
IP addr (2-hop) IP addr (1-hop) Total cost
--- 19:58:41.634309 ----- TOPOLOGY
Source IP addr  Dest IP addr      LQ      ETX
```

Figure 5.6 Configuration of OLSRD:

INSTALLATION OF EQUIPMENT

6.1 Installation of Equipment

The controller board chosen for testing is practical enough to be installed in a car's dashboard. The LCD Screen must be placed at a position whereby it is most easy for a driver to view his information and to send messages as quickly as possible, for example, in an emergency situation. For users, they may vary the processors for their own requirements and suitability. For example, someone who is savvier with mobile phones, may even install the application in a cell phone and use it as his OBU Device.

While testing the functionality of the system, OBU was installed in rear part of the CAR and attached the LCD on dashboard such that it is easy for the driver to view and use it's touch panel. Snapshots of the equipment and vehicle are shown below.

Figure 6.1 to 6.3 show how the equipment was installed.

In Figure 6.1, it is demonstrated that the GPS dongle was attached on the rooftop of the car. This was the most suitable position as best reception of signals from GPS satellites is achieved. Testing proved the theory.



Figure 6.1 Installation of GPS dongle

Figure 6.2 shows the installation of the OBU, the main unit, which basically contained the Intel atom board. This was placed in the rear part of the car as there was enough space there. One may even place it under the rear seat or install it in the bonnet, according to whatever suits him best and for his specific vehicle design. The testing was done in the position shown in figure 6.2 and good results were achieved, as will be explained in chapter 7.



Figure 6.2 Installation of OBU

Figure 6.3 then shows the installation of LCD. This, as can be seen, was attached to the dashboard of the car. This was a most suitable position as it is very easy for the driver to view updated maps and use the touch screen promptly, when ever required.



Figure 6.3 Installation of LCD

The Intel atom board uses 240 V power supply, so to, use it inside the car, a power inverter was used which converted 240 V to 12 V .The power was then driven by the car's engine itself.

Figure 6.4 is a screen shot of the 12V-220v Converter while figure 6.5 shows the lighter used to interface with this converter. This lighter can be installed the socket present in every car , whereby, people generally charge their mobile phones and laptops. Testing was done using this very equipment and everything was observed

to be found working fine and in an efficient manner.



Figure 6.4 220V-12V Power inverter

A three in one power socket, easily available in market, powered up the LCD, the controller board, and if required, a serial port GPS as well.



car cigar lighter

Figure 6.5 Car lighter

Figure 6.6 is just a close up of the installed OBU.



Figure 6.6 Close view of OBU

It is illustrated that the equipment is of appropriate size and suitable to be installed in any car.

CHAPTER 7

FUTURE ENHANCEMENTS AND CONCLUSION

7.1 Future Enhancements

The project is developed keeping an undergrad scope in mind. Programming for three nodes is involved. But for commercial purposes, it is very much scalable to be extended to a number of vehicles. Moreover, the WiFi Module must be replaced with a DSRC one, as it is most recommended for vehicular environments.

Two aspects of Fleet management are covered, these being, vehicle telematics (tracking and diagnostics) and safety management. The project can be extended to include the aspects of speed management and fuel management.

Speed Management may include sensors recording the speed of a vehicle at periodic intervals and then relaying the information to the application which routes it towards the management server. The programming for sending and relaying information is already embedded in the application. The speed measurement function only would need to be added.

Fuel Management shall include functions such as Tracking fuel consumption for every vehicle, Providing an ongoing analysis of vehicle miles per gallon of fuel ,an ongoing analysis of cost per mile or hour and an ongoing analysis of total operating and maintenance costs and sending this all information to the Management Servers. Sensors for measurement of fuel at periodic intervals and programming to alert warning at low fuel instances, calculations involving

analysis if costs etc would be needed. The system could further be made even more efficient by displaying graphs on the management server as well.

Other possible future enhancements include addition of cryptographic techniques to ensure safe data delivery and avoid any bad guy intrusions. Another future aspect is extending the fleet management system such that more than one communication zones are involved. Nodes in different cities' may be tested for communication.

Working under a typical VANET environment, the system was observed to be highly efficient for traffic load management and resource management.

Communication to the management server was reliable (i.e. no data loss was observed) and efficient enough. The three node scenario was developed such that, it is scalable enough to be extended to a number of nodes and additions and modifications in the functionality / application is very much implementable and generic.

A lot of future enhancements are possible. Data ranges may be extended by replacing the Wi-Fi module with DSRC. Noise observed during voice chat may be reduced by implementing any echo-reduction and noise reduction algorithms in asterisk software or, by simply using a better quality phone.

For constructing a better Fleet Management System, functionalities such as speed and fuel management may as well be implemented in future.

7.2 Conclusion

Figure 7.1 is the frontend graphical user interface (GUI) of the VANET application at the node. Right half of the GUI shows the LDM of the operator. In the GUI of the node, each node finds its own position in the centre of the map, whereas in the GUI of management server, deployment of all nodes can be seen in colored dots (red blue & green). Blue line in LDM of Figure 7.1 shows the direction to the nearest node or management server access point. The screen could be scrolled accordingly using touch feature.

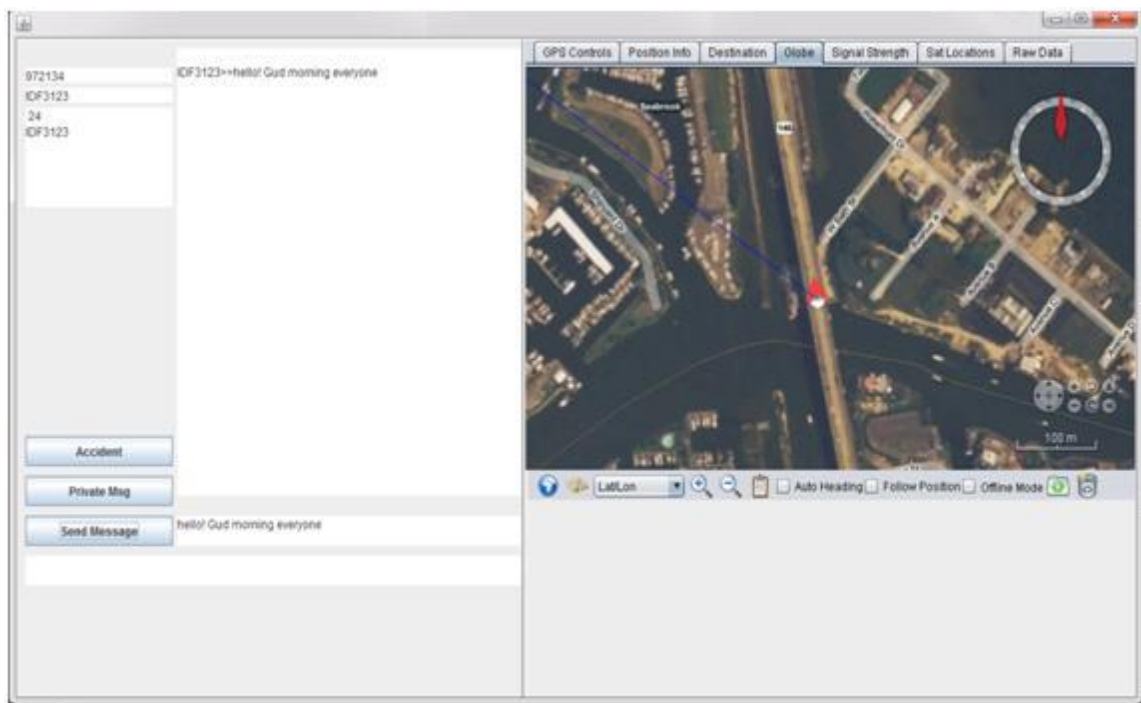


Figure 7.1 The front end of GUI at each node

Left half of the GUI shows the online client directory for voice and text chat as well as interface for emergency preconfigured alarms. The right half of the left portion shows the ongoing text chat. The black portion with green writing is used to display the emergency alarms received from management server or nearby vehicle nodes.

The hardware independent development on two different types of mother boards made the design highly generic for implementation on any network. However, it was observed that development on ARM-11 board requires more expertise due to its peculiar nature.

Due the unavailability of DSRC modules at reasonable budgets, Wi-Fi was used as the communication mode and its range was observed up to maximum of 150 meters. However for voice chat, quality of voice started degrading after around 70m.

All nodes registered successfully with Asterisk and successful calls were established among all nodes using Wi-Fi and management server using WiMAX. Initially location updates were shared with the periodic delay of 3 seconds which could be reduced to 1 second according to type of network. Considering the maximum speed of 120 kmph, a node actual position may vary to an acceptable limit of 30 meters only. The variation is not significant as the application on the nodes does not involve accident avoidance scenario. The provision of emergency response and other local updates may well be coordinated in advance according to 30 meters error limit.

The application demonstrates the concept of VANET involving OBU's , RSU's and Servers. It meets the objectives of providing fleet management functions using the concept of Adhoc Networking. A Local Dynamic Map displays the locations of vehicles using offline maps and updates itself periodically. A driver can view the prevailing road scenarios on a friendly GUI and may use text or voice chat features to communicate with any node that he wishes to. The routing

protocol performs the function of relaying multi hop messages in an efficient manner.

In a complicated and commercial environment several RSU's will be present, connected to each other forming an intranet. They may be connected to the servers' intranet through any mechanism and the need for GPRS and Public IP shall then be eliminated.

APPENDIX A

Code of the Application

Coding has been done in JAVA. There are two main classes which drive the application, one is of the application front end and the other is of the local dynamic map. Both classes are lengthy as they perform complex functions. The main methods and functions would be illustrated in this appendix; however detailed code can be made available on demand at any time.

APPLICATION

First part of the code sets up how the GUI looks like, after that functionality to the buttons on the front end is provided.

The front end looks like as shown below;

Screen shot

Accident alert button:

```
a.addActionListener(
```

```
newActionListener()
```

```
{
```

```
    @Override
```

```

public void actionPerformed( ActionEvent event )
    {
try // create and send packet
    {

        byte [] b = new byte[] {(byte)255,(byte)255,(byte)255,(byte)255};
System.out.println( "Sending Accident Alert" );
String cordinate = null;//event.getActionCommand();

cordinate = "Longitude " + cordLat.getText()+"::"+ " Latitude " +
cordLon.getText();//event.getActionCommand();

        String message = ("A:" + lineId + "::all: has met an accident at "
            + "::"+GpsInspector.tx_lat+"::"+GpsInspector.tx_lon);

byte data[] = message.getBytes(); // convert to bytes
        // create sendPacket

DatagramPacket sendPacket = new DatagramPacket( data,
data.length, InetAddress.getByAddress(b), 11345 );
socket.send(sendPacket ); // send packet

System.out.println( "Accident Alert sent" );
notes.append( "Accident Alert sent\n");

```

```
notes.setCaretPosition(notes.getDocument().getLength());
```

```
        } // end try  
catch ( IOExceptionioException )  
    {  
  
    System.out.println(ioException.toString() + "\n" );  
  
        } // end catch  
    } // end actionPerformed  
// end inner class  
}  
  
); // end call to addActionListener
```

Voice chat button:

This button opens up the soft phone “twinkle”

```
tw.addActionListener(  
  
newActionListener(  
  

```

```
newActionListener(  
  

```

```

    {
        @Override
    public void actionPerformed( ActionEvent event )
        {
    try {
            String cmd = "twinkle";

            Runtime run = Runtime.getRuntime() ;

            Process pr = run.exec(cmd);
    try {
    pr.waitFor();

            } catch (InterruptedException ex) {
    Logger.getLogger(FileInputGUI.class.getName()).log(Level.SEVERE, null, ex);
            }

            } // end actionPerformed

            // end inner class

    catch (IOException ex) {
    Logger.getLogger(FileInputGUI.class.getName()).log(Level.SEVERE, null, ex);
            }

            } // end actionPerformed

            // end inner class
        }

        ); // end call to addActionListener

```

Send message button:

This button sends the messages typed in the text box to all the nodes in the network.

```
b.addActionListener(
```

```
newActionListener()
```

```
{
```

```
    @Override
```

```
public void actionPerformed( ActionEvent event )
```

```
{
```

```
try // create and send packet
```

```
{
```

```
byte [] b = new byte[] {(byte)255,(byte)255,(byte)255,(byte)255};
```

```
System.out.println( "about to send my packet" );
```

```
notes.append("about to send my packet\n" );
```

```
notes.setCaretPosition(notes.getDocument().getLength());
```

```
String rawMessage = enter.getText();//event.getActionCommand();
```

```

        String message = ("M::" + lineId + "::all:" + rawMessage+":: :: ");
System.out.println( "sending this" + message );
    notes.append("sending this" + message+"\n" );
    notes.setCaretPosition(notes.getDocument().getLength());
    byte data[] = message.getBytes(); // convert to bytes
        // create sendPacket
    DatagramPacket sendPacket = new DatagramPacket( data,
    data.length, InetAddress.getByAddress(b), 11345 );
    socket.send(sendPacket ); // send packet
    System.out.println( "sent" );
        } // end try
    catch ( IOException ioException )
    {
        System.out.print("this is catch statement");

        System.out.print(ioException);

        System.out.println(ioException.toString() + "\n" );
        } // end catch
    } // end actionPerformed
// end inner class
}

```

```
); // end call to addActionListener
```

Private chat button

This button sends the private message to any desired node.

```
c.addActionListener(
newActionListener()
{
    @Override
    public void actionPerformed( ActionEvent event )
    {
        try // create and send packet
        {

            String ad = JOptionPane.showInputDialog("Private Chat Address","Enter
Vehicle Address here!!!");

            String msg = JOptionPane.showInputDialog("Input Message","Enter
Message here!!");

            byte [] b = new byte[] {(byte)255,(byte)255,(byte)255,(byte)255};

            System.out.println( "about to send my packet" );
```



```

    notes.append( "about to send my packet\n" );
    notes.setCaretPosition(notes.getDocument().getLength());

        String message = ("P::" + lineId + "::"+ ad + "::" + msg+":: :: ");
    System.out.println( "sending this" + message );
    notes.append("sending this" + message+"\n" );
    notes.setCaretPosition(notes.getDocument().getLength());
    byte data[] = message.getBytes(); // convert to bytes

        // create sendPacket
    DatagramPacket sendPacket = new DatagramPacket( data,
    data.length, InetAddress.getByAddress(b), 11345 );
    socket.send(sendPacket ); // send packet
    System.out.println( "sent" );
    notes.append( "Packet sent\n" );
    notes.setCaretPosition(notes.getDocument().getLength());

        } // end try
    catch ( IOException ioException )
        {

    System.out.println(ioException.toString() + "\n" );

        } // end catch
    } // end actionPerformed
// end inner class
}

```

```
); // end call to addActionListener
```

After this it was given reception functionality to the application

This “wait for packets” functions keeps listening to the predefined sockets and keeps updating the applications. The information this function looks for includes, public messages, private messages, accident alerts, coordinate information and vehicle id’s.

```
public void waitForPackets()
{
    new Thread(new Runnable() {
        public void run() {
            while ( true )
                {
                    try // receive packet, display contents, return copy to client
                    {
                        byte data[] = new byte[ 100 ]; // set up packet
                            // wait for public chat msgs

                        DatagramPacketreceivePacket =
                            newDatagramPacket( data, data.length );
                        if(socket == null)
```

```

        {
socket = new DatagramSocket(11345);
System.out.println("Variabe is null");
        }

socket.receive(receivePacket ); // wait to receive packet
System.out.println("wait for packets function");
notes.append("wait for packets function\n");
notes.setCaretPosition(notes.getDocument().getLength());
                String aLine = new String(receivePacket.getData());
                Scanner scanner = new Scanner(aLine);

scanner.useDelimiter("::");
if ( scanner.hasNext() ){
                String type = scanner.next();
                String sID = scanner.next();
                String rID = scanner.next();
                String msg = scanner.next();
                String lat = scanner.next();
                String lon = scanner.next();

if("M".equals(type))
        {
displayArea.append("\n" + sID + ">>" + msg);
displayArea.setCaretPosition(displayArea.getDocument().getLength());
        }
}

```

```

else if ("I".equals(type))
    {
if(sID.equals("CAR_A"))
    {
list.append(sID+"\n");

Lat_r=lat;

Lon_r=lon;

list.setCaretPosition(list.getDocument().getLength());
    }
else if(sID.equals("CAR_B"))
    {
list.append(sID+"\n");

list.setCaretPosition(list.getDocument().getLength());

        Lat_r1=lat;

        Lon_r1=lon;

other_cords.setText(sID + "-- "+lat+"."+lon+"\n");
    }
else if(sID.equals("CAR_C"))
    {

list.append(sID+"\n");

list.setCaretPosition(list.getDocument().getLength());

```

```

        Lat_r2=lat;
        Lon_r2=lon;
other_cords.setText(sID + "-- "+lat+": "+lon+"\n");

    }

test = sID;

        //list.append(sID + "\n");
other_cords.setText(sID + "-- "+lat+": "+lon+"\n");
    }

else if ("A".equals(type))
    {

alert.setText("\n"+ sID + msg+"Longitude: "+lat+", Latitude: "+lon);

    }

else if ("P".equals(type))
    {

        String lineID = id.getText();

if(rID.equals(lineID))
    {

Toolkit.getDefaultToolkit().beep();

JOptionPane.showMessageDialog(count, msg, sID + "to you", WIDTH, null);

    }

    }

else if("G".equals(type)){

```

```

if("own".equals(sID))
    {
        //cord.setText(rID);
    }
    //put else to treat other vehicle coordinates
    }
}
} // end try
catch ( IOExceptionioException )
    {

System.out.println("Exception:" + ioException.toString());
    } // end catch
    // end while

try {
Thread.sleep(3000);

    } catch (InterruptedException ex) {

Logger.getLogger(FileInputGUI.class.getName()).log(Level.SEVERE, null, ex);
    }

}

```

```

        }
    }).start();
}

```

Now ,the on board unit was enabled to transmit its own coordinates and id to the neighboring vehicles once it joins their network.

```

public void txID(){
    new Thread(new Runnable() {
        public void run() {
            inti = 1;
            int j = 0;
            while (i<11)
                {
                    i++;
                    System.out.print(i);
                    try {
                        cordLat.setText("Long:"+GpsInspector.tx_lat);
                        cordLon.setText("Lat :"+GpsInspector.tx_lon);
                        nots.append("....Reading GPS");
                        nots.setCaretPosition(nots.getDocument().getLength());
                        Thread.sleep(3000);
                    }
                }
        }
    }).start();
}

```

```

        } catch (InterruptedException ex) {
Logger.getLogger(Input.class.getName()).log(Level.SEVERE, null, ex);
        }

j++;

try // create and send packet
    {
        // byte [] b = new byte[] {(byte)10,(byte)42,(byte)43,(byte)255};
byte [] b = new byte[] {(byte)255,(byte)255,(byte)255,(byte)255};

        String rawMessage = lineId;//event.getActionCommand();

        String message,message1,message2,message3;

        message = ("l::" + lineId + "::all::" + "id"+"::"+
GpsInspector.tx_lat+"::"+GpsInspector.tx_lon);
byte data[] = message.getBytes(); // convert to bytes

        // create sendPacket

DatagramPacket sendPacket = new DatagramPacket( data,
data.length, InetAddress.getByAddress(b), 11345 );

if(socket == null){

socket = new DatagramSocket(11345);

System.out.println("Variabe is null2");

        }

```



```

socket.send(sendPacket ); // send packet

Thread.sleep(5000);

        } // end try

catch (InterruptedException ex) {

Logger.getLogger(FileInputGUI.class.getName()).log(Level.SEVERE, null, ex);

        }          catch ( IOExceptionioException )

        {

System.out.println(ioException.toString() + "\n" );

        } // end catch

                //end tx id op

if(i==10)

        {

//to avoid null pointer exception

i=1;

tlat = 33.5766;

tlon = 73.0619;

        }

        }

        }

}).start(); //end of new ruunable thread

```

```
} //end txID()
```

After this, the distance covered by the vehicle is calculated and the time from the system clock is read and appended into the relevant text box. Finally time wise mileage of the vehicle is recorded into a text file.

BIBLIOGRAPHY

- [1] 'Wireless Networks for Vehicular Support', "In-depth tutorial and information," <http://what-when-how.com/information-science-and-technology/wireless-networks-for-vehicular-support-information-science/>
- [2] Lee Armstrong, "DSRC Standard programs worldwide," <http://www.leearmstrong.com/Dsrc/DSRCHomeset.htm>, ArmstrongInc
- [3] JuergenHaas, "Linux Guide," http://linux.about.com/cs/linux101/a/linux_2.htm
- [4] J.Haerri, Filali, F., and Bonnet, C, "Performance Comparison of AODV and OLSR in VANETs Urban Environments Under Realistic Mobility Patterns," in Proc. 5th IFIP Mediterranean Ad-Hoc Networking Workshop, June 14-17, Lipari, Italy, 2006
- [5] C. D.Wang, and J. P. Thompson, "Apparatus and Method for Motion Detection and Tracking of Objects in a Region for Collision Avoidance Utilizing a Real-Time Adaptive Probabilistic Neural Network", U.S. patent no. 5,613,039, 1997
- [6] Reducing Highway Fatalities, Federal Highway Administration, U.S. Department of Transportation, <http://safety.fhwa.dot.gov/facts/road factsheet.htm>
- [7] K. Bilstrup, E. Uhlemann and E. G. Ström, "Medium Access Control in Vehicular Networks Based on the Upcoming IEEE802.11p Standard," in Proceedings of. World Congress on ITS, New York City, NY, Nov. 2008, 12 pages," 2008.