Department of Electronics and
Electrical Communications Engineering
Faculty of Engineering - Cairo University

ADVANCED DRIVING ASSISTANT SYSTEM BASED ON V2X COMMUNICATION TECHNOLOGY

A Graduation Project Report Submitted to
the Faculty of Engineering at Cairo University
in Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science
in Electronics and Communications Engineering

By

Abdelrahman Ahmed Ali
Ahmed Essam Abdullah
Moataz El-Sherbiny Abdullah
Mohamed Ali El Sayed
Mohamed Raafat El Sayed
Mostafa Farag Abd El-Bary

Under supervision of

Associate Prof. Hassan Mostafa

Dr.Amin Nassar

Faculty of Engineering, Cairo University

Giza, Egypt

August 2020

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

V2X	Vehicle to Everything
V2V	Vehicle-To-Vehicle Communication
V2I	Vehicle-To-Infrastructure Communication
V2P	Vehicle-To-Pedestrian Communication
V2N	Vehicle-To-Network Communication
NIST	National Institute of Standards and Technology
SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a Service
HTTP	Hyper Text Transfer Protocol
REST	Representational State Transfer
GPL	GNU General Public License
SDK	Software development kit
RFS	Root FILE System
LTS	Long Term Support
DQL	Data query language
DML	Data manipulating language
DBMS	Database management system
UML	Unified Modeling Language
GUI	Graphical user interface
SQL	Structure Query Language
ERD	Entity–relationship diagram
RSU	Road Side Units
OBU	On Board Unit
C-V2X	Cellular Vehicle –To- Everything

CM	Car Maintenance
WAVE	Wireless Access in Vehicular Environments
SDR	Software defined radio
DSRC	Dedicated Short Range Communication
IVC	Inter-Vehicle Communication
VANETs	Vehicular Ad Hoc Networks
DCC	Decentralized Congestion Control
QoS	Quality of Service
DCF	Distributed Coordination Function
FPGA	Field-Programmable Gate Array
GPPs	General Purpose Processors.
USRP	Universal Software Radio Peripheral
ITS	The Intelligent Transportation Systems
SIMD	Single Instruction Multiple Data
AWGN	Additive White Gaussian Noise
SNR	Signal to noise ratio
FFT	Fast Fourier transform
IFFT	inverse Fast Fourier transform
PHY	Physical
MAC	Medium Access Control
IoT	Internet of Things
MQTT	Message Queueing Telemetry Transport
TLS	Transport Layer Security
SSL	Secure Sockets Layer
SCADA	Supervisory Control and Data Acquisition
PSID	Provider Service Identifier
GPS	Global Positioning System

WSMP	WAVE Short Message Protocol
SCH	Service Channel
CCH	Control Channel
BSSID	Basic Service Set Identification
RTS	Request To Send
CTS	Clear To Send
CSMA/CA	Carrier Sense Multiple Access/Collision Avoidance
EDCA	Enhanced Distributed Channel Access
DIFS	Distributed coordination Function Interframe Space
AIFS	Adoption of arbitration Interframe Spacing
CRC	Cyclic Redundancy Check
PLCP	Physical Layer Convergence Procedure
OFDM	Orthogonal Frequency Division Multiplexing
API	Application Programming Interface
IBM	International Business Machines
IT	Information Technology
OT	Operational Technology
GNSS	Global Navigation Satellite System
NMEA	National Marine Electronics Association
NTRIP	Networked Transport of RTCM via Internet Protocol
EEBL	Electronics Emergency Braking Light
RWW	Road Work Warning

ACKNOWLEDGEMENT

We have to thank everyone who contributed in this project to reach these great results. The whole V2X team members thank the supervisors, engineers and our colleagues who helped us to achieve this great achievement and we specially thank

Dr Hassan Mosrafa

Dr Amin Nassar

Eng. Bishoy Wassfey

Eng. Abdelrahman Sobeih

for their great effort in this project.

ABSTRACT

Most accidents occur because the driver can only see the Sudden changes on the Road when reach it, there is a lot of changes on the road, The driver must follow every dangerous movement on the road, and these dangers hide among many of the signals that the driver watches, These signals are made by hundreds of cars around the driver, causing distraction and discomfort while driving, and this may lead to not noticing some very dangerous signals on the road due to the stress and distraction the driver was exposed to. As a result, there has to be a solution, and that solution makes care able to communicate with everything on the road another car, infrastructure, pedestrians and buildings, and alert the driver of the changes on the road and alert the driver also of the sudden changes before reaching it.

One of the technological advances that could solve this problem is vehicle to everything communication. This report will include more information about V2X communication, its benefits and its market nowadays. Next, a problem statement and techniques that have been used in the project to solve the problem will be explained. After that, the project design is discussed along with the tools used as well as the importance of each tool in our project. Then, the actual implementation of our project along with the testing methods and results are furtherly explained. Finally, the next phases of the project and relation between the project and new technologies are discussed.

CHAPTER 1: INTRODUCTION

1.1. Road accidents

Road safety becomes a major public health concern when the statistics show that more than 3,000 people around the world succumb to death daily due to road traffic injury, Also, road crashes lead to the global economic losses as estimated in road traffic injury costs of US$518 billion per year The huge economic losses are an economic burden for developing countries. It is reflected that the road crash costs are estimated to be US$ 100 billion in developing countries which is twice the annual amount of development aid to such countries Considering within southeast Asian countries, the economic growth rate of Thailand continues to move upward with an aggravating road traffic situation due to the heavy negative impact of a higher level of motorization. Over 130,000 fatalities and nearly 500,000 people were permanently disabled due to road crashes over past decades The economic losses due to the road crashes are; therefore, considerably high, costing approximately US$2,500 million per year (about US$0.3 million per hour), or 3.4 percent of the Gross National Product (GNP).[1]

The collection and use of accurate and comprehensive data related to road accidents is very important to road safety management The road accident data are necessary not only for statistical analysis in setting priority targets but also for in-depth study in identifying the contributory factors to have a better understanding of the chain-of-events. Having the inconsistencies in the aims of the police and the road safety engineers, the data analysis and its interpretation usually does not result in proper countermeasures. Sometimes a lack of proper knowledge of crash and proper training of the police officers in charge of systematic data collection procedures from a crash scene adds to the diverging nature of the role of the police and the road safety professionals. These problems have become a burning issue for developing countries addressing road safety without completed crash data due to the negligence of the concerned authorities.

Road accidents are the most unwanted thing to happen to a road user, though they happen quite often. The most unfortunate thing is that we don't learn from our mistakes on the road. Most of the road users are quite well aware of the general rules and safety measures while using roads but it is only the laxity on part of road users, which cause accidents and crashes. Main cause of accidents and crashes are due to human errors. We are elaborating some of the common behavior of humans which results in accidents. [1]

Over Speeding

Most of the fatal accidents occur due to over speeding. It is a natural psyche of humans to excel. If given a chance man is sure to achieve infinity in speed. But when we are sharing the road with other users we will always remain behind some or other vehicle. Increase in speed multiplies the risk of accident and severity of injury during accidents. Faster vehicles are more prone to accident than the slower one and the severity of accident will also be more in case of faster severity of accident will also be more in case of faster vehicles. Higher the speed, greater the risk. At high speed the vehicle needs greater distance to stop i.e. braking distance. A slower vehicle comes to halt immediately while a faster one takes a long way to stop and also skids a long distance due to the law of notion. A vehicle moving on high speed will have greater impact during the crash and hence will cause more injuries. The ability to judge the forthcoming events also gets reduced while driving at faster speed which causes error in judgment and finally a crash. [2]

Drunken Driving

Consumption of alcohol to celebrate any occasion is common. But when mixed with driving it turns celebration into a misfortune. Alcohol reduces concentration. It decreases reaction time of a human body. Limbs take more to react to the instructions of the brain. It hampers vision due to dizziness. Alcohol dampens fear and encourages humans to take risks. All these factors while driving cause accidents and many times it proves fatal. For every increase of 0.05 blood alcohol concentration, the risk of accident doubles. Apart from alcohol many drugs, medicines also affect the skills and concentration necessary for driving. First of all, we recommend not to consume alcohol. But if you feel your merrymaking is not complete without booze, do not drive under the influence of alcohol. Ask a teetotaler friend to drop you home. [2]

Distraction to Driver

Though distraction while driving could be minor but it can cause major accidents.

Distractions could be outside or inside the vehicle. The major distraction nowadays is talking on a mobile phone while driving. Act of talking on the phone occupies a major portion of the brain and the smaller part handles the driving skills. This division of the brain hampers reaction time and ability of judgement. This is one of the reasons for crashes. One should not attend to telephone calls while driving. If the call is urgent one should pull out beside the road and attend the call. Some of the distractions on road are:

1. Adjusting mirrors while driving
2. Stereo/Radio in vehicle
3. Animals on the road

4. Banners and billboards.

The driver should not be distracted due to these things and reduce speed to remain safe during diversions and other kinds of outside distractions. [2]

Red Light jumping

It is a common sight at road intersections that vehicles cross without caring for the light. The main motive behind Red light jumping is saving time. The common conception is that stopping at a red signal is wastage of time and fuel. Studies have shown that traffic signals followed properly by all drivers saves time and commuters reach their destination safely and timely. A red light jumper not only jeopardizes his life but also the safety of other road users. This act by one driver incites another driver to attempt it and finally causes chaos at crossing. This chaos at intersections is the main cause of traffic jams. Eventually everybody gets late to their destinations. It has also been seen that the red light jumper crosses the intersection with greater speed to avoid crash and challan but it hampers his ability to judge the ongoing traffic and quite often crashes. [2]

Avoiding Safety Gears like seat belts and helmets

Use of a seat belt in a four-wheeler is now mandatory and not wearing a seat belt invites penalty, same in the case of helmets for two wheeler drivers. Wearing seat belts and helmets has been brought under law after proven studies that these two things reduce the severity of injury during accidents. Wearing seat belts and helmets doubles the chances of survival in a serious accident. Safety Gears keep you intact and safe in case of accidents. Two wheeler deaths have been drastically reduced after use of helmets has been made mandatory. One should use safety gears of prescribed standard and tie them properly for optimum safety. [2]

1.2. V2X Definition

Vehicle-to-everything (V2X) is a technology that allows vehicles to communicate with moving parts of the traffic system around them. Also known as connected-vehicle-to-everything communication, it has several components.

One component of this technology is called vehicle-to-vehicle (V2V) which allows vehicles to communicate with one another. Another component is vehicle to infrastructure (V2I) which allows vehicles to communicate with external systems such as street lights, buildings, and even cyclists or pedestrians. Vehicle-to-pedestrian (V2P), and vehicle-to-network (V2N) communications.

IEEE 802.11p: The original V2X standard is based on a Wi-Fi offshoot, IEEE 802.11p (part of the IEEE's WAVE, or Wireless Access for Vehicular Environments program), running in the unlicensed 5.9GHz frequency band. IEEE 802.11p, which was finalized in 2012, underpins Dedicated Short-Range Communications (DSRC) in the US, and ITS-G5 in the European Cooperative Intelligent Transport Systems (C-ITS) initiative.

V2X communication via 802.11p goes beyond line-of-sight-limited sensors such as cameras, radar and LIDAR, and covers V2V and V2I use cases such as collision warnings, speed limit alerts, and electronic parking and toll payments.

Functional characteristics of 802.11p include short range (under 1km), low latency (~2ms) and high reliability -- according to the US Department of Transportation, it "works in high vehicle speed mobility conditions and delivers performance immune to extreme weather conditions (e.g. rain, fog, snow etc.)". Essentially, 802.11p extends a vehicle's ability to 'see' the environment around it, even in adverse weather.

1.2.1. History

In 1999 the US Federal Communications Commission (FCC) allocated 75 MHz in the spectrum of 5.850-5.925 GHz for intelligent transport systems. Since then the US Department of Transportation (USDOT) has been working with a range of stakeholders on V2X. In 2012 a pre-deployment project was implemented in Ann Arbor, Michigan. 2800 vehicles covering cars, motorcycles, buses and HGV of different brands took part using equipment by different manufacturers. The US National Highway Traffic Safety Administration (NHTSA) saw this model deployment as proof that road safety could be improved and that WAVE standard technology was interoperable. In August 2014 NHTSA published a report arguing vehicle-to-vehicle technology was technically proven as ready for deployment. On 20 August 2014 the NHTSA published an Advance Notice of Proposed Rulemaking (ANPRM) in the Federal Register, arguing that the safety benefits of V2X communication could only be achieved if a significant part of the vehicles fleet was equipped. Because of the lack of an immediate benefit

for early adopters, the NHTSA proposed a mandatory introduction. On 25 June 2015 the US House of Representatives held a hearing on the matter, where again the NHTSA, as well as other stakeholders argued the case for V2X.

To acquire EU-wide spectrum, radio applications require a harmonized standard, in case of ITS-G5 ETSI EN 302 571, first published in 2008. A harmonized standard in turn requires an ETSI System Reference Document, here ETSI TR 101 788. Commission Decision 2008/671/EC harmonizes the use of the 5 875-5 905 MHz frequency band for transport safety ITS applications. In 2010 the ITS Directive 2010/40/EU was adopted. It aims to assure that ITS applications are interoperable and can operate across national borders, it defines priority areas for secondary legislation, which cover V2X and requires technologies to be mature. In 2014 the European Commission's industry stakeholder "C-ITS Deployment Platform" started working on a regulatory framework for V2X in the EU. It identified key approaches to an EU-wide V2X security Public Key infrastructure (PKI) and data protection, as well as facilitating a mitigation standard to prevent radio interference between ITS-G5 based V2X and road charging systems. The European Commission recognized ITS-G5 as the initial communication technology in its 5G Action Plan and the accompanying explanatory document, to form a communication environment consisting of ITS-G5 and cellular communication as envisioned by EU Member States. Various pre-deployment projects exist at EU or EU Member State level, such as SCOOP@F, the Testfeld Telematics, the digital testbed Autobahn, the Rotterdam-Vienna ITS Corridor, Nordic Way, COMPASS4D or C-ROADS. There exist real scenarios of implementation V2X standard as well. The first commercial project where V2X standard is used for Intersection movement assist use-case. It has been realized in Brno City / Czech Republic where 80 pcs of cross intersections are controlled by V2X communication standard from public transport vehicles of the municipality Brno.

1.2.2. Understanding Vehicle-to-Everything (V2X)

V2X communications systems are mainly used for the purpose of increasing safety and preventing collisions. In a traditional vehicle, V2X systems can convey important information to the driver regarding inclement weather, nearby accidents, road conditions, and the dangerous activities of nearby vehicles. In autonomous vehicles, V2X provides extra information to a vehicle's existing navigation system.V2X uses a short-range wireless signal to communicate with compatible systems, and this signal is resistant to interference and inclement weather. In addition to safety benefits, V2X technology serves other purposes like integrating automatic payments for tolls, parking, and similar fees. From an economic standpoint, the market potential for V2X technology is huge. Since 2016, production has risen exponentially.

1.2.3. V2X Market

The V2X market is relatively nascent, and many of the system's benefits will not be fully realized until the market expands. In order for a V2X vehicle to communicate with another vehicle or roadside object that object must also use V2X technology. Many traffic systems don't have V2X systems in place, which means that vehicles that do have the technology cannot communicate with them. However, as V2X systems become more common, vehicles are expected to be able to communicate not only with traffic systems but also with cyclists and even pedestrians who carry V2X devices.

Vehicle-to-everything technology is expected to grow significantly over the next 20 years, as many top V2X suppliers, such as Delphi, Denso, Qualcomm, and Continental, have large-scale integration plans. Many new car models use some form of V2X technology, especially luxury brands. Vehicle-to-everything systems are expected to eventually be added to lower-priced vehicles as well.

Vehicle-to-everything technology will be particularly prominent in autonomous vehicles, which scan the immediate environment for risks and make decisions based on this data. Vehicle-to-everything systems also transcend harsh weather conditions that impact autonomous vehicles.

1.2.4. Real World Example of Vehicle-to-Everything (V2X)

The 5G Automotive Association (5GAA) is an international organization of companies from the automotive, technology, and telecommunications industries devoted to developing V2X technology. It was founded in 2016 by eight corporations: AUDI AG, BMW Group, Daimler AG, Ericsson, Huawei, Intel, Nokia, and Qualcomm Incorporated. It now has over 100 member companies around the world.

1.3. Objective of the study

The objective of this study is to design a system based on V2X communication technology , and also implement different use cases using this system to solve the problem of road injuries , road traffic and make the driving journey easily and more comfortable, the objective also to design dashboard and implement on board unit to get data from real car and transfer this data between real cars and test use cases, and design Admin panel to control the system and test the use cases and all the system and make sure that the system working efficiency and achieve the system goals.

1.4. Limitations of the study

While working on the project, coved virus appeared and spread, which put us in quarantine, and it became difficult for members of the team to meet, and it became difficult to reach some components, and this created a set of sudden restrictions while working on the project.

There are other restrictions from the beginning of the project, for example the time because it was necessary to fully work the system and test it, within a period of six months, so it was necessary to choose some of the most important things to accomplish first and design the backbone of the system, which can be modified by anyone easily and without any complication.

There were other restrictions on the cost of the project as we want to design a usable system in the market. Therefore, all available alternatives were searched that perform the same task but at a lower cost.

1.5. Thesis organization

This chapter is an overview about road accidents and the reasons that affect it with a V2X definition and importance. The objective of this study and its limitation is clearly discussed in the chapter.

The following chapters discusses the V2X proposed in this work and its implementation in software and hardware and the results. This Thesis is organized as follows:

Chapter 2 provides information about road accidents statistics and the major importance of V2X to the world with the results and benefits of implementing V2X ADAS. It also provides a background of the WAVE protocol stack and the cellular V2X technology based on 5G. Then it discusses an overview of the tools used in the proposed work such as MQTT protocols, elements of the cloud, web services and embedded Linux image. The chapter ends with explaining the chosen use cases, their implementation, why they were chosen and their results and benefits.

Chapter 3 provides a system overview with a detailed discussion on the used hardware and software tools, where it also discusses the V2V communication based on GNU radio-MQTT and V2N communication represented on cloud. It illustrates the importance of the Admin Panel in the proposed work, where the final part of the system architecture is the Admin Panel which has a significant importance in the chosen use cases.

Chapter 4 provides the results of the use cases discussed in this study.

Chapter 5 provides a brief conclusion of the system and suggestions for the future work.

CHAPTER 2: LITERATURE REVIEW

2.1. Introduction

V2X technology enables cars to communicate with their surroundings and makes driving safer and more efficient for everyone. By making the invisible visible, it warns the driver of road hazards, helping reduce traffic injuries and fatalities. In addition to that, it helps to optimize traffic flow. In order to have full knowledge of V2X, its implementation, how to improve its performance and provide sufficient solutions, some elements and tools must be covered. This chapter is an overview about the needed tools, the solutions proposed in this study and an introduction to their implementation.

2.2. Road accidents statistics

Age-standardized death rates from road accidents, measured as the annual number of deaths per 100,000 people. Deaths include those from drivers and passengers, motorcyclists, cyclists and pedestrians. Age-standardized assumes a constant population age and structure to allow for comparisons between countries and with time. This is shown in Figure 1.

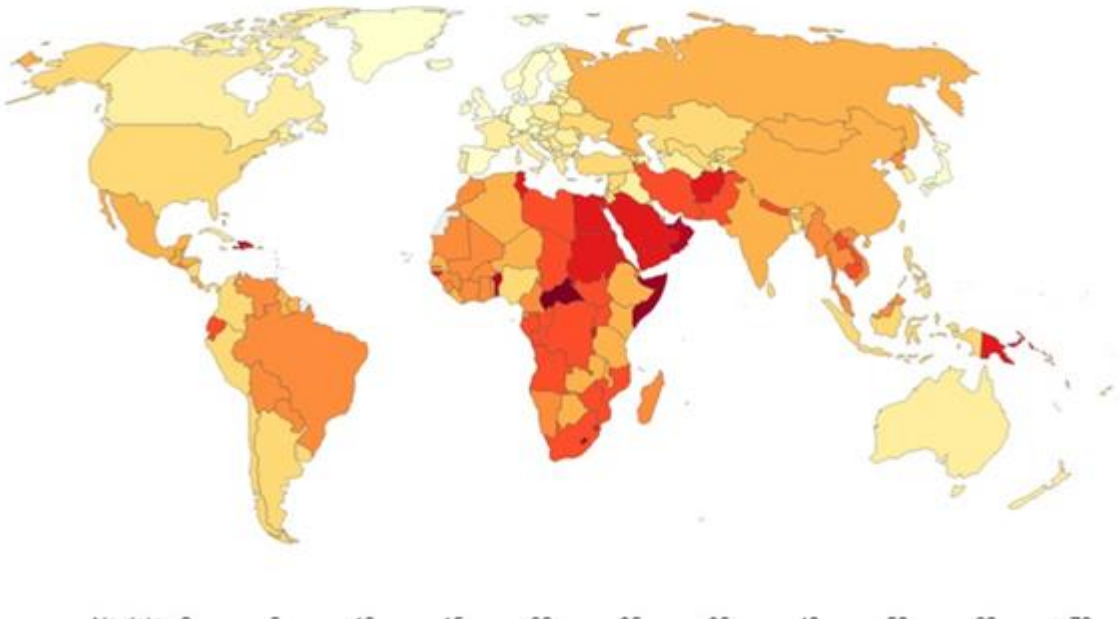

Figure 1: Road incident death rates 2017

More than 90% of all fatal road crashes occur in low and middle-income countries. These countries have less than half of the world's vehicles running on their roads, yet due to multiple factors (lack of awareness, careless driving, and so on) provoke more car accidents than all other countries combined. [1]

Car crash statistics show that about 1.35 million people die on the roads per annum. According to statistical data, nearly a million and a half people are killed in road crashes each year. On average, this means that approximately 3,700 people die each day in road crashes around the world. This data is drawn up for crashes involving cars for the most part, as well as busses and trucks and other participants in global traffic like: pedestrians, cyclists, and motorcyclists.

Car accident deaths are the 8th leading cause of death globally. Traffic fatalities stats show this to be the first cause of death among people ages 5–29.Approximately half a million people under 25 dying on the roads each year, accidents are turning into a modern global menace. Fatal car wrecks and road accidents cost countries worldwide about 3% of their GDP. Estimates vary, as some countries have costs estimated at about 1–2% of their Gross Domestic Product, while others experience up to 3% expenses. In cash terms, the total amount of incurred costs from such accidents is estimated at $518 billion. [2]

Data refers to the specific cause of death, which is distinguished from risk factors for death, such as air pollution, diet and other lifestyle factors. This is shown in Figure 2 by cause of death as the percentage of total deaths.

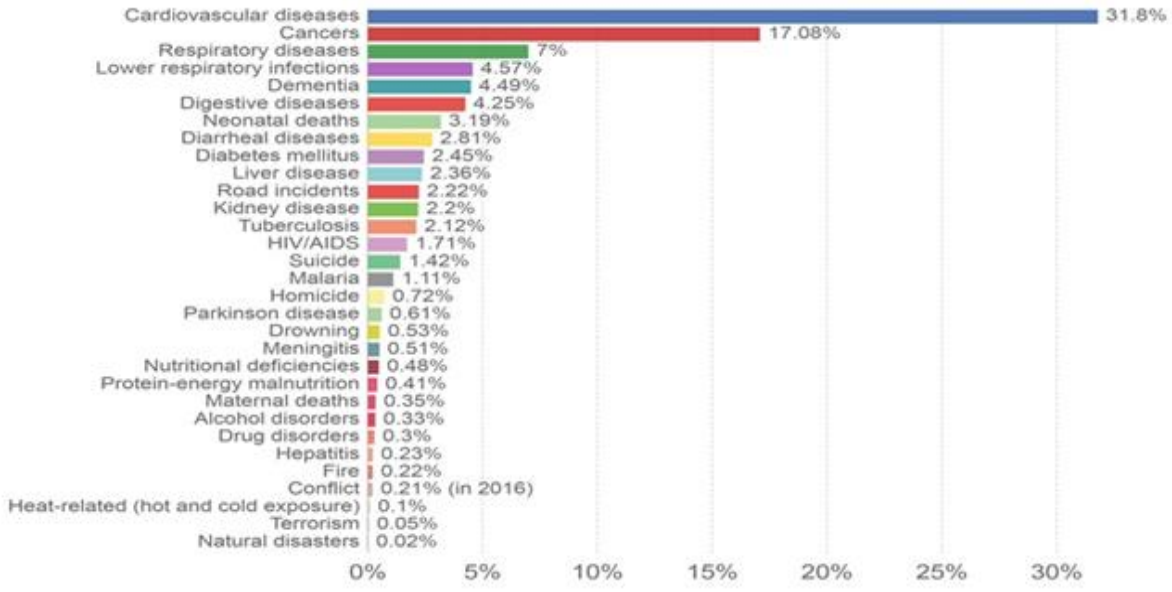

Figure 2: Share of deaths by cause 2017

2.3. Significance of V2X ADAS

Advanced Driver Assistance Systems (ADAS) are clearly making a difference in vehicle safety as more cars include them and as drivers learn to use them. However, further gains in safety are possible by adding a communications capability to every vehicle and the surrounding infrastructure.

With V2V technology, cars talk with each other by exchanging data wirelessly over an unlicensed spectrum called the Dedicated Short Range Communications (DSRC) band, which is similar to Wi-Fi technology, or by using 5G technology. The V2V transportation safety and efficiency applications run the same on either network. V2V technology enables cars to relay real-time information about their location and heading to prevent traffic collisions. It also reduces traffic congestion, thereby shortening drive times and improving the overall transportation environment.

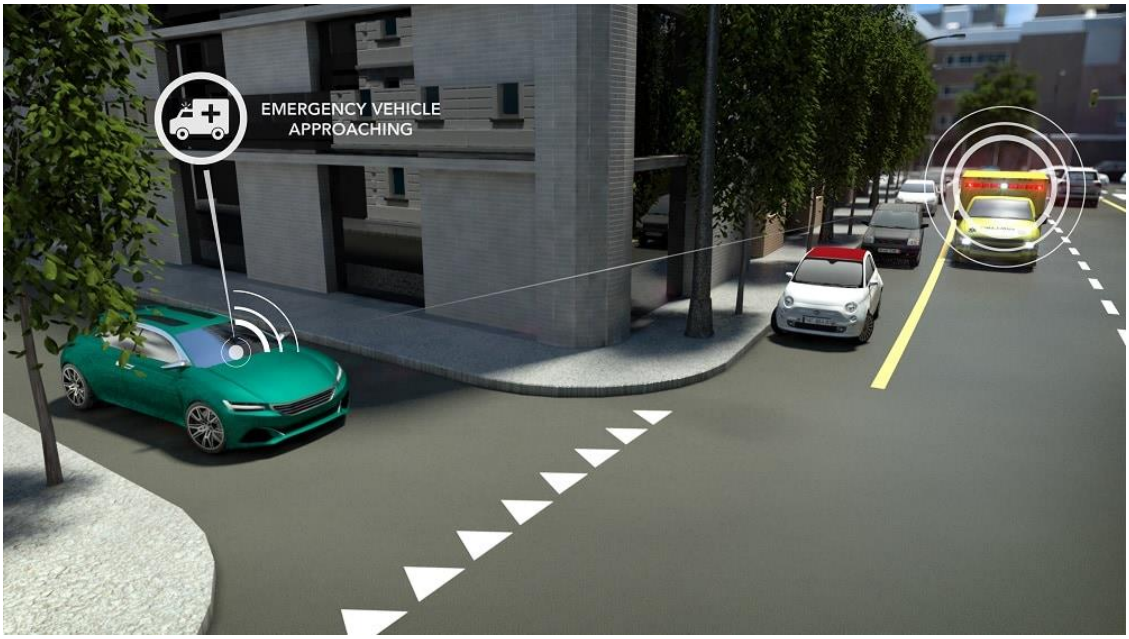

Figure 3: V2V technology example

With V2I technology enables the wireless exchange of data between vehicles and roadway infrastructure, with such key applications as intersection safety, speed management, e-tolling, transit safety and operations, and work-zone or low-bridge warnings. This technology has the potential to transform infrastructure equipment — including stoplights and weather information systems — into smart infrastructure by incorporating algorithms that recognize high-risk situations while they are developing and respond with driver warnings and countermeasures.

Figure 4: V2I technology example

With V2P technology enables pedestrians and bicyclists to be active participants in a V2X environment by using their smartphone to send and receive alerts, communicating with transit infrastructure like crosswalk signals if individuals need more time or alerting cars that a pedestrian is in the upcoming crosswalk or that a bicyclist is in the adjacent bike lane.

Figure 5: V2P technology example

Autonomous cars are, as the name suggests, designed to be independent. They're loaded with advanced sensors that constantly tell them about their immediate driving environment. V2X is one of the many sensors being installed in new cars, trucks, and streetcars.

However, V2X sensors in vehicles and infrastructure offer considerably more information about the driving environment than other sensors, and even provide predictive information. This is possible because V2X-enabled cars and trucks receive more precise road conditions, traffic information, and so forth before the car or truck even reaches a specific area, allowing the driver or auto-pilot to make the right decision every time. V2X not only provides insights for real-time driving, it also tells you about the entire journey from Point A to Point B.

Unlike other sensors, V2X sensors can be easily added into existing cars and trucks, similar to the way consumers first adopted GPS portable navigation devices — aftermarket devices that sit on the vehicle's dashboard. This is important because not every car is going to turn into an autonomous vehicle overnight. As more V2X-enabled cars hit the streets and the V2X network becomes stronger, more information can be shared between autonomous and legacy cars. V2X users benefit from the network effect principle: V2X is designed to be a collaborative technology that gets stronger with each node that's added to the network, like the internet. As a result, V2X lays an important foundation for autonomous cars, along with benefiting the 253 million mostly non-autonomous cars and trucks that travel on U.S. roads today.

As the government and industry continue to move forward, implementing new technologies in the transportation system and moving toward fully autonomous cars, it must be done in a safe manner, one that protects the public and the integrity of our transportation system while advancing innovation to meet the future needs of industry and society. Everyone working in this exciting field has a responsibility to ensure that automated and non-automated vehicles can safely co-exist on the same roadways.

While V2X technology is proven to enhance vehicle and pedestrian safety today, it also lays a foundation of safeguards that will help make Level 3 autonomous cars a reality. Cars talking to other cars, to roadside infrastructure, and to bicyclists and pedestrians creates a collaborative transportation environment in which the collective data benefits all. Today, we see V2X technology being deployed in Smart Cities like Tampa, Florida, and being planned for automotive manufacturers' 2018 and 2019 model year cars and trucks. With the technology advancing across the board, it's only a matter of time before it reaches mainstream adoption.

After discussing why the world needs the V2X ADAS, the results and the benefits of implementing V2X ADAS will be discussed. Here is how it makes driving better for all of us.

1- Increased traffic safety

Each year more than 1.2 million people die in traffic accidents worldwide. And most of the accidents are caused by human error. Vehicle-to-X technology (V2X) helps to reduce the number

of deaths by making the invisible, visible. An electronic emergency brake light, for example, warns the driver that a vehicle that is nearby but not visible is beginning to brake. V2X can also help to detect road hazards or vulnerable pedestrians and cyclists. Smart cars can press brake before human drivers react, and alert the vehicles behind them to reduce their speed. The US DOT estimates that V2X would save >1,000 lives / year and reduce 2.3 million non-fatal injuries.

2- Time savings

V2X technology helps optimize traffic flow and reduce traffic congestion. This saves time for everyone. When cars communicate with road signs, traffic lights, other vehicles and traffic management centers, they automatically know the best route from A to B. For example, traffic lights can tell cars at which speed they have to travel in order to catch the green light. Connected cars can also provide transport authorities with real-time traffic data for better road management and improved infrastructure planning.

3- Money savings

A more efficient, safer transportation system can help to save money: According to US DOT, V2X would save society $871 billion annually in the U.S. Blocked roads lower productivity, cause delays in supply chains and increase the cost of doing business. In Europe, congested roads cost nearly 100 billion euros annually. With V2X technology, vehicles can detect congestion-causing factors in advance, and react accordingly. Real-time data can be combined with simulation models to optimize routes and make journeys faster and more efficient.

4- Environmental factors

V2X helps lessen the environmental impact of transportation. Platooning, for example, allows self-driving cars or trucks to follow each other with very little distance between the vehicles. This technique can lead to less fuel consumption and CO_2 emissions. Cars traveling in free-flow conditions also use less fuel, so they create fewer emissions.

5- Convenience

With V2X comes a new level of comfort as it allows drivers to see well ahead and sense hazards, traffic jams, or road blocks far beyond the driver's line of sight. This will help to make car travel more fun and convenient than ever before.

2.4. WAVE – Traditional implementation of V2X

2.4.1. Definition

A wireless access in vehicular environments (WAVE) system provides interoperable, efficient, and reliable radio communications in support of applications offering safety and convenience in an intelligent transportation system (ITS). In many of the ITS application scenarios, the WAVE system is designed to provide vehicles with the direct connectivity to other vehicles (V2V), to roadside (V2R), or to infrastructures (V2I) through dedicated short-range communications (DSRC). In the USA, DSRC is used to refer to radio spectrum in the 5.9 GHz band or technologies associated with WAVE. While outside the USA, DSRC may refer to a distinct radio technology operating at 5.8 GHz. [4]

The architecture and operations of a WAVE system are based on the IEEE 1609 series of standards (IEEE, 2014a) and the IEEE 802.11p standard, which is now incorporated in IEEE Std 802.11-2012. IEEE Std 1609.0 is a comprehensive guide describing the WAVE architecture and services necessary for a WAVE system. This guide is meant to be used in conjunction with IEEE Std 1609.4 (multichannel operations), IEEE Std 1609.3 (networking services), IEEE Std 1609.2 (Security Services for Applications and Management Messages), IEEE Std 1609.11 (Over-the-Air Electronic Payment Data Exchange Protocol for ITS), IEEE Std 1609.12 (Identifier Allocations), IEEE Std 802.11 (operations outside the context of a basic service set), and other WAVE standards that may be developed to specify higher layer, application, or features. [4]

A WAVE device is a device compliant to the IEEE 1609 standards and IEEE Std 802.11 and supporting the information exchange with other WAVE devices. Typical WAVE devices are onboard units (OBUs) mounted in onboard equipment in vehicles and roadside units (RSUs) that generally operate when stationary, which can be installed in roadside poles, traffic lights, road signs, or roadside electronic cabinets. WAVE devices adopt orthogonal frequency-division multiplexing (OFDM) and achieve a communication range of approximate 1, 000 ft.

2.4.2. Historical Background

United States of America (USA)

Intelligent transportation systems (ITSs) have been developed throughout the world since the early 1990s to improve transportation safety, traffic efficiency, and user comfort and convenience. The coordination among moving vehicles and roadside infrastructures enables high-performance traffic management, pollution reduction, and energy conservation as well.

To implement such intelligent transportation systems, wireless communications and networking have been recognized, from the beginning, as a cornerstone and play an essential role in the evolution of such systems. The IEEE has developed the WAVE architecture and associated standards to promote and standardize the widely application of ITSs. A brief context for the IEEE WAVE standards and related activities are introduced as follows.

Back in 1991, the US Congress mandated the creation of the US ITS program called Intelligent Vehicle Highway Systems (IVHS) in the 1991 Intermodal Surface Transportation Efficiency Act (ISTEA). The IVHS program is administered by the US Department of Transportation (DOT) and supported by the nonprofit organization, Intelligent Transportation Society of America (ITSA). The program aimed at applying advanced concepts and technologies in areas of communications, navigation, and information systems to reduce traffic congestion, increase transportation efficiency, enhance mobility, improve highway safety, and reduce harm to the environment caused by automobiles (Sweeney, 1993). The USDOT renamed the IVHS program into ITS program to clarify the multimodal intent.

In the mid-1990s, a national systems architecture, i.e., the National ITS Architecture, was developed by the ITS program office. Furthermore, the related ITS standards were established by USDOT in 1996 to encourage the widespread use of ITS technologies in the USA.

In 1997, the Transportation Equity Act for the twenty-first century (TEA-21) was passed, which retained ISTEA's essential features while boosting investments to promote the deployment of the USDOT's ITS program.

In 1999, the US Federal Communications Commission (FCC) allocated 75 MHz in the 5.850–5.925 GHz band to DSRC uses (FCC, 2003), in response to the petition of the ITS America in 1997.

In 2004, the FCC published the DSRC Report and Order, FCC 03-324 (FCC, 2004), which established standard licensing and service rules in the ITS Radio Service in the 5.9 GHz band. The published FCC 03-324 adopted the standard for the physical and MAC layers (ASTM's E2213-03 ASTM 2003), which was developed by the American Society for Testing and Materials (ASTM) based on IEEE 802.11.

In November 2004, the IEEE 802.11 Task Group p was formed to develop an amendment to the IEEE 802.11 standard to include WAVE. The specifications for higher layers in the protocol suite were developed by another IEEE team, working group 1609.

In 2006 and 2007, a set of IEEE 1609 standards, as well as IEEE Std 802.11p, were managed for trial use to demonstrate the performance of WAVE architecture and protocols. The trial-used

IEEE 1609 standards include IEEE Std 1609.4–2006 (IEEE, 2006b), IEEE Std 1609.3–2007 (IEEE, 2007), IEEE Std 1609.2–2006 (IEEE, 2006a), and IEEE Std 1609.1–2006 (IEEE, 2006c).

Since 2010, a set of full-use IEEE 1609 standards have been published. The currently active full-use standards include IEEE 1609.0-2013 (IEEE, 2014a), IEEE 1609.2-2016 (IEEE, 2013, 2016a), IEEE 1609.3-2016 (IEEE, 2010, 2012b, 2014b, 2016b), IEEE 1609.4-2016 (IEEE, 2011b, 2016d), IEEE 1609.11-2010 (IEEE, 2011a), and IEEE 1609.12-2016 (IEEE, 2012a, 2016c).

Europe

To acquire EU-wide spectrum, radio applications require a harmonised standard, in case of ITS-G5 ETSI EN 302 571,first published in 2008. A harmonised standard in turn requires an ETSI System Reference Document, here ETSI TR 101 788. Commission Decision 2008/671/EC harmonises the use of the 5 875-5 905 MHz frequency band for transport safety ITS applications. In 2010 the ITS Directive 2010/40/EU was adopted. It aims to assure that ITS applications are interoperable and can operate across national borders, it defines priority areas for secondary legislation, which cover V2X and requires technologies to be mature. In 2014 the European Commission's industry stakeholder "C-ITS Deployment Platform" started working on a regulatory framework for V2X in the EU. It identified key approaches to an EU-wide V2X security Public Key infrastructure (PKI) and data protection, as well as facilitating a mitigation standard to prevent radio interference between ITS-G5 based V2X and road charging systems. The European Commission recognised ITS-G5 as the initial communication technology in its 5G Action Plan and the accompanying explanatory document,to form a communication environment consisting of ITS-G5 and cellular communication as envisioned by EU Member States. Various pre-deployment projects exist at EU or EU Member State level, such as the Testfeld Telematik, the digital testbed Autobahn, the Rotterdam-Vienna ITS Corridor, Nordic Way, COMPASS4D or C-ROADS.There exist real scenarios of implementation V2X standard as well. The first commercial project where V2X standard is used for Intersection movement assist use-case. It has been realized in Brno City / Czech Republic where 80 pcs of cross intersections are controlled by V2X communication standard from public transport vehicles of municipality Brno. [6]

2.4.3. Spectrum Allocation

The 75 MHz bandwidth allocated by the US FCC for DSRC usage is divided into seven 10 MHz operation channels, and the remaining 5 MHz bandwidth is reserved.

As illustrated in Fig. , the spectrum from 5.855 to 5.865 GHz is defined as Channel 172, while the spectrum from 5.865 to 5.875 GHz is defined as Channel 174, and so on, until Channel 184. Among all the defined seven channels, Channel 178 is used as Control Channel (CCH), while the 20-MHz Channel 175, covering Channel 174 and Channel 176, and Channel 181, merged by Channel 180 and 182, are specified as service channel (SCH). The rest Channel 172 and Channel 184 are kept for public safety applications. Specifically, according to FCC 06-110 document (FCC, 2006), Channel 172 is designated for "vehicle-to-vehicle safety communications for accident avoidance and mitigation and safety of life and property applications." Channel 184 is specified for "high-power, longer-distance communications to be used for public safety applications involving safety of life and property, including road intersection collision mitigation."

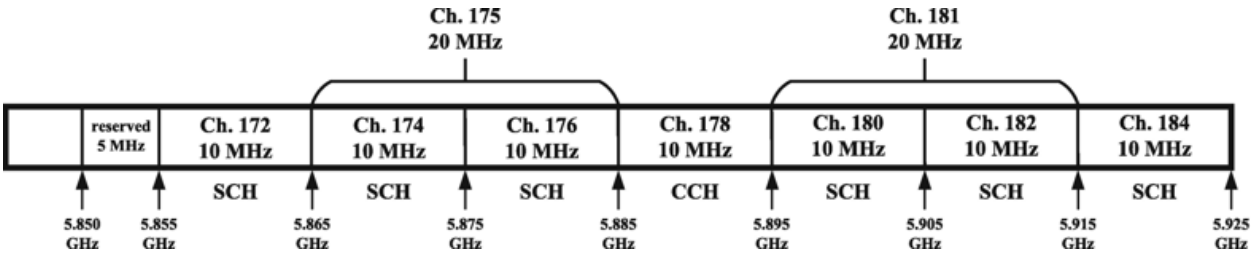

Figure 6: DSRC Spectrum Allocation in USA

The IEEE WAVE standards specify two kinds of radio channel, i.e., a control channel and several service channels. The CCH 178 is used only for system management messages and WAVE Short Message Protocol (WSMP) messages. The WSMP is a protocol for rapid exchange of the messages subject to intermittent radio connectivity, which will be described in the next subsection. To reducing the delivery delay, WAVE Short Messages (WSMs) are transmitted with controllable physical characteristics and thus could be transmitted with minimal channel capacity. As a result, WSMs are allowed to use both CCH and SCHs.

The service channels are used for application data transfers as well as IPv6 traffic.

Spectrum allocation for C-ITS in various countries is shown in the following table. Due to the standardization of V2X in 802.11p preceding C-V2X standardization in 3GPP, spectrum allocation was originally intended for the 802.11p based system. However, the regulations are technology neutral so that the deployment of C-V2X is not excluded.

Table 1 : Spectrum Allocation for DSRC

Country	Spectrum (MHz)	Allocated Bandwidth (MHz)
Australia	5855-5925	70
China	5905-5925 (trials)	20
Europe	5875-5905	30
Japan	755.5-764.5 and 5770-5850	9 and 80
Korea	5855-5925	70
Singapore	5875-5925	50
USA	5850-5925	75

2.4.4. Protocol Stack

The WAVE protocol stack and the mapping of these protocol elements to IEEE WAVE standards are illustrated in Fig 7.A data plane with protocols carrying higher layer information and a management plane for security and management functions are defined.

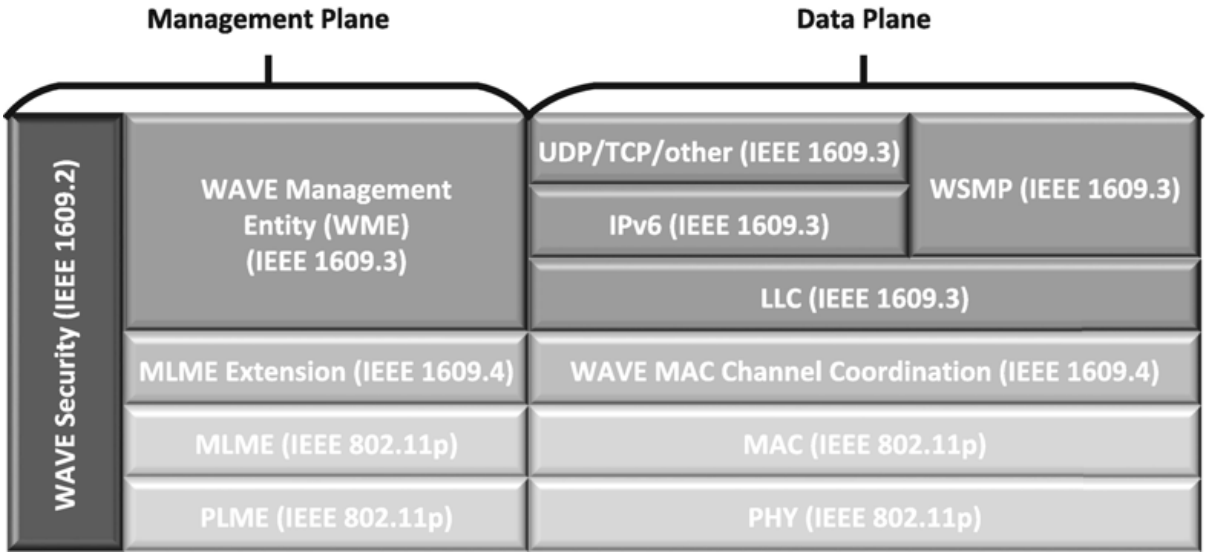

Figure 7: WAVE protocol Stack

The full-use IEEE 1609 standards and IEEE 802.11p, which has been incorporated in IEEE 802.11-2012, collectively define the WAVE protocol stack.

IEEE 1609.3 specifies two data plane protocol stacks, the Internet Protocol Version 6 (IPv6) standard and the WSMP, which are both based on the common set of lower layer protocols including physical (PHY), medium access control (MAC) and logical link control (LLC). IEEE 1609.3 specifies the use of two Ethertype values, IPv6 or WSMP, in the Ethertype field in the LLC header. IEEE 1609.3 also specifies a WAVE Management Entity (WME).

IEEE 802.11p, as an amendment to IEEE 802.11, specifies several physical layers (PHYs) and one medium access control (MAC) sublayer.

IEEE 1609.4 specifies multichannel operations in the data plane and extensions to the IEEE 802.11 MAC sublayer management entity (MLME).

Security services for WAVE applications and management messages are defined by IEEE 1609.2.

IEEE 1609.11 specifies the first application-level protocol for Over-the-Air Electronic Payment Data exchange in ITS. IEEE 1609.12 standardized the used identifiers and the allocated values of the identifiers in WAVE systems.

IEEE 1609.0 is a guide indicating the necessary services and architecture for WAVE system. [7]

2.4.5. Physical Layer Operations

At the physical layer (PHY), the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standard sets the specifications for frame transmissions and receptions over the air. This includes signal modulations, frame formats, and the way different elements of incoming frames need to be interpreted at the receivers. [6]

2.4.5.1. Orthogonal Frequency Division Multiplexing

IEEE 802.11a radios use orthogonal frequency division multiplexing (OFDM) to transmit over the air. OFDM divides the spectrum into a number of narrow - band subchannels, each carrying part of the information. Frequencies for each subchannel and transmission timings are chosen so that subchannel transmissions do not interfere with each other, hence the name “ orthogonal frequency. ” Each subchannel operates at low symbol rates, allowing interposition of guard intervals between subsequent symbols. These guard intervals make it possible to virtually eliminate intersymbol interference (ISI), resulting in high reliability at high data rates. OFDM is a simple yet efficient way to deal with signal distortion caused by multipath effects. [6]

2.4.5.2. Modulation and Coding Rates

Data rates for IEEE 802.11 radios are determined by modulation schemes (variations in periodic waveforms) and coding rates adopted for data frame transmissions. A modulation scheme is a

method to vary periodic waveforms to convey information over the channel. Binary phase – shift keying (BPSK) modulation conveys 1 bit per periodic waveform per subcarrier. Since IEEE 802.11a radios concurrently transmit data over 48 subcarriers, BPSK modulation allows each OFDM symbol to carry 48 bits of information. The coding rate represents the fraction of the total carried bits used for actual data bits. The rest are redundancy used to correct errors in the reception process. With a typical 20 MHz IEEE 802.11a channel, it takes 4 μ s to send each periodic waveform, including the guard interval, to prevent intersymbol interference. The combination of BPSK modulation with 1/2 coding rate results in a 6 Mbps data rate for IEEE 802.11a radios. As shown in Table, higher modulation and coding rate combinations result in higher data rates. However, higher data rates require cleaner signal at the receivers because frame reception is more prone to errors at higher data rates.

Table 2: Modulation and coding rates

Modulation	Coded Bits per Subcarrier	Coded Bits per OFDM Symbol	Coding Rate	Data Bits per OFDM Symbol	Data Rate for 20 MHz Channel (Mbps)
BPSK	1	48	1/2	24	6
BPSK	1	48	3/4	36	9
QPSK	2	96	1/2	48	12
QPSK	2	96	3/4	72	18
16-QAM	4	192	1/2	96	24
16-QAM	4	192	3/4	144	36
64-QAM	6	288	2/3	192	48
64-QAM	6	288	3/4	216	54

2.4.5.3. VANET Transceivers

VANET transceivers constitute a consolidated topic: OFDM radio transceivers have been developed for years and the Standard specifies both the signal characteristics and the signal processing blocks in great detail.

As a result, today not only industrial components are available, but also lab-made implementations, leveraging Software-Defined Radio (SDR) tools and paving the way to the experimental testing of new solutions (both backward-compatible and disruptive ones). [7]

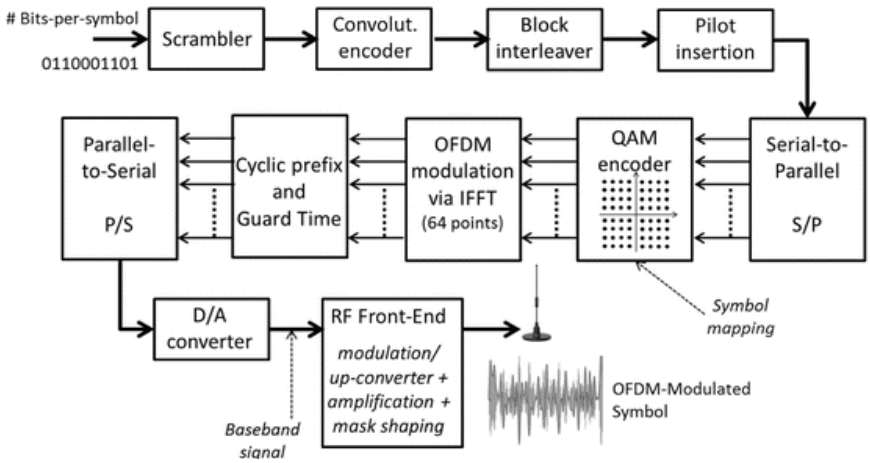

Figure 8: The building blocks of an IEEE 802.11 transmitter

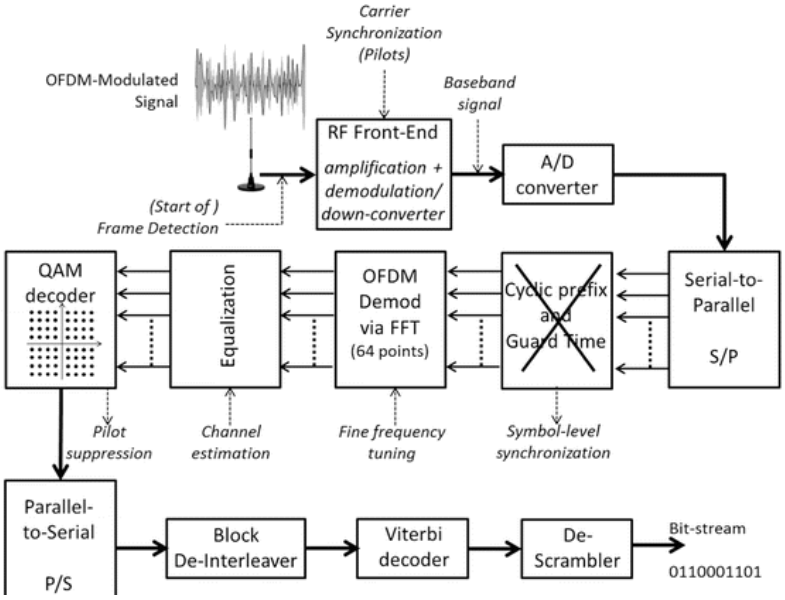

Figure 9: The building blocks of an IEEE 802.11 receiver

2.4.5.4. Frame Reception

2.4.5.4.1. Frame Preamble and Physical Layer Convergence Procedure Header

An IEEE 802.11 sender may select any modulation and coding rate combinations defined in the standard to transmit a frame. A receiver needs to know whether the signal detected through the channel corresponds to a frame or just noise, the frame duration length, and what modulation and coding rate combination is being used by the sender, in order to successfully receive a frame.

Every IEEE 802.11 frame starts with a known bit sequence called the frame preamble. The purpose of the frame preamble is to notify receivers of the imminent arrival of a frame and assist them to lock on to the signal. The frame preamble is followed by the physical layer convergence procedure (PLCP) header, which contains details on the frame payload (frame body), including frame length, modulation, and coding rate. The frame preamble and the Signal portion of the PLCP header are BPSK modulated in almost all IEEE 802.11 radio configurations. While the frame preamble has a zero coding rate, the Signal portion of the PLCP header is coded at 1/2 rate. The frame payload follows the PLCP header and can be modulated according to any of the eight modulation and coding combinations illustrated in Table. The sender indicates which modulation and coding combination is to be used to demodulate the frame payload in the Rate data field of the PLCP header. [7]

2.4.5.4.2. Frame Body

When an IEEE 802.11 radio is listening for incoming frames, it continuously looks for the known pattern of the frame preamble by demodulating the received signal using BPSK. When the frame preamble pattern is detected, the receiver attempts to decode the PLCP header. Upon success, the receiver demodulates incoming waveforms according to the frame modulation, coding, and duration indicated in the PLCP header. The resulting raw bits are passed to the media access control (MAC) layer, where a cyclic redundancy check (CRC) determines whether the frame was successfully received. While a radio is transmitting over the channel, it is not able to receive any incoming frames. Also, the radio would not be able to receive any incoming frames if it missed the frame preamble and PLCP header for any reason, because it would have no information on how to receive the frame body. Similarly, when a radio is already receiving a frame, it is not able to receive another incoming frame, because it would treat the preamble of the new incoming frame as part of the frame it is already demodulating. Furthermore, if the new incoming frame has a strong enough signal, it will collide with the earlier frame and prevent the earlier frame's successful reception. Also, since the frame body can be transmitted with a different modulation and coding rate combination, a receiver might be able to successfully receive and decode the frame preamble and PLCP header, yet fail to receive the frame body, depending on signal quality and interference. [7]

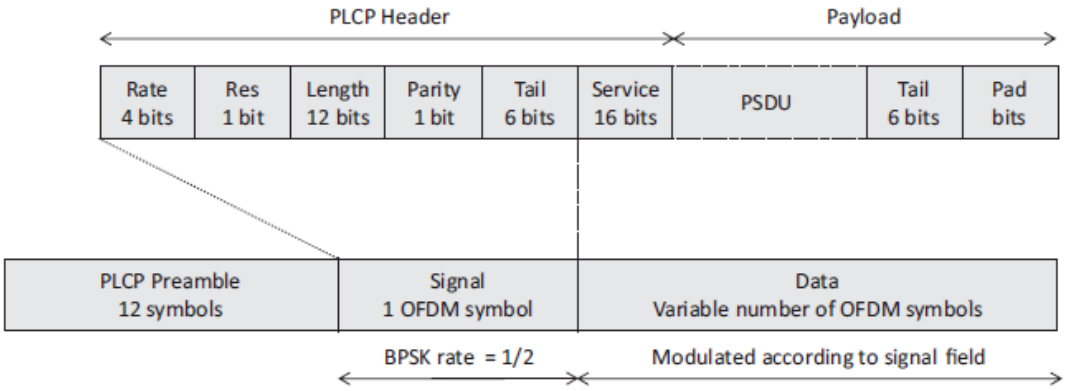

Figure 10: OFDM PLCP preamble, header, and data. PSDU, physical layer service data unit.

2.4.5.4.3. Frame Body Capture

IEEE 802.11 specifies a robust process for searching for and decoding the frame preamble and PLCP header. If a new frame arrives when a radio is still receiving the frame preamble and PLCP header of an earlier frame, the radio may choose to lock onto the new frame if it has sufficiently higher power than the earlier frame. Furthermore, it is sometimes possible to capture a new incoming frame during the body reception of an earlier one. This mechanism, called frame body capture, is not part of the IEEE 802.11 standard but is implemented in some radio chipsets as an optional feature.

With frame body capture, the PHY continuously monitors the received signal strength. When a sudden sharp rise (e.g., greater than 10 dB) is detected, the receiver assumes the arrival of a new frame with a stronger signal. It then abandons the previous frame and attempts to decode the frame preamble and PLCP header of the new incoming frame. Upon success, it starts reception of the new frame.

The frame body capture mechanism is intended for typical Wi - Fi usage environments, such as homes and offices. Incoming frames associated with stronger signals are preferred over others because they are more likely originated by nearby nodes. Frame body capture can be useful for vehicle safety communications as well. A vehicle may prefer messages from closer vehicles over messages from vehicles that are further away, because nearby vehicles will more likely impose immediate risks than faraway vehicles. Frame body capture techniques can also be used to reduce the negative impact of hidden terminal effects. [7]

2.4.5.5. PHY Amendments

The IEEE 802.11p PHY is an extension to the IEEE 802.11a PHY. It specifies minimal changes required for operations at 5.9 GHz. This allows us to build IEEE 802.11p chipsets without the high costs associated with the development of new radio chipsets. Furthermore, since IEEE 802.11a radios operate at 5.8 GHz, it is possible to build IEEE 802.11p radios with existing IEEE 802.11a chipsets reconfigured to operate in the nearby 5.9 GHz band.

2.4.5.5.1. Channel Width

The IEEE 802.11p PHY is based on the same OFDM scheme as IEEE 802.11a, but it operates over 10 MHz channels instead of the 20 MHz channels used by IEEE 802.11a devices. Ten megahertz (10 MHz) channels can be implemented by doubling the OFDM timing parameters and halving the frequency parameters used for 20 MHz IEEE 802.11a transmissions. This also entails doubling the size of the guard intervals between transmitted symbols.

Downscaling channel width is motivated by a recent study demonstrating that the guard interval used for transmitting over a 20 MHz channel is not long enough to offset the worst - case excess delay spread introduced by multipath effects in vehicular environments [CHCS08] . That is, the guard interval is not long enough to prevent intersymbol interference within one radio's own transmissions.

The excess delay spread varies with the environment. In a suburban environment, empirical measurements show that 90% of the excess delays introduced by multipath effects are lower than 0.6 μ s. This value increases to 1.4 μ s on highways and reaches 1.5 μ s in rural environments. Since the OFDM guard interval needs to be longer than the excess delay spread, the 0.8 μ s guard interval defined for IEEE 802.11a over 20 MHz channels would not be sufficient in a vehicle communications environment. For this reason, IEEE 802.11p uses a channel width of 10 MHz with 1.6 μ s guard intervals.

[SCHB07] indicate 8.5 MHz as the theoretical optimal channel width for WAVE, offering highest protection against excess delay spread. However, for ease of implementation with existing IEEE 802.11a chipsets, using 10 MHz wide channels in WAVE is a reasonable choice.

2.4.5.5.2. Spectrum Masks

Spectrum masks are used to limit excessive radiation from transmitting radios at frequencies beyond the intended bandwidth. Spectrum masks aim to protect adjacent channels from interference. IEEE 802.11p amendments specify four masks for the 5.9 GHz DSRC spectrum, corresponding to class A, B, C, and D operations. Class C is expected to be adopted for vehicle safety communications. Table 6.2 indicates spectrum mask requirements for each class at different offsets from channel center. For each 10 MHz channel, the transmitted spectrum shall have a 0 dBr bandwidth not exceeding 9 MHz [FCC04]. The spectrum masks defined for WAVE are more stringent than the ones for current IEEE 802.11 radios.

2.4.5.5.3. Improved Receiver Performance

Vehicles in adjacent lanes have been shown to interfere with each other if they are operating in two adjacent channels [RBKL07]. For example, vehicle V 1 transmitting over channel 176 could cause interference, preventing vehicle V2 in the adjacent lane (2.5 m apart) from receiving messages sent over channel 178 by another vehicle V3. Cross - channel interference is a well - known issue in wireless communications. The most effective and proper solution to this problem is through channel management policies. The definition of these policies is outside of the scope of IEEE 802.11 protocols. Nevertheless, given the even higher relevance of this issue in vehicular environments, IEEE 802.11p recommends improved receiver performance for adjacent channel rejections.

2.4.6. Media Access Control Layer Operations

At the media access control (MAC) layer, the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standard specifies channel access operations using carrier sensing multiple access with collision avoidance (CSMA/CA). This scheme includes a back - off algorithm and a handshake protocol designed to reduce frame collisions introduced by contention for media access and to address hidden terminal effects. The MAC layer filters out incoming frames that are not meant for the receiving node. [8]

2.4.6.1. Carrier Sensing Multiple Access with Collision Avoidance

Using CSMA/CA, a station must sense channel status before sending a frame. As long as the medium is busy, the MAC layer must wait for the medium to become idle again. The station may only transmit the frame if the medium is continuously idle for a distributed coordination

function interframe space (DIFS) interval plus an additional time randomly calculated by the MAC layer back - off algorithm. This random back - off delay is intended to reduce the likelihood of collisions when multiple stations are competing for medium access. If the channel is found busy during the DIFS interval or the back - off delay, the station should defer its transmission.

The CSMA/CA scheme does not require any coordination among nodes, is simple to implement, and offers good performance over moderately loaded channels. The IEEE 802.11 MAC layer adopts enhanced distributed channel access (EDCA) to offer four different traffi c classes or access categories. EDCA permits to differentiate services and prioritize outgoing traffi c at the MAC layer. Traffic prioritization is mainly provided through the adoption of arbitration interframe spacing (AIFS) intervals, which vary with the access category. The IEEE 802.11 MAC layer maintains separate ECDA queues, one for each access category. Frames assigned to these queues compete with each other for access to the medium. [8]

2.4.6.1.1. Hidden Terminal Effects

The hidden terminal problem arises when two nodes cannot sense each other's signals directly, but can both sense the signals from a third node. Under these conditions, the two nodes cannot determine whether the other node is transmitting. When hidden terminal effects are present, the CSMA/CA scheme alone cannot effectively mitigate frame collisions over the channel.

Figure 11 illustrates an example of hidden terminal effect in a vehicular ad hoc network. In this example, both vehicle A and vehicle C can sense vehicle B. However, they cannot sense each other. Before transmitting a frame, vehicle A must sense the channel first to make sure no other vehicles are currently transmitting. Since vehicle A cannot sense vehicle C, it may assume that the channel is clear when, in fact, vehicle C is transmitting. Hidden terminals permit simultaneous attempts to access the medium, resulting in collisions over the channel.

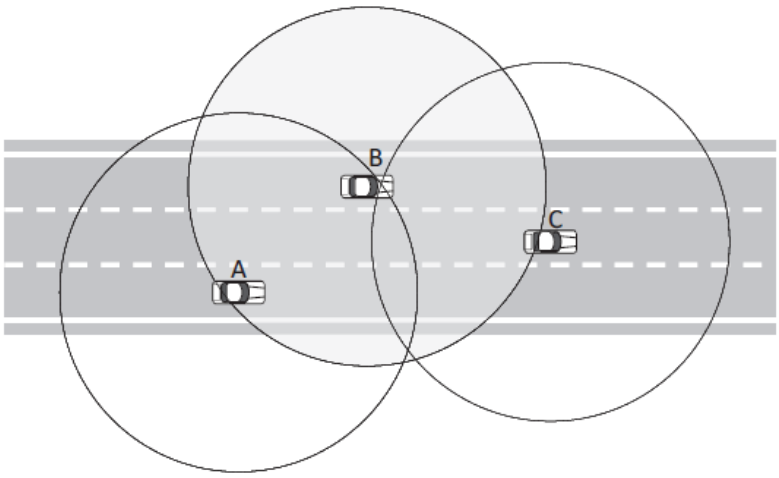

Figure 11: Hidden Terminal Effect

The IEEE 802.11 MAC layer may use a handshake protocol to reduce frame collisions caused by hidden terminals. Before transmitting a data frame, a node sends a Request to Send (RTS) frame indicating the intended destination. The destination of the RTS frame replies with a Clear to Send (CTS) frame. Any other nodes receiving the RTS or CTS frame should wait before attempting transmissions over the channel for a time duration indicated by the RTS and CTS frames.

In the previous example, vehicle A sends an RTS frame. Vehicle B replies with a CTS frame, indicating that the channel is clear. Vehicle C receives the CTS frame from B and therefore waits before attempting any further transmissions.

The RTS/CTS handshake only takes place before the transmission of packets exceeding a certain threshold, whereas smaller packets are sent immediately. It can be viewed as virtual carrier sensing that enables a node to sense another node when the two nodes cannot receive each other's signals directly. The RTS/CTS handshake is not used for broadcast and thus is unlikely to have an impact on most vehicle safety communications.

2.4.6.2. Basic Service Set

An IEEE 802.11 basic service set (BSS) is a group of radios configured to communicate with each other over the air - link. Transmissions from outsider BSSs are filtered out. An infrastructure BSS contains a station serving as the access point for the group. The IEEE 802.11 ad hoc mode, which does not require an access point, is called independent BSS (IBSS).

Each BSS is identified with a 48 - bit basic service set identification (BSSID) at the MAC layer. For an infrastructure BSS, the BSSID will be the 48 – bit MAC address of the access point. A special case of the BSSID is the wildcard BSSID, where all bits are set to " 1. " Joining a BSS requires executing time - consuming operations such as scanning, authentication, and association. Scanning can be passive (listening to discover beacon frames) or active (sending probe request frames). Authentication may require the use of shared keys and encryption functions. The association process consists of further exchanges of information between the node and the access point. [8]

2.4.7. MAC Layer Amendments

The key purpose of IEEE 802.11p MAC layer amendments is to enable efficient communication setups while significantly reducing the associated overheads.

Unlike physical layer (PHY) amendments, modifications to the MAC layer typically result in changes only to the radio software. This gives us relatively more freedom in the design of these amendments.

Traditional IEEE 802.11 MAC layer operations are too time consuming for wireless access in vehicular environments (WAVE). Many WAVE use cases cannot tolerate the delays introduced by operations such as scanning for BSS beacons, authentication, and association.

A key MAC amendment introduced to address this issue is the ability to send data frames outside the context of a BSS (OCB). This mechanism is available to WAVE radios that are not members in a BSS. These stations are allowed to transmit and receive data frames using a wildcard BSSID value without any previous setups. This means vehicles can immediately communicate with each other without any additional overheads as long as they operate in the same dedicated short - range communications (DSRC) channel and use the wildcard BSSID.

2.4.8. WAVE Upper Layers

In the United States, the 5.9 GHz dedicated short - range communications (DSRC) spectrum is organized into seven 10 - MHz channels. This makes it possible for a single wireless access in vehicular environments (WAVE) radio to engage in concurrent operations over multiple DSRC channels. For instance, a WAVE radio can broadcast safety messages over the control channel (CCH) (#178) while supporting nonsafety applications over a service channel (SCH) #182.

DSRC multichannel operations are specified in the Institute of Electrical and Electronics Engineers (IEEE) 1609.4 standard. The standard defines a time division scheme that allows WAVE radios to switch from one DSRC channel to another. This mechanism allows different applications to communicate simultaneously over separated physical channels. The IEEE 1609.4 time division scheme is independent of the spectrum and is therefore directly applicable to other frequency bands.

Vehicle safety applications interact with the WAVE protocol stack through the WAVE Short Message Protocol (WSMP), specified in the IEEE 1609.3 standard. WSMP offers the source the means to specify frame transmission parameters. [8]

2.4.8.1. DSRC Multichannel Operations

The first version of the IEEE 1609.4 standard was issued at the end of 2006 [IEEE06] . Version D1.6 was issued in 2010 [IEEE10a] . With respect to the WAVE protocol stack, IEEE

1609.4 sits directly on top of IEEE 802.11p at the upper media access control (MAC) layer, as shown in Figure 12.

The IEEE 1609.4 standard specifies a time division scheme for multichannel operations, and defines synchronization intervals timing, channel switching, and channel routing.

One approach to multichannel operations is to use different radios to communicate over different channels. The IEEE 1609.4 standard, however, assumes a single radio operating over multiple DSRC channels. The protocol does not include mechanisms for multiradio operations or for future coexistence of single - radio and multi - radio solutions.

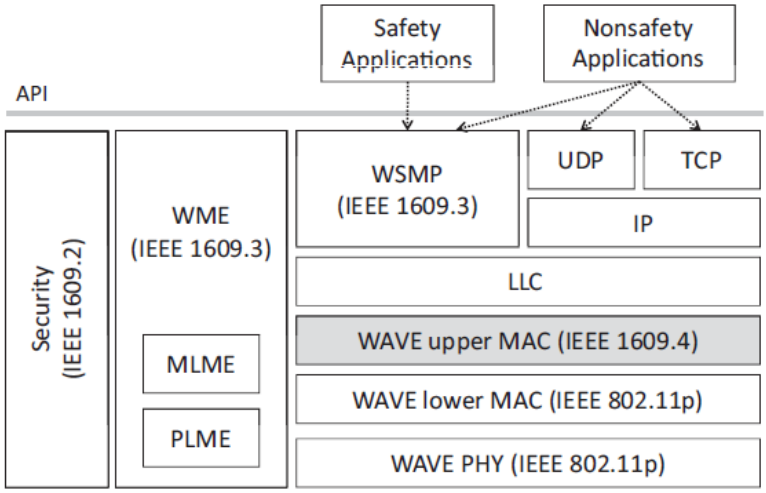

Figure 12: IEEE 1609.4 in WAVE protocol Stack

2.4.8.2. Time Synchronization

The IEEE 1609.4 channel switching scheme requires all devices to maintain synchronization with boundaries of seconds within a common time reference.

For instance, existing implementations exploit the pulse per second (PPS) signal issued by some Global Positioning System (GPS) devices to keep precise timing.

The standard allows radios that do not have direct access to precise timing sources to acquire timing information from other WAVE radios. For instance, this can be achieved by reading the timing information included in WAVE Timing Advertisement frames sent by WAVE upper layer protocols. To become a provider of timing information to other devices, a WAVE radio must meet some minimum Coordinated Universal Time (UTC) synchronization requirements.

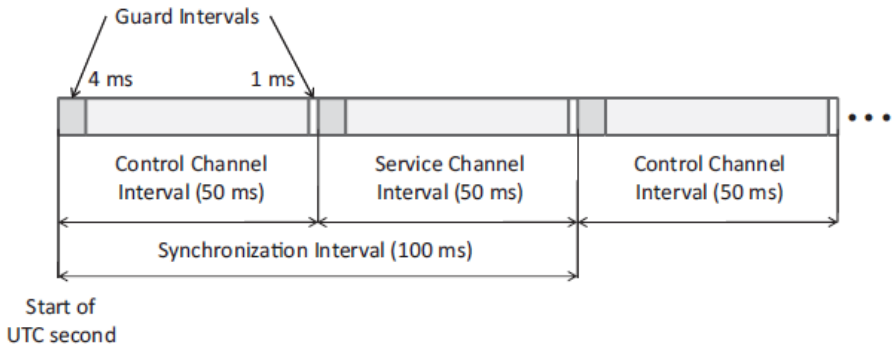

Figure 13: Synchronization, CCH, SCH, and guard intervals

2.4.8.3. Synchronization Intervals

The IEEE 1609.4 time division scheme is based on synchronization intervals.

Each second comprises 10 100 - ms synchronization intervals (Figure 13). Each synchronization interval is divided into a CCH interval followed by an SCH interval. Each CCH interval marks the beginning of a UTC second or multiples of 100 ms thereafter.

In its original form, the IEEE 1609.4 standard assumes WAVE radios use the CCH to broadcast safety messages and advertisements for services available on an SCH. The standard permits the transmission of safety messages over the CCH during SCH intervals. However, vehicles equipped with a single radio tuned to an SCH will not be able to receive these messages.

The standard allows WAVE radios not to engage in channel switching. However, WAVE radios that are permanently tuned to the CCH are expected to keep track of the beginning and end of synchronization intervals. This allows these radios to suspend transmitting messages over the CCH during SCH intervals if necessary. This may happen, for example, when a radio expects that some nearby radios may be using the SCHs to support different applications.

2.4.8.4. Guard Intervals

IEEE 1609.4 defines front guard intervals at the beginning and rear guard intervals at the end of each control or SCH interval.

A front guard interval is defined as the sum of synchronization tolerance and maximum channel switching time. Synchronization tolerance is the expected precision of a device's internal clock, with respect to its ability to align with the UTC time. Maximum channel switching time is the overhead introduced by the operation of switching from one physical channel to another.

The purpose of rear guard intervals is to allow extra time to complete frame reception before switching to another channel. WAVE radios are not permitted to transmit any frames during rear guard intervals. With current radio chipset technology, 4 - ms front guard intervals and 1 – ms rear guard intervals are sufficient to protect the communications. [8]

2.4.8.5. Channel Switching

At the beginning of a control or SCH guard interval, a WAVE radio suspends MAC layer activities. The radio waits until the end of the guard interval and then starts communications over the channel or resumes activities suspended at the end of the previous cycle.

A radio cannot initiate frame transmission during a guard interval. If frame transmission has not been completed before the start of the next guard interval, a radio may drop the frame and abort the process. However, the IEEE 1609.4 standard suggests that implementations should minimize frame drops through careful transmissions scheduling.

Data frames that are not transmitted during a control or SCH interval are stored in local queues until the next cycle. IEEE 1609.4 defines a mechanism to assign an expiration time to individual data frames. This allows the MAC layer to purge data frames that expired before having a chance of being transmitted.

The standard requires radios to declare channel busy during guard intervals so that transmissions at the beginning of the next channel interval are scheduled through the back - off mechanism. If a front guard interval begins while a radio is still engaged in frame reception, the process is abandoned with loss of the frame.

2.4.9. WAVE Short Message Protocol

The WSMP is positioned between the data link and the application layers (Figure 14). This corresponds to the network and transport layers in an Internet Protocol (IP) network. WSMP offers an interface to DSRC safety applications.

Unlike IP networks, DSRC safety communications do not require routing functions. Typically, the communications take place over a single hop. Frames sent over the DSRC link are not fragmented by the network and there is no need for reassembly at the destination. Functions to increase packet reception reliability, such as those offered by the Transmission Control Protocol (TCP), are not required in communication scenarios dominated by pervasive broadcast. Therefore, WSMP is designed to support a few basic services.

WSMP essentially offers the means for the source to specify transmission parameters for individual frames, including channel number, data rate, and transmission power. Also, it allows the sender to indicate which application the message is intended for, facilitating demultiplexing at the destination. This is done through the use of the Provider Service Identifier (PSID) field in the WSMP header. PSIDs are conceptually similar to TCP port numbers. Safety messages share a single PSID, so the PSID mechanism is intended for the coexistence of safety and nonsafety applications.

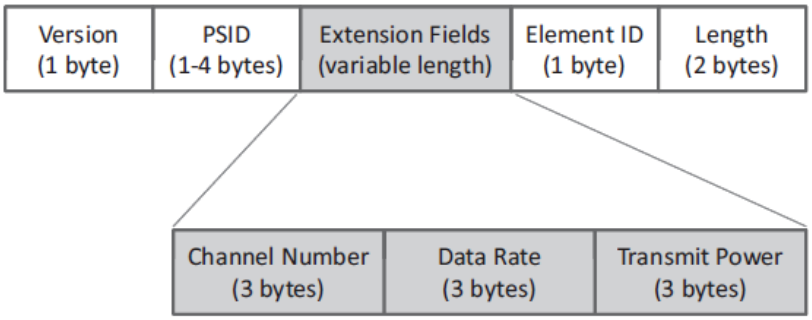

Figure 14: WSMP Header

WSMP is specified in the IEEE 1609.3 standard [IEEE10b]. The current version of the IEEE 1609.3 standard is version 2. Figure 9.10 illustrates the WAVE Short Message header. The first byte of the header contains information about the protocol version (4 bits are reserved for future use). The second header field describes the PSID associated with the packet. The PSID has a variable length to allow for variable numbers of PSIDs.

The Extension Fields have a variable length to accommodate the description of several elements. Three such elements are specified in the current IEEE 1609.3 standard: channel number, data rate, and transmission power. The Element ID field indicates the end of the Extension Fields. The last 2 bytes of the WSMP header describe the length (expressed in bytes) of the message payload.

2.5. Cellular V2X (C-V2X)

2.5.1. Preface

The Cellular V2X (C-V2X) is a 3GPP standard describing a technology to achieve the V2X requirements. C-V2X is an alternative to 802.11p, the IEEE specified standard for V2V and other forms of V2X communications. Pre-commercial C-V2X deployments have recently gained considerable momentum with support from multiple automakers.

2.5.2. History

Cellular V2X uses 3GPP standardized 4G LTE or 5G mobile cellular connectivity to send and receive signals from a vehicle to other vehicles, pedestrians or to fixed objects such as traffic lights in its surroundings. It commonly uses the 5.9 GHz frequency band to communicate – this being the officially designated intelligent transportation system (ITS) frequency in most countries. C-V2X can function without network assistance and has a range that exceeds a mile. In 2014, 3GPP Release 13 spurred studies to test the applicability of the then current standards to V2X. This resulted in the 3GPP Release 14 specifications for C-V2X communications, finalized in 2017. 3GPP Release 15 introduced 5G for V2N use-cases, and 3GPP Release 16 includes work on 5G NR direct communications for V2V/V2I. [16]

C-V2X was developed within the 3rd Generation Partnership Project (3GPP),[1] to replace the US promoted Dedicated short-range communications (DSRC) and the Europe originated Cooperative Intelligent Transport Systems (C-ITS) As such standards are decisive steps towards the target autonomous driving and clues to market influence, especially as the National Highway Traffic Safety Administration (NHTSA) plans to propose the compulsory introduction of vehicle-to-everything technology off 2020 for all US vehicles.

In Europe, the EU announced in July 2019 that it was adopting a technology-neutral approach to C-ITS, leaving the way forward for 4G, 5G and other advanced technologies to be part of V2X applications and services.

In the United States, the Federal Communications Commission proposed late in 2019 that 20 MHz and possibly 30 MHz of the 5.9 GHz band be allocated to C-V2X.

2.5.3-System Implementation

2.5.3.1. C-V2X Architecture

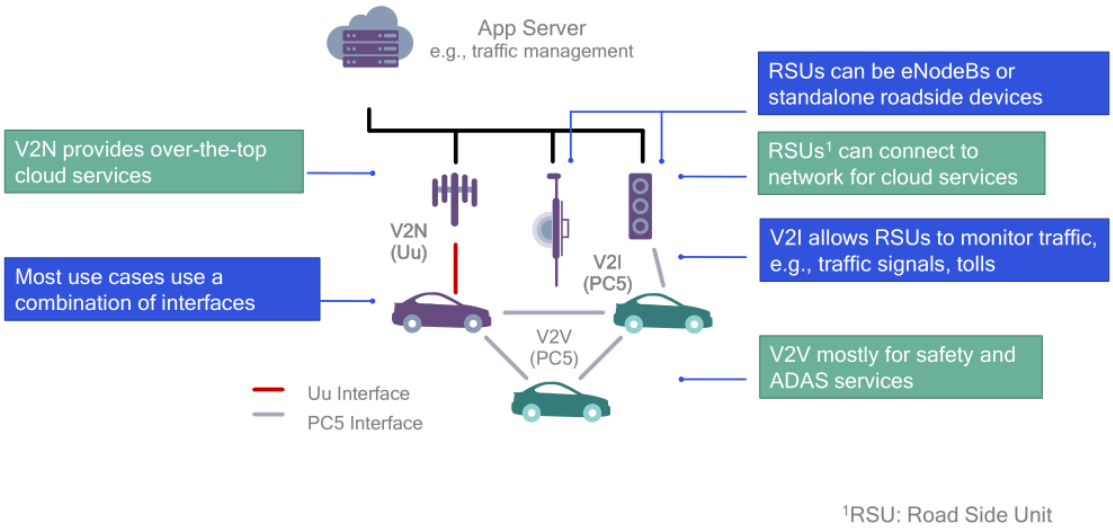

Figure 15: System Architecture

V2X communications consists of four types of communications:

• Vehicle-to-vehicle (V2V)

• Vehicle –to-infrastructure (V2I)

• Vehicle-to-network, (V2N)

• Vehicle-to-pedestrian (V2P)

Cellular-V2X defines a new air interface called PC5 for V2V, V2I communication.

V2N is still over the legacy LTE Uu air interface and provides over the top cloud services. [16]

2.5.3.2. C-V2X Communication Interface

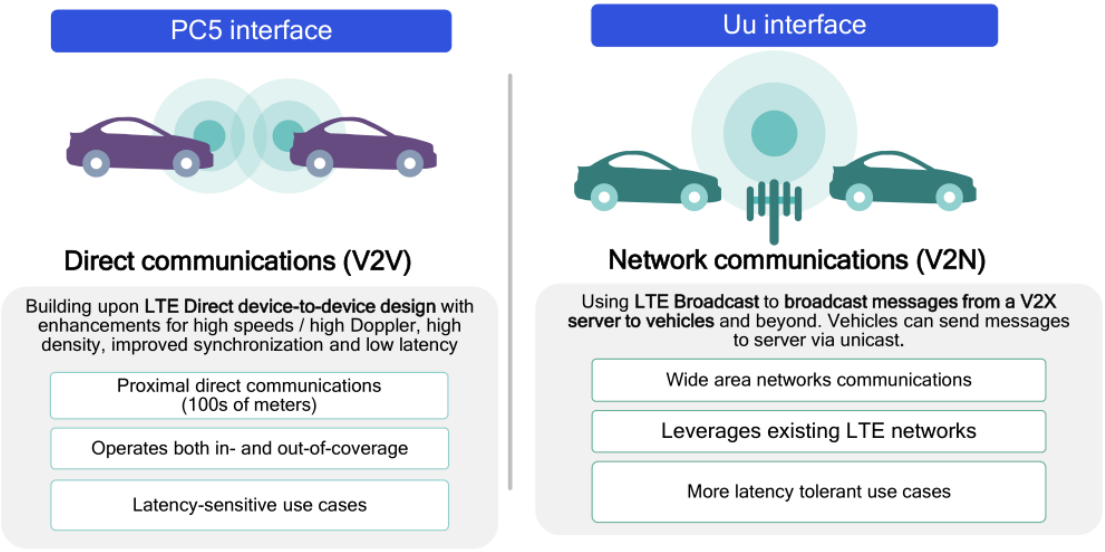

Figure 16: Communication Interfaces

C-V2X defines two Complimentary Transmission Modes:

1) Direct safety communication independent of cellular network via PC5 interface

   It's between vehicles (V2V), between vehicles and infrastructure (V2I), and vehicles and other road users, such as cyclists and pedestrians (V2P). In this mode, C-V2X operates in the 5.9 GHz frequency band – the ITS (intelligent transport system) spectrum that has been identified and harmonized internationally for safety purposes. In this mode, C-V2X works independently of the cellular networks.

   The ITS 5.9 GHz spectrum band has been set aside by governments worldwide to enable vehicles to talk to each other using dedicated frequencies that won't be subject to interference. Using this band, C-V2X can support direct low latency connections over short distances, without the involvement of the cellular network. Like 802.11p, C-V2X employs the global navigation satellite system (GNSS) to determine the position of the vehicle and to synchronize communications between vehicles and with roadside infrastructure. In this mode, no SIM card is required, as the vehicle doesn't need to connect to the cellular network. The vehicle and its driver remain anonymous, as no cellular subscription is required for direct safety communications.

   C-V2X and 802.11p can co-exist in the ITS spectrum by employing different channels within the 5.9 GHz band. Just 10MHz of spectrum in the 5.9GHz band is required to

support basic safety services, while 70MHz could support advanced safety services, such as sharing large amounts of data collected by on-vehicle sensors.

2) Network communications for complementary services via Uu interface

C-V2X employs the conventional mobile network to enable the vehicle to receive information about road conditions and traffic in the area. In this mode, C-V2X operates in spectrum that has been licensed to mobile operators to provide connectivity to their customers.

C-V2X can also support vehicle-to-network (V2N) applications delivered over commercially-licensed cellular spectrum. This mode can be used to provide network assistance for safety-related features, as well as commercial services, requiring the involvement of a mobile operator, providing access to cloud-based data or information. This mode also enables C-V2X to harness the data security and privacy of mobile networks.

Time-critical services can be supported by edge computing – the deployment of computer servers and data analytics on the edge of the network.

Developed to be both deployable in the near term and future-proof, C-V2X is versatile enough to support both today's use cases, and those of tomorrow. Compatible with 4G and 5G cellular networks, it is intended to be both scalable and interoperable. In time, C-V2X will support advanced driver assistance systems (ADAS) where vehicles can cooperate, coordinate and share information collected by sensors, and ultimately, connected automated driving (CAD).

2.5.4. C-V2X based on 5G

5G will support a broad range of V2X and non-V2X use cases, including enhanced Mobile Broadband (eMBB), massive Internet of Things (mIoT) and mission-critical services. These use cases have diverse requirements, including high data rates for eMBB, low power consumption and high scalability for mIoT, and ultra-low latency and high-reliability communications (URLLC) for mission-critical services. Meanwhile, the ever-increasing mobile data traffic volume means that 5G must support more spectrum bands and types, from sub-6GHz to mmWave, as well as licensed, shared and unlicensed allocations. To satisfy these diverse and sometimes stringent requirements under one platform, the 5G air interface must be highly capable, with the flexibility to adapt to target use cases and spectrum. The 5G system is introduced in 3GPP Release 15. The 5G radio interface is referred to as 5G New Radio (NR). This new radio interface will support an array of advanced features such as scalable Orthogonal Frequency Division Multiplexing (OFDM) numerology, flexible self-contained Time Division

Duplex (TDD) sub-frame, massive Multiple-Input Multiple-Output (MIMO), diverse spectrum bands and bandwidth, V2X and many others. These features together deliver a radio interface with much lower latencies, enhanced reliability, higher spectral efficiency and greater data throughput. [17]

The 5G system also is evolving toward an end-to-end infrastructure capable of delivering a consistent user quality of experience in a heterogeneous environment across a wide variety of use cases, using multi-access including 5G radio but also 3G-4G, Wi-Fi, LPWAN and fixed networks. A number of technologies are introduced, such as NFV/Software Defined Networking (SDN), Control-User Plane Separation (CUPS), MEC, network slicing, automation and Development and Operations (DevOps). These allow more flexibility and move from the current monolithic core network to a shared pool of virtual resources allocated dynamically for different types of services. [17]

5G networks also will use MEC. As defined by ETSI MEC, a couple of new concepts and technologies are introduced. The first is a cloud hosting environment at the edge of the network that enables the deployment of virtualized network functions and third-party applications closer to the user and devices. Second is a set of Application Program Interfaces (APIs), mainly radio and location APIs exposed to third-party applications. Virtualization is also an important technology utilized in the implementation of MEC. The MEC hosting infrastructure consists of hardware resources and a virtualization layer. The virtualization manager provides, in brief, an infrastructure-as-a-service (IaaS) capability. [17]

Network slicing allows network operators to define different types of services and allocate dynamically proper resources to support this end-to-end service by configuring different network segments. By using slicing technology, one physical network can be divided in multiple virtual networks, each supporting different service requirements or even different customers. Virtualization, combined with SDN and slicing, enables the network to allocate resources to different slices in a dynamic fashion and reroute dynamically the traffic through the different virtual functions of each slice. The amount of resources allocated can adapt based on traffic conditions. Moreover, traffic from one slice does not impact traffic from other slices. In addition, management of a slice can be delegated to third parties.

Last but not least, management of 5G systems will be much more automated. Starting with continuous integration/continuous delivery (CI/CD), new network functions developed by suppliers are carried over to network operators via programmatic interfaces. Next, those functions are integrated/tested and deployed in live networks through a standard automation environment that will take care of onboarding, instantiation, lifecycle management, update/upgrade and termination. This DevOps system will be associated with analytics that will

collect events to monitor the network and services and provide recommendations or trigger actions to reconfigure the network automatically.

2.5.5. C-V2X and DSRC Comparison

This section compares C-V2X and DSRC, with a focus on enabling vehicular safety for longer range and enhanced reliability use cases, consistent performance in congested situations and enabling advanced vehicular communication use cases.

C-V2X has several key advantages over DSRC, including:

- Longer range and enhanced reliability, resulting in enhanced safety
- More consistent performance under traffic congestions
- Evolution path towards 5G for emerging applications
- Better technical advantage over DSRC

Longer C-V2X Range and better reliability for Enhanced Safety

Based on link level simulation analysis, 27 C-V2X can achieve Line of sight (LOS) and non-LOS (NLOS) V2V ranges of 443 m and 107 m, respectively, compared to 240 m and 60 m, respectively, for DSRC. Longer range can be directly translated into earlier alerts and better visibility of unexpected and potentially dangerous situations. It also allows vehicles to travel at higher speeds while still being able to stop in time to avoid hazardous conditions.

Figure 3 illustrates a scenario where a disabled vehicle behind a blind curve is transmitting alerts to approaching vehicles under both icy and normal road conditions. If DSRC is used, an approaching vehicle must maintain a speed below 28 mph and 46 mph for icy and normal road conditions, respectively, to stop in time to avoid accident after receiving an alert. With C-V2X, the incoming vehicle receives the alert earlier at a longer distance away. Therefore, it can stop before reaching the disabled vehicle even if it is traveling at higher speed, (for example, 38 mph and 63 mph, for icy and normal road conditions, respectively).

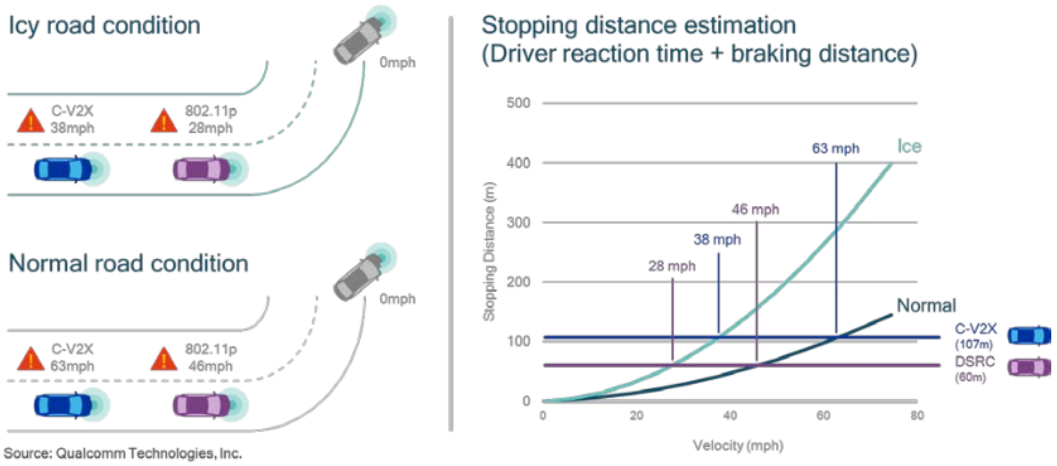

Figure 17: Disabled Vehicle after Blind Curve Use Case

Figure 4 illustrates a do-not-pass use case scenario where a vehicle following a large truck has limited visibility of the opposite traffic. At the same time, a second vehicle is approaching the first vehicle from the adjacent lane. The higher the vehicle speeds, the faster the two vehicles approach each other and the more dangerous is the situation if the first vehicle selected to pass the truck.

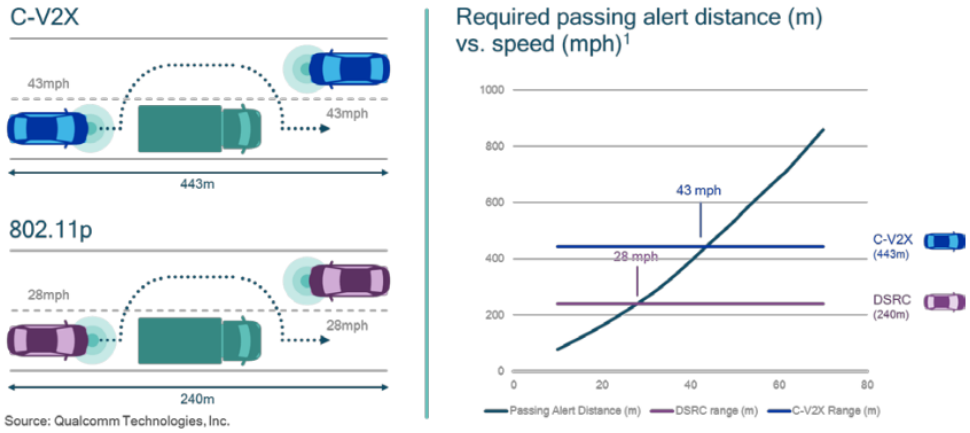

Figure 18: Do-Not-Pass Warning Use Case

With V2V communication, the second vehicle can send warning alerts, which are used by the first vehicle to decide whether it should pass the truck. Like the first example above, the longer C-V2X range allows the first vehicle to receive the alerts earlier, thus allowing it to safely pass the truck even if it is traveling at a higher speed compared to the case when DSRC is used.

Consistent C-V2X Performance Even in Congestion Conditions

Safety messages are generally transmitted more or less periodically for a given duration. C-V2X is designed to leverage these quasi-periodic traffic arrival patterns to deterministically pre-allocate resources for subsequent traffic arrivals. This semi-persistent scheduling mechanism ensures that resources are available when traffic arrives. There is no need to go through resource contention procedures for subsequent traffic, thus allowing C-V2X to maintain low latency even as vehicle density increases.

In addition, to improve channel access when traffic load is high, C-V2X selects the best resource, rather than the first available one, for traffic transmission. A vehicle with pending traffic first measures the relative energy levels of the available radio resource blocks averaged over a short period of time. It then ranks the radio resource blocks and selects one for transmission among those with the lowest relative energy levels. This least-energy resource selection scheme delivers better signal quality in the presence of other transmitting vehicles. Figure 5 (a) illustrates a scenario where C-V2X allows vehicles A and B to transmit simultaneously to their respective recipients. Figure 5 (b) illustrates the corresponding DSRC scenario when vehicle A is transmitting, and vehicle B backs off to avoid collision. C-V2X enables graceful degradation (decrease in range) in the face of congestion, while in DSRC, more abrupt changes in packet transmission rate occur in time, in extreme cases leading to lack of service.

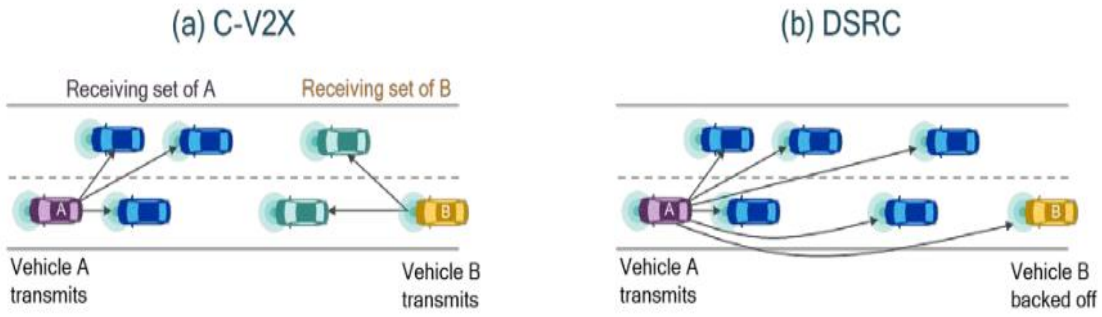

Figure 19: Least Energy Resource Selection Enables Sustained Communication with Increased Vehicle Density

Evolution Path toward 5G

Release 14 C-V2X and DSRC support low latency and reliable exchange of messages among vehicles and the infrastructure to enhance safety and efficiency. However, this is just the beginning of the evolution toward future vehicles that are expected to be increasingly autonomous with many advanced safety and non-safety functionalities. This evolution will call for additional V2X communication capabilities, such as higher capacity, ultra-low latency, ultra-

high reliability, longer range and higher data rates. Continuous advancements in V2X technologies is needed to meet emerging requirements.

Better technical advantage over DSRC

The technical differences will be discussed in the table below.

Table 3: C-V2X Technical Advantages over IEEE 802.11p

	C-V2X: PC5 802.11p C-V2X: PC5 Advantage	C-V2X: PC5 802.11p C-V2X: PC5 Advantage	C-V2X: PC5 802.11p C-V2X: PC5 Advantage
Synchronization	Synchronous	Asynchronous	Spectral Efficiency. Synchronization enables0020time division multiplexing (TDM) and lowers channel access overhead.
Resource Multiplexing Across Vehicles	FDM and Time Division Multiplexing (TDM) Possible	TDM Only	Frequency Division Multiplexing allows for larger link budget and therefore longer range – or more reliable performance at the same range.
Channel Coding	Turbo	Convolutional	Coding gain from turbo codes leads to longer range – or more reliable performance at the same range.
Retransmission	Hybrid Automatic Repeat Request (HARQ)	No HARQ	Leads to longer range – or more reliable performance at the same range.
Waveform	SC-FDM	OFDM	Allows for more transmit power with the same power amplifier. Leads to longer range – or more reliable performance at the same range.
Resource Selection	Semi- persistent transmission with relative energy-based selection.	Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA)	Optimizes resource selection with selection of close to ‘best’ resource with no contention overheads. By contrast 802.11p protocol selects the first “good enough” resource and requires contention overhead.

2.5.6. Conclusion

Summarizing, C-V2X is expected to be instrumental in transforming connected transportation services throughout the globe. C-V2X direct communications can operate both in ITS spectrum and in commercial cellular spectrum, combined with network-based C-V2X communications operating on existing and future cellular networks. The newly formed 5GAA is a clear proof of the existence of a strong ecosystem of leading automotive and telecommunication companies which is confident that a technically superior standards-based, cost-effective and scalable access technology from the cellular industry will carry C-ITS and Connected Vehicle applications well into the 5G era and beyond.

2.6. Proposed work

A system based on implementation of vehicle-to-everything (V2X) communication technology through the presence of on board units inside the vehicle and road side units to collect all needed data about cars and roads then uploading these information to an application server to process them then transmit the data back to the vehicles to give the driver all the needed information and warnings is proposed.

The system introduces many use cases as emergency braking warnings, road work hazard, car maintenance and many other use cases that can be added using software only.

The communication between vehicles will be based on Wireless Access in Vehicular Environment (WAVE) using IEEE 802.11p standard and the communication with the application server will be through LTE networks (4G)

All on board units and roadside units is implemented using hardware components which are mainly raspberry pi interfaced with some modules as OBD, GPS, USRP which is a kit belongs to software defined radios to act as transceivers and (LCD Touch Screen) to be the user interface between the driver and the system.

2.7. MQTT

2.7.1. MQTT Definition

MQTT (MQ Telemetry Transport) is a lightweight messaging protocol that provides resource-constrained network clients with a simple way to distribute telemetry information. The protocol, which uses a publish/subscribe communication pattern, is used for machine-to-machine (M2M) communication and plays an important role in the internet of things (IoT).

The MQTT protocol is a good choice for wireless networks that experience varying levels of latency due to occasional bandwidth constraints or unreliable connections.

The MQTT protocol surrounds two subjects: a client and a broker. An MQTT broker is a server, while the clients are the connected devices. When a device (or client) wants to send data to a server (or broker) it is called a publish. When the operation is reversed, it is called a subscribe.

If the connection from a subscribing client to a broker is broken, then the broker will buffer messages and push them out to the subscriber when it is back online. If the connection from the publishing client to the broker is disconnected without notice, then the broker can close the connection and send subscribers a cached message with instructions from the publisher.

2.7.2. History of MQTT Protocol

MQTT was created by Dr. Andy Stanford-Clark of IBM and Arlen Nipper of Arcom -- now Eurotech -- in 1999. MQTT was created as a cost-effective and reliable way to connect monitoring devices used in the oil and gas industries to remote enterprise servers. When challenged with finding a way to push data from pipeline sensors in the desert to off-site supervisory control and data acquisition (SCADA) systems, they decided upon a TCP/IP-based publish/subscribe topology that would be event-driven to keep satellite link transmission costs down. [9]

Although MQTT is still closely associated with IBM, it is now an open protocol that is overseen by the Organization for the Advancement of Structured Information Standards (OASIS).

Though the name suggests it, MQTT is not part of the original IBM MQSeries; however, as of version 7.1, it is available in WebSphere MQ. MQTT was previously known as the SCADA protocol, MQ Integrator SCADA Device Protocol (MQIsdp) and WebSphere MQTT (WMQTT), although all these variations have fallen out of use. [9]

2.7.3. MQTT Mechanism

An MQTT session is divided into four stages: connection, authentication, communication and termination. A client starts by creating a Transmission Control Protocol/Internet Protocol (TCP/IP) connection to the broker by using either a standard port or a custom port defined by the broker's operators. When creating the connection, it is important to recognize that the server might continue an old session if it is provided with a reused client identity.

The standard ports are 1883 for nonencrypted communication and 8883 for encrypted communication – using Secure Sockets Layer (SSL)/Transport Layer Security (TLS). During the SSL/TLS handshake, the client validates the server certificate and authenticates the server. The client may also provide a client certificate to the broker during the handshake. The broker can use this to authenticate the client. While not specifically part of the MQTT specification, it has become customary for brokers to support client authentication with SSL/TLS client-side certificates. [9]

Because the MQTT protocol aims to be a protocol for resource-constrained and IoT devices, SSL/TLS might not always be an option and, in some cases, might not be desired. On such occasions, authentication is presented as a cleartext username and password, which are sent by the client to the server, as part of the CONNECT/CONNACK packet sequence. In addition, some brokers, especially open brokers published on the internet, will accept anonymous clients. In such cases, the username and password are simply left blank. [9]

MQTT is called a lightweight protocol because all its messages have a small code footprint. Each message consists of a fixed header (2 bytes) an optional variable header, a message payload that is limited to 256 megabytes (MB) of information and a quality of service (QoS) level.

During the communication phase, a client can perform publish, subscribe, unsubscribe and ping operations. The publish operation sends a binary block of data (the content) to a topic that is defined by the publisher.

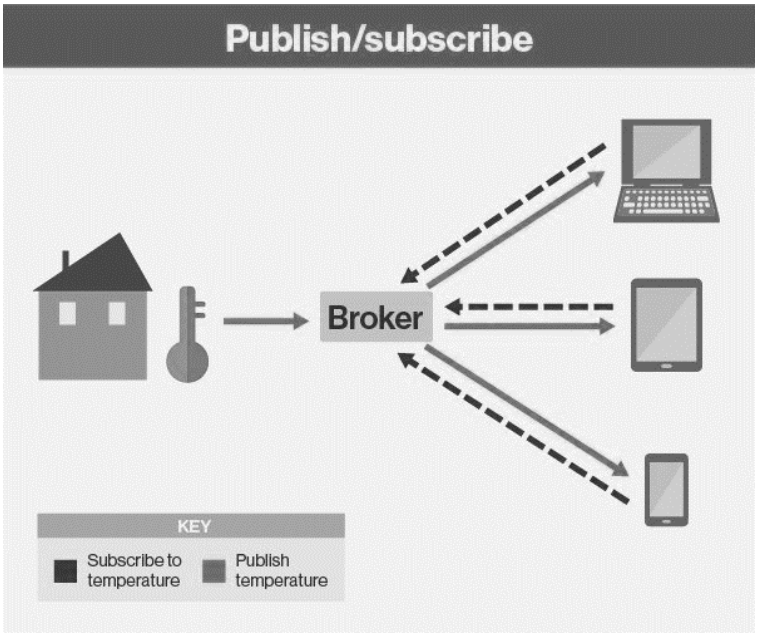

Figure 20: MQTT Publish Subscribe Model

MQTT supports message binary large objects (BLOBs) up to 256 MB in size. The format of the content will be application-specific. Topic subscriptions are made using a SUBSCRIBE/SUBACK packet pair, and unsubscribing is similarly performed using an UNSUBSCRIBE/UNSUBACK packet pair.

Topic strings form a natural topic tree with the use of a special delimiter character, the forward slash (/). A client can subscribe to -- and unsubscribe from -- entire branches in the topic tree with the use of special wild-card characters. There are two wild-card characters: a single-level wild-card character, the plus character (+); and a multilevel wild-card character, the hash character (#). A special topic character, the dollar character ($), excludes a topic from any root wild-card subscriptions. Typically, $ is used to transport server-specific or system messages.

Another operation a client can perform during the communication phase is to ping the broker server using a PINGREQ/PINGRESP packet sequence. This packet sequence roughly translates to ARE YOU ALIVE/YES I AM ALIVE. This operation has no other function than to maintain a live connection and ensure the TCP connection has not been shut down by a gateway or router.

When a publisher or subscriber wants to terminate an MQTT session, it sends a DISCONNECT message to the broker and then closes the connection. This is called a graceful shutdown because it gives the client the ability to easily reconnect by providing its client identity and resuming where it left off.

The disconnection should happen suddenly without time for a publisher to send a DISCONNECT message, the broker may send subscribers a message from the publisher that the broker has previously cached. The message, which is called a last will and testament, provides subscribers with instructions for what to do if the publisher dies unexpectedly.[9]

2.7.4. Quality of service levels

QoS refers to an agreement between the sender of a message and the message's recipient. QoS will define the guarantee of delivery in referring to a specific message. QoS acts as a key feature in MQTT, giving the client the ability to choose between three levels of service.

The three different QoS levels determine how the content is managed by the MQTT protocol. Although higher levels of QoS are more reliable, they have more latency and bandwidth requirements, so subscribing clients can specify the highest QoS level they would like to receive.

The simplest QoS level is unacknowledged service. This QoS level uses a PUBLISH packet sequence; the publisher sends a message to the broker one time, and the broker passes the message to subscribers one time. There is no mechanism in place to make sure the message has been received correctly, and the broker does not save the message. This QoS level may also be referred to as at most once, QoS0 or fire and forget.

The second QoS level is acknowledged service. This QoS level uses a PUBLISH/PUBACK packet sequence between the publisher and its broker, as well as between the broker and subscribers. An acknowledgement packet verifies that content has been received, and a retry mechanism will send the original content again if an acknowledgement is not received in a timely manner. This may result in the subscriber receiving multiple copies of the same message. This QoS level may also be referred to as at least once or QoS1.

The third QoS level is assured service. This QoS level delivers the message with two pairs of packets. The first pair is called PUBLISH/PUBREC, and the second pair is called PUBREL/PUBCOMP. The two pairs ensure that, regardless of the number of retries, the message will only be delivered once. This QoS level may also be referred to as exactly once or QoS2. [9]

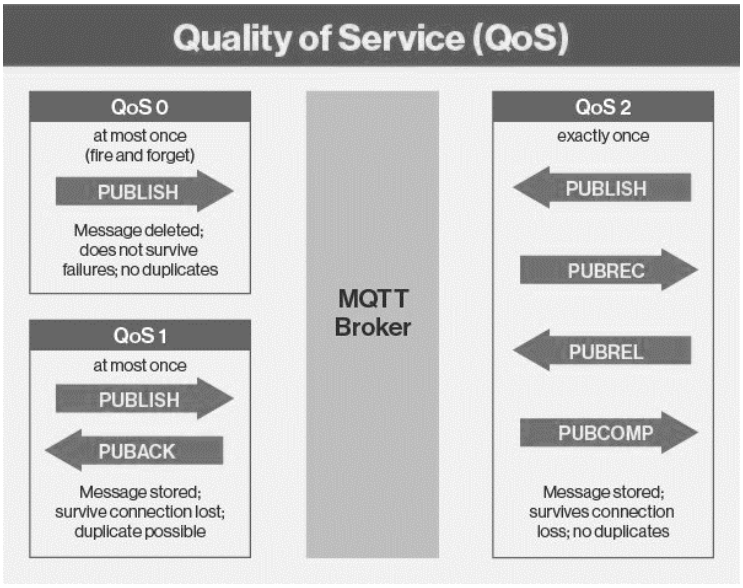

Figure 21: MQTT Quality of Service Levels

2.7.5. MQTT Specifications

MQTT has different specifications depending on the specific version. Version 5.0 superseded the last version of MQTT, version 3.1.1. Some newer specifications, as defined by OASIS, include the following:

- The use of publish/subscribe message patterns.
- A mechanism that can notify users when abnormal disconnections occur.
- The three levels of message delivery: at most once, at least once and exactly once.
- The minimization of transport overhead and protocol exchanges to reduce network traffic.
- An agnostic messaging transport referring to the content of the payload.

2.7.6. MQTT Advantages

MQTT has a few distinct advantages when compared to competing protocols. Advantages include the following:

- Efficient data transmission and quick to implement due to its being a lightweight protocol.
- Low network usage, due to minimized data packets.
- Efficient distribution of data.
- Successful implementation of remote sensing and control.
- Fast and efficient message delivery.
- Usage of small amounts of power, which is good for the connected devices.
- Reduction of network bandwidth.

2.7.7. MQTT in IoT

MQTT is one of the most commonly used protocols concerning IoT. MQTT enables resource-constrained IoT devices to send, or publish, information about a given topic to a server that functions as an MQTT message broker. The broker then pushes the information out to those clients that have previously subscribed to the topic. To a human, a topic looks like a hierarchical file path. Clients can subscribe to a specific level of a topic's hierarchy or use a wild-card character to subscribe to multiple levels.

2.7.8. MQTT protocol applications and use cases

Facebook currently uses MQTT for its Messenger app, not only because the protocol conserves battery power during mobile phone-to-phone messaging, but also because the protocol enables messages to be delivered efficiently in milliseconds (ms), despite inconsistent internet connections across the globe.

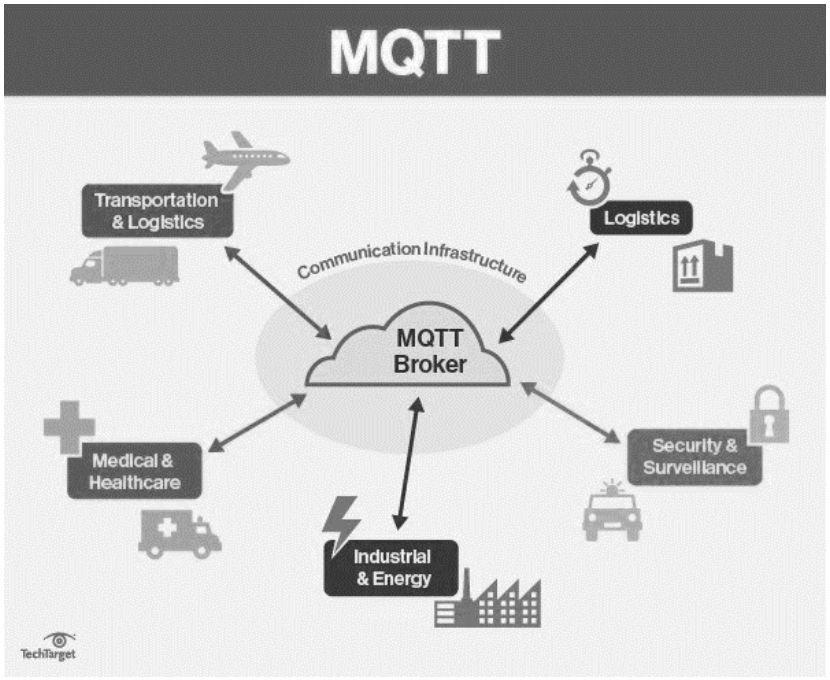

Figure 22: MQTT Protocol Vertical Uses

Most major cloud services providers, including Amazon Web Services (AWS), Google Cloud, IBM Cloud and Microsoft Azure, support MQTT.

MQTT is well suited to applications using M2M and IoT devices for purposes such as real-time analytics, preventative maintenance and monitoring in environments, including smart homes, healthcare, logistics, industry and manufacturing.

2.7.9. MQTT Challenges

Because the MQTT protocol was not designed with security in mind, the protocol has traditionally been used in secure back-end networks for application-specific purposes. MQTT's topic structure can easily form a huge tree, and there's no clear way to divide a tree into smaller logical domains that can be federated. This makes it difficult to create a globally scalable MQTT network because, as the size of the topic tree grows, the complexity increases.

Another negative aspect of MQTT is its lack of interoperability. Because message payloads are binary, with no information as to how they are encoded, problems can arise -- especially in open architectures where different applications from different manufacturers are supposed to work seamlessly with each other.

As touched upon previously, MQTT has minimal authentication features built into the protocol. Usernames and passwords are sent in cleartext, and any form of secure use of MQTT must employ SSL/TLS, which, unfortunately, is not a lightweight protocol.

Authenticating clients with client-side certificates is not a simple process, and there's no way in MQTT to control who owns a topic and who can publish information on it, except using proprietary, out-of-band means. This makes it easy to inject harmful messages into the network, either willfully or by mistake.

Furthermore, there's no way for the message receiver to know who sent the original message unless that information is contained in the actual message. Security features that have to be implemented on top of MQTT in a proprietary fashion increase the code footprint and make implementations more difficult.

2.8. Web Service

2.8.1. Cloud

"The cloud" refers to servers that are accessed over the Internet, and the software and databases that run on those servers. Cloud servers are located in data centers all over the world. By using cloud computing, users and companies don't have to manage physical servers themselves or run software applications on their own machines.

As defined in The NIST Definition of Cloud Computing technical report

“Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.” [10]

2.8.1.1. Essential Characteristics

- On-demand self-service

Provision computing capabilities as needed automatically without requiring human interaction with the service provider.

- Broad network access

Capabilities are available over the network and accessed through standard mechanisms (Web standard).

- Resource pooling

The providers compute resources that are pooled to serve multiple consumers using a multi-tenant model, with different resources dynamically assigned and reassigned according to demand.

Multi-tenant means that a single instance of software and its supporting infrastructure serves multiple customers, each customer shares the software application and shares a single database, each tenant data is isolated and remains invisible to other tenants. [10]

- Rapid elasticity

Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outwards and inwards demand.

- Measured service

Resources usage can be monitored, controlled and reported, providing transparency for both provider and consumer of the utilized service.

2.8.1.2. Service Models

- Software as a Service (SaaS)
- Platform as a Service (PaaS)
- Infrastructure as a Service (IaaS)

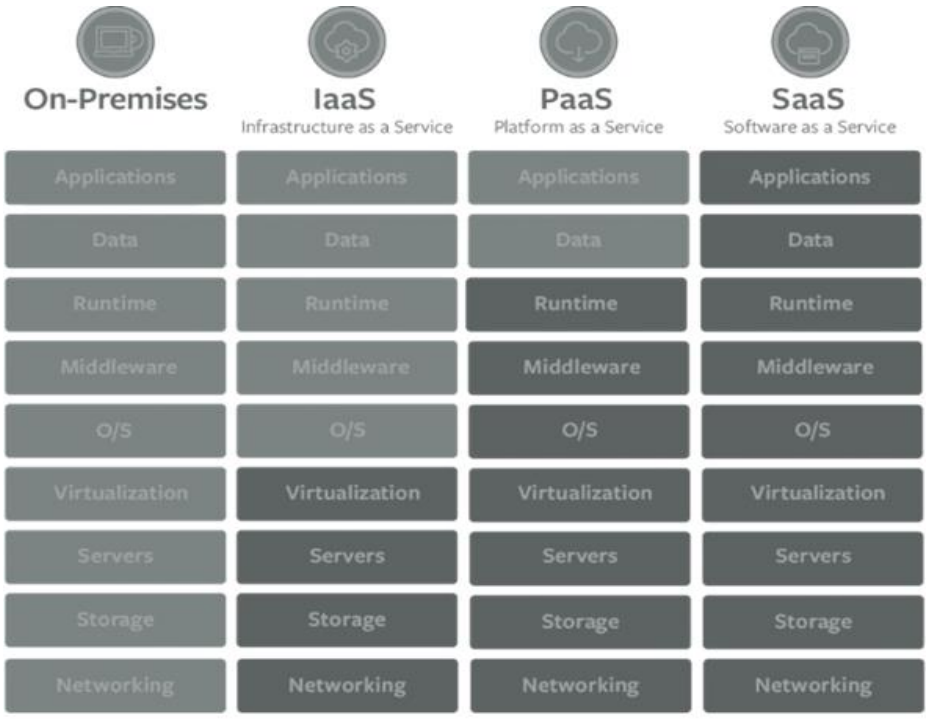

Figure 23: Services Model

2.8.2. Web Service API over HTTP

It is a way of communication between two systems, Provider system and Client system communicating through HTTP protocol and exchange data through different forms, the objective to build a Web service API based on REST architecture is to provide resource data based on the request with the network call by the external clients.

2.8.2.1. Benefits of using Web Service

· Exposing the Existing Function on the network

Web services allow you to expose the functionality of your existing code over the network that can be activated using HTTP requests. Once it requested the other applications could make use of it.

· Interoperability

Web service allows various applications to communicate with each other sharing data for example a Java written application can talk to Python web services and vice versa.

· Standardized Protocol

Web services use standardized industry standard protocol for the communication between different applications

· Low Cost Communication

Low Cost internet can be used to implement Web services as it works over HTTP protocol.

2.8.3. HTTP Protocol

Stands for Hyper Text Transfer Protocol. Is a request and response text based protocol used by the World Wide Web. Which allows the fetching of resources, such as HTML documents. It is the foundation of any data exchange on the Web and it is a client-server protocol, which means the recipient initiates requests. [12]

This protocol defines how messages transmitted and formatted to control web server actions take in response.

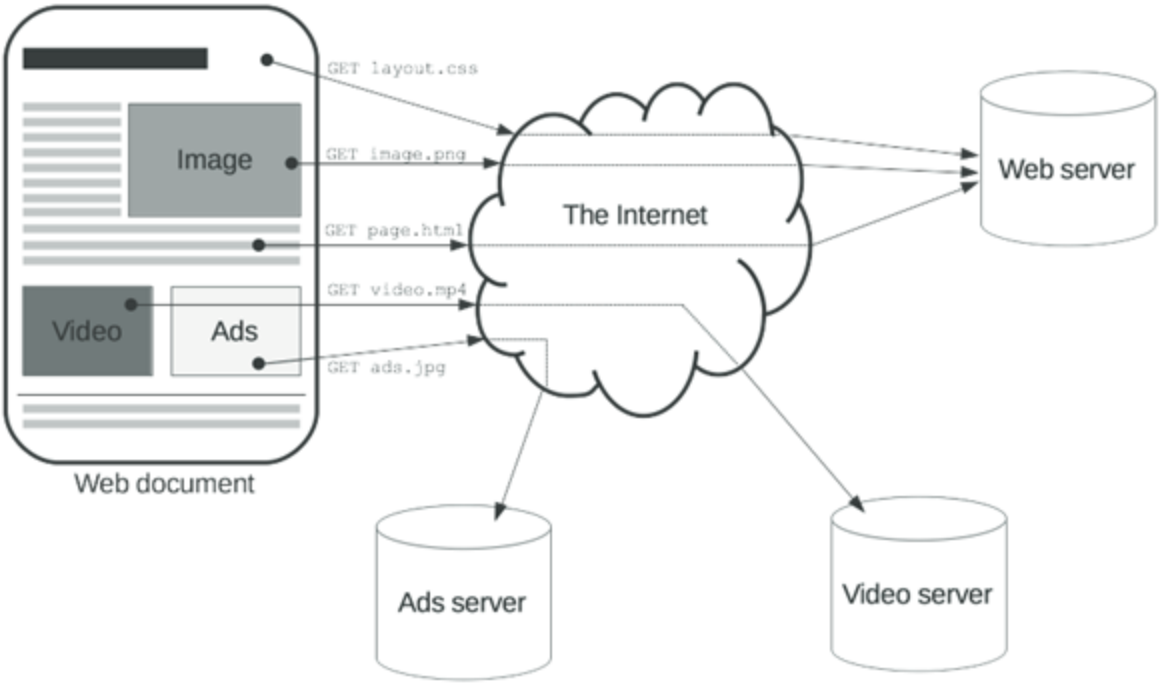

Figure 24: HTTP protocol overview

2.8.3.1. HTTP Flow:

- A client sends an HTTP request
- A web server receives the request
- The server run an application to process the request
- The server returns an HTTP response
- The client receives the response

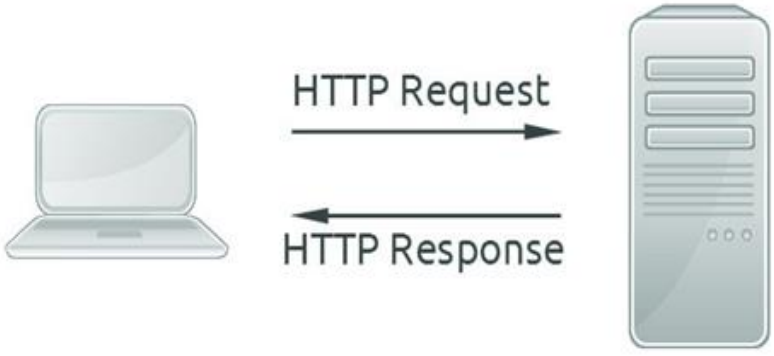

Figure 25: HTTP Protocol Flow

2.8.3.2. Common features controllable with HTTP:

- Caching

How documents are cached can be controlled by HTTP. The server can instruct proxies and clients, about what to cache and for how long. The client can instruct intermediate cache proxies to ignore the stored document.

- Authentication

Some pages may be protected so that only specific users can access them. Basic authentication may be provided by HTTP, either using the WWW-Authenticate and similar headers, or by setting a specific session using HTTP cookies.

- Relaxing the origin constraint

To prevent snooping and other privacy invasions, Web browsers enforce strict separation between Web sites. Only pages from the same origin can access all the information of a Web page. Though such constraint is a burden to the server, HTTP headers can relax this strict separation on the server side, allowing a document to become a patchwork of information sourced from different domains; there could even be security-related reasons to do so.

- Sessions

Using HTTP cookies allows you to link requests with the state of the server. This creates sessions, despite basic HTTP being a state-less protocol. This is useful not only for e-commerce shopping baskets, but also for any site allowing user configuration of the output.

2.8.3.3. HTTP Messages:

- Requests
    - An HTTP method (GET, POST …) that defines the operations the client wants to perform.
    - HTTP protocol version
    - The path of the resources to fetch
    - Optional Header attributes like
        - Content Type: The format of Content
        - Content Length: Size of content
        - Expires: when to consider stale
        - Cookies: Passenger data in the request

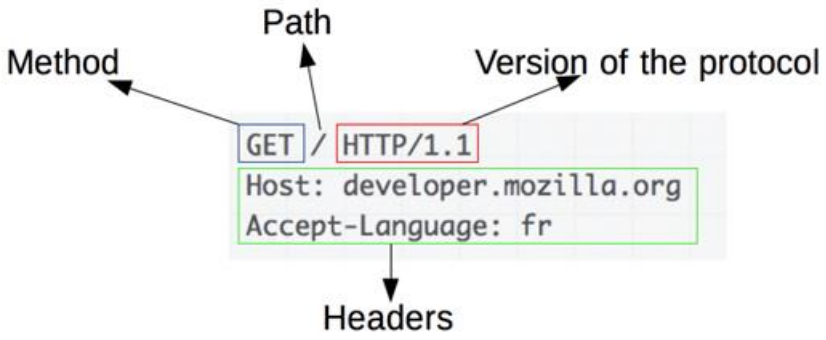

Figure 26: HTTP Request Header

- Responses
  - HTTP protocol version they follow.
  - A status code that indicates if the request was successful, or not and why.
  - A status message, a non-authoritative short description of the status code.
  - Optionally, a body containing the fetched resource.

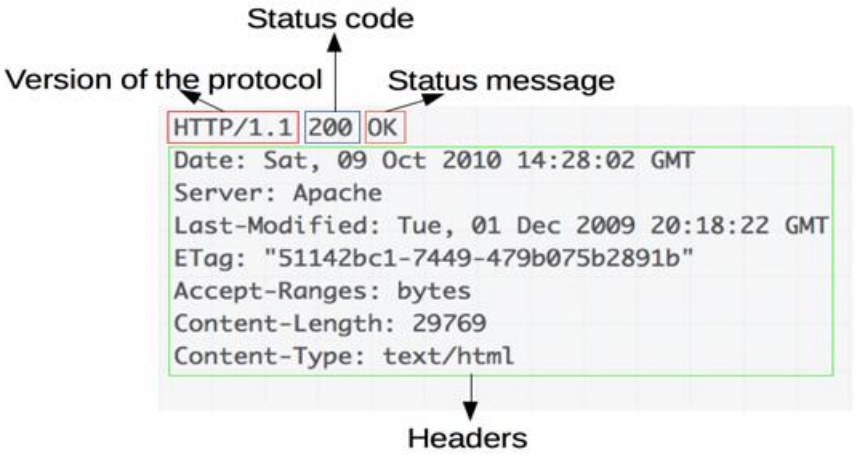

Figure 27: HTTP Response Header

2.8.4. REST Architecture

REST stands for Representational State Transfer is an architectural style that enables communication between systems working over HTTP protocol to clarify communication and resources management. The concept of REST is defined by certain rules, constraints or principles. The system, application, services or whatever satisfies these REST principles are called RESTful.

2.8.4.1. Historical background

Roy Fielding defined REST in his 2000 PhD dissertation "Architectural Styles and the Design of Network-based Software Architectures" at UC Irvine. He developed the REST architectural style in parallel with HTTP 1.1 of 1996–1999. [12]

2.8.4.2. REST constraints

- Separation of client and server

Separation of concerns is the principle behind the client-server constraints. By separating the user interface concerns from the data storage concerns, we improve the portability of the user interface across multiple platforms and improve scalability by simplifying the server components. [12]

- Server requests are stateless

Each request must be treated as if it was the first request the server has ever seen from this client, every request contains all needed data to operate successfully.

- Cache

Response messages from server to client are explicitly labeled as cacheable or non-cacheable. Response can be cached by the client if the information does not changed in the server.

- Uniform Interface

RESTful architecture must have a uniform interface between all clients and servers for example a server must not require a different way of accessing data if the client is smartphone or PC.

- Layered system

Layered system means that a client can have access to an endpoint that relies on another endpoint without having to understand all the underline implementations.

- Code on demand

Code on demand is an optional constraint for REST application, but it opens a possibility for code like JavaScript, for example from the server to be sent to client for execution.

2.9. Embedded Linux Image

Linux was first released into an unsuspecting world in the summer of 1991. Initially the spare-time hobby of a Finnish computer scientist by the name of Linus Torvalds, Linux was at first accessible only in software source code form to those with enough expertise to build and install it. Early enthusiasts (most also developers themselves by necessity) exploited the growth of the Internet in the early 1990s as a means to build online communities and drive development forward. These communities helped to build the first Linux software distributions, containing all the software components needed to install and use a Linux system without requiring users to be technical experts.

2.9.1. Historical background

In the 60s and early 70s, programmers were used to distribute the source code with their programs. Unix, a family of multitasking, multi user computer operating systems, was developed in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others. However, things started to change in the late 70s/early 80s when manufacturers stopped to provide source code. [13]

On September 27, 1983, Richard M. Stallman announced the GNU Project. The GNU Project aims at developing a complete operating system offering the freedoms to run, study, modify and distribute it.

In 1991, in Finland, a student, Linus Torvalds, began a project that later became the Linux kernel. He wrote the program specifically for the hardware he was using 80386 processor. Development was done on MINIX using the GNU C compiler.

As Torvalds wrote in his book 'Just for Fun', he eventually ended up writing an operating system kernel. On 25 August 1991 (age 21), he announced this system in a Usenet posting to the newsgroup "comp.os.minix."

In 1992, he suggested releasing the kernel under the GNU General Public License. In the middle of December 1992, he published version 0.99 using the GNU GPL. [13]

2.9.2. Definitions

2.9.2.1. What Is Linux?

Technically speaking, Linux refers only to an operating system kernel originally written by Linus Torvalds. The Linux kernel provides a variety of core system facilities required for any system based upon Linux to operate correctly.

2.9.2.2. What is Embedded Linux?

Embedded Linux is the usage of the Linux kernel and various open-source components in embedded systems.

2.9.3. Reasons for choosing Linux

Nowadays, Linux based operating systems are used in embedded applications with rich features, offering resources to develop value-adding software features, including networking, graphical and audio features.

2.9.3.1. Code Availability

Code availability relates to the fact that Linux's source code and all build tools are available without any access restrictions. The most important Linux components, including the kernel itself, are distributed under the GNU General Public License (GPL). [14]

2.9.3.2. Quality and reliability of code

- Modularity and structure

Each separate functionality should be found in a separate module, and the file layout of the project should reflect this. Within each module, complex functionality is subdivided in an adequate number of independent functions.

- Readability

The code should be readable and easy to fix.

- Extensibility

Adding new feature should be straightforward. As the code are modular and structured.

- Configurability

It should be possible to select which features from the code should be part of the final application. This selection should be easy to carry out.

2.9.3.3. Re-using components

The key advantage of Linux and open-source in embedded systems is the ability to reuse components. The open-source ecosystem already provides many components for standard

features, from hardware support to network protocols, going through multimedia, graphic, cryptographic libraries, etc. Allowing focus to add value added features to your product.

2.9.3.4. Hardware support

Linux supports many platform and hardware devices, many of hardware vendors offering the supported drivers to Linux systems and offer BSP (Board support packages) as open source components.

2.9.3.5. Low cost

Free software can be duplicated on as many devices as you want, free of charge. If your embedded system uses only free software, you can reduce the cost of software licenses to zero. Even the development tools are free, unless you choose a commercial embedded Linux edition.

2.9.3.6. Licensing

Licensing enables programmers to do with Linux what they could only dream of doing with proprietary software. In essence, you can use, modify, and redistribute the software with only the restriction of providing the same rights to your recipients.

2.9.3.7. Vendor Independent

Vendor independence means that you do not need to rely on any sole vendor to get Linux or to use it. Furthermore, if you are displeased with a vendor, you can switch, because the licenses under which Linux is distributed provide you the same rights as the vendors.

2.9.3.8. Community support

Open-source software components are developed by communities of developers and users. This community can provide high-quality support: you can directly contact the main developers of the component you are using. The likelihood of getting an answer doesn't depend what company you work for.

2.9.4. Building Embedded Linux-based System

A very first solution to build embedded Linux system is to do it yourself from scratch:

- Determine system component including cross toolchain.
- Configure and build the kernel.
- Build the root filesystem.
- Configure and build boot software.

This solution offers a full control in the build process, but needed to solve dependencies between software components, and needed to reduce the product development cycle, due to the Market competition.

Therefore, the community need tools to help in building systems that can solve software dependencies issue without giving up reliability.

2.9.4.1. Embedded Distribution Tools overview

Table 4: Other tools comparison

Tool	Advantages	Disadvantages
From scratch	Full control on the build process.	Need to solve all dependences manually. Limited to simple configuration and resources.
Buildroot	Small, Simple and gives quick results. Adapted for small-embedded devices.	Not perfect if advanced functionalities and multiple machines support needed.
OpenWRT	Based on Buildroot	Used for Embedded network devices like routers.
Scratchbox		Deprecated
Angstrom	Inherit from Open Embedded project. Share the same task engine with Yocto project and same kind of meta data	Last release 2 years ago

After discussing the advantages and disadvantages between different Embedded Distribution tools the tool that will be worked with to build our embedded Linux system is The Yocto Project that will be mentioned in the next section.

2.9.5. The Yocto Project

It is not a Linux distribution, as explained on the Yocto Project website:

"The Yocto Project is an open source collaboration project that provides templates, tools and methods to help you create custom Linux-based systems for embedded products regardless of the hardware architecture."

Around November 2010, the Linux Foundation announced that this entire work would continue under the banner of the Yocto Project as a project sponsored by the Linux Foundation (with Richard Purdie, Fellow of the Linux Foundation, as Architect). It was then established that the Yocto Project and OpenEmbedded would coordinate on a core set of package metadata called OE-Core, combining the best of both Poky and OpenEmbedded with an increased use of layering for additional components. [15]

2.9.5.1. Why Yocto

- Yocto is hosted by the Linux Foundation, and is supported by major actors like silicon vendors (Texas Instruments, Intel, Freescale), major tools vendors (Wind River, MontaVista, Mentor Graphics), embedded consultants etc....
- Build systems automate the process of building a target system, including the kernel, and sometimes the toolchain.
- Develop using one common Linux OS for different architectures.
- Re-use your software stack with future devices.
- Base your work on a validated collection of software and libraries.
- No need to solve all dependencies manually.
- Yocto inherits from the Open Embedded project.

2.9.5.2. Yocto Project Component

The main component in Yocto project are:

- Poky: The reference distribution.
- BitBake: the task executer (Scheduler).
- Open–Embedded Core: Inherited recipes and classes from the Open Embedded Project.
- The BSP Layer: Board support packages.

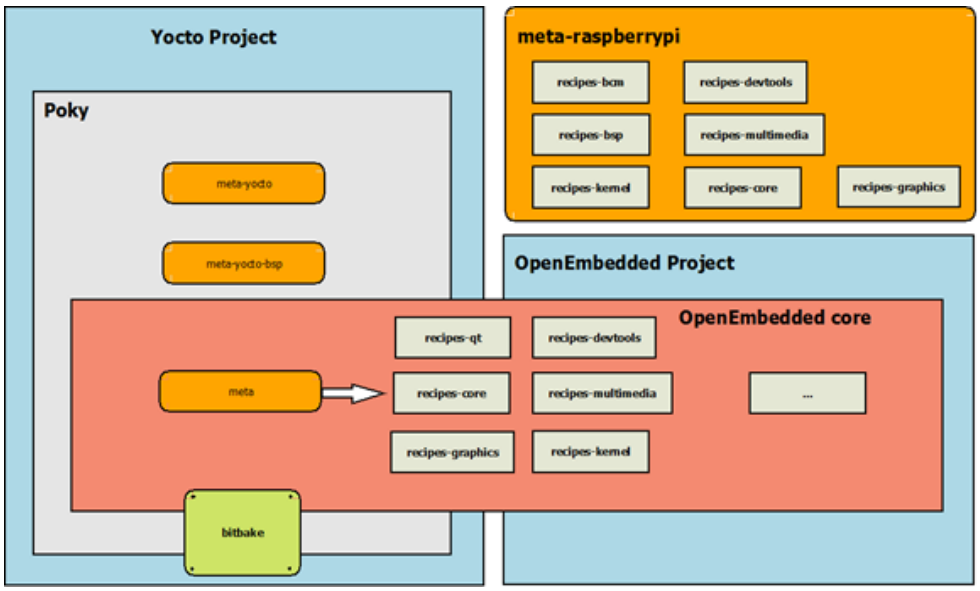

Figure 28: Yocto Project Components

2.9.5.2.1. Poky

Poky is the reference Yocto Project distribution. It contains some of the basic components (called the build system) of OpenEmbedded and a set of metadata for creating embedded distributions for a number of targets. It is platform independent and performs cross compiling using the BitBake tool (a task scheduler), OpenEmbedded-Core, and a default set of metadata, as shown in the following figure. It provides the mechanism to build and combine thousands of distributed open source projects.

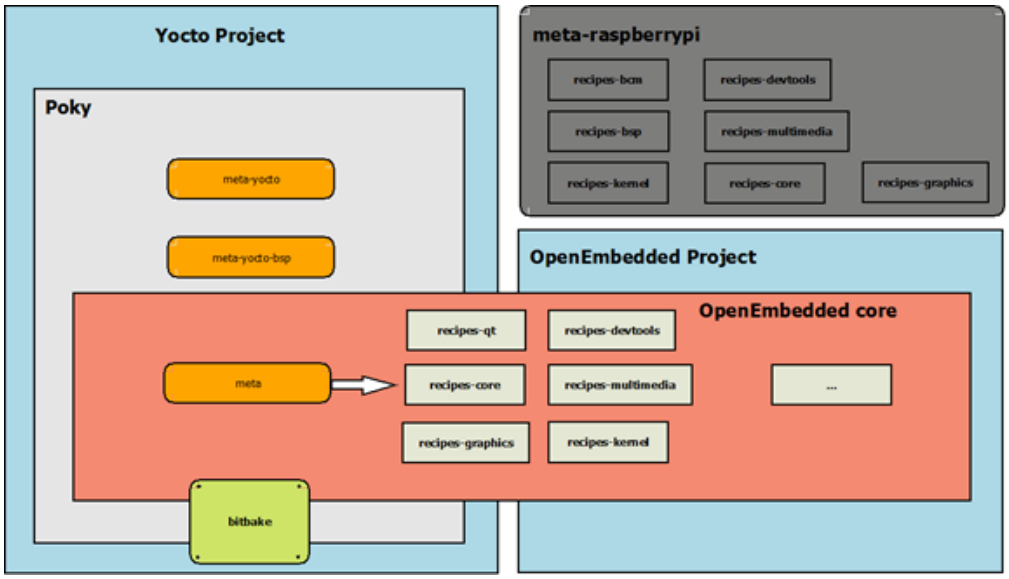

Figure 29: Poky Highlight

2.9.5.2.2. BiBake

BitBake is a task scheduler (like GNU Make) that parses shell and Python scripts. The code parsed generates and runs tasks (configure, compile, and so on), which are sets of steps ordered according to the code's dependencies.

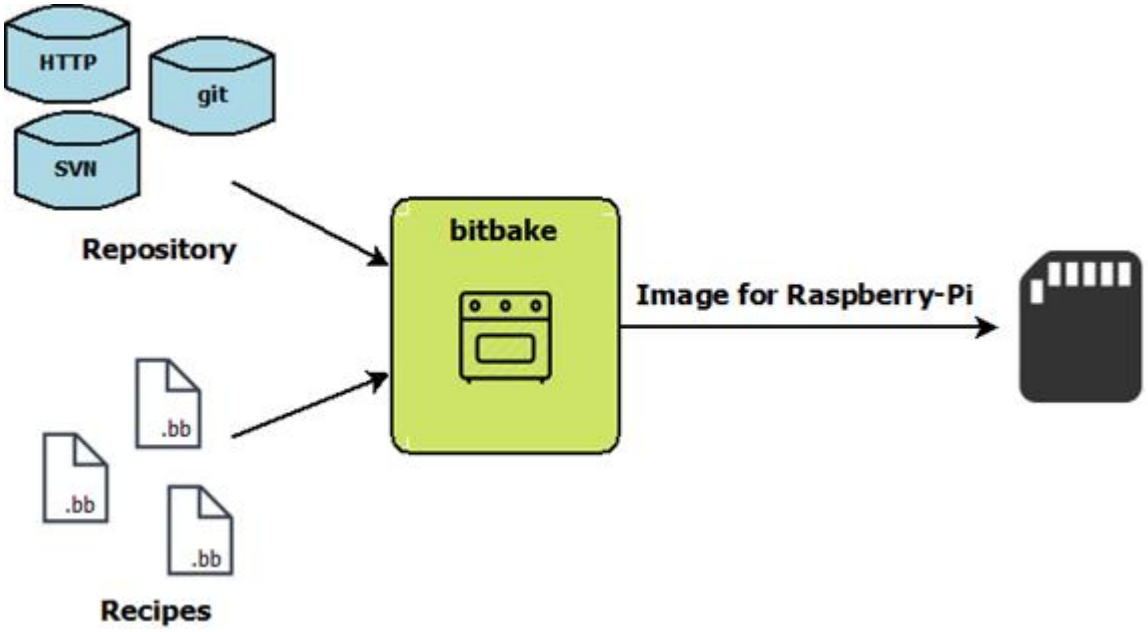

Figure 30: BitBake Diagram

Here are some points taken directly from the BitBake user manual:

· BitBake executes tasks according to the provided metadata, which builds up the tasks. Metadata is stored in recipe (.bb), configuration (.conf), and class (.bbclass) files and provides BitBake with instructions on what tasks to run and the dependencies between those tasks.

· BitBake includes a fetcher library for obtaining source code from various places, such as source control systems or websites.

· The instructions for each unit to be built (such as a piece of software) are known as recipe files and contain all the information about the unit (dependencies, source file locations, checksums, description, and so on).

· BitBake includes a client/server abstraction, can be used from a command line or as a service over XMLRPC, and has several different user interfaces.

2.9.5.2.3. OpenEmbedded-Core

The OpenEmbedded-Core metadata collection (meta in the following diagram) provides the engine of the Poky build tool. It is designed to provide the core features (several recipes). It provides support for six different processor architectures (ARM, x86, x86-64, PowerPC, MIPS, and MIPS64), supporting only QEMU-emulated machines.

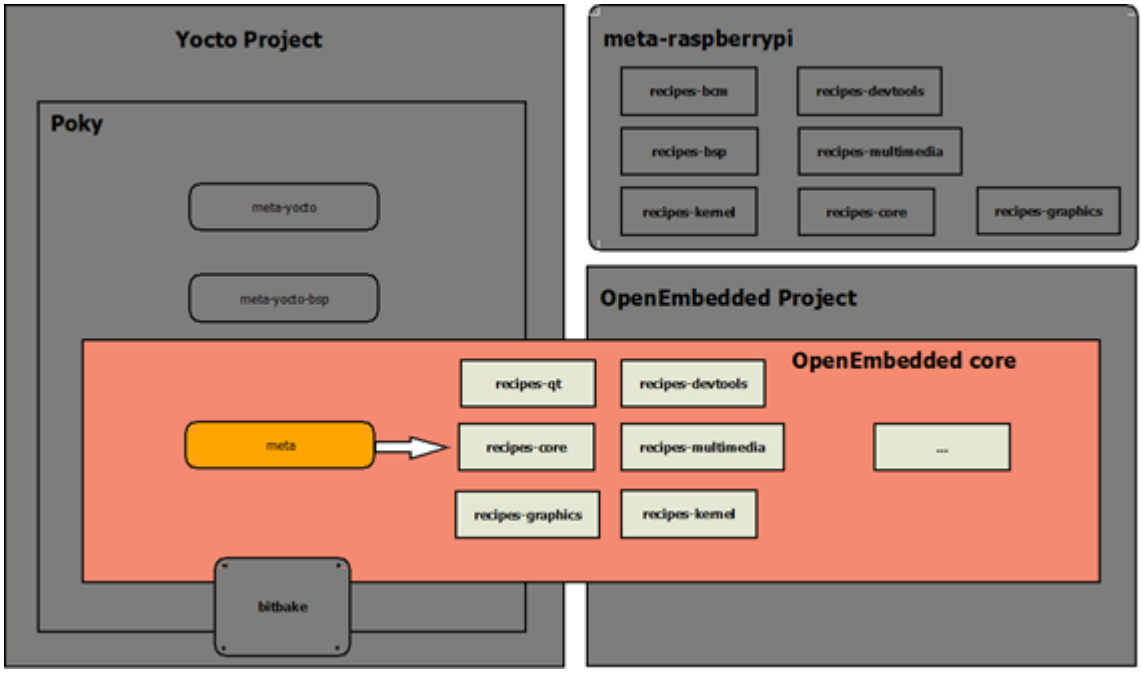

Figure 31: OpenEmbedded-core

2.9.5.2.4. The BSP Layers

A Board Support Package (BSP) is a collection of information that defines how to support a particular hardware device, set of devices, or hardware platform. The BSP includes information about the hardware features present on the device and kernel configuration information along with any additional hardware drivers required. The BSP also lists any additional software components required in addition to a generic Linux software stack for both essential and optional platform features. [15]

One of the BSP layers examples used in our project is meta-raspberrypi, which provided raspberry pi vendors to offer recipes for raspberry pi hardware kit.

2.9.5.3. Yocto project workflow

The following figure represents the high-level diagram for Yocto project development workflow

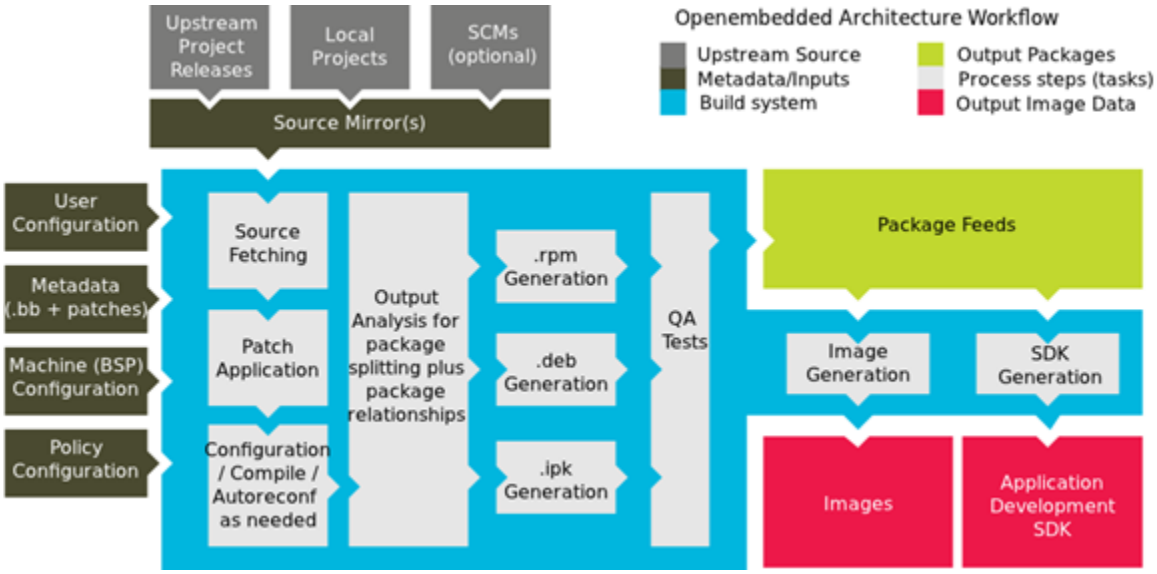

Figure 32: Yocto project workflow

- User Configuration

This metadata can be used to control the build process.

- Metadata layers

These are various layers that provide software, machine, and distribution metadata.

  - Configurations files (.conf)
  - Recipes files (.bb and .bbappend)
  - Classes (.bbclass)
  - Include files(.inc)

- Source files

These contain upstream releases, local projects, and source control management (Git, SVN, and so on). [15]

- Build system

These are processes under the control of BitBake. This block expands on how BitBake fetches source files, applies patches, completes compilation, analyzes output for package generation, creates and tests packages, generates images, and generates cross-development tools.

- Package feeds

These are directories containing output packages (RPM, DEB, or IPK), which are subsequently used in the construction of an image or SDK produced by the build system. These feeds can also be copied and shared using a web server or other means to facilitate extending or updating existing images on devices at runtime if runtime package management is enabled.

- Images

The pieces that compose the operating system, such as the kernel image, bootloader, and rootfs.

Example the final image contains an OS component that is ready to burn in a storage device to boot in specific hardware.

- Application development SDK

These cross-development tools are produced along with an image or separately with BitBake.

An SDK is a set of tools that could be used outside Yocto/OE. These tools generally include a compiler, linker, debugger, libraries, and external headers. This set of compilation tools is called a toolchain.

2.9.5.4. Developing in debian

2.9.5.4.1. Developing cycle

a. Build a base image using debootstrap, create a root filesystem from scratch.

Debootstarp is a tool which will install a Debian base system into a subdirectory of another, already installed system. It doesn't require an installation CD, just access to a Debian repository.

b. Install only the packages you need

Install the packages that fit main application dependencies.

c. Use a cross-compiler to build your own applications and copy to RFS-dir.

Cross compiling the main application to the desired architecture using a crosstool chain.

d. Build FS (ext4, etc) or disk / UBI images using some tools and scripting

2.9.5.4.2. Limitations

a. Only limited number of HW architectures supported

b. SDCard / UBI / etc. image generation, as debian based systems normally not optimized for working with SD cards

c. Image footprint (Large image size compared to Yocto)

2.9.5.5. Conclusion

Table 5: Yocto vs Debian based systems

Tool	Advantages	Disadvantages
Yocto	· Optimized for minimal memory and storage. · Small footprint · reduced attack surface · SDK generation (cross toolchain)	· Short support time (No LTS) · Recipes from different layers might be incompatible · Security tracking/updates need to be done
Debian based systems	· Well managed packages · Security tracking · Is best for prototype and proof of concept	· Large footprint · By default not optimized to work with SD cards · Only limited number of HW architecture support

2.10. Use Cases

Mobility requirements make transportation systems a fundamental and relevant aspect of our lives. Moreover, the appearance of modern Intelligent Transportation Systems based on vehicular networks, which add wireless communication capabilities to the current vehicles, requires the deployment of efficient roadside communication infrastructure.

The number of vehicles on the roads drastically increases every year, where traffic accidents represent a serious drama in our society. Therefore, safety acquires a special attention when designing transportation systems. Governments are increasingly establishing restrictive regulations to improve safety on roads, so that current roads are designed to be safer. Moreover, the automotive industry adds new safety elements inside vehicles, (e.g. airbags, stability control systems, antilock brake systems, etc.). However, the number of accidents still increases every year all over the world, which causes the number of fatalities to be also higher.

2.10.1. Road Works Warning (RWW)

A close look at the accidents shows that many of the deaths occurred during the time between the accident and the arrival of medical assistance. If more than 60 minutes have elapsed by the time the patient arrives at the operating table, the chances of survival fall sharply. So Traffic jams and obstruction due to the accident and obstacles on the road affects the patients greatly and increases the percentage of death. Moreover, due to high speed vehicles, existent car accidents could cause more casualties.

The proposed solution uses a Road Side Unit (RSU) which is a communication node installed within the roads and infrastructure. Where the capacity of vehicles that communicate with an infrastructure depends on the number and radio coverage of existing RSUs in the nearby area. When an accident occurs, the nearby RSU detects the problem and forwards a message with the required information such as its ID to the Admin Panel and the existing vehicles in its coverage zone. Then the Admin Panel forwards the location of the accident to the nearby vehicles with the problem definition. Other Vehicle's drivers are informed with the message through the OBU. The driver then builds a decision either to stop to avoid the accident or to change the current road to decrease the traffic and obstructions which helps the driver to reach his destination easily.

This solution won't stop accidents but will help decrease casualties due to an existent accident, prevent traffic jams and help the fast arrival of medical assistants and safe arrival of passengers and other drivers to their destination.

2.10.2. Electronic Emergency Braking Light (EEBL)

Because of the slow reaction of the human factor to avoid a car in front of it, which leads to a high rate of road accidents, whose statistics indicate that 1.35 million people die on the roads annually. Car accident deaths are the 8th leading cause of death globally. Since a fraction of a second is very important in saving lives, this use case has been chosen.

Emergency brake use case limits the speed of a vehicle, for example, when the speed is above 80 kilometers per hour and suddenly it drops below 10 kilometers per hour, here an emergency braking event has occurred in the car, whether the driver presses the brakes or a car accident occurred, then this car sends a warning message to other cars behind it to inform them of the emergency braking event that occurred and the warning message appears on the drivers screen.

This use case helps the driver avoid an accident by displaying a warning message on his screen to enable him to take a faster reaction which is a major factor helping to reduce the percentage of road accidents.

2.10.3. Vehicle Health Report

In many industries inclusive of the automotive vehicle industry, vehicle maintenance has become more important. It is hard to diagnose failure in advance in the vehicle industry because of the limited availability of sensors and some of the designing exertions. Vehicle systems are complex both in hardware and software so their maintenance is challenging. It is reported by European Commission that there will be a 50% increment in transport vehicles within 20 years. It will require effective strategies to keep up the vehicle performance. Vehicles having very complex structures need an effective maintenance strategy.

Vehicle health is used to track vehicles and driver's status, where drivers could face some issues such as internal car issues, tire damage or fuel shortage. The existence of those obstructions could lead to waste of driver's time and existence of traffic congestion. When an issue occurs with the vehicle, the driver uses the dashboard to send the issue to the admin panel and request for the nearest car maintenance information using his location. The information of the nearest car maintenance to the driver's location appears on the driver's screen.

The admin panel can forward the existence of this issue to the rest of the system to decrease traffic congestion while the driver saves time by calling for an experienced technician. Vehicle health provides best solutions when issues occur in remote areas where it is hard to find car maintenance and it provides the driver with enough information to solve this situation.

CHAPTER 3: MATERIAL AND METHODOLOGY

3.1. System Architecture

3.1.1. Overview

The system consists of four main parts, these parts are On-Board unit (OBU), Road Side Unit (RAU), cloud and control panel,

Each part performs a specific mission, but in a continuous and integrated manner with the rest of the parts, and we have three types of communications, V2V communication, V2I communication, and V2N communications.

The role and primary mission of each part will be clarified below, and how it helps in the effective functioning of the system.

3.1.2. OBU

Everything starts from this unit, this unit can be considered as the cell that makes up the body of the system, and without its quality and efficiency work, the system will be weak and useless, as it is distinguished by its multiple primary and subsidiary tasks because it is the first point of contact between the system and the data, The main task of this unit is to collect the necessary data from the car and help the driver to communicate with the system easily, and to make the necessary alerts and directions required during the driving trip, as it can be used in the future to pay violations immediately according to the system set by the country that uses this system.

It consists of Raspberry pi which connected to GPS to get the location and connected to the USB modem to transfer data to the cloud and connected to a touch screen to display the system and the services it provides easy to the user and connected to OBD to transfer the car data such as the fuel and speed condition.

3.1.3. RSU

These units are the eyes and ears of the system, each OBU is specific to the car only and may have malfunction while driving or was intentionally disabled, but this unit is controlled by the system itself, then the data will be collected accurately, which makes the system operate in a secured manner.

The main task of this unit is to launch steady warnings on the road and important in a situation such as a road work use case and works to collect data from cars or direct and alert and continuously alert.

It consists of Raspberry pi which is connected to GPS to get the location and connected to the USB modem to transfer data to the cloud.

3.1.4. Cloud

One of the most important parts of the system, because it is considered the direct contact point between the collected data and the control panel, which monitors and organizes the communication, as it is the main means of transferring data between cars and the RSU.

In our project, there are two interfaces to reach cloud:

1- MQTT
2- RESTful web services API over HTTP protocol

Will be discussed in the next parts.

3.1.5. Admin panel:

Admin panel's (control room) purpose is production control, and serves as a central space where a large physical facility or physically dispersed service can be monitored and controlled. Control rooms for vital facilities are typically tightly secured and inaccessible to the general public. Multiple electronic displays and control panels are usually present, and there may also be a large wall-sized display area visible from all locations within the space. Some control rooms are themselves under continuous video surveillance and recording, for security and personnel accountability purposes.

It is considered the upper hand in judging the system and the smooth running of things without complications or problems.

3.2. Hardware

3.2.1. Raspberry pi 3B

The Raspberry Pi is a single-board, low-cost, high-performance computer first developed in the UK by the Raspberry Pi Foundation. Not only has it helped bring the joy of electronics and computer programming to people around the world, but it has also become a staple of the maker community.

Raspberry Pi 3B is used as the main component of On-Board Unit (OBU) and Roadside Unit (RSU) due to its high performance and affordable cost as it is compatible with both software tools and other hardware components.

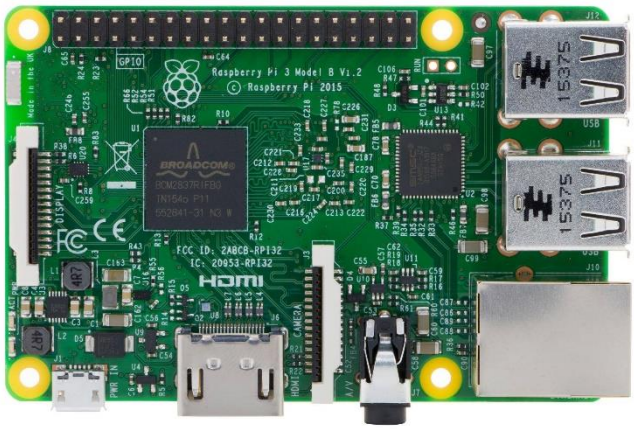

Figure 33: Raspberry Pi

3.2.2. GPS

GPS u-blox neo-6m is an essential component in both OBU and RSU to retrieve the locations of the vehicles and RSUs.

Figure 34: GPS u-blox neo-6m

3.2.3. USB Modem

It is a major interface with the Raspberry pi in both OBU and RSU to allow the internet access over LTE

Figure 35: USB Modem

3.2.4. LCD Display (Touch Screen)

It has an important function as it shows the messages and warnings to the driver; it allows the driver to interact with the whole system as well.

There is no specific model for the touchscreen as long as it can interface with the Raspberry pi,

Figure 36: LCD Display

3.2.5. OBD II

On-board diagnostics (OBD) is an automotive term referring to a vehicle's self-diagnostic and reporting capability. OBD systems give the vehicle owner or repair technician access to the status of the various vehicle sub-systems. The amount of diagnostic information available via OBD has varied widely since its introduction in the early 1980s versions of on-board vehicle computers. Early versions of OBD would simply illuminate a malfunction indicator light or "idiot light" if a problem was detected but would not provide any information as to the nature of the problem. Modern OBD implementations use a standardized digital communications port to provide real-time data in addition to a standardized series of diagnostic trouble codes, or DTCs, which allow a person to rapidly identify and remedy malfunctions within the vehicle.

As it was mentioned, an OBD II adaptor is very important to access the CAN bus of the vehicle to retrieve all the needed data such as speed, RPMs and fuel level.

The used OBD II adaptor is ELM 327 as it can be easily connected to the Raspberry pi and its price as affordable as well.

Figure 37: On-board diagnostics

3.3. Software tools

3.3.1 Node-RED

Node-RED is a programming tool for wiring together hardware devices, APIs and online services. Primarily, it is a visual tool designed for the Internet of Things, but it can also be used for other applications to very quickly assemble flows of various services. Node-RED is open source and was originally created by the IBM Emerging Technology organization. It is included in IBM's Bluemix (a Platform-as-a-Service or PaaS) IoT starter application package.

Node-RED provides a browser-based flow editor that makes it easy to wire together flows using the wide range of nodes in the palette. Flows can be then deployed to the runtime in a single-click. JavaScript functions can be created within the editor using a rich text editor. A built-in library allows you to save useful functions, templates or flows for re-use.

The light-weight runtime is built on Node.js, taking full advantage of its event-driven, non-blocking model. This makes it ideal to run at the edge of the network on low-cost hardware such as the Raspberry Pi as well as in the cloud. The flows created in Node-RED are stored using JSON which can be easily imported and exported for sharing with others. At present, Node-RED is a JS Foundation project. Node-RED enables users to stitch together Web services and hardware by replacing common low-level coding tasks (like a simple service talking to a serial port), and this can be done with a visual drag-drop interface. Various components in Node-RED are connected together to create a flow. Most of the code needed is created automatically.

Node-RED is a powerful tool for building IoT applications and services. Its genesis was triggered exactly by this need to rapidly prototype IoT applications and Node-RED was created as an open source project by the IBM Emerging Technologies group, in particular by two researchers, Nick O'Leary and Dave Conway-Jones. They created Node-RED initially as a tool for themselves as they were working on IoT projects and were "looking for a way to simplify the process of hooking together systems and sensors when building proof-of-concept technologies for customers."

Node-RED has been more open and popular as open source software for both edge and cloud environments. An initial version of Node-RED was released as an open source project in late 2013 and built up a small but active user and developer group during 2014:

- March 2014: Released in QCon London.
- June 2014: Included in the catalog on IBM Cloud.
- November 2015: Pre-installed in Raspberry Pi.
- October 2016: Moved to Linux Foundation.
- March 2018: Downloaded 50,000 times a month.

When the IBM folks created Node-RED, they were mostly focused on the Internet of Things, i.e. connecting devices to processing and processing to devices. As a tool for rapid application development for the IoT, Node-RED is both powerful and flexible.

The three most important reasons for the importance of Node-RED:

- Rapid development for IoT applications.
- Standard technologies in industrial IoT.
- Open community.

The major features of Node-RED are:

- It supports browser-based flow editing.
- As it is built on Node.js, it supports a lightweight runtime environment along with the event driven and non-blocking model.
- You can run it locally (Docker support, etc.).
- It can run in the cloud environment like Bluemix, AWS, MS-Azure, etc.
- The various flows created in Node-RED are stored using JSON, which can be easily imported and exported for sharing with others.
- It can easily fit on most widely used devices like Raspberry Pi, BeagleBone Black, Arduino, Android based devices, etc.

3.3.2 GNU Radio

GNU Radio is a free software development toolkit that provides signal processing blocks to implement software-defined radios and signal-processing systems. It can be used with external RF hardware to create software-defined radios, or without hardware in a simulation-like environment. It is widely used in hobbyist, academic, and commercial environments to support both wireless communications research and real-world radio systems.

The GNU Radio software provides the framework and tools to build and run software radio or just general signal-processing applications. The GNU Radio applications themselves are generally known as "flow graphs", which are a series of signal processing blocks connected together, thus describing a data flow.

As with all software-defined radio systems, configurability is a key feature. Instead of using different radios designed for specific but disparate purposes, a single, general-purpose, radio can be used as the radio front-end, and the signal-processing software (here, GNU Radio), handles the processing specific to the radio application.

These flow graphs can be written in either C++ or the Python programming language. The GNU Radio infrastructure is written entirely in C++, and many of the user tools are written in Python.

A major advantage of GNU Radio, especially when compared to other SDR frameworks, is that it is hardware agnostic, i.e., it is not developed to be used with a specific radio front end. For historic reasons, the devices from Ettus Research are well supported, as the company was working closely with the GNU Radio community.

Also GNU Radio itself is not tuned towards a particular application. Instead, it provides a solid base that can be used to realize any technology. GNU Radio, therefore, does not come with extensive implementations for wireless standards.

In fact, an exemplary implementation for digital TV is the only technology specific code. Being a general-purpose framework, GNU Radio already served as the base for research in many areas, including satellite communications, cognitive radio, cooperative diversity, multi-antenna systems, localization, and radio astronomy. Application-specific functionality for these use-cases is implemented in so-called Out-Of-Tree (OOT) modules, which extend GNU Radio with custom blocks. Our IEEE 802.11p transceiver is one example of such an OOT module.

The central component of GNU Radio is a block. Typically, a block implements a specific signal processing task like a filter, an FFT, a modulator, or a synchronization algorithm.

3.3.3 U-Center

U-Center is U-Blox's powerful GNSS evaluation and visualization tool which can be downloaded free-of-charge from our website (www.U-Blox.com). This user guide provides a description of the features of this software. It allows end users to assess and test U-Blox GNSS positioning chips and modules for navigation and positioning performance.

The purpose of U-Center is to enable users to:

• Conduct performance tests on U-Blox and other GNSS devices.

• Configure U-Blox GNSS positioning chips and modules.

• Update the firmware on GNSS modules.

• Test the added performance provided by U-Blox's Assist Now service.

U-Center evaluation software provides system integrators and end users with a quick and simple way to interface with U-Blox GNSS chipsets, modules and boards. It enables easy evaluation, performance testing, development and debugging of GNSS positioning chips and modules. U-Center allows easy connection to U-Blox products and provides a suite of features to view, log, and analyze performance.

The major features include:

• Support for U-Blox's receivers using U-Blox positioning technology. U-Center can communicate with these receivers using either the UBX protocol or the NMEA-0183 standard protocol.

• Support for receivers that utilize standard NMEA strings.

• U-Center presents all the information collected during the operation of the GNSS device. All aspects of GNSS data (position, velocity, time, satellite tracking, etc.) can be monitored and logged under various test scenarios for the evaluation of a receiver. U-Center software allows analysis of the collected data in order to investigate performance issues such as accuracy, road test position and trajectory, satellite tracking, time to first fix, etc. All processed data can be captured in ASCII format and ported into popular spreadsheets for creating additional plots and statistics.

• Camera View: photographic data can be stored in the log file together with the navigation data and later be replayed in the application.

• Export data files to Google Earth and Google Maps.

• Supports (Multiple GNSS) Assist Now Online and Assist Now Offline.

• Data recording and playback function.

• Structural and graphical data visualization in real-time.

• Export functionality to standard PC applications.

• Docking views (real-time cockpit instruments): Satellite constellation, compass, clock, altimeter, speedometer, GNSS and satellite information views.

• Download firmware updates to GNSS positioning modules.

• Support for NTRIP server and NTRIP client functionality.

• Google Earth server support.

• SQLITE database support

3.4. Vehicle-To-Vehicle Communication (V2V) communication

In recent years, as an instance of vehicular ad-hoc network, the telematics integrating the utility of telecommunications and informatics has dramatically promoted the development of wireless communications in vehicular environment, intelligent transportation system, and automotive electronics industry. The further extension of WAVE (Wireless Access in Vehicular Environments) technology is dedicated short range communication (DSRC). These networks are characterized by rapidly changing topologies and shorter connection lifetime, and one of the most essential goals of the emerging DSRC based V2V communication standards is to enable road safety applications that can save thousands of lives. Safety-related applications are primarily geared toward avoiding the risk of car accidents such as cooperative collision warning,

lane change, emergency vehicle approaching warning, pre-crash sensing, traffic violation warning.

-Progress towards ubiquitous V2V systems will need to be conducted in stages, starting with low-risk, simple implementations, and learning from these to plan and design wider systemic deployments.

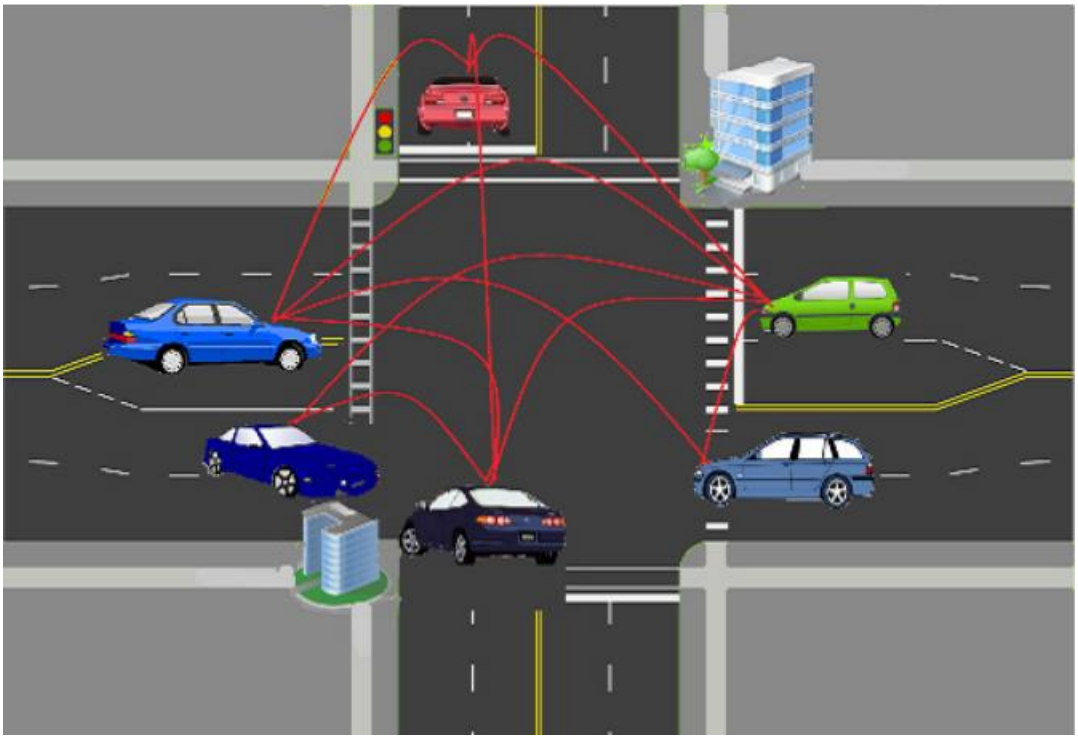

Figure 38: Vehicles communicating each other with

3.4.1. V2V implementation with GNU Radio and USRP

3.4.1.1. Transceiver Overview

Inter-Vehicle Communication (IVC) is the basis for Vehicular Ad Hoc Networks (VANETs), networks of cars communicating directly with each other and possibly with infrastructure nodes. Once successfully deployed, IVC is the basis for a multitude of applications ranging from safety, e.g., intersection collision warning systems, over efficiency, e.g., traffic information systems, to entertainment applications. Besides the use of cellular networks such as UMTS or LTE for IVC, Dedicated Short Range Communications (DSRC) based on IEEE 802.11p are considered a key technology.

Two important steps towards the realization of these networks were made: First, in 1999, the regulatory bodies ECC and FCC reserved five and seven 10 MHz channels in the 5 GHz band exclusively for VANETs in Europe and the U.S., respectively. Secondly, the standardization of the IEEE 802.11p PHY and MAC layer has been completed. Higher layer protocols have been standardized in the context of ETSI ITS G5, including Decentralized Congestion Control (DCC), and the Wireless Access in Vehicular Environment (WAVE) communication stack, including multi-channel operation.

IEEE 802.11p combines the quality of service extensions defined in IEEE 802.11e and the OFDM PHY of IEEE 802.11a. To account for the high mobility and the challenging environment in VANETs, all timings of the IEEE 802.11a were doubled, resulting in a channel bandwidth of 10 MHz Sticking very close to the successful IEEE 802.11a standard has the obvious advantage that the same chips can be used for applications in VANETs and, thus, the costs can be, at least in theory, reduced considerably.

Now experimental validation is needed and Fifield tests are currently ongoing in Europe and the U.S. The current situation is that, on the one hand, DSRC radios for practical experiments are either commercial grade solutions developed for Fifield tests, which are rather expensive and, most importantly, do not allow to change parts of the PHY or MAC of the underlying IEEE 802.11p implementation. Examples are the Cohda Wireless MK2 or the Denso WSU. On the other side of the spectrum, adapted stacks for the popular Atheros chipset AR5414A-B2B (e.g., used in the Unex DCMA-86P) have been developed in several projects. This allows to use very cheap hardware solutions, but they are bound to the capabilities of the chipset, and not fully validated in Fifield tests. Finally, one conclusion of the last Dagstuhl Seminar on IVC was that the reproducibility of measurements with different hardware platforms is very limited. All present researchers agreed that a fully Open Source stack would be of huge benefits.

We close this gap by providing an Open Source simulation and experimentation framework for IEEE 802.11p for SDR systems. We implemented the system for GNU Radio, a real-time signal processing framework for SDR. In our experiments, we relied on the widely used USRP b200 from Etuis Research. The core of our framework is a modular OFDM transceiver, which extends our previous work on an OFDM receiver [10], by now supporting all four modulation schemes BPSK, QPSK, QAM-16, and even QAM-64. The system is implemented fully in software and runs on a typical laptop computer. Given that the design goals for IEEE 802.11a were set to support low mobility indoor environments, it is unclear if and to what extent IEEE 802.11p allows robust and reliable communication in vehicular networks. As the properties of the PHY and MAC implementation are easy to change in an SDR solution, our system provides all necessary features to experiment with protocol changes in a flexible lab environment. In particular, our implementation supports to connect the transmitter and receiver via a simulated channel within GNU Radio to study the system behavior before performing real measurements.

In addition, our SDR based IEEE 802.11p transceiver can be regarded as a first step towards SDR based systems to be deployed in real cars. We believe this will be a necessary step as the product cycle of typical cars is very long and installed systems for IVC need to be able to be changed by means of software updates only.

With an SDR solution, even PHY and MAC protocol updates become feasible.

Our main contributions can be summarized as follows:

- The first SDR based transceiver for IEEE 802.11p has been presented, which has been implemented fully in software in GNU Radio. Performance measurements confirm that our system runs even on simple laptop computers without problems.

- Our system can be used for simulation of new PHY or MAC versions by connecting transmitter and receiver within GNU Radio. We show that the simulations match very well to a widely accepted error model for IEEE 802.11p networks.

- The SDR transceiver is validated by comparing it with a commercial grade Unex DCMA-86P2 radio. Our measurement results indicate that the SDR solution performs exactly as expected.

3.4.1.2. RELATED WORK

Experimental research in VANETs can be done on many different layers depending on which aspect of the transceiver system is to be studied. On the highest layers one can utilize hardware prototypes like the Cohda Wireless MK2 or the NEC Link bird IEEE 802.11p system that was used in the CVIS project. These solutions implement the complete communication stack and, thus, lend themselves well to study actual applications like safety and efficiency systems, or information dissemination strategies like adaptive beaconing. The MK2, for example,

implements the whole IEEE 802.11p stack, including the IEEE 802.11e based Quality of Service (QoS) extensions and provides an SDK that allows implementation of WAVE based applications.

-**The drawback** of these solutions is that the PHY and (at least parts of) the MAC layer are implemented in hardware and thus fixed, i.e., it is not possible to investigate alternative PHY and MAC algorithms.

- On the MAC layer, the impact of different channel access schemes, i.e., the Distributed Coordination Function (DCF) can be studied. This topic received great attention in the WiFi research community and, since IEEE 802.11p is very similar to IEEE 802.11a, the concepts can also be used in VANETs.

-To experiment with MAC layer protocols, the community came up with several approaches.

-The spectrum ranges from systems that provide access to the timings, medium access, and acknowledgement functionality, up to completely replacing the MAC with a fully programmable state machine.

-The physical layer represents the bottom of the communication stack, and is typically the most difficult to study with commercial devices. Experimentation and testing of new ideas and solutions is not possible, since algorithms are implemented in hardware and thus fixed.

-SDR systems provide the possibility to investigate the physical layer, since they replace the transceiver chip with general purpose hardware and therefore can encode, modulate, and transmit arbitrary electromagnetic signals.

-In terms of hardware, there are basically two architectures for SDR systems that differ in where the signal processing is implemented.

-First, everything can be implemented on a Field-Programmable Gate Array (FPGA).

-Second, one can implement the signal processing on General

Purpose Processors (GPPs) of a normal PC. This approach is used by Microsoft's Sora platform as well as the

-Universal Software Radio Peripheral (USRP) of Ettus Research, which is typically used with the GNU Radio signal processing framework.

3.4.1.3. TRANSCEIVER STRUCTUR

To enable experimental wireless research in VANETs, we implemented an SDR-based IEEE 802.11p transceiver.

- An SDR system consists of a software part, where the signal processing algorithms are implemented, and a hardware part that is responsible for up and down conversion of the analog wave form. On the software side, we implemented the transceiver based on GNU Radio, a GPP based real-time signal processing framework.

- On the hardware side, we used the Ettus Research USRP b200 on the Intelligent Transportation Systems (ITS) frequency band around 5.9 GHz. The FPGA image of the b200 takes care of down sampling, filtering, and removing of the DC offset.

-The transceiver software consists basically of send and receive chains and code that switches between the two states.

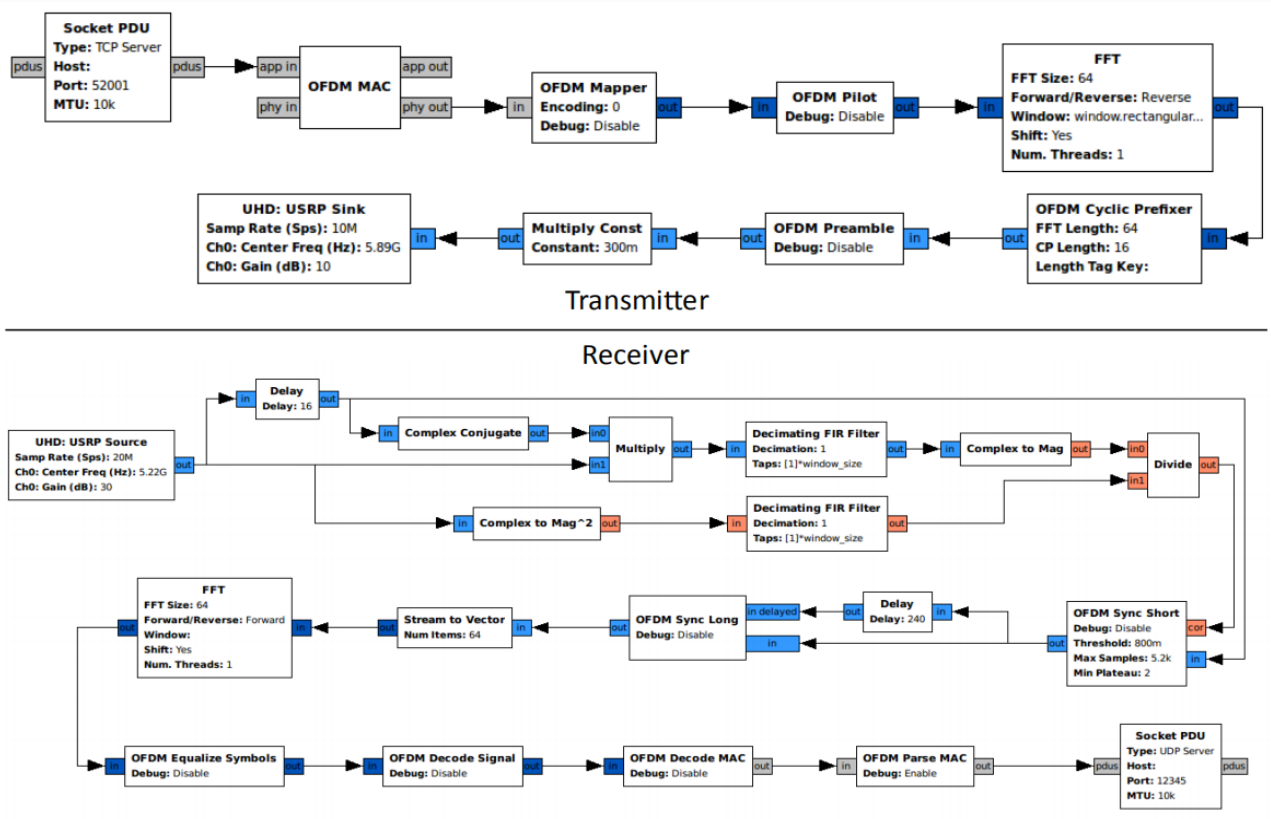

Figure 39: Transceiver GNU Radio Block Diagram

3.4.1.3.1. TRANSCEIVER (Transmitter part)

An IEEE 802.11p transmitter for the N210 and GNU Radio has already been presented by Fuxjager. Since the transmitter was implemented partly in Python and based on an incompatible version of GNU Radio that lacked many of the recently introduced features that enable seamless packet based operation, we implemented it from scratch. Regarding its use in a transceiver, our transmitter implementation also has the important advantage that it supports variable packet sizes and allows to specify the encoding on a per packet basis.

In contrast to the receive chain, there are nearly no design decisions to make on transmitter side since modulation and encoding are fully specified in the standard.

-The only parameter that we chose is the transition width of the window function, which we set to 1. This window mainly asserts that the output signal honors the spectral mask defined in the standard and, thus, the signal decays fast in the frequency domain, limiting adjacent channel interference.

An overview of the transmitter structure as exposed to the GNU Radio Companion is depicted in the top half of Figure 34. The Companion provides a graphical user interface to setup and configure signal processing flow graphs. The implementation of the transmitter is straightforward and includes mainly encoding, Fast Fourier Transformation and addition of the cyclic prefix.

-A major concern regarding the transmitter might be the carrier sensing mechanism, which is fundamental for both unicast and broadcast transmissions. IEEE 802.11p uses the extended DCF defined in the IEEE 802.11e amendment, which adds support for QoS. Due to latencies incurred by the communication between PC and the USRP it is not possible.

-to implement the carrier sensing logic in software, since this introduces a large blind spot between the time the medium is sensed and when it is finally accessed. This issue can, however, be circumvented by implementing the CSMA/CA on the FPGA, which is possible since the SDR has an on-board flash where it can store a frame and handle the channel access in hardware. A basic CSMA implementation for the N210, b200 with a non-standard PHY that demonstrates this possibility has been presented in.

3.4.1.3.1.2. TRANSCEIVER (Receiver Part)

On the receiver side we build on our implementation presented in. In contrast to the transmitter, the performance of the receiver is crucial since it has to catch up with the incoming sample stream. The required computational speed is achieved with the help of Single Instruction Multiple Data (SIMD) instructions as provided by GNU Radio's Factorized Library of Kernels (VOLK). Instead of iterative calculation, SIMD instructions act on vectors of data and thus, provide a significant speed up.

-Our receiver has a modular and easy to use structure as depicted in the bottom half of Figure 34. At first, the 2013 IEEE Vehicular Networking Conference autocorrelation coefficient is calculated in order to detect the cyclic pattern of the short preamble of OFDM frames, which is used for frame detection by the *OFDM Sync Short* block. The following blocks are responsible for frame alignment, frequency offset correction, channel estimation and actually decoding the payload. On the receiver side the modularity is important since we expect a main application of the framework to be the investigation and comparison of different receive algorithms, for instance, different channel estimation strategies.

-With a modular concept it is easy to exchange algorithms and study the impact on the performance. For the evaluations presented in this paper, we also implemented support for QAM-16 and QAM-64 encodings. This means that meanwhile all modulation and coding schemes defined in the standard are supported.

-As with the transmitter, the receiver experiences communications latency between PC and USRP. Unfortunately, this makes it impossible to comply with the tough timing constraints of RTS/CTS and acknowledgement frames that are used for unicast transmissions. If unicast transmissions have to be investigated, it is possible to work around this limitation by disabling retries due to missing acknowledgements and by setting the RTS/CTS threshold to infinity.

- However, such investigations would most likely be better served with full FPGA based solutions like WARP. Nevertheless, a vast array of VANET applications do not use acknowledged unicast transmission. Rather, the broadcast characteristic of the medium is exploited to effectively disseminate information. These broadcast transmissions are fully supported by our framework.

3.4.1.4. SIMULATION

Since we now have a complete transceiver system in GNU Radio, we can use it to study the performance of the employed algorithms by means of simulation. A simulative performance evaluation is desirable since it allows to reproduce the experiments and also to compare the systems with independent research results. To show the correctness of the implementation and that we achieve reasonable performance we conducted simulations over an Additive White Gaussian Noise (AWGN) channel. Again, we set the packet size to 133 B and send 10000 packets each, for different encodings and SNRs.

-The simulative determined packet delivery ratios are shown in Figure 2. The error bars depict the confidence intervals with a confidence level of 0.95. We can see that the curves corresponding to the different encodings are reasonable in the sense that higher order modulations require a higher SNR.

-More interesting is, however, the comparison with independent measurements and simulations. Implement a similar transceiver system in the discrete event simulator ns3. This implementation was used for detailed simulations of physical layer effects in WiFi networks. This transceiver was not real-time capable and not intended for use in an SDR environment. However, to show the correctness of the implementation and prove that their implementation produces meaningful results, the authors also present packet error curves, which match very well with the results presented in Figure 2.

Simulations in an SDR environment that can be backed up with real over the air measurements have a high potential. There are two major use cases where they are particularly beneficial. First, since one important application for the transceiver is the investigation and comparison of different receive algorithms, the simulation can be used to validate implementation of these algorithms under controlled circumstances. Once the implementations of the algorithms are validated by simulations, the very same systems and implementations can be used for over the air measurements.

-The second major use case for simulation is the investigation of wireless channels with different characteristics. There is a large body of channel measurements and characterizations in the VANET community. Most of the measurements are made with sophisticated channel sounders that allow to measure the characteristics in great detail. That way, features like delay spread and coherence time are measured and different channel models that capture the characteristics of different environments are proposed [25]. These channels can be easily implemented in GNU Radio and used to study the performance of different receive algorithms under reproducible conditions.

3.4.1.5. PERFORMANCE COMPARISON WITH COMMERCIAL IEEE 802.11P DEVICES

Even though we showed in the above mentioned simulations that the implemented algorithm works nicely in theory, we still owe a proof that the SDR system also works well with real hardware and produces results that are similar to commercial IEEE 802.11p devices. This is a crucial part for the evaluations since it shows that the implementation is indeed usable for research: producing reasonable results not only by means of simulations but also over the air with all the hardware impairments like frequency offsets, clock drifts and imperfect channel filters. These measurement results have been presented for the receiver. To also provide these error curves for the transmitter, we connected the USRP via cable and attenuators to a Unex DCMA-86P2, a commercial IEEE 802.11p capable Wi-Fi card. These cards are based on an Atheros transceiver chip and can be operated in IEEE 802.11p mode on the frequency band reserved for ISM applications (with minor modifications to the Linux kernel like removal of regulatory restrictions and the implementation of an interface to change the signal bandwidth to 10 MHz).

All measurements are performed on channel 172 with a center frequency of 5.86 GHz, again a packet size of 133 B, and a rate of 30 packets per second. Like in the case of the simulations, we send 10000 packets per configuration. At first, we investigate the transmit side and send frames with the SDR and receive them with the commercial Wi-Fi card. The resulting error curves are depicted in Figure 3. The SNR is measured on the receiving side by the Unex devices. When put in monitor mode, the cards annotate each received frame with metadata including signal and noise levels. To directly compare the SDR with a commercial device, we repeated the same measurements with another Unex device as sender. Since we experienced deviating results with different Wi-Fi cards, we used the same receiver for both measurements. The error curves of these measurements are shown in Figure 36.

-We can see that the results for both devices match very closely except for the QAM-64 3/4 encoding where we experience worse performance with the SDR. Since we do not see such an effect in simulations and since the systems works well for the other cases, we are reasonably sure that the sample stream we generate is correct. Furthermore, we experienced no underruns, i.e., we were able to stream the samples to the device so that the device did not stall which would destroy the physical wave form. With these observations, we expect the deviation in the results to be caused by hardware imperfections. Candidates that might cause such a behavior are oscillator drift and, more likely, non-linearity in the amplifier that might slightly disturb the signal, which might lead to packet errors especially in higher order modulations.

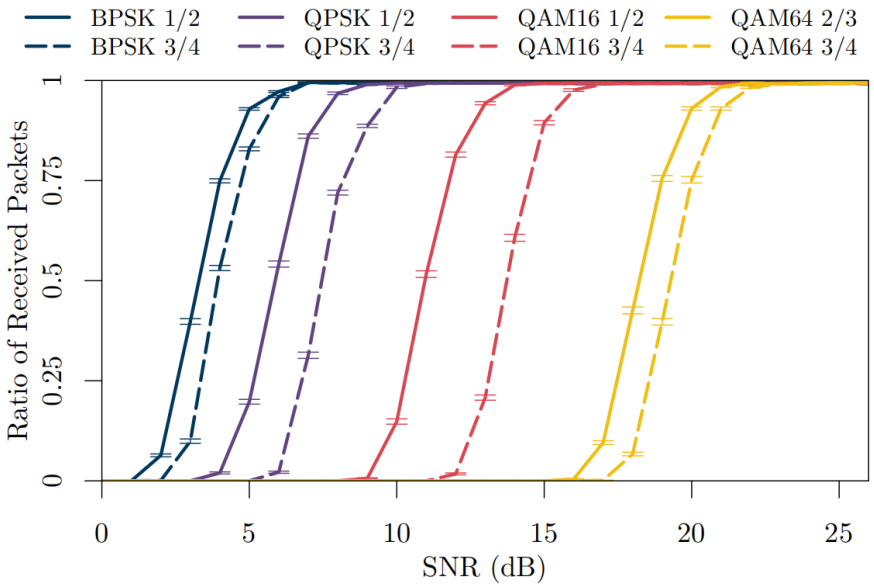

Figure 40: Simulative determined packet delivery ratio of 133 B sized packets

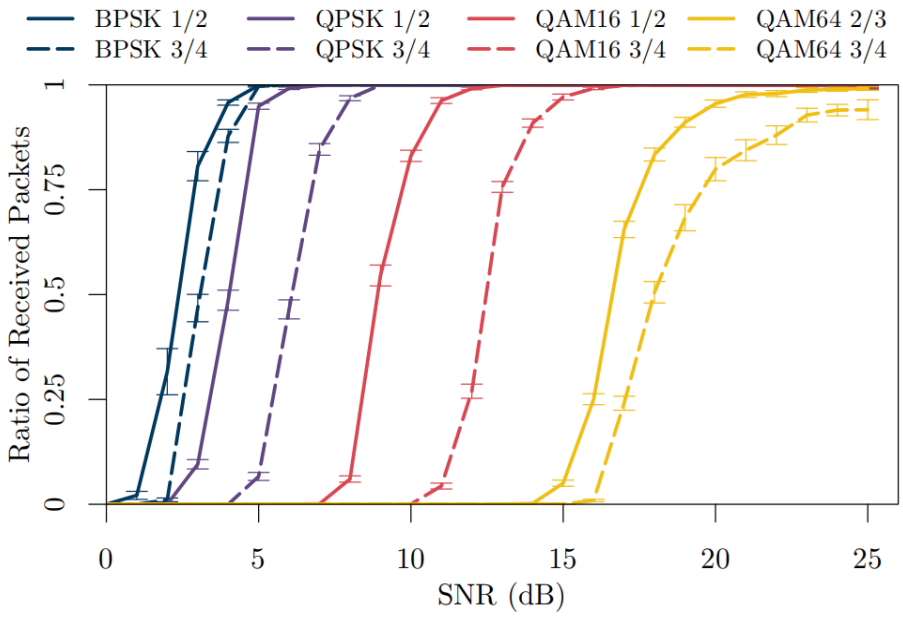

Figure 41: Packet delivery rate of frames sent from the SDR and received with

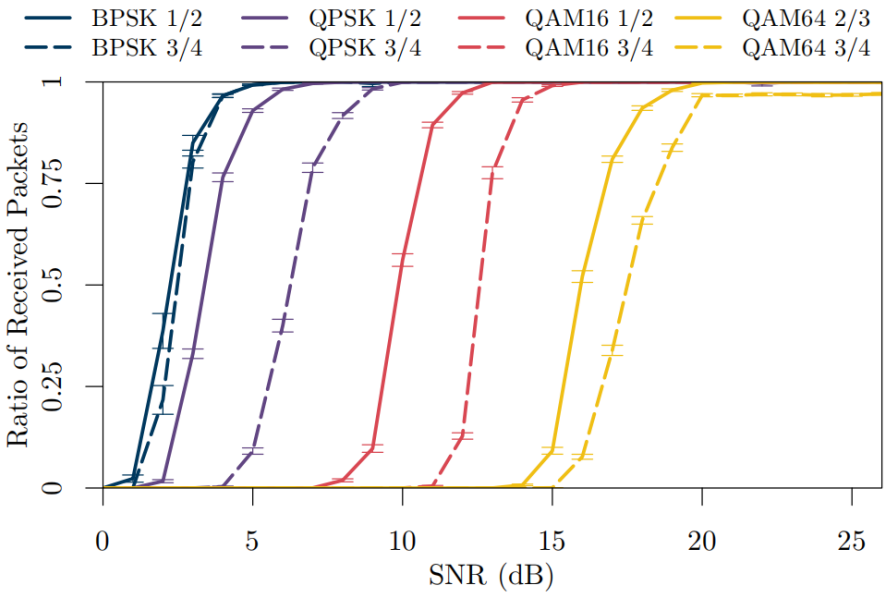

Figure 42: Packet delivery rate for two commercial grade IEEE 802.11p devices

3.4.1.6. GNU Radio block diagrams

3.4.1.6.1. WIFI physical hierarchy

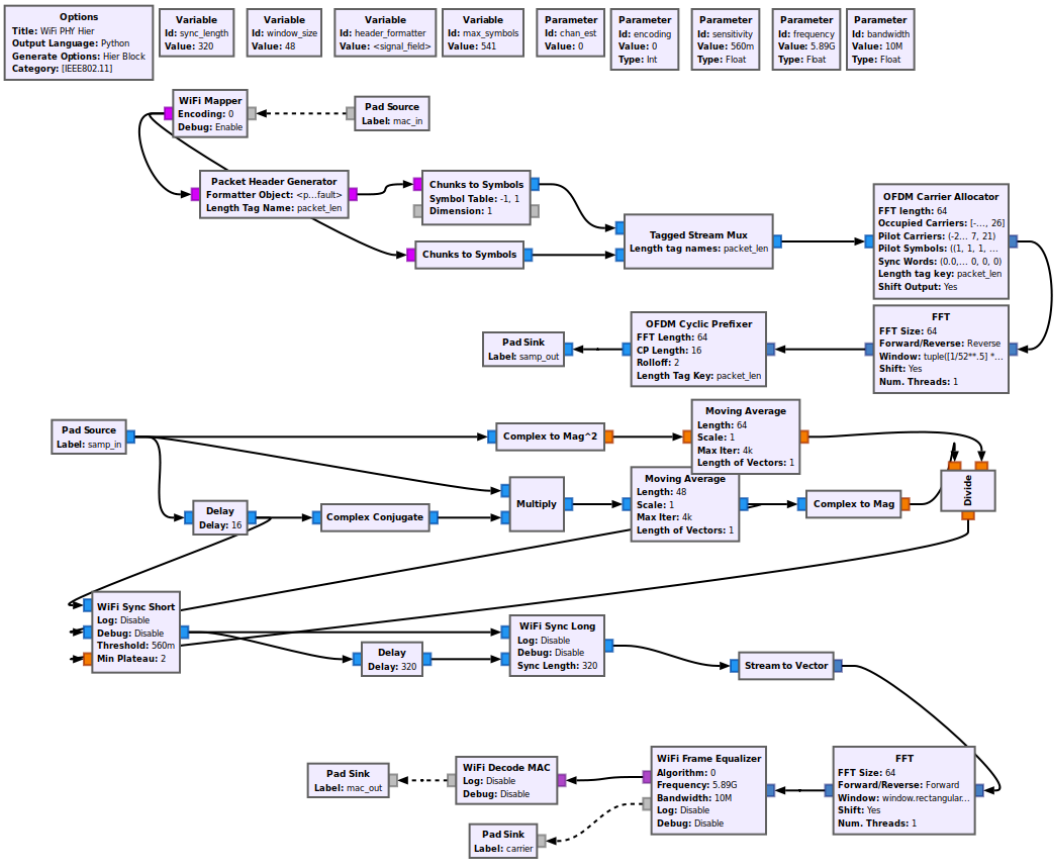

Figure 43: Wi-Fi physical hierarchy flow graph

The previous block diagram shows the blocks of the physical hierarchy in details and as explained in the standard. The Wi-Fi Mapper does the functions of the PLCP; scrambling, interleaving and splitting the data into symbols. The OFDM carrier allocator puts the symbols into the destined subcarriers, adds the pilots and prepares the OFDM symbol for the IFFT block to perform inverse fast Fourier transform. At the end, before transmitting, the cyclic prefix is added through the OFDM Cyclic pre-fixer block.

3.4.1.6.2. Transmitter flow graph

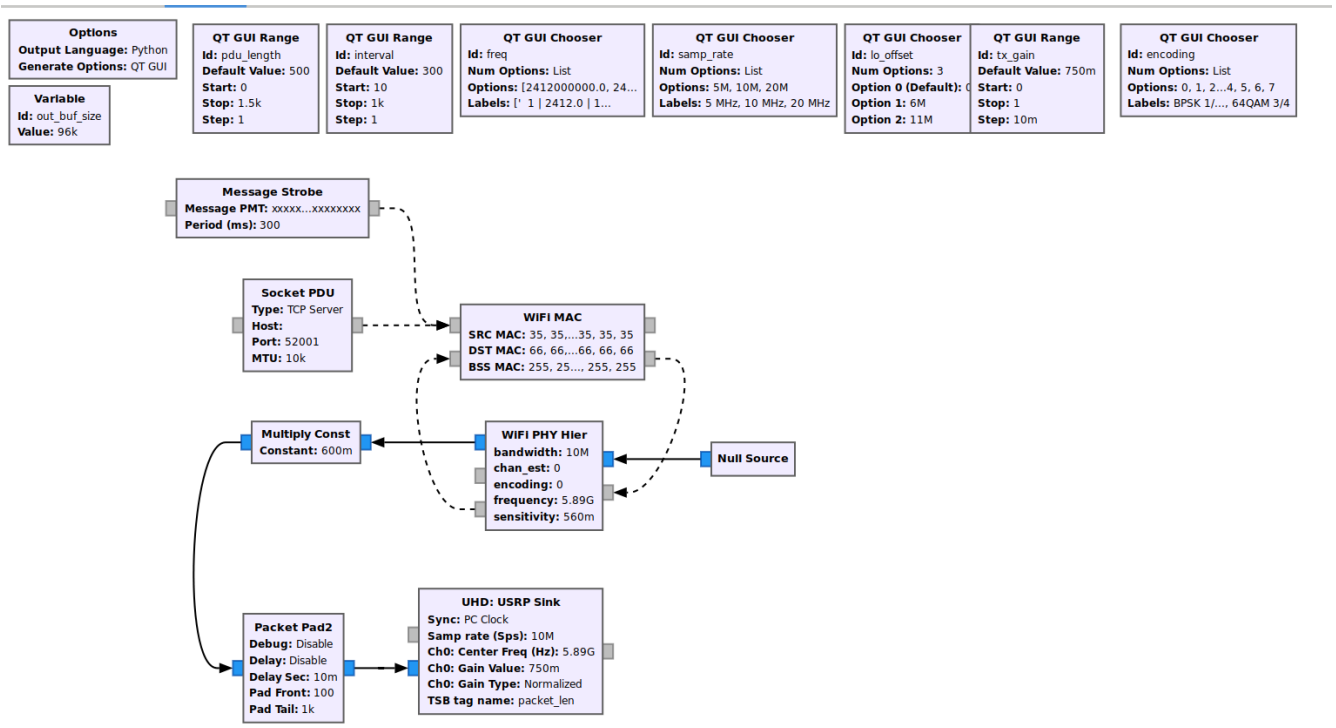

Figure 44: Transmitter flow graph

The physical hierarchy is all inserted into a single block called Wi-Fi PHY hierarchy. This block diagram shows the insertion of data into the MAC layer, then into the physical layer up to the USRP block which puts the data on the channel to be sent.

3.4.1.6.3. Receiver flow graph

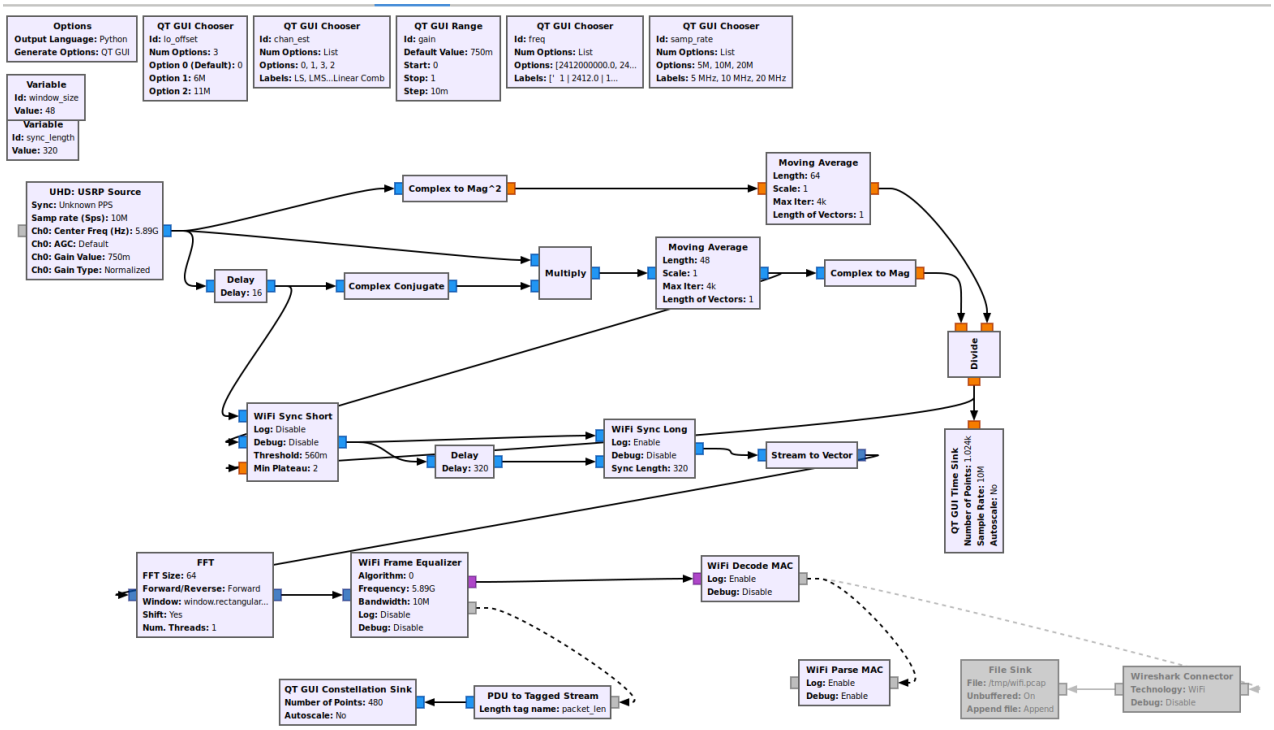

Figure 45: Receiver flow graph

This block diagram takes the receiver part of the physical hierarchy to receive the data from the channel first using the USRP block, then recover the data from the channel and start de-modulating, de-interleaving and de-scrambling the data.

3.4.1.6..4. Transceiver flow graph

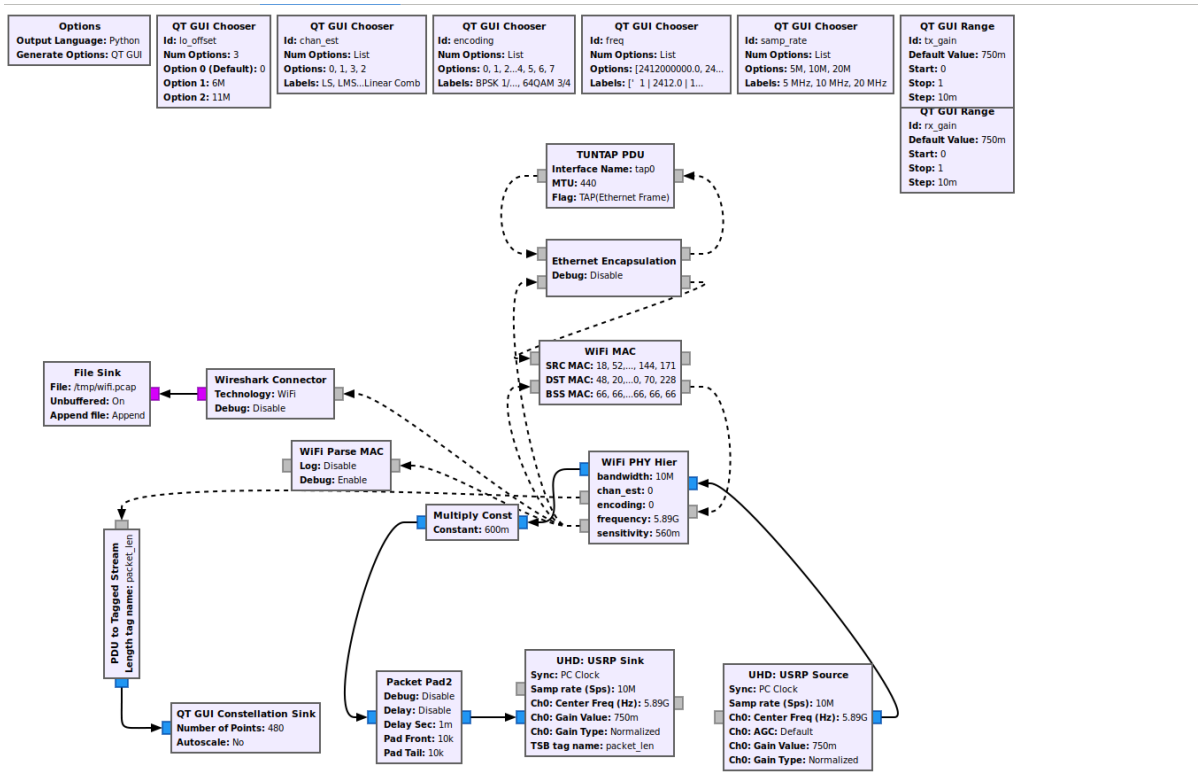

Figure 46: Transceiver flow graph

To show both sides in one block diagram, this block diagram shows the transmitter, the channel and the receiver. The channel here is created using USRP.

3.4.2. V2V implementation through MQTT

In this part Vehicle-To-Vehicle Communication was implemented by using internet (MQTT) instate of WAVE (Wireless Access in Vehicular Environments) technology is dedicated short range communication (DSRC). Due to covid-19 pandemic that makes trying to use hardware (USRP) was difficult .so the way to make Vehicles to communicate with each other through internet is the most available and acceptable way in this time. It gives acceptable results isn't better that DSRC and it isn't considered real time application.

Using MQTT in Vehicle-To-Vehicle Communication to split map into regions, each region with topic named by this region which make all cars in this region able to publish and subscribe on this topic which make them visible to each other. Then by using function to calculate the distance between two cars to detect is car in specific range with another car and interested with this information or not.

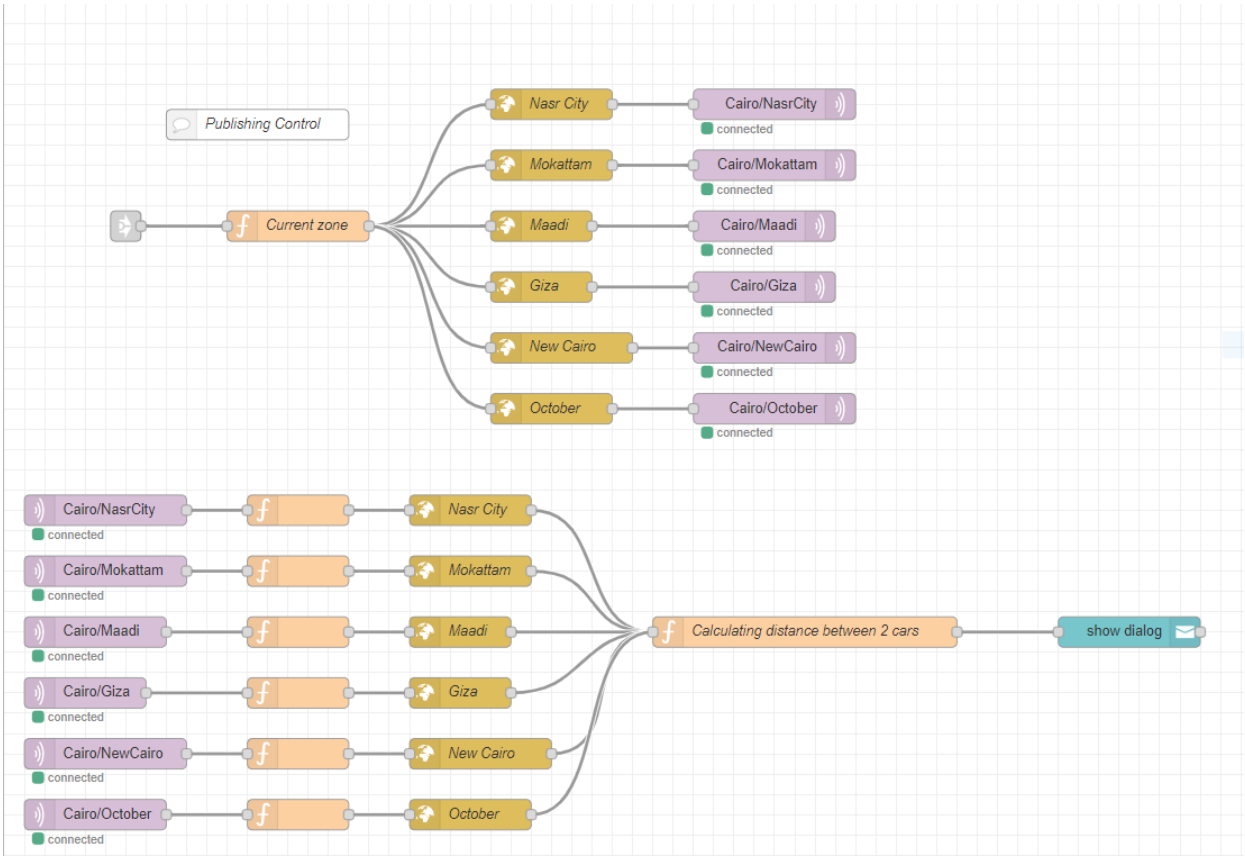

Figure 47: V2V implementation through MQTT

3.5. Cloud

In our project, there are two interfaces to reach cloud

- MQTT interface
- RESTful web services API over HTTP protocol

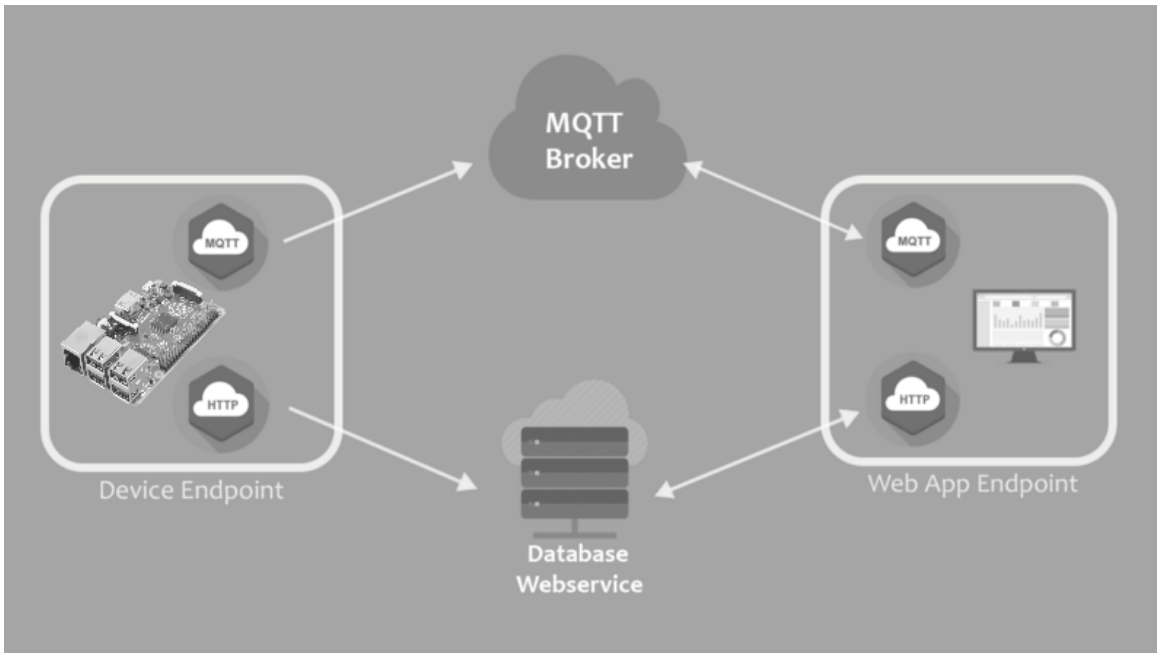

Figure 48: Cloud Interfaces

3.5.1. MQTT interface

MQTT is used to provide instant event notifications for the distributed applications, as a communication system built on MQTT supports the publish/subscribe model, in which any message published to a topic is immediately received by all subscribers on that topic.

It’s designed for constrained devices and low-bandwidth, high latency or unreliable network, so it's a good choice for building event notification systems.

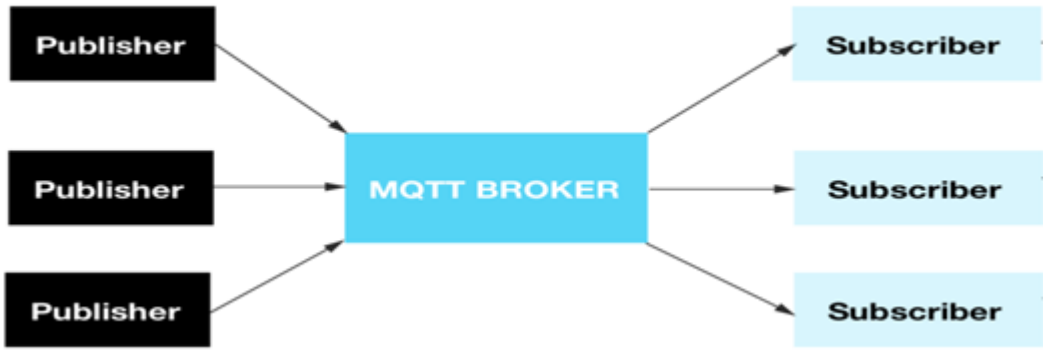

Figure 49: MQTT pub/sub Model

3.5.2 RESTful web services API architecture

As mentioned before, the objective is to develop a RESTful API offering Database services (CRUD operation) using PHP programming language over HTTP/1.1 protocol.

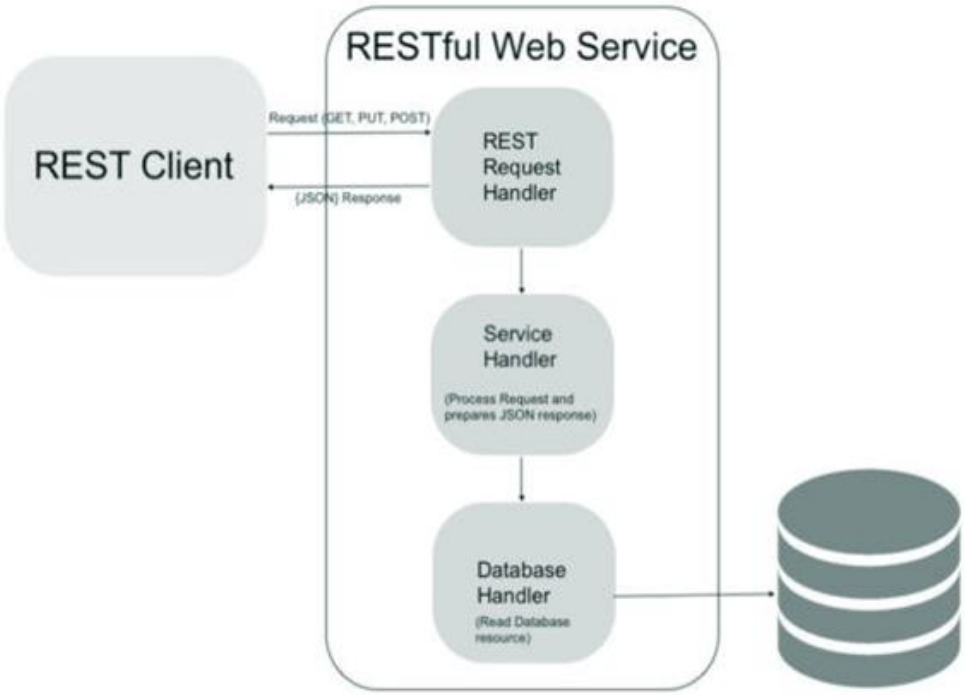

Figure 50: RESTful web services API architecture

As shown in Figure 50, the RESTful API consists of three layers

- **REST Request Handler**

  This layer is responsible for managing the HTTP request, get HTTP method in addition to getting request parameters.

- **Service handler**

  This layer is responsible for execute the HTTP request getting HTTP header parameters for example getting serialization format and serialize data

  - Serialization is the process of translating data structure or object into format that can be stored.

  - In our API the serialization format used is JSON

- **Database Handler**

  This layer is responsible for executing database DQL (Data query language) and DML (data manipulating language). In addition to parsing data from database objects into programming language objects.

3.5.3. RESTful web services UML sequence diagram

Sequence diagrams are used to show how objects in a program interact with each other to complete tasks. In Figure 51, a sequence diagram is used to visualize how our application interacts between its layers to complete requests of getting vehicle information.

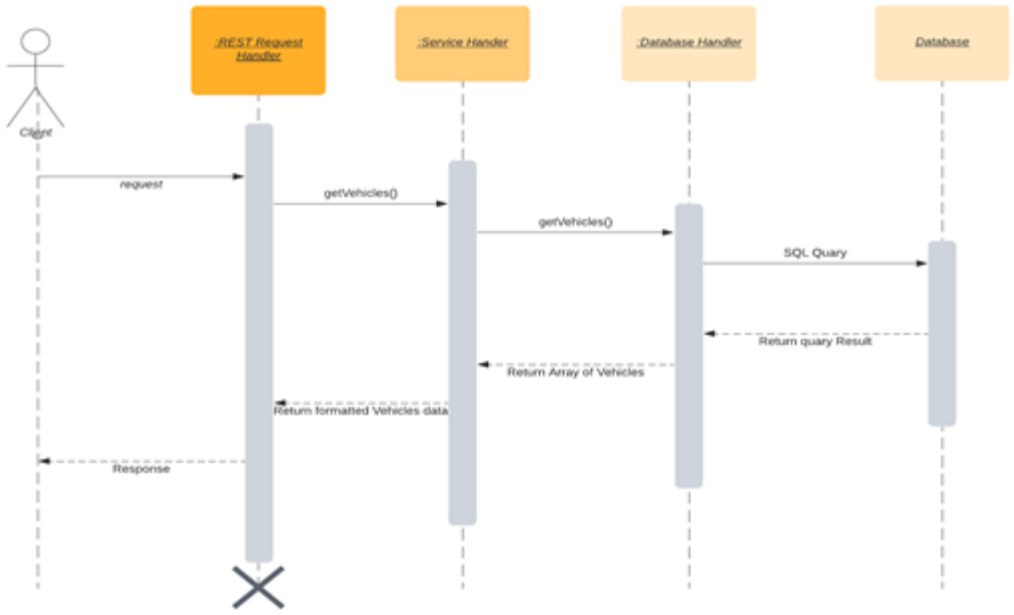

Figure 51: UML sequence diagram

As a client, send HTTP request to our RESTful API application passing through the design layers REST Request handler, Service Handler and Database Handler. Then execute the database query in Database management system (DBMS), and return to Database handler layer to parse database object to PHP language object then serialize data to Jason format in Service Handler layer and return the response to client, which contains desired data.

3.5.4. RESTful web services UML Class diagram

The UML Class diagram is a graphical notation used to construct and visualize object oriented systems. A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system

By showing the systems:

- Classes
- Class attributes
- Class operations

The relationships among objects

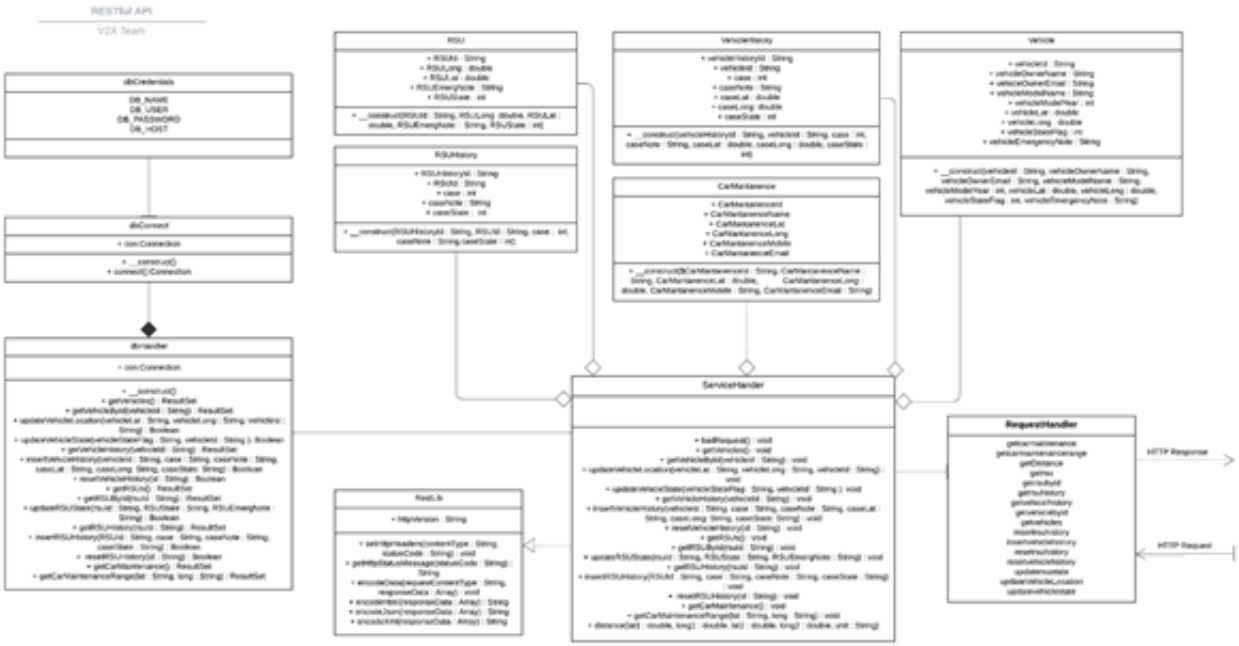

Figure 52: UML class diagram

3.5.5. Testing RESTful API

Testing RESTful API is done by a software testing tool called postman, which is used to test RESTful API applications by offering a friendly GUI show HTTP packets. To be able to check the header attributes and the data returned in the body of the packet.

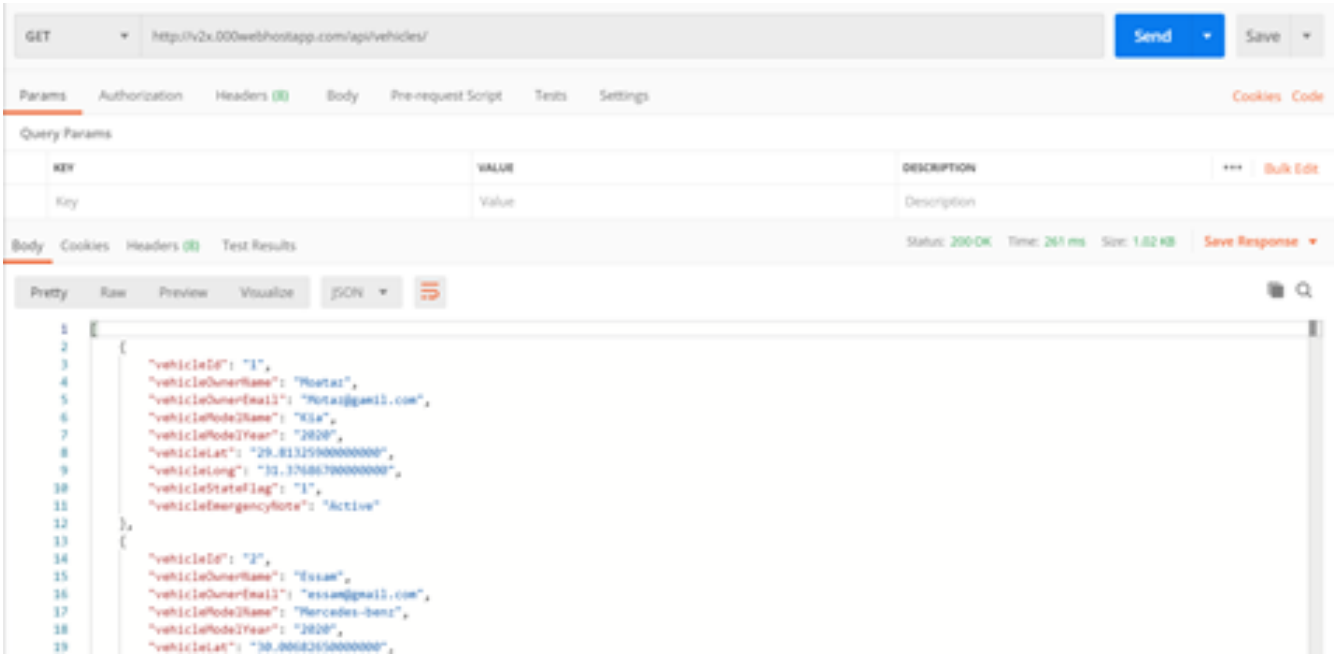

Figure 53: Postman software GUI

3.5.6. Database

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).

MySQL is a database management system used in our project, it is a relational database that uses Structure Query Language for write and query data.

Relational database means the data is stored as well as retrieved in the form of relations (tables).

Structure Query Language (SQL)

Structured Query Language is a standard Database language, which is used to create, maintain and retrieve the relational database.

The queries to deal with relational database can be categories as:

- **Data Definition Language**

  It is used to define the structure of the database. e.g; CREATE TABLE, ADD COLUMN, DROP COLUMN and so on.

- **Data Manipulation Language**

  It is used to manipulate data in the relations. e.g.; INSERT, DELETE, UPDATE and so on.

- **Data Query Language**

  It is used to extract the data from the relations. e.g.; SELECT

3.5.6.1. Database ERD

An entity–relationship model describes interrelated things of interest in a specific domain of knowledge. A basic ER model is composed of entity types and specifies relationships that can exist between entities.

In Figure 9, the entity–relationship model describes the table level design of our database.

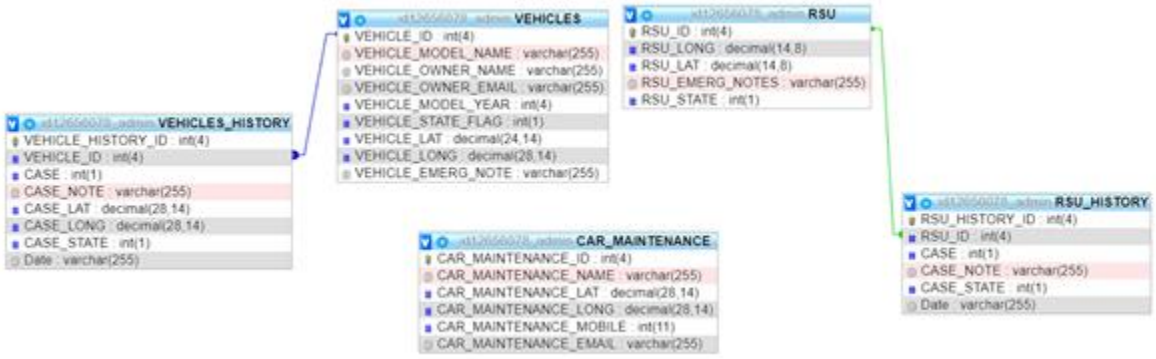

Figure 54: ERD diagram

3.6. Admin Panel

3.6.1. Overview

Admin panel's (control room) purpose is production control, and serves as a central space where a large physical facility or physically dispersed service can be monitored and controlled. Control rooms for vital facilities are typically tightly secured and inaccessible to the general public. Multiple electronic displays and control panels are usually present, and there may also be a large wall-sized display area visible from all locations within the space. Some control rooms are themselves under continuous video surveillance and recording, for security and personnel accountability purposes. Examples of control rooms

-Nuclear power plants and other power-generating stations.

-Major transportation facilities, such as bridges, tunnels, canals, airports, and rapid transit systems, may have 24-hour manned control rooms to monitor and report on traffic congestion, and to respond to emergencies.

-Computerized data centers, which often serve remote users in multiple time zones worldwide.

-Emergency services, including police, fire service and emergency medical service.

3.6.2. Contents

V2x system is a network and operating system with Major purpose to get all things in this system is connected with each other, to make all these things are controlled, it is required to have a control panel, to monitor and report on traffic flow, congestion and respond to the problems that occurred.

In this system we have in admin panel three taps

- System map.
- Road side unit (RSU) tap.
- Control unit

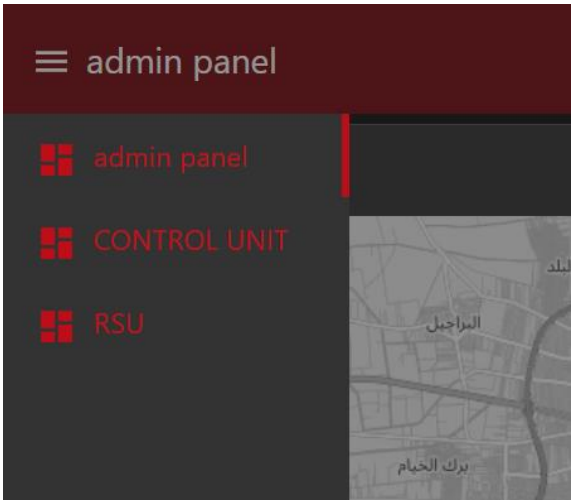

Figure 55: Admin Panel Contents

3.6.3. System Map

System map it is a tap (window) on admin panel is used to monitor the overall system which contains cars, roadside units (RSU) and car maintenance to make the admin can be able to see all system and be aware of problems that happen. And also show the number of active roadside units (RSU) **red** building pin marks and the number of active car maintenance (CM) **blue** building pin marks

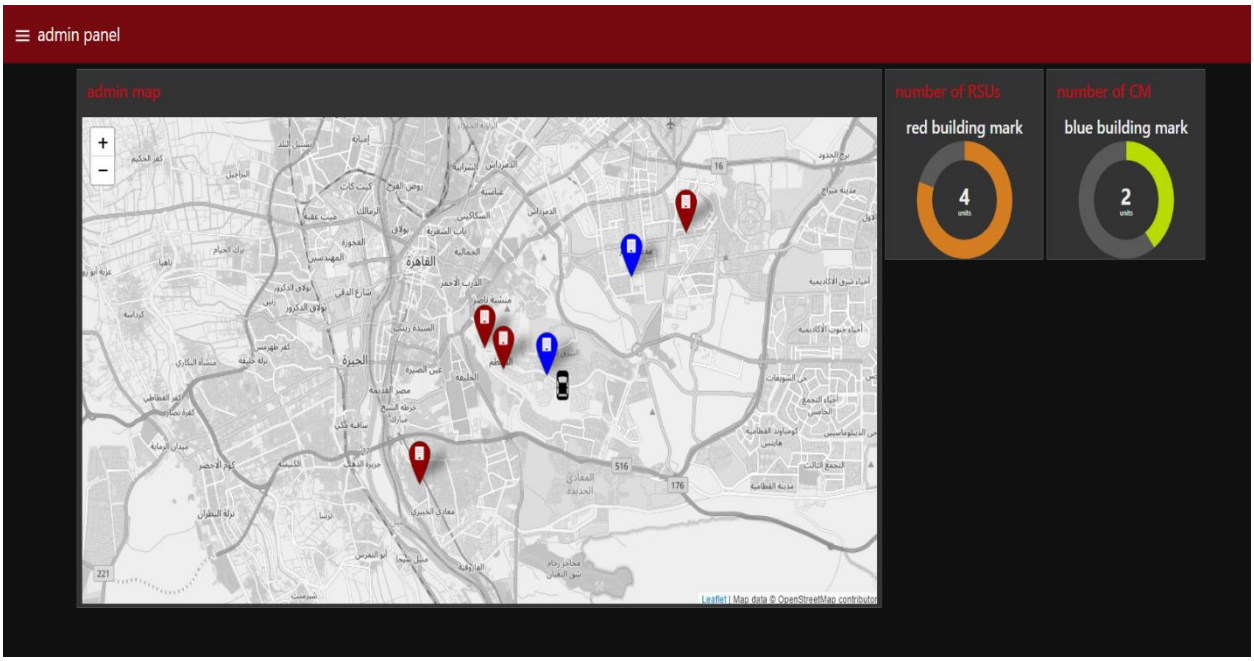

Figure 56: System map

3.6.3.1. System Map Flow

This flow is divided into four flows with two different types of data. Data from database and other from MQTT. Four flows are:

1-Getting car maintenance information location and status (open or closed) And also get the number of CM in the system and printing it on admin map

This is done by HTTP request restful API.

To call function from database to get all CM information stored in system Database (HTTP request). Then this data pass to CM function block to analyze this data and storing it.

2-Getting roadside unit info locations, status and emergency note that indicate for road health and reporting this problem if this happen

3-Getting car locations by MQTT and storing it. Showing on map if the car has a problem

4-Showing all this information collected from these flows above and print it on map.

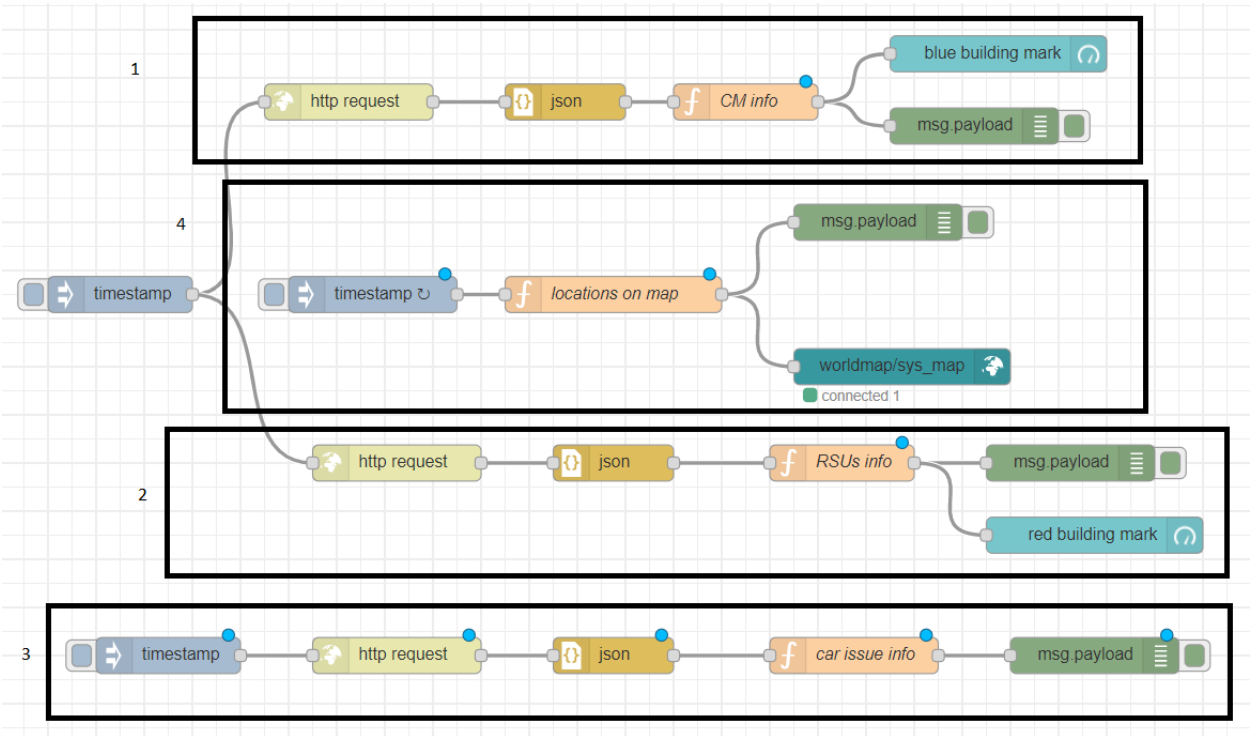

Figure 57: System map flow

3.6.4. Control Unit

In this system control unit is designed for cars and all information about it, location, and health report this tap is divided into four small taps

1- Search for information.it is used to make the admin or control map be able to get all information about specific car.

2- Car location on map.

3- Car information: This tap contains all car information stored in the database like car ID, model name, and year of the vehicle and owner Email.

4- Car issue report for showing car health. Active if this car is good without reporting any problem or damage if there is some kind of problem.

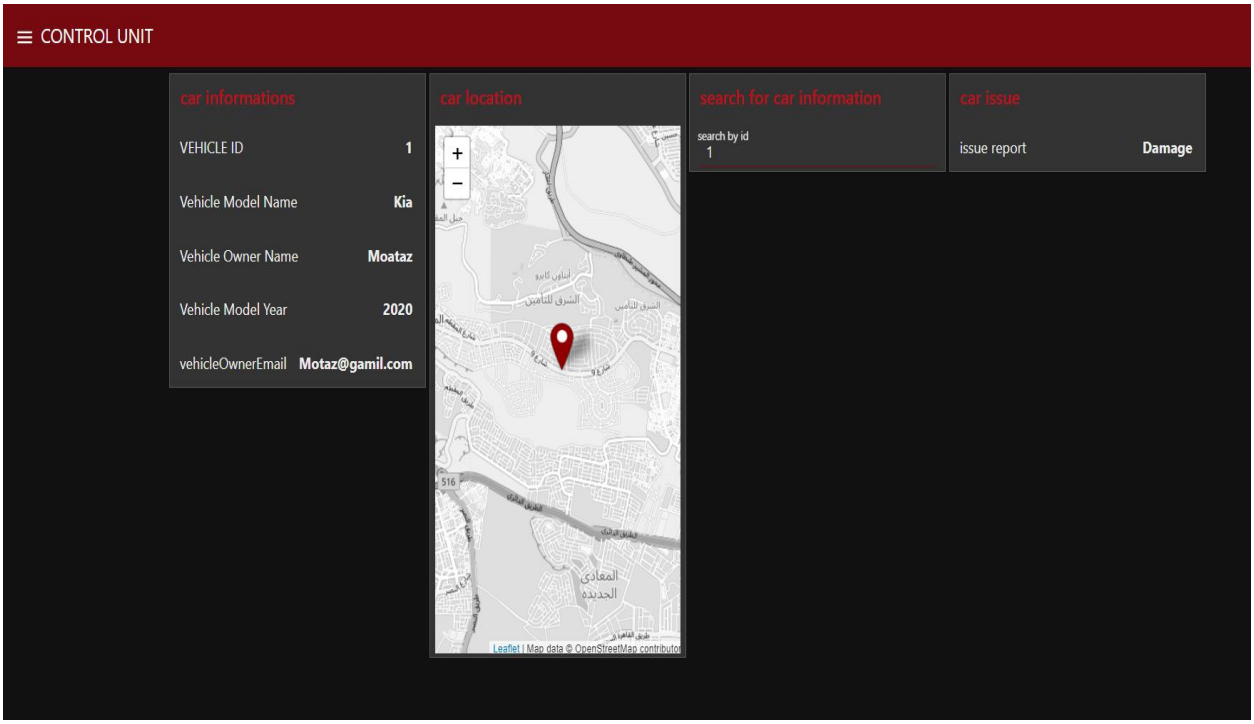

Figure 58: Control unit

3.6.4.1. Control Unit Flow Graph

Control unit flow is divided into two flows first one is getting the desired Car ID from the admin control unit graphical user interface and storing this ID to make an HTTP request through restful API with this Id of the desired car to get all information about this car from data that stored in the system database, the output of the HTTP request is Jason format then by parse block is converting it to JavaScript object. After getting the data from database data passed to getting a car info function block to analyze this data and printing it on GUI which is data of the desired car.

Figure 59: HTTP request database output for car info

The second part is to get the location of this car on map.

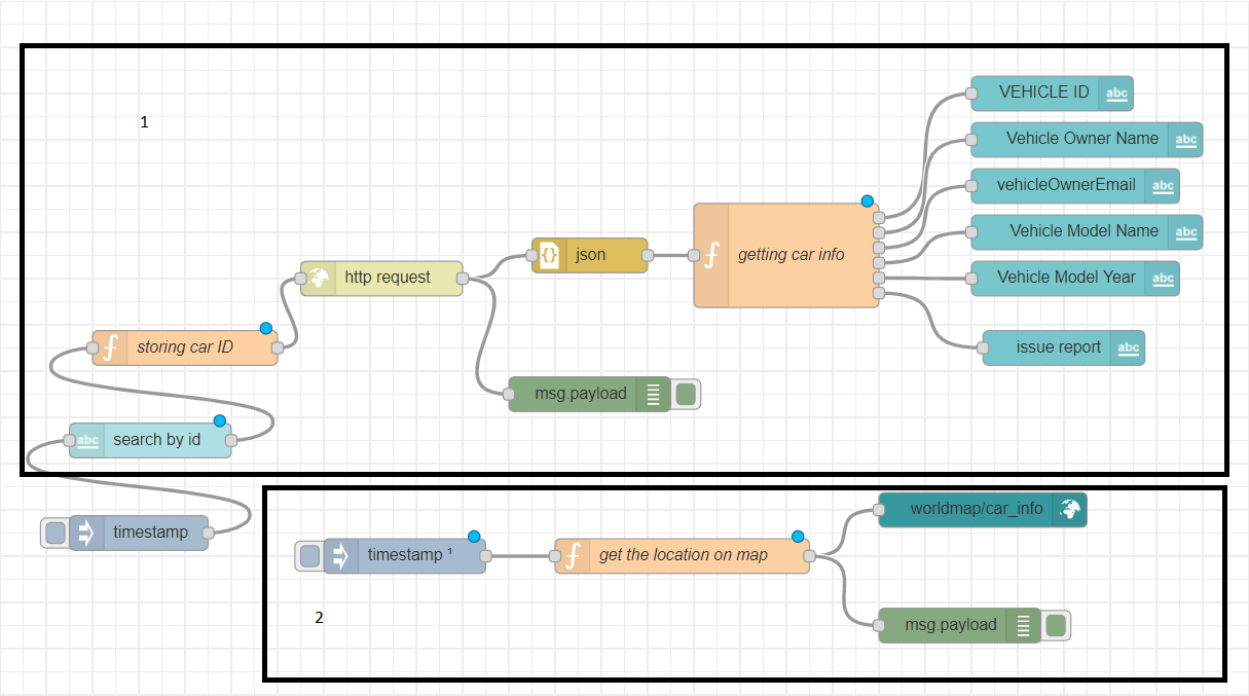

Figure 60: Control unit flow

3.6.5. RSU

In this system Roadside unit is designed for RSU and all information about it, location, and status. This tap is divided into four small taps

1- Search for information.it is used to make the admin or control map be able to get all information about specific Road side unit (RSU).

2- RSU location on map.

3-

3- RSU information: This tap contains all RSU information stored in the database like RSU ID, LAT, LONG, RSU state and RSU emergency note.

4- Car issue report for showing RSU warnings.

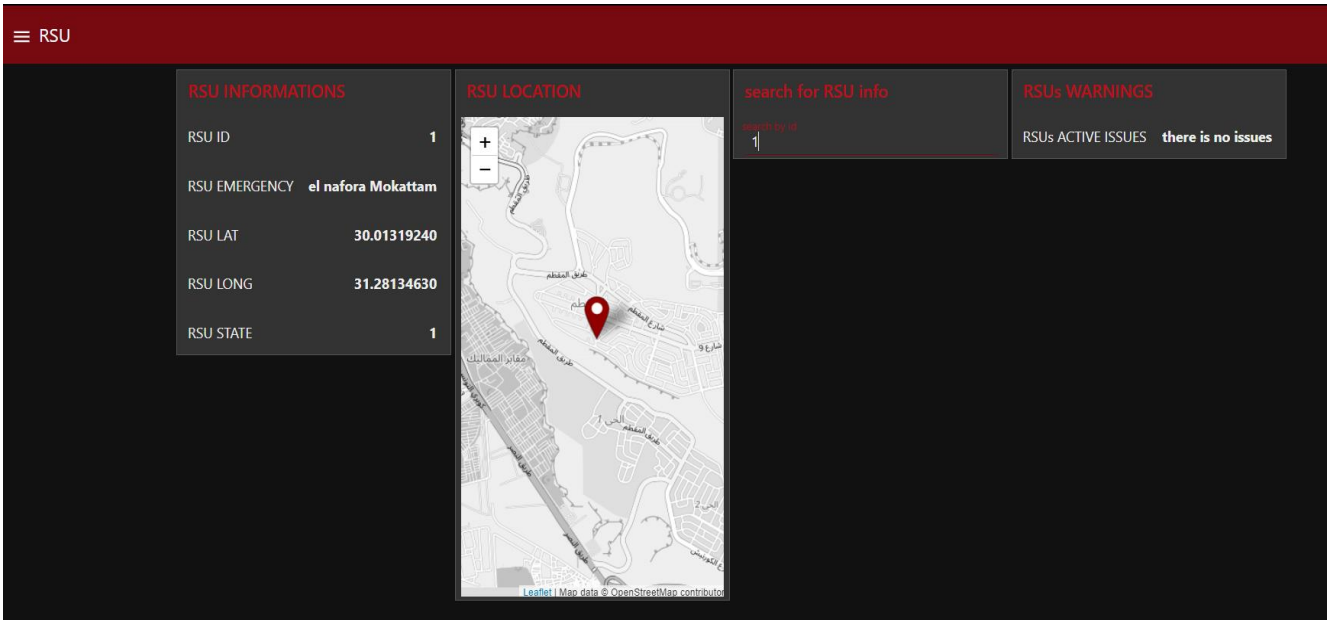

Figure 61: Roadside unit (RSU)

3.6.5.1. Roadside Unit Flow Graph

RSU flow is divided into three flows first one is getting the desired RSU ID from the admin RSU graphical user interface and storing this ID to make an HTTP request through restful API with this Id of the desired RSU to get all information about this RSU from data that stored in the system database, the output of the HTTP request is Jason format then by parse block is converting it to JavaScript object. After getting the data from database data passed to getting a RSU info function block to analyze this data and printing it on GUI which is data of the desired RSU.

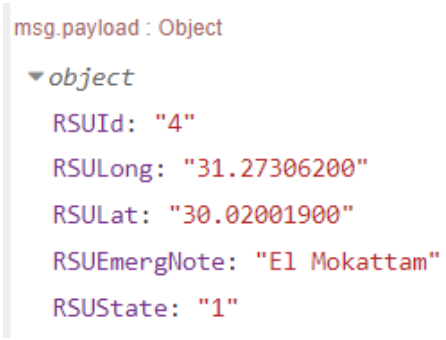

Figure 62: HTTP request database for RSU info

The second part is to get the location of this car on map.

The third part is getting warnings by MQTT.

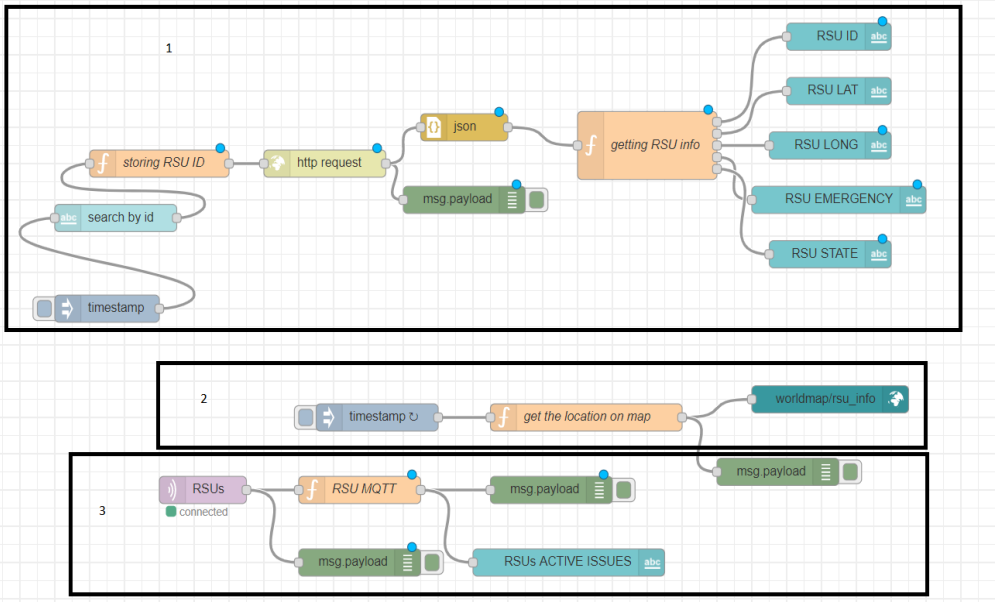

Figure 63: RSU flow

3.6.6. MQTT Flow Graph

This flow graph is designed to get all car locations (latitude and longitude) by using MQTT.

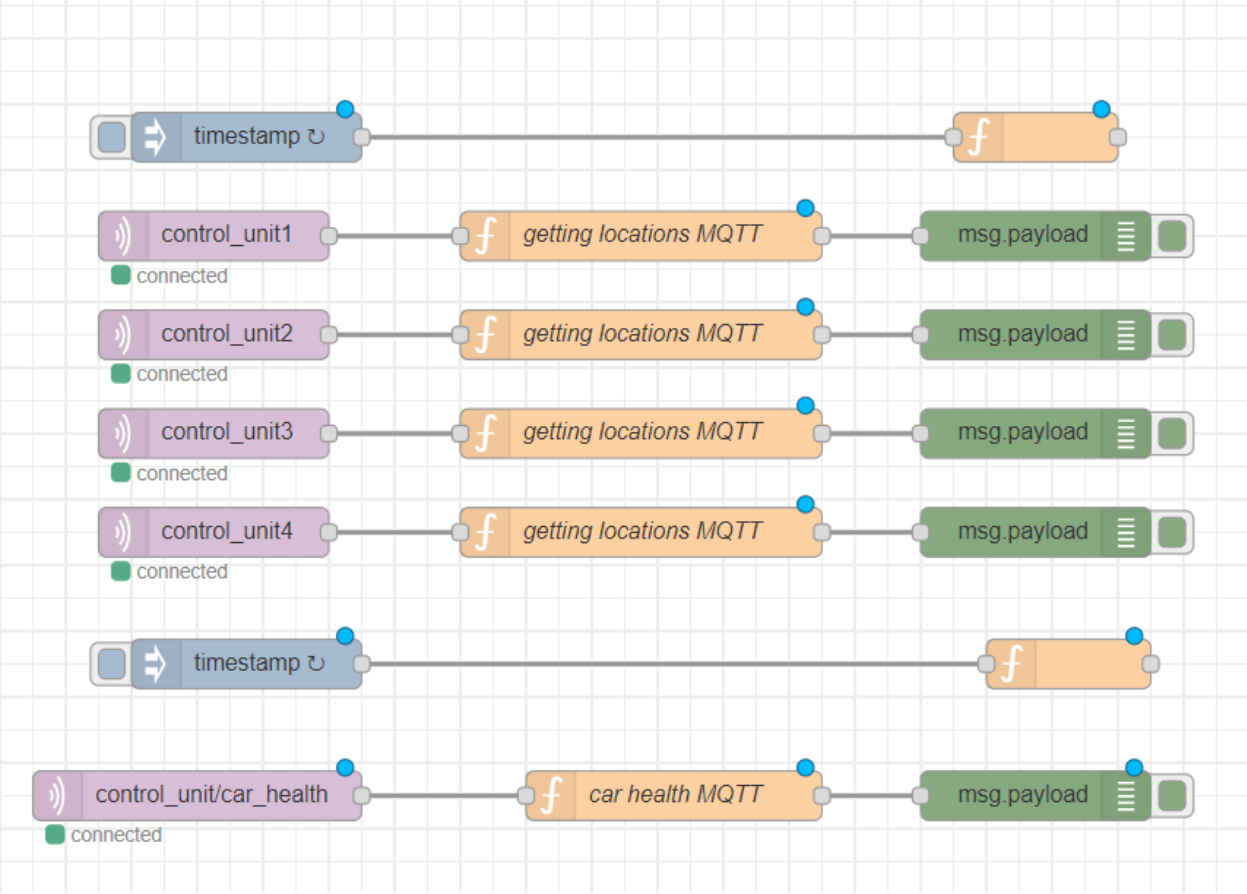

Figure 64: MQTT flow graph

CHAPTER 4: RESULTS AND DISCUSSION

4.1. Electronic Emergency Braking Light Results

4.1.1. Preface

The Electronic Emergency Braking Light (EEBL) Warning enables a vehicle to broadcast a self-generated emergency brake event to surrounding vehicles. Upon receiving the event information, the receiving vehicle determines the relevance of the event and if appropriate provides a warning to the driver in order to avoid a crash. This application is particularly useful when the driver's line of sight is obstructed by other vehicles or due to bad weather conditions (e.g., fog, heavy rain)

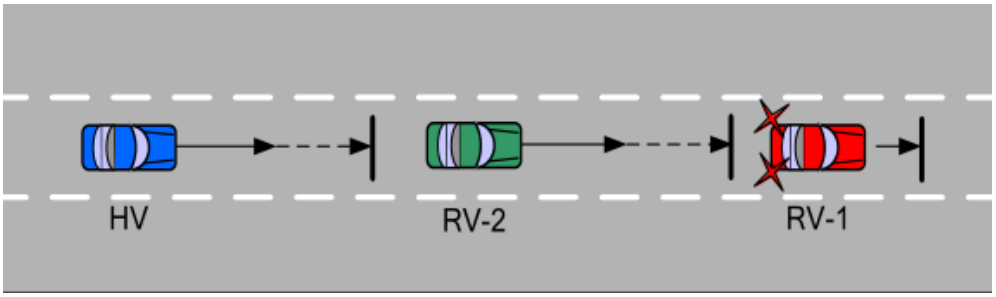

Figure 65: Electronic Emergency Braking Light

4.1.2. EEBL Flow Description

Requirement: The needed data for this use case are speed, latitude and longitude as shown in figure below.

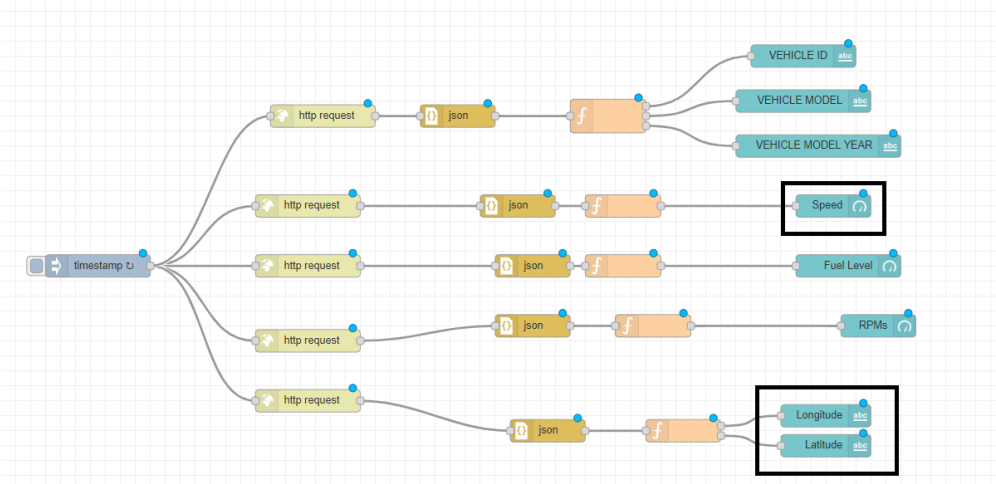

Figure 66: MQTT for acquiring data

The mechanism of this use case: By acquiring the speed of the vehicle continuously, if the value of the speed decreased by a certain value within a certain small time then driver must have braked suddenly and a waning is generated and sent from this vehicle to all the surrounding vehicles.

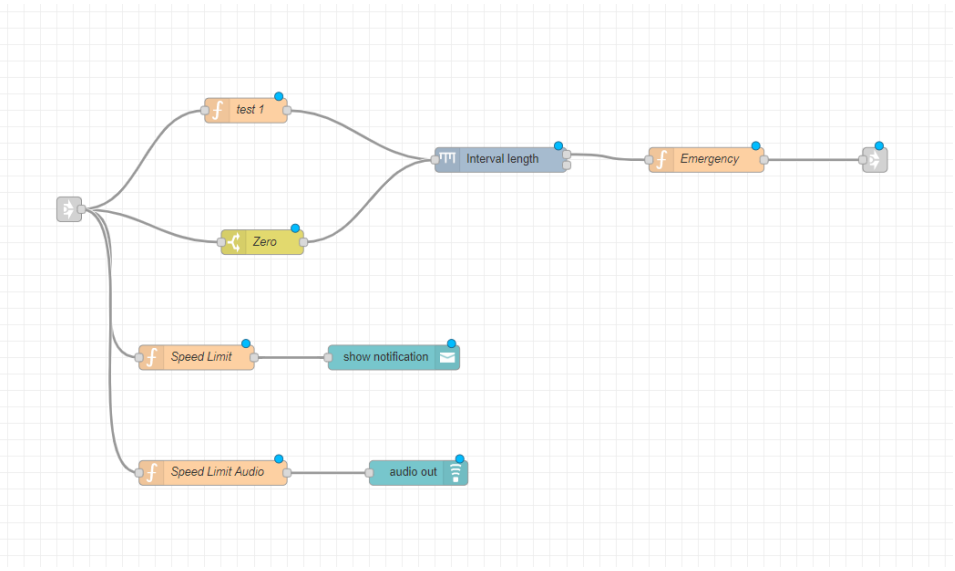

Figure 67: EEBL mechanism flow

Publish/Subscribe Control: In order to control the publish/subscribe method, the regions is divided into sub-regions each region has a different topic and the published/subscribed vehicle change its topic dependent on its location. As the received warning should be in the same area where the actual event happened.

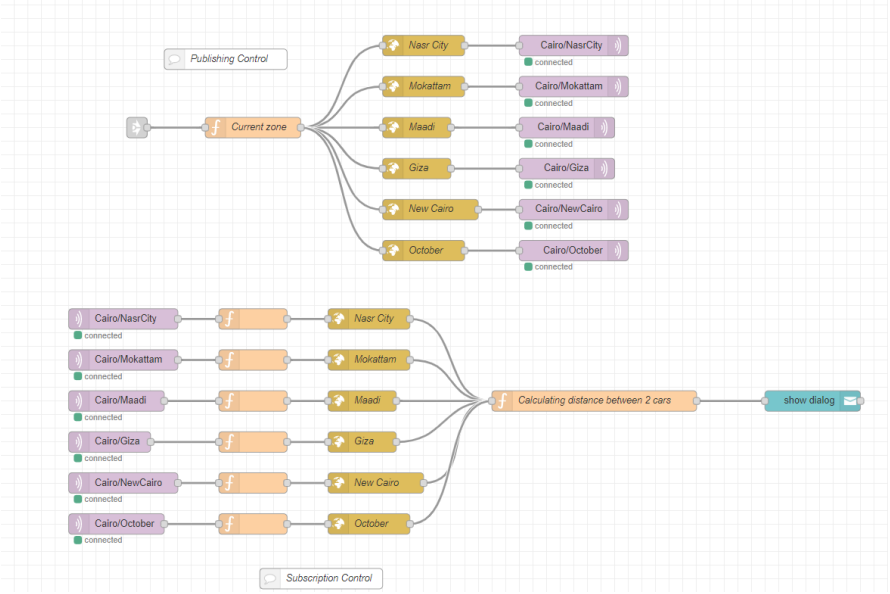

Figure 68: Pub/Sub region control

4.1.3. EEBL Warning

As soon as the event is generated the distance between the stopped vehicle and the other moving vehicles is calculated and a warning appears in the dashboard of the moving vehicle as in figure 70.

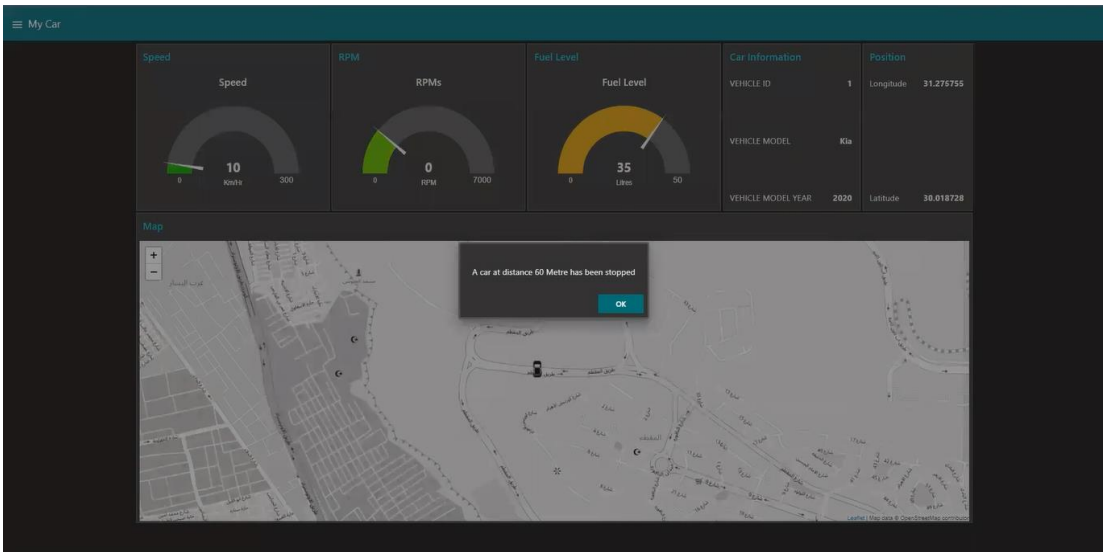

Figure 69: EEBL Warning

4.1.4. Comparison between different technologies

The Crash Avoidance Metrics Partnership (CAMP) Vehicle Safety Consortium Communications (VSCC) comprising BMW, Daimler Chrysler, Ford, GM, Kia, Nissan, Toyota, and Volkswagen, in partnership with USDOT, proposed more than 57 application scenarios about Connected Vehicle, like safety applications, non-safety applications, high potential benefit safety applications, and other applications. For the Electronic Emergency Braking Light (EEBL) applications, the latency was decided to be around 100 millisecond.

Table 6: DSRC vs C-V2X vs Proposed Implementation

DSRC	For this use case results a latency from 10 milliseconds to 140 milliseconds depending on the speed of the vehicle and the environment.
C-V2X (LTE)	For this use case results a latency much higher than 100 milliseconds mainly because of Doppler Effect and cellular handoff of LTE network but with higher coverage area than DSRC.
Proposed Implementation	For this use case results a latency around 140 milliseconds with a high coverage area which is kind of acceptable for safety application.

4.2. Road Works Warning

4.2.1. Preface

Road works occur when part of the road, or in rare cases, the entire road, has to be occupied for work relating to the road, most often in the case of road surface repairs.

When the road side unit detects that there is an issue with the road, it instantaneously reports the road works warning to both on board units and cloud infrastructure which is reported to administrators in the admin dashboard.

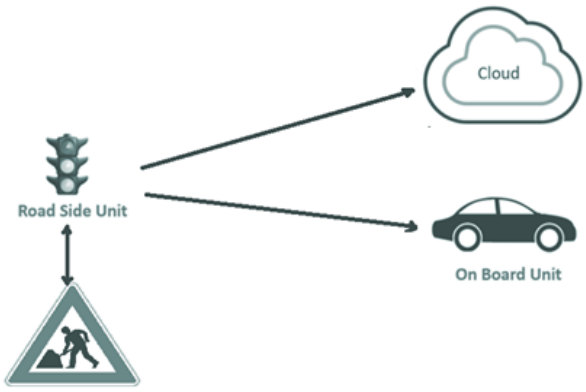

Figure 70: Road works warning use case

4.2.2. Admin Dashboard Warning

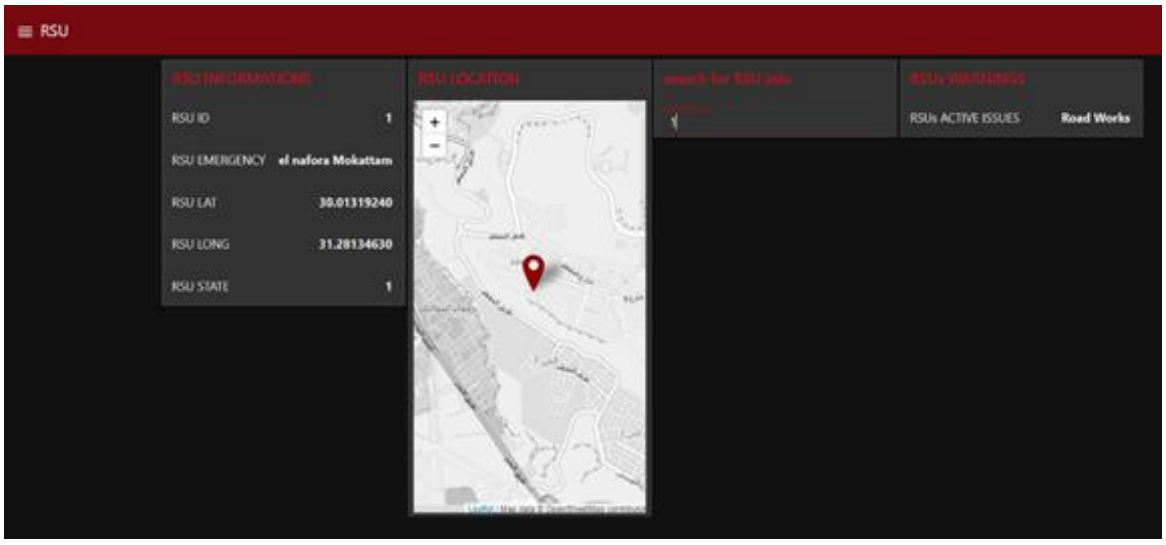

Figure 71: Snapshot of admin dashboard for RWW

The admin dashboard shows the detailed data of the road side unit, and contains the state of RSU which detected that there is a road works warning, to give the operators a warning to take appropriate action.

4.2.3. OBU Dashboard (Vehicle Dashboard)

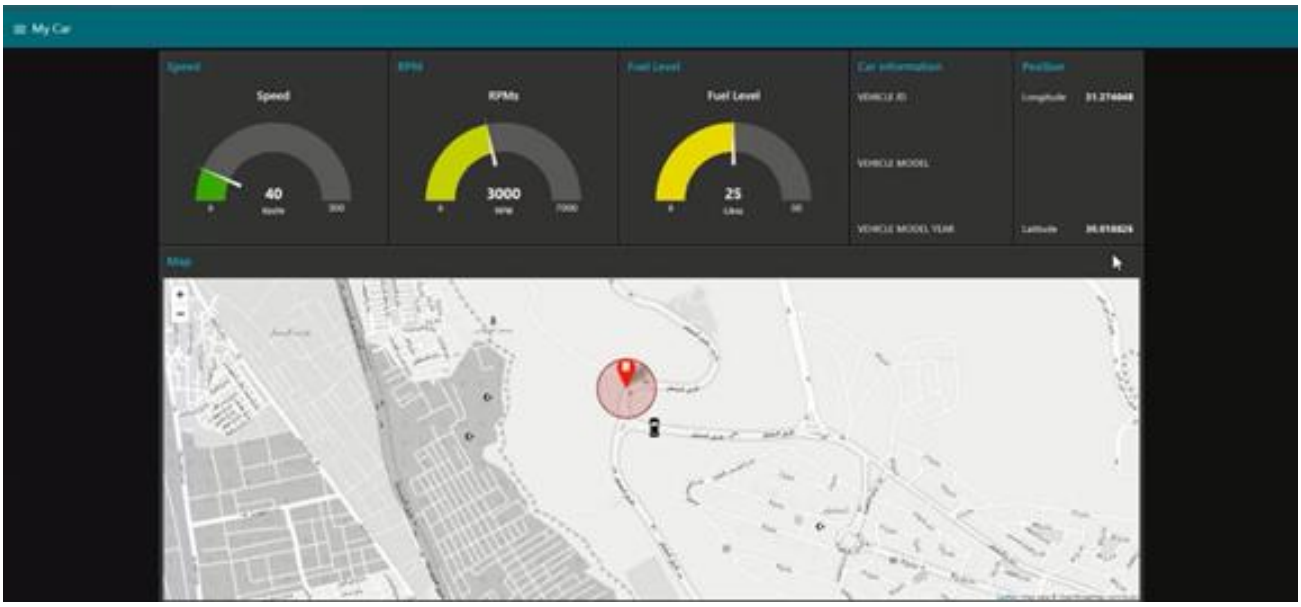

Figure 72: snapshot of the OBU dashboard for RWW

The red mark in the dashboard map referring to there is a roadworks warning in the marked road that gives the driver a warning to have a chance to avoid this road and avoid high traffic points.

4.3. Car Maintenance Results

4.3.1. Preface

Imagine if you got lost in a city you do not its people or routes and your car had broken, the car maintenance use case will make you call help immediately. It also solve the traffic congestion as if a car stopped in the middle of the street it will block the way causing traffic congestion.

4.3.2. Car Maintenance Flow Description

When a problem occurs in the car and the car has to stop, the driver reports the problem to the admin panel and his vehicle's issue will appear on the management map and he sends his current location to the database to find out the information of the nearest car maintenance to his location and communicates with it. The flow graph in Figure 1 illustrates how the issue is sent to the Admin panel and how to obtain the information about the nearest car maintenance.

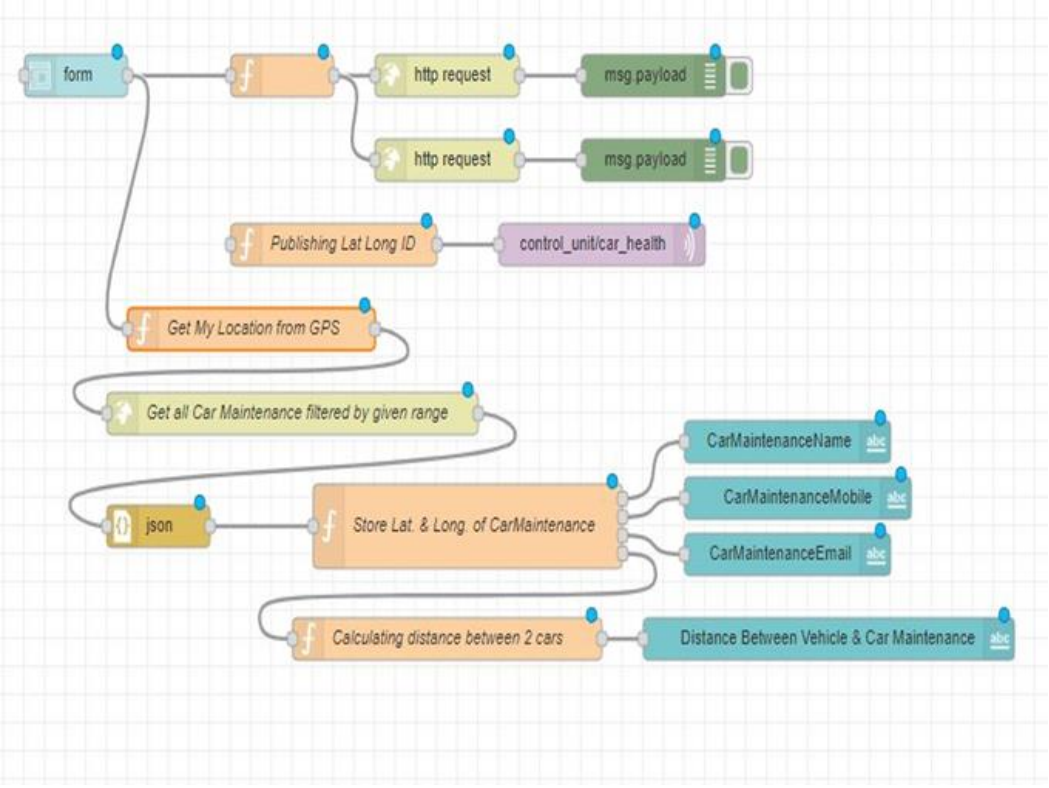

Figure 73: Vehicle Health flow graph

4.3.3. Car Maintenance Results

When the driver uses the dashboard to submit the issue as shown in Figure 2 and send it to the admin panel, the color of his vehicle on map will change from black to red as shown in Figure 12 in the Admin Panel results section. At the same time the driver will receive on his dashboard the information of the nearest car maintenance such as Name, Email, Mobile and the distance between Vehicle and nearest car maintenance as shown in Figure 2 then the driver can communicate with car maintenance as soon as possible.

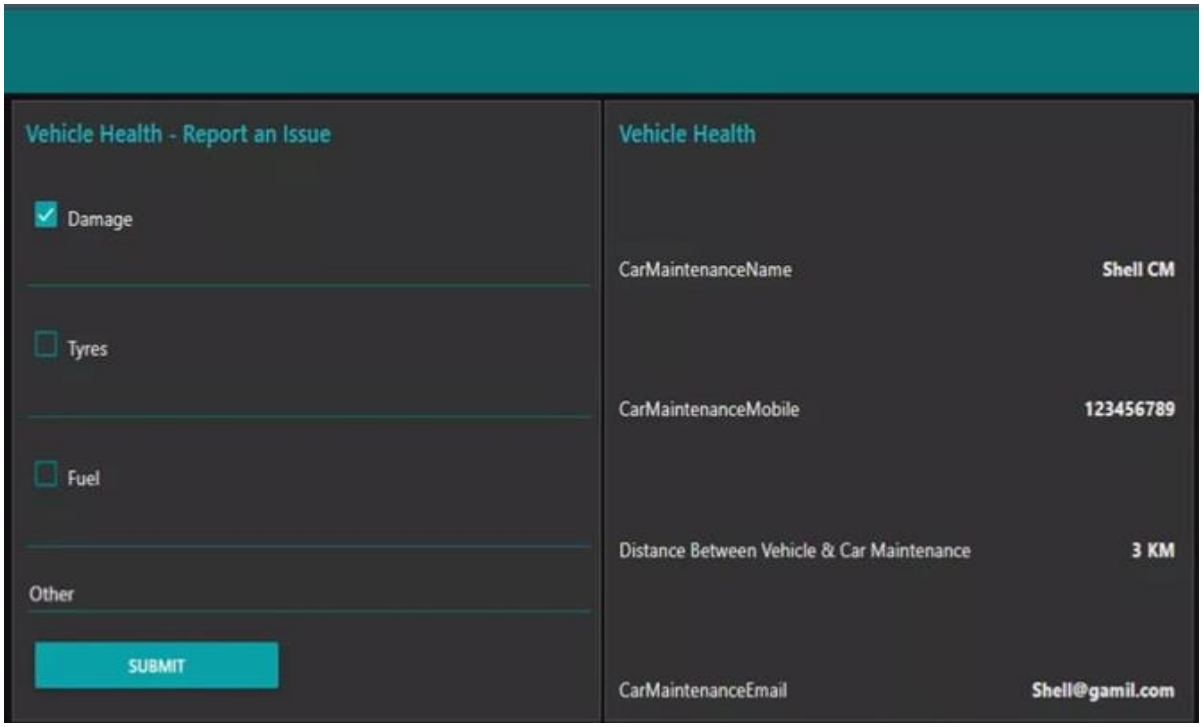

Figure 74: Vehicle Health dashboard

CHAPTER 5: CONCLUSION

5.1. Conclusion

To conclude, this thesis firstly discussed the problem statements that highlight the need of V2X communication systems and its need in markets nowadays, No doubts that the new cars' generation is going through a massive development to self- driving cars which increases the importance of V2X communications.

Our role in the project was design V2X system with available technologies and achieve low latency, Throughout our work, we took into consideration that V2X is a new technology and the system should be compatible with all the coming technologies, after understanding all technical aspects, the implementation of use cases have been did using some tools and the collected data from the car, then the dashboards of the system have been designed to make it so easy to use the system, then admin panel has been designed to test the results of the different use cases and control the system. Finally, take into consideration that this project can be improved using the coming technologies and the number of use cases can be expanded easily because of the compatibility of the system to the user requirements and the new technologies.

5.2. Future work

Future work of this project is very important to keep pace with the continuous development in technology, the future work of the project is divided into two levels.

5.2.1. LEVEL ONE

In this level future work will be in the system itself, system latency can be improved using new technology like 5G for example, and the number of the use cases can be expanded if the city that use this system want new use case or more than one new case, and all of that can be done easily using some improvement in software without any complexity changes in hardware.

5.2.2. LEVEL TWO

This level will done using the system or the improvement system, the future work here don't in the system but depend on the information that have been collected from the system, if all the data that have been collected from the system stored in dataset, this data set can be used in the self-driving cars, and after some years the Admin panel won't need any human control, and all of that can be done using deep learning with collected datasets.

REFERENCES

[1] Global, regional, and national age-sex-specific mortality for 282 causes of death in 195 countries and territories, 1980–2017: a systematic analysis for the Global Burden of Disease Study 2017.

[2] Global health estimates 2015: deaths by cause, age, sex, by country and by region, 2000–2015. World Health Organization, Geneva2017

[3] B. Gallagher, H. Akatsuka, and H. Suzuki, “Wireless Communications for Vehicle Safety: Radio Link Performance andWireless Connectivity Methods,” *IEEE Vehicular Technology Magazine*, vol. 1, no. 4, pp. 4–24, Dec. 2006.

[4] G. Bansal , J. Kenney , and A. Weinfi eld : “ Cross - Validation of DSRC Radio Testbed and NS - 2 Simulation Platform for Vehicular Safety Communications , ” IEEE Vehicular Technology Conference (VTC Fall), San Francisco, CA, 2011 .

[5] G. Sklivanitis, A. Gannon, S. N. Batalama, and D. A. Pados, “AddressingNext-Generation Wireless Challenges with Commercial Software-DefinedRadio Platforms,” IEEE Communications Magazine, vol. 54, no. 1, 59–67,Jan. 2016. DOI: 10.1109/mcom.2016.7378427.

[6] L. Cheng , B. E. Henty , R. Cooper , D. D. Stancil , and F. Bai : “ Multi - Path Propagation Measurements for Vehicular Networks at 5.9 GHz , ” IEEE Wireless Communications and Networking Conference (WCNC), pp. 1239 – 1244 , Las Vegas, NV, 2008 .

[7] M. Gast : 802.11 Wireless Networks: The Defi nitive Guide , 2nd edition , O ’ Reilly Media , 2006 .

[8] Lewis, Karen (October 17, 2016). "Node-RED visual programming for the Internet of Things (IoT) is now a JS Foundation Project". IBM Internet of Things blog. IBM. Retrieved February 7, 2017.

[9] "OASIS MQ Telemetry Transport (MQTT) Technical Committee". *OASIS*. Retrieved May 9, 2014.

[10] "Web Services Architecture". World Wide Web Consortium. 11 February 2004. Relationship to the World Wide Web and REST Architectures. Retrieved 29 September 2016.

[11] Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST)". Architectural Styles and the Design of Network-based Software Architectures (Ph.D.). University of California, Irvine.

[12] "Yocto Project Linux Kernel Development Manual". www.yoctoproject.org. Retrieved 2018-07-31.

[13] Building Embedded Linux Systems, Karim Yaghmour, Jon Masters, Gilad Ben-Yossef, and Philippe Gerum, second edition, August 2008

[14] Embedded Linux Projects Using Yocto Project Cookbook, Alex González, March 2015

[15] Christian Charreyre. Using Yocto Project to build rich and reliable embedded Linux distributions. Embedded Real Time Software (ERTS'14), Feb 2014

[16] 5G Automotive Association, "The case for cellular V2X for safety and cooperative driving", White Paper, November 2016

[17] Blasco et. al, "3GPP LTE enhancements for V2V and comparison to IEEE 802.11p", 11th ITS European Congress, Glasgow, Scotland, June 2016.