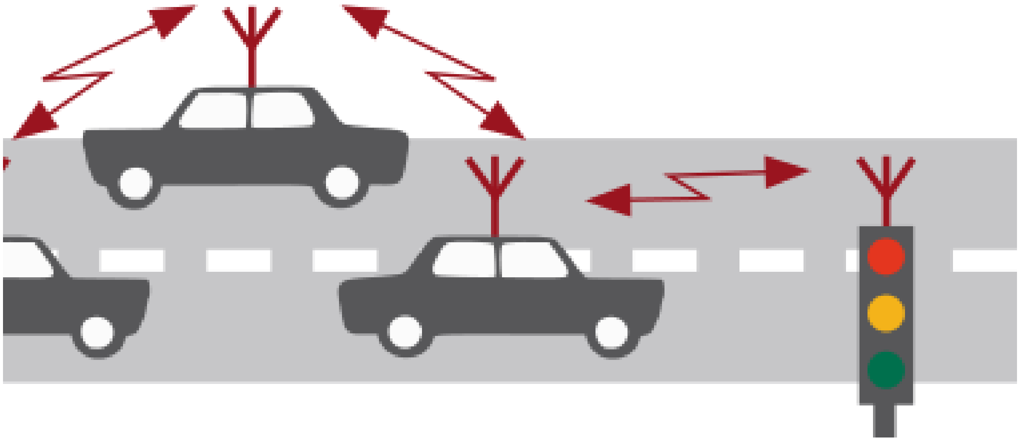

Evaluation of Vehicle to Everything Environments: C-ITS Simulation Package

Rui Miguel Pereira Pacheco

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Alberto Manuel Ramos da Cunha

Examination Committee

Chairperson: Prof. Nuno João Neves Mamede
Supervisor: Prof. Alberto Manuel Ramos da Cunha
Member of the Committee: Prof. Teresa Maria Sá Ferreira Vazão Vasques

June 2019

Acknowledgments

I would like to thank my family for believing in me when doubt came and for providing me with an unbiased and critical environment where my geniality and creativity could flourish without dogma and obtuse knowledge. A special thanks to my parents and grandparents who always provided with valuable life experiences that I'll be sure to carry through my life.

I would like to recognize the open source community as a concept and the people who contribute to it daily. They've made my work possible. A special mention to Raphael Rielb, Christina Obermaier and Christian Facchi from Technische Hochschule Ingolstadt for their support and crucial sharing of knowledge. I would also like to acknowledge my dissertation supervisor Prof. Alberto Cunha for his insight, support and sharing of knowledge that has made this Thesis possible.

Last but not least, to all my friends, I hope I can help you find purpose in your life as you've helped me find my own over and over again throughout the years. To more years to come and shall our friendship flourish even in our last days.

They who have the predisposition and the capacity to hold in their head two opposing ideas at once, and then, without panicking or simply settling for one alternative or the other are able to creatively resolve the tension between those two ideas by generating a new one that contains elements of the others but is superior to both.

Roger Martin

Abstract

In a very near future, intelligent vehicles will interact directly with each other and with the road infrastructure. This integration of information and communication technologies with the transport infrastructure lays the foundations for Cooperative Intelligent Transport Systems (C-ITS) applications. These applications run within each intelligent vehicle and are essential to aided and autonomous driving, endorsing efficient, safe and environmental-friendly transport networks that promote the citizen's quality of life via helping the driver in making more informed decisions based on micro and macro traffic situations.

For C-ITS to become a reality, efforts on standardising the underlying architecture of communications used by C-ITS applications were made by the European Telecommunications Standards Institute (ETSI). These efforts were followed by the European Union's Platform for the Deployment of C-ITS (C-ITS Platform) that presented an application bundle expected to be available on future roads in the short term.

In this document, we deploy and enhance a realistic and scalable simulation tool stack with a C-ITS application bundle compliant with ETSI standards.

Keywords

C-ITS; VANET; C-ITS applications; Day 1 Services.

Resumo

Num futuro muito próximo, veículos inteligentes irão interagir diretamente entre si e com a infra-estrutura rodoviária. Esta integração de tecnologias de informação e comunicação com a infra-estrutura de transporte estabelece as bases para as aplicações dos Sistemas de Transporte Inteligente Cooperativos (C-ITS). Essas aplicações são executadas dentro de cada veículo inteligente e são essenciais para a condução assistida e autônoma, apoiando redes de transporte eficientes, seguras e amigas do ambiente que promovem a qualidade de vida dos cidadãos, ajudando o condutor a tomar decisões mais informadas com base em situações de micro e macro tráfego.

Para que o C-ITS se torne uma realidade, os esforços para padronizar a arquitetura subjacente das comunicações usadas pelos aplicativos C-ITS foram feitos pelo Instituto Europeu de Padrões de Telecomunicações (ETSI). Esses esforços foram seguidos pela Plataforma da União Europeia para a Implantação do C-ITS (Plataforma C-ITS), que apresentou um pacote de aplicativos que deverá estar disponível nas futuras estradas num curto prazo.

Neste documento, implantamos e aprimoramos uma parafernália de ferramentas de simulação realista e escalável com um pacote de aplicativos C-ITS em conformidade com os padrões ETSI.

Palavras Chave

C-ITS; VANET; Aplicações C-ITS; Serviços de Dia 1.

Contents

List of Figures

List of Tables

Listings

Acronyms

ASN.1	Abstract Syntax Notation One
API	Application Programming Interface
C-ITS	Cooperative Intelligent Transport System
CAM	Cooperative Awareness Message
CPU	Central Processing Unit
DENM	Decentralized Environmental Notification Message
ETSI	European Telecommunications Standards Institute
GLOSA	Green Light Optimal Speed Advisory
GPU	Graphics Processing Unit
GUI	Guided User Interface
IDE	Integrated Development Environment
OS	Operating System
OSI	Open System Interconnection
OMNeT++	Objective Modular Network Tested in C++
RAM	Random Access Memory
RSU	Road Side Unit
SUMO	Simulation of Urban Mobility
SSD	Solid State Drive
TCD	Test Case Document

V2I	Vehicle-to-Infrastructure
V2P	Vehicle-to-Pedestrian
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
IST	Instituto Superior Tecnico

1

Introduction

Contents

1.1 Motivation

Smart cities are already being shaped up today and there have been major efforts to empower with intelligence the road infrastructure and its users. A new trend has appeared where a car is more than a tool, capable of actively aiding its driver mitigate some risks related to human interaction and increasing road safety for everyone.

By empowering a vehicle with the intelligence and technology to not only make informed decisions based on the data collected through its sensors, but also by information which could be gathered by communicating with the road infrastructure and other cars driving on it, we're empowering it's decision capabilities. Tomorrow's car will represent a step change in form and function, combining the intelligence of driver-assisted vehicles with the potential of vehicular communications.

This vision has led to the advent of Cooperative Intelligent Transportation Systems (Cooperative Intelligent Transport System (C-ITS)) which rely on vehicular communication technologies to enable applications that could potentially improve road safety, traffic efficiency, and introduce new entertainment and business models.

As these systems will be dealing with road safety, thorough validation has to be made, but due to the multidisciplinary areas involved, real world C-ITS testing is complex and resource intensive. This is where computer generated simulations shine, reducing simulation costs and complexity while offering a realistically simulated controlled environment.

1.2 Research Aims and Objectives

The Departamento de Engenharia Informática (DEI) of Instituto Superior Técnico, in Lisbon, is studying the development of a small scale and controlled environment where C-ITS applications can be tested in real world. To enhance that initiative, this research aims to lay the foundations for research on C-ITS enabled environments by using a computer simulation package before investing in a real-world endeavour.

We will combine a vehicular network simulator (Objective Modular Network Tested in C++ (OMNeT++)) with a road simulator (Simulation of Urban Mobility (SUMO)) into simulation tool stack aiming to realistically emulate real-world scenarios using the European Telecommunications Standards Institute (ETSI) standardised architecture & protocol framework for vehicle-to-everything (V2X) communication. We will also implement the Day 1 Services bundle which is composed by a set of C-ITS applications which, because of their expected benefits and maturity of technology were considered by the European Union's C-ITS Platform as high priority and to be deployed in the short term. The bundle will be tested in scenarios ranging from simple use cases related to micro traffic control, to more complex scenarios where cooperation between multiple entities can be used to solve macro traffic problems.

We aim to build a teaching support tool able to help computer, transportation and mobility engineering students, interested in transports and mobility sector, to learn the organisation of C-ITS systems.

The thesis work is mainly concerned with the implementation and evaluation of a simulation platform able to simulate several important use cases of future intelligent roads. The thesis work is organised in three phases: definition, implementation and evaluation.

The definition phase has two main objectives. The former is the characterisation of a set of messages that applications use to exchange information. The latter is the definition of the Day 1 Services bundle's applications.

The second phase is the implementation which is twofold. Foremost, we will implement the simulation platform and discuss its architecture with detail and afterwards, starting from the previously defined bundle, we will define a set of test case documents which are composed by scenarios and sub-sets of the bundle to be implemented into the simulator.

Lastly, the evaluation phase's main scope, the analysis of the simulation software characteristics and the test case document's implementation analysis.

1.3 Organisation of the Document

This thesis is is organised as follows:

Chapter 1

- Chapter 2 - **State of the Art** description of the current methodologies, tools and standards adopted by the C-ITS sector
- Chapter 3 - **Simulation Platform Architecture** description of the chosen simulation stack and its development state
- Chapter 4 - **Using the Simulation Platform** description of relevant C-ITS features provided by the software package, enriched by detailed examples on how to utilise them
- Chapter 5 - **Evaluation** presents the chosen evaluation scenarios and applications, details the test case documents for each and the gathered results and showcases an analysis on the scalability of the chosen software
- Chapter 6 - **Conclusions & Future Work** finalises the thesis organising our findings related to the simulation stack, how it can contribute to the C-ITS research community and some future work ideas building on what has been accomplished.

2

State of the Art

Contents

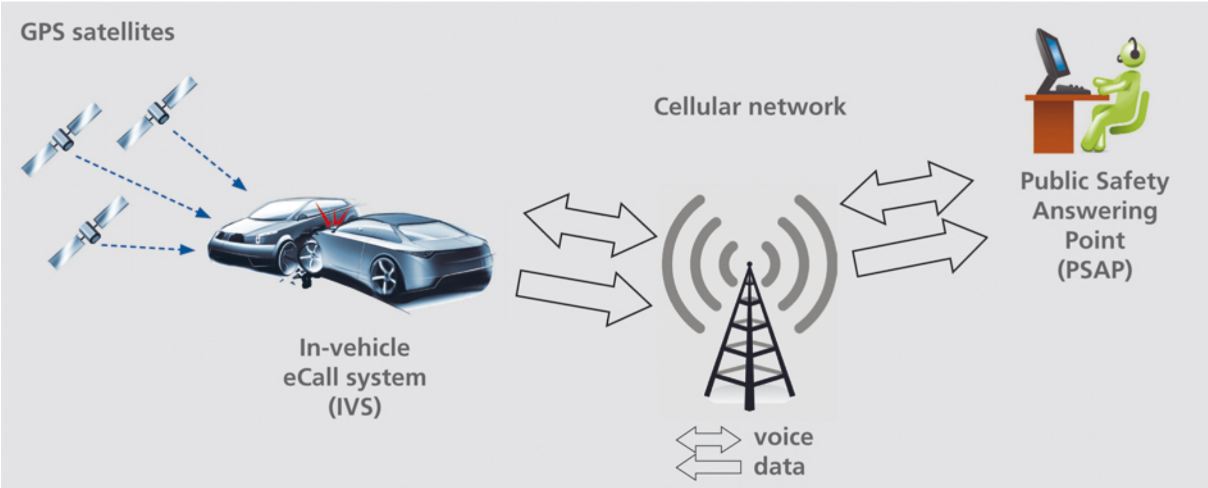

Figure 2.1: eCall architecture [1]

2.1 Intelligent Transport System

An ITS system is made up by isolated components capable of gathering, via their sensors, useful information for aiding traffic management by enabling road users to make a safer and smarter use of transport networks.

As of March 2018 onward, vehicles sold within the European Union must be compliant with an ITS application named eCall [12]. This isolated ITS component automatically establishes an emergency call whenever a dangerous incident occurs - i.e. car crash, burst tire in the highway. This voice call also carries the location of the incident to a trained operator so as to provide adequate assistance.

2.2 Cooperative Intelligent Transport System

Cooperative Intelligent Transport Systems are made up by empowering ITS components with standardised interaction capabilities, allowing road vehicles to cooperatively communicate with other vehicles, traffic signals and road infrastructure as well as with other road users.

With the increased information available, these systems can potentially decrease road fatalities, improve the capacity of roads, diminish the carbon footprint of road transport and enhance the user experience during travels by enabling road users and traffic managers to share and coordinate their actions [13].

According to the EU's C-ITS's 2016 report [14]: "Given the expected benefits and considering the overall relatively moderated costs linked to deployment, there is a strong interest in enabling a fast move at European scale that will translate into market production and early deployment."

Figure 2.2: Cooperative Intelligent Transport System [2]

2.3 C-ITS Architecture

Each agent in a C-ITS system is an ITS station (ITS-S) able to sense its surroundings via the built-in sensors. The ability to gather information on its surroundings has paved way to cooperative applications where two or more ITS stations can exchange messages depicting relevant events.

The interconnection between two ITS stations can be achieved through the usage of Vehicle-to-Everything (V2X) technologies which enable standardised message exchange. According to the involved entities, those technologies are classified as follows:

- **Vehicle-to-Vehicle (V2V):** interconnection used by 2 or more vehicles form a vehicular ad hoc network (VANET) that they use to communicate.
- **Vehicle-to-Infrastructure (V2I):** interconnection used by vehicles to communicate with roadside units (RSU). The opposite communication is defined as Infrastructure-to-Vehicle (I2V).
- **Vehicle-to-Pedestrian (V2P)):** interconnection between vehicles and pedestrians via their personal ITS stations and vice versa.

Due to the diverse potential areas of business in C-ITS systems, many competing stakeholders such as car manufacturers, telecom operators as well as road infrastructure managers and owners are interested in its development. Thus, a common framework of standards and protocols was needed to interconnect different stakeholder's products and adequate C-ITS scalability.

For this reason, protocol architectures have been developed by USA, Japan and EU to support C-ITS applications [15]. This project focuses on following the European Standards Organisation (ETSI) guidelines [3] for global use in C-ITS protocol and architecture standards. This decision was made as it is the most popular protocol within the research community.

Thus, the proposed reference protocol stack in ITS stations (ITS-S) is on figure 2.3.

The reference architecture is inspired by the Open System Interconnection (OSI) layered model and is made up by the following layers:

- **Access** is responsible for the transmission of digital data bits from the the sending (source) ITS station over network communications to the receiving (destination) ITS station.
- **Networking & Transport** is responsible for routing the message over the network so it reaches its desired destination and it also implements transport protocols suitable for different kinds of application needs.
- **Facilities** provides support to applications allowing for inter-application share of generic functions and data.

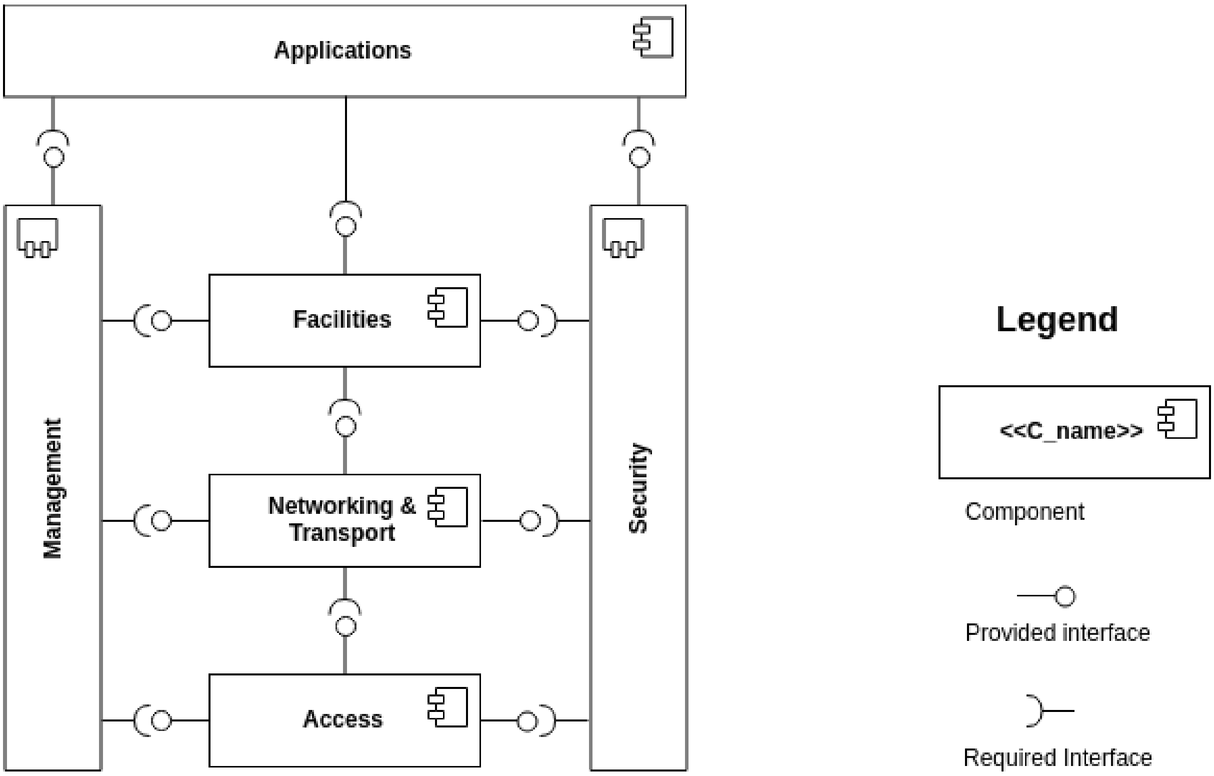

Figure 2.3: ETSI ITS-S architecture

- **Security** enables secure communication within each ITS station and between either two.
- **Management** deals with all communications within each ITS station modules.
- **Applications** enables the communication between C-ITS applications through the usage of the aforementioned services.

2.4 ETSI Message Set

As described in section 2.3, the European Telecommunications Standards Institute (ETSI) has proposed [16] a middleware layer (called facilities) responsible for supporting common communication requirements of many ITS services. These requirements are Periodic Status Exchange (PSE) and Asynchronous Notifications (AN) [17]. As such, the facilities layer defines types of messages exchanged within the cooperative environment that address each of the requirements, respectively: Cooperative Awareness Message (CAM) (Cooperative Awareness Message) and Decentralized Environmental Notification Message (DENM) (Distributed Environmental Notification Message).

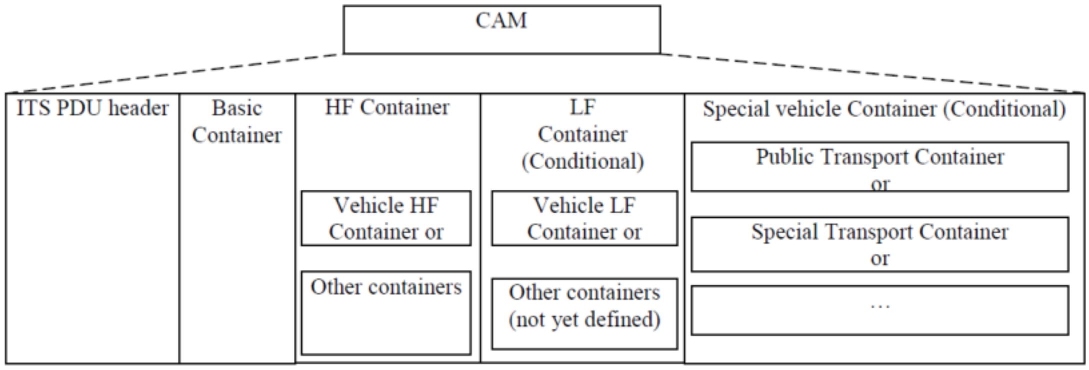

Figure 2.4: General structure of a CAM [3]

2.4.1 Cooperative Awareness Message (CAM)

CAM messages are a kind of heartbeat messages periodically broadcasted by each ITS-S to its neighbours to create and maintain awareness of ITS-S's characteristics in the vicinity. They contain status and attribute information of the originating ITS-S such as its type, position, velocity and acceleration [18]. The details of the message structure is depicted in figure 2.4 and described as follows:

ITS PDU header: contains protocol version, C-ITS station id, message id and a time-stamp from when it was sent.

Basic Container: contains C-ITS station type (vehicle or RSU) and the geographic position.

High Frequency Container: contains fast-changing (dynamic) status information of the vehicle (speed, heading, etc).

Low Frequency Container: contains static or slow-changing station data (e.g. lighting conditions)

Special Vehicle Container: provides further status information for special vehicles (e.g. rescue vehicles like a police car of ambulance).

2.4.2 Decentralised Environmental Notification Message (DENM)

DENM messages are event-triggered messages broadcasted to alert road users of a hazardous event. These messages contain descriptive information about the signalled event such as its type, location, cause and according to the specific situation, they can be re-transmitted multiple times while the event is relevant [18].

As per figure 2.5, the message structure is composed by:

ITS PDU header: contains protocol version, C-ITS station id, message id and a time-stamp from when it was sent.

Management Container: contains basic information about the signalled event (position, time-stamp,

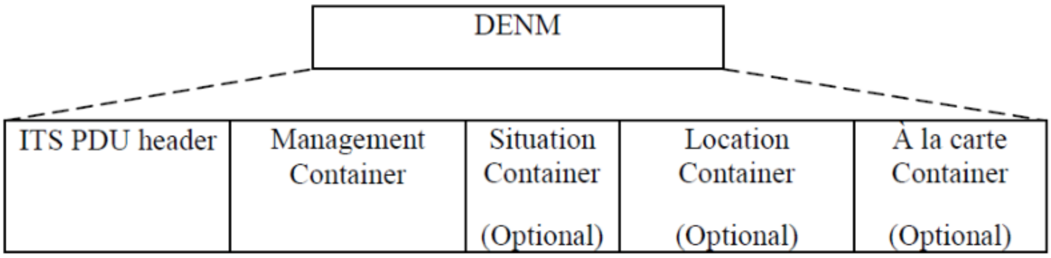

Figure 2.5: General structure of a DENM [3]

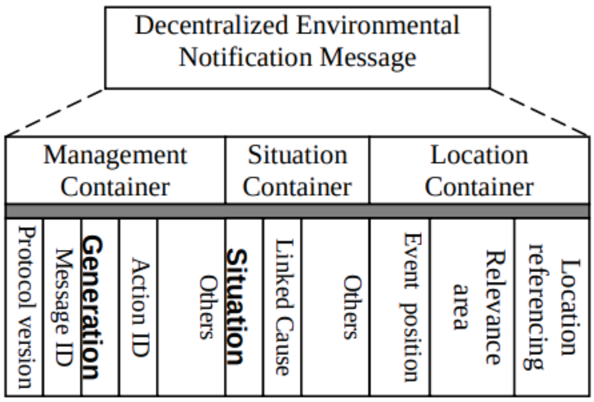

Figure 2.6: DENM container's properties [3]

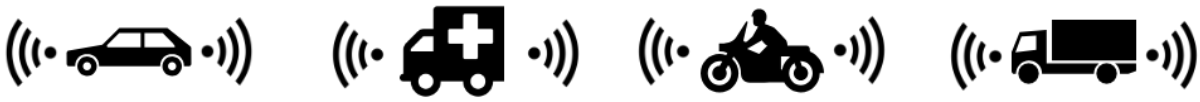

Figure 2.7: Vehicle Subsystem examples, left to right: Car (regular vehicle), Ambulance (emergency vehicle), Motorcycle (regular vehicle), Truck (heavy vehicle)

etc).

Situation Container: contains further information that describes the detected event (event type, cause code, etc).

Location Container: contains the location of the relevant event.

À la carte Container: contains additional customisable information useful for application specific logic.

2.5 Cooperative Environment

According to ETSI guidelines [3], the cooperative environment housing a C-ITS system is made up by the following ITS sub-systems:

- Vehicle sub-system
- Roadside sub-system
- Personal sub-system
- Central sub-system

Each of these sub-systems contains ITS stations of its domain. Some ITS stations, according to their functional requirements, do not employ the full C-ITS architecture [19].

2.5.1 Vehicle sub-system

This domain is made up by motorised road vehicles capable of providing the full-stack C-ITS architecture, thus enabling their interaction with other ITS stations.

2.5.2 Roadside sub-system

This domain is composed by all Road Side Unit (RSU) capable of providing the full-stack C-ITS architecture, thus enabling their interaction with other ITS stations. They also include an ITS-station (ITS-S) border router which enables its communication with the central sub-system via a different protocol stack.

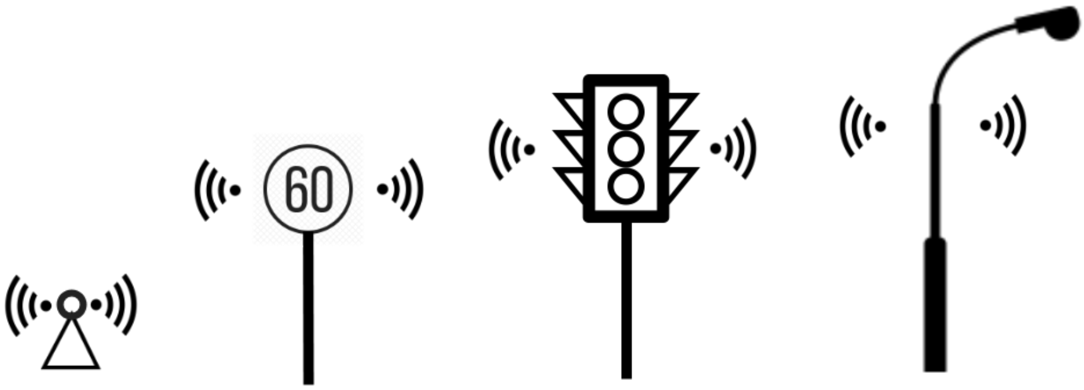

Figure 2.8: Roadside Subsystem examples, left to right: RSU, Speed Limit Sign RSU, Traffic Light RSU, Lamp Post RSU

Figure 2.9: Personal Subsystem: Person carrying a mobile phone (personal ITS-S).

2.5.3 Personal sub-system

This domain is made up by all hand-held devices capable of providing the application and communication functionality required to interact with ITS station. They are called Personal ITS stations and they can be PDA's, mobile phones, etc.

2.5.4 Central sub-system

This domain is made up by all systems which gather information from C-ITS environment communications and are designed to be used by a central C-ITS entity enabling macro traffic management.

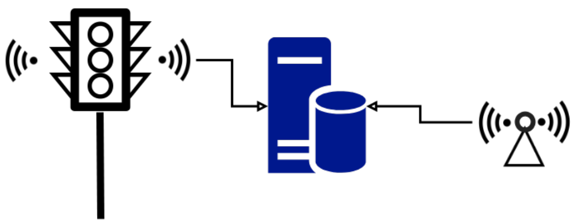

Figure 2.10: Central Subsystem architecture: RSU's communicating with central traffic management

2.6 C-ITS Application Classes

A cooperative environment can only be so useful as the applications which rely on it to provide increased intelligence to road users. This cooperative environment is made possible by C-ITS applications running within every ITS-S.

According to ETSI guidelines [20], C-ITS applications are classified as one of four application classes [14]: Road Safety, Traffic Efficiency, Local Services and Internet Services. Dependent on how much these applications rely on the communication services, application classes impose more or less strict requirements with respect to reliability, security, latency, and other performance parameters. Any application belongs to one application class and, within that class, to one group (if applicable).

2.6.1 Road Safety Class

They are driving assistance applications that handle emergency situations, they can be divided into 2 sub-groups:

- **Cooperative awareness** aims to cooperatively act on micro traffic management - related to individual driving decisions.

- **Road hazard warning** helps the driver sense the surrounding environment and alert him to possible dangers.

2.6.2 Traffic Efficiency Class

They are traffic management applications that improve the capacity and utilisation of road infrastructure. They can be divided into 2 sub-groups.

- **Speed management** aids the driver in choosing the best cruising speed.

- **Cooperative navigation** aims to cooperatively act on macro traffic management - related to driving decisions made by masses.

2.6.3 Local and Internet Services Class

They advertise and provide on-demand information to passing vehicles on either a commercial or non-commercial basis. Local services are provided within the C-ITS network infrastructure and Internet services are acquired from providers in the world wide web.

Table 2.1: Day 1 Services Bundle Classification

Day 1 Services		
Class	Group	Name
Road Safety	Road Hazard Warning	Electronic emergency brake light
		Weather conditions
		Slow or stationary vehicle(s)
		Traffic jam ahead
		Hazardous location
		Road works
		Signal violation intersection safety
	Cooperative Awareness	Emergency vehicle approaching
		In-vehicle signage
Traffic Efficiency	Speed Management	In-vehicle speed limits
		Green light optimal speed advisory (GLOSA)
		Shock wave damping
	Cooperative Navigation	Traffic signal priority request by designated vehicles
		Probe vehicle data

2.7 C-ITS Platform

The Platform for the Deployment of Cooperative Intelligent Transport Systems in the European Union (C-ITS Platform) was created by the European Commission services (DG MOVE) in November 2014 with the intention of strategically defining investment plans that stimulate the emergence of business models, foster interoperability between stakeholders and discuss public-private stakeholder cooperation.

The first phase gathered public and private stakeholders, representing all of the key stakeholders along the value chain including public authorities, vehicle manufacturers, suppliers, service providers, telecommunications companies, etc.

Its final product was a report [14] about a shared vision on the inter-operable deployment of Cooperative Intelligent Transport Systems in the European Union. This report included the common technical framework necessary for the deployment of C-ITS, within whom, a list of Day 1 Services bundle which, because of their expected societal benefits and the maturity of technology, are expected to be available in the short term. The bundled services are classified as depicted in Table 2.1.

Additionally, ETSI has defined specific requirements [21] related to some applications included on the Day 1 Services bundle, those requirements are depicted in Table 2.2.

Table 2.2: ETSI Day 1 Services requirements

Name	V2X Technology (V2V, V2I/I2V, V2P)	Critical Actuation Time	Minimum Message Broadcast Frequency (1Hz = 1 message/second)
Electronic emergency brake light	V2V	100ms	10Hz
Slow or stationary vehicle(s)	V2V + V2I + I2V	100ms	10Hz
Hazardous location	V2I + I2V	n.d.	10Hz
Road works	I2V	100ms	2Hz
Signal violation intersection safety	I2V	100ms	10Hz
Emergency vehicle approaching	V2V + V2I + I2V (authenticated messages)	100ms	10Hz
In-vehicle signage	I2V	500ms	1Hz
In-vehicle speed limits	I2V	n.d.	1-10Hz

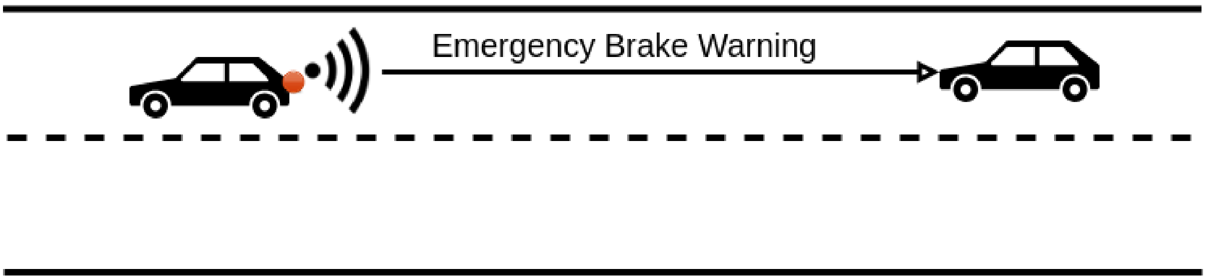

Figure 2.11: Electronic Emergency Brake Light usage example

2.7.1 Road Safety Services - Road Hazard Warning

- **Electronic Emergency Brake Light**

  Enhances the safety in a dense driving environment by warning the driver of a hard braking event in front. It aims to avoid rear end collisions which can occur if a vehicle driving ahead suddenly brakes on highways, especially in dense driving situations or in situations where the driver's view is obstructed by other vehicles or bad weather conditions. The driver will be warned before realising the vehicle ahead is braking hard, especially if there are vehicles in between. It does so by broadcasting a self-generated emergency brake event to surrounding vehicles (V2V) and surrounding infrastructure (V2I).

- **Weather Conditions**

  Warns the driver about adverse weather conditions such as heavy rain, hail or winds. This is especially useful to motorcycles which are more susceptible to strong winds [22]. In this application, weather stations spread around the road communicate with the central sub-system which then relays important weather conditions to geographically interested roadside infrastructure that

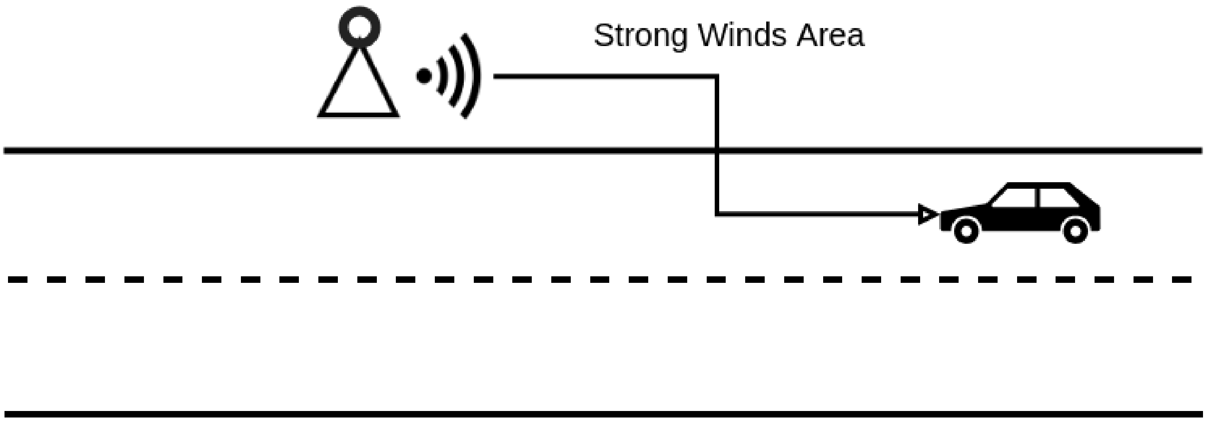

Figure 2.12: Weather Conditions example scenario

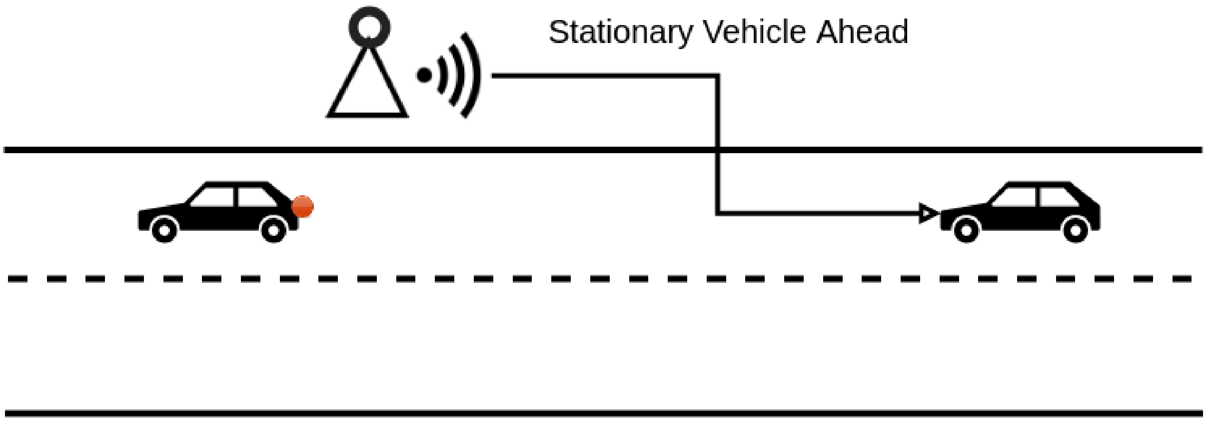

Figure 2.13: Example stationary vehicle ahead scenario

broadcast it to nearby vehicles (I2V). Another implementation could be envisioned where the vehicle's sensors are able to gather relevant weather information, relay it to RSUs via V2I and then the RSUs themselves could broadcast to nearby vehicles via I2V.

- **Slow of Stationary Vehicle(s)**

  Informs the driver to the presence of a slow vehicle in a given road via I2V. Especially useful in mountain roads with blind corners where the line of sight of the driver is severely handicapped. This application could, in conjunction with a routing application running within the vehicle, improvement the traffic fluidity by encouraging another itinerary if possible.

- **Traffic Jam Ahead**

  Informs the driver to the presence of a traffic jam in a given road via I2V. This application could, in conjunction with a routing application running within the vehicle, improvement the traffic fluidity by encouraging another itinerary if possible, much similarly to what is done in navigation applications that take into account traffic jams like WAZE [23] and Google Maps [24].

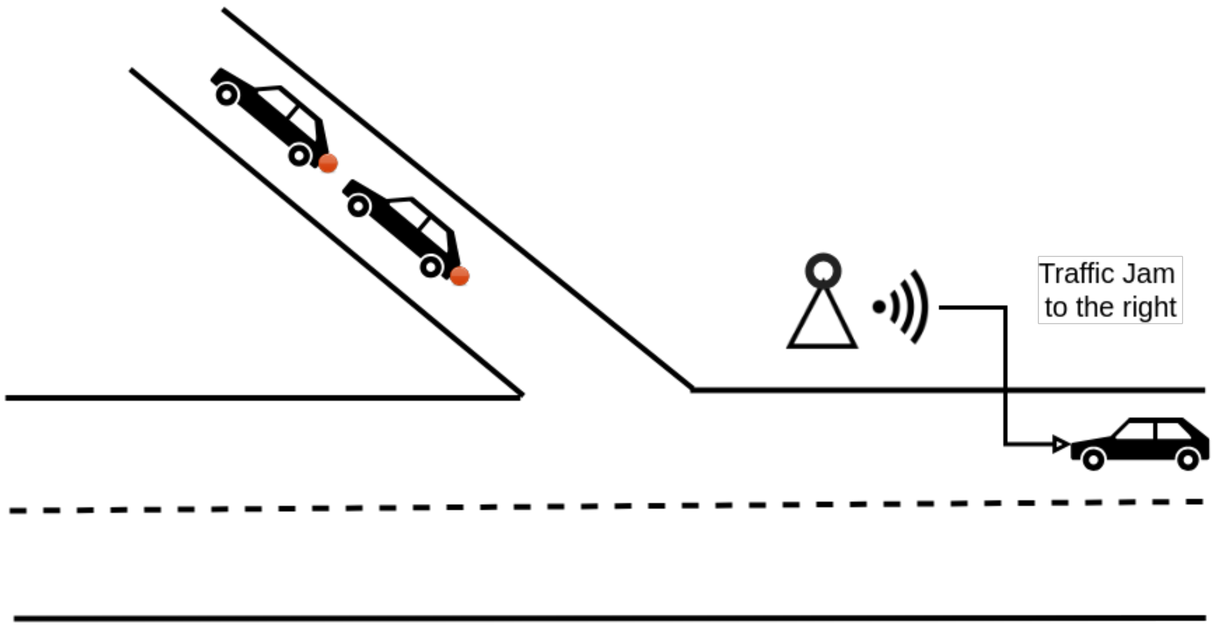

Figure 2.14: Traffic Jam Ahead example scenario

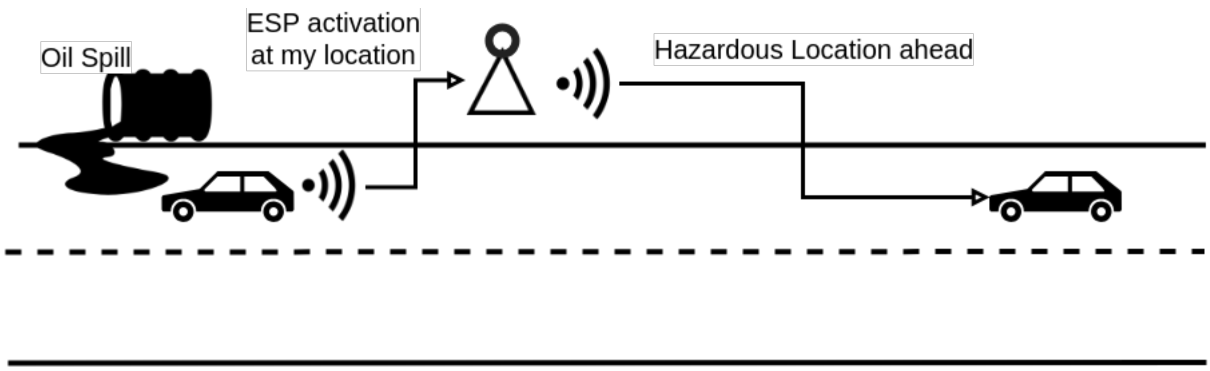

Figure 2.15: Hazardous Location example scenario

- **Hazardous Location**

  A vehicle detects a hazardous location by analysing available in-vehicle sensor information such as the vehicle Electronic Stability Program (ESP) which can detect slippery spots on the road [25]. The vehicle can then broadcast this information on a hazardous location to its environment via V2I and V2V. The receiving RSUs relay the area's location to incoming vehicles so they can take preventive measures when going through it.

- **Road Works**

  Construction sites and temporary maintenance working areas are accident black spots, because static traffic signs are ignored or realised too late. To augment security, a Road Works Warning message is sent by a road works trailer (a movable RSU) via I2V which depicts relevant information

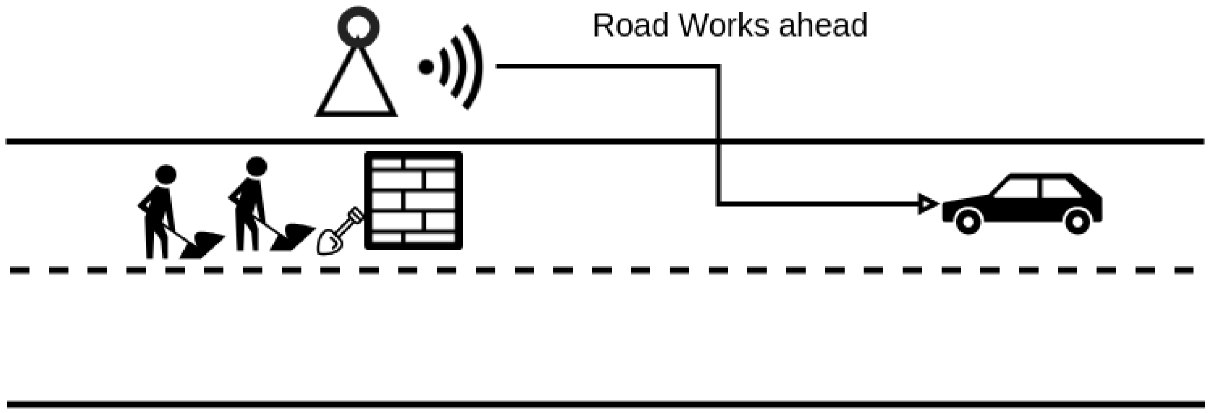

Figure 2.16: Road Works example scenario

about temporary road policy such as temporary maximum speed, affected lanes, etc.

- **Signal Violation Intersection Safety**

  A detecting ITS station (RSU) signals to affected road users (I2V) that a vehicle has violated a road signal, thus there is an increased risk of an accident.

2.7.2 Road Safety Services - Cooperative Awareness

- **Emergency Vehicle Approaching**

  Allows an active emergency vehicle to indicate its presence so vehicles in its path can give way and free an emergency corridor. Especially useful in urban traffic situations where the emergency vehicle must go through a high traffic road junction.

- **In-vehicle Signage**

  Information on current valid traffic signs is broadcasted by nearby RSUs via (I2V) and given to the driver. This application aims to solve one major challenge related to autonomous driving that is gathering valid traffic signage via image processing, which has its limitations, namely whenever the signage is degraded or its visibility is obstructed by taller vehicles. As an example, traffic information like overtaking not permitted in the next 100m can be used by autonomous driving applications in the orchestration of their driving plan.

2.7.3 Traffic Efficiency Services - Speed Management

- **In-vehicle Speed Limits**

  A capable Road Side Unit broadcasts the current local speed limits (regulatory and contextual) to nearby vehicles (I2V). Its main goal is to improve road safety.

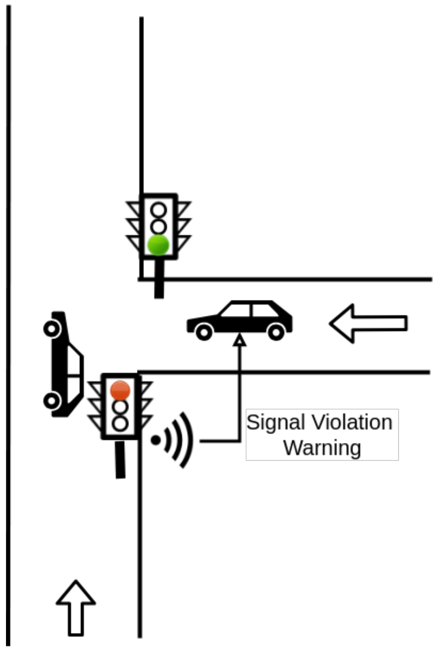

Figure 2.17: Signal Violation Intersection Safety example scenario

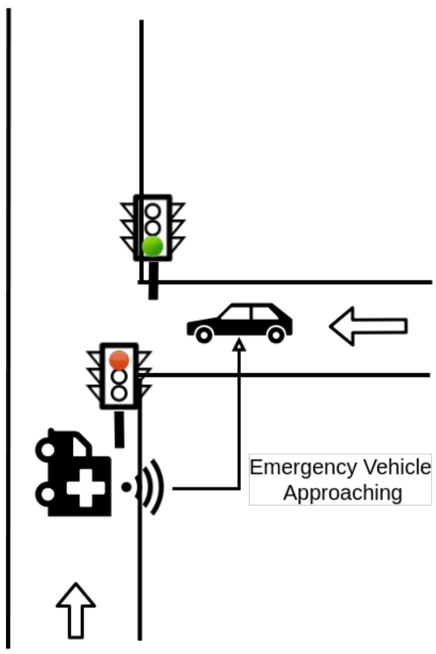

Figure 2.18: Emergency Vehicle Approaching example scenario

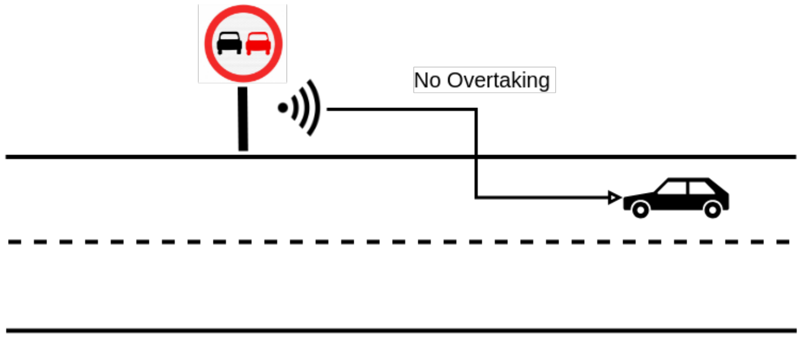

Figure 2.19: In-vehicle Signage example scenario

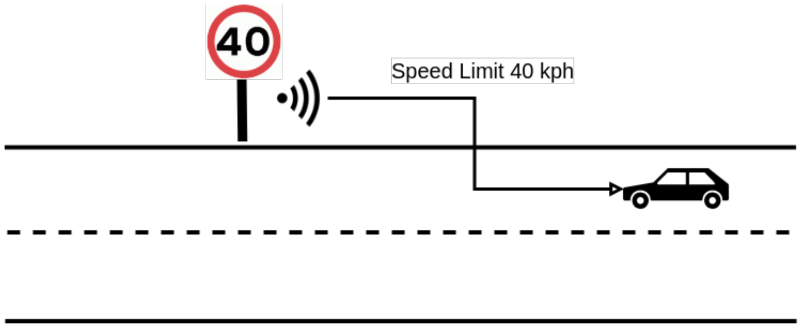

Figure 2.20: In-vehicle Speed Limits example scenario

- **Green light Optimal Speed Advisory (GLOSA)**

  Green Light Optimal Speed Advisory (GLOSA) systems provide drivers with optimal speed advisory that allow them to pass traffic lights during green interval. Two approaches exist, in a single-segment approach, the system provides vehicles with the optimal speed for the segment ahead of the nearest traffic signals. In a multi-segment approach, several signals in a sequence on a vehicle's route are taken into account. The application runs within the RSUs and the traffic management system (central sub-system). RSU's broadcast (I2V) the speed advisory for a given zone right before a traffic light for single-segment approach. For multi-segment approach, RSUs broadcast (I2V) the speed advisory upon individual vehicle route reception (V2I), as this approach considers individual vehicle route.

  An implementation of GLOSA was made [26] in VSimRTI showcasing an 80% reduction in stop time (where the car is still) and up to 7% reduction in fuel consumption in a high traffic scenario.

- **Shock Wave Damping**

  Perturbations in dense traffic (e.g. sudden breaking/acceleration, lane-switching) generate spreading shock waves that lead to long-term disruption of traffic flow.

  Shock Wave Damping can reduce the harmonisation of traffic flows by preventing the formation of shock waves and damping their development. To do so, RSU's within a given zone gather

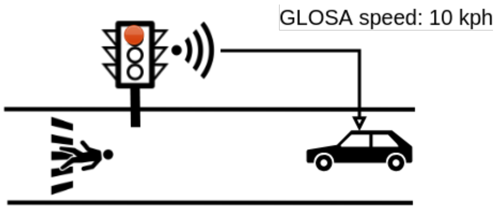

Figure 2.21: GLOSA example scenario

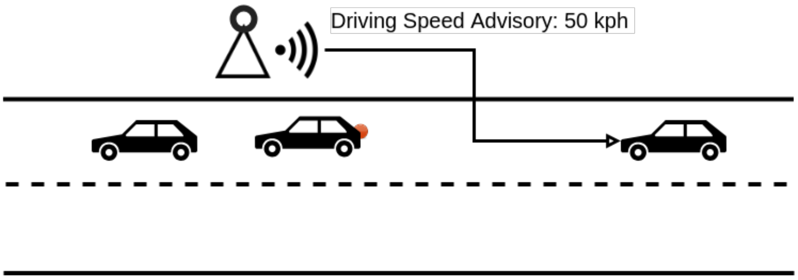

Figure 2.22: Shock Wave Damping example scenario

traffic-related information (V2I) specific to that zone, relay it to the traffic management infrastructure (central sub-system) which then uses that information to calculate an optimal driving speed advisory. The calculated speed advisory is then broadcasted via I2V and given to the driver.

2.7.4 Traffic Efficiency Services - Cooperative Navigation

- **Traffic Signal Priority Request by Designated Vehicles**

  An emergency vehicle announces its location and desired itinerary to road traffic regulatory RSUs (traffic signals) via V2I. The traffic management infrastructure (central sub-system) makes arrangements to the traffic regulatory signage to prioritise the emergency vehicle's itinerary.

- **Probe Vehicle Data**

  Vehicles continuously broadcast Cooperative Awareness Messages (CAM) depicting their location, speed, weather conditions, etc. These messages are received by RSUs (V2I) which can relay them to the traffic management infrastructure (central sub-system), allowing for a real-time gathering of road data statistics and enabling for dynamic adaptability of road infrastructure. For instance, dynamic traffic light control requires traffic detection to function. Without detectors, only fixed time control of green/yellow/red light phases is possible. By monitoring an intersection approach continuously the C-ITS system is aware of approaching vehicles as they frequently transmit CAM messages, thus making arrangements to optimise traffic light phases.

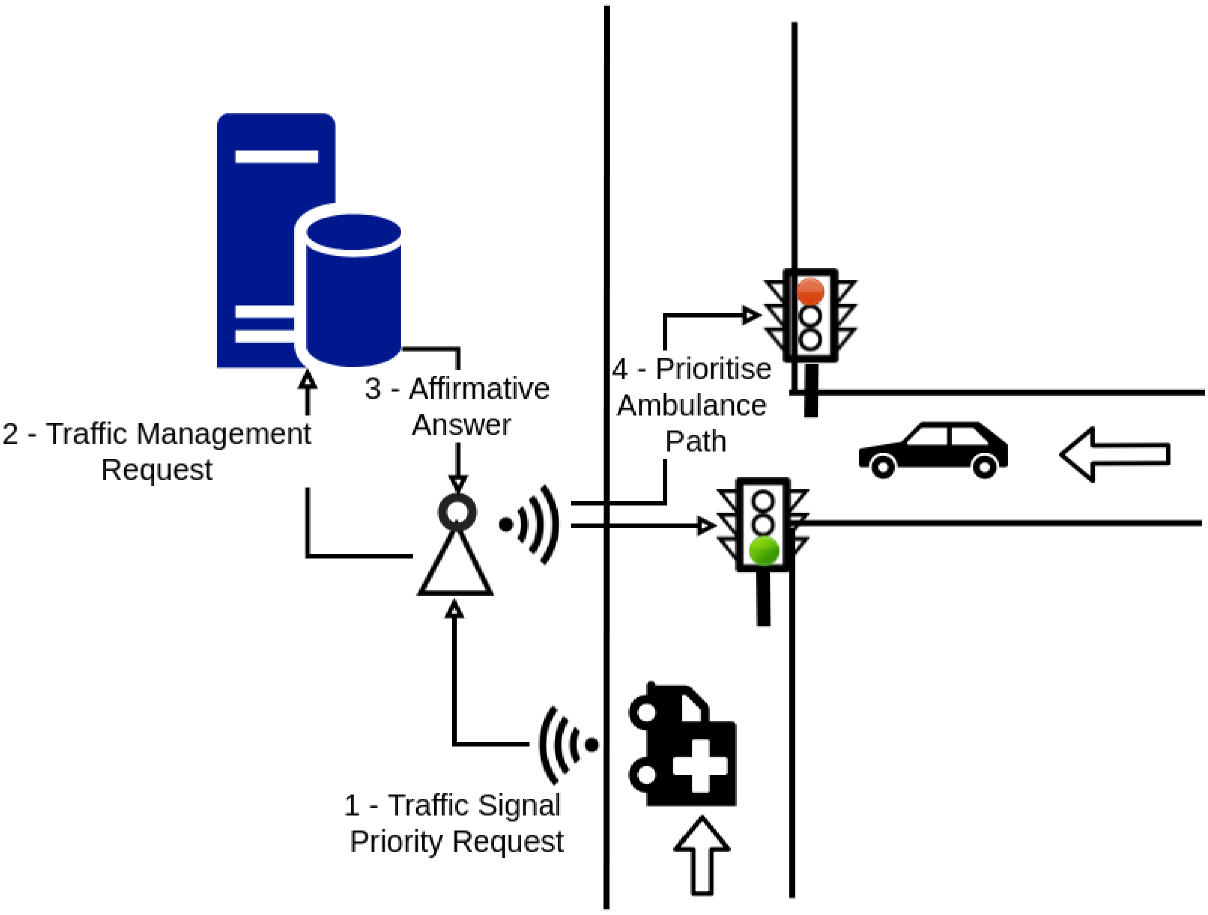

Figure 2.23: Traffic Signal Priority Request by Designated Vehicles example scenario

2.8 Cooperative Vehicular Simulation

Simulator tools have been preferred over outdoor experiment as they play a vital role in imitating real world scenarios with reduced costs. They offer complete control over the testing scenario, allowing for macro and micro level analysis.

In this section, we will review various publicly available VANET simulators used by the research community, comparing them to our specific simulation needs. We will only consider *libre software* [27] due to their copyright nature and primarily due to the ability to freely access and modify the simulator's source code.

VANET simulation software can be divided into three different categories, they are (a) vehicular mobility generators, (b) network simulators, and (c) VANET simulators.

2.8.1 Vehicular Mobility Generators

Their aim is to generate realistic vehicular mobility traces used as an input for a network simulator, these traces depict the location of each vehicle at every time instant for the entire simulation time. The inputs of the mobility generator include the road map and specific scenario parameters such as maximum vehicular speed for any given road, the rate of vehicle arrivals and departures, etc.

2.8.2 Network Simulators

Their aim is to dynamically build network communication topologies between two moving nodes based on their location retrieved from the vehicular mobility generator trace. In the course of the simulation, the network simulator dynamically creates ad hoc networks between nodes in range, allowing their inter-connection and exchange of messages. Realistic network simulators are especially useful to evaluate different physical level communication standards (e.g. LTE, IEEE 802.11p) as they can simulate wave collisions with structures such as buildings [28].

2.8.3 VANET Simulators

A simulator (or software suite) that utilises mobility and network simulators in conjunction to reproduce a C-ITS environment is called a VANET simulator. Initially, mobility and network simulators were different scientific investigative areas, as such, they were not created with inter-communication in mind, even worse, they were designed to work separately [4]. Fortunately, due to recent interest in VANETS, network simulators became able to load mobility scenarios as depicted in Fig. 2.24. However, they must be loaded prior to the simulation and no modification is allowed after the simulation has begun.

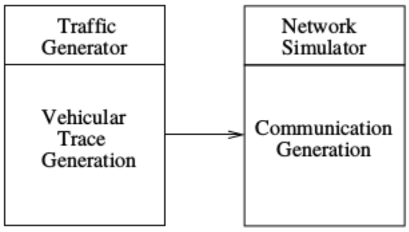

Figure 2.24: Interaction between Network and Traffic Simulators: The Isolated Case [4]

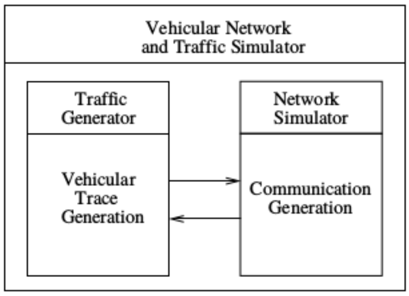

Figure 2.25: Interaction between Network and Traffic Simulators: The Integrated Case [4]

The research community then took a different approach, giving birth to simplistic all-in-one discrete event simulators that closely link vehicular mobility generators with simplistic network simulators where the lack of elaborated standardised protocol stacks was compensated by a native collaboration between the networking and the mobility worlds, as depicted in Fig. 2.25.

Another approach, taken by the research community, was to federate existing network simulators and mobility models through communication interfaces [4], seen in Fig. 2.26.

An article [29] by Oanh Tran Thi Kim *et al* compared all the mentioned VANET simulator architectures with greater detail.

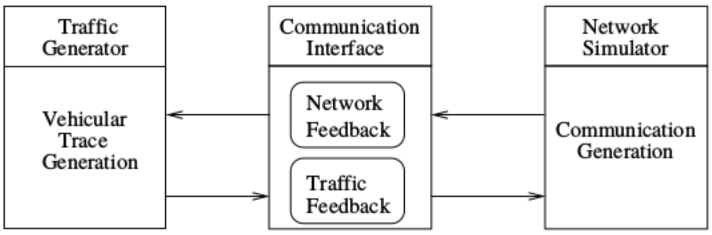

Figure 2.26: Interaction between Network and Traffic Simulators: The Federated Case [4]

3

Simulation Platform Architecture

Contents

3.1 Simulation Requirements

This project's goal is to simulate C-ITS applications where message exchange between two vehicles can alter, for example, their speed whenever there is an hazardous location warning and/or itinerary in the case of a traffic jam ahead message. Therefore, on our VANET simulation, the vehicular and network simulator must be closely linked as the dynamics of traffic could change abruptly. The chosen software suite must support the following requirements:

- Capable of realistic vehicular mobility and V2X communications simulation in urban and motorway scenarios.
- Capable of individual vehicle decision making based on C-ITS applications running within each station, allowing for real-time vehicle parameter tweaking such as speed and route.
- Real-time updated Guided User Interface (GUI) capable of showcasing V2X communications.
- Gathering of statistics related to the simulation such as number of messages exchanged by a certain vehicle with a certain RSU, average speed of a given vehicle, etc.

Additionally, the following features are greatly appreciated:

- Easy to use and setup.
- Rich documentation and examples related to VANET simulation.
- Rich GUI allowing for simulation parameters tweaking.
- Ability to import scenarios including speed limits, lane counts, access and turn restrictions, etc.
- Available implementations of IEEE 802.11p standards [30].
- Available implementations for domain specific models in vehicular environment such as RSUs, ITS-s, traffic light management systems, collision events, etc.

3.2 Evaluation of VANET Simulators

As result of the increased interest by the scientific community in V2X and VANET simulation software, simulation tools that closely link vehicular and network simulators needed for realistic C-ITS application simulations were created.

Taken into consideration various surveys and comparative studies of VANETS simulators [31] [4] [32] [33], the ones that answer the majority of our simulation requirements defined in section 3.1 were:

Table 3.1: VANET Simulators Comparative Overview

	VEINS	VSimRTI	VANETsim	Anylogic
Open Source	yes	no	yes	no
Realistic Vehicle Mobility	yes	yes	yes	yes
Implemented V2X Communication	yes	yes	yes	yes
Individual Vehicle Decision Making	yes	yes	yes	no
Real-Time GUI	yes	yes	yes	yes
Statistics Gathering	yes	no	yes	no
Ease of Use and Setup	very hard	medium	easy	easy
Documentation	medium	good	bad	medium
V2X Examples	bad	good	bad	bad
GUI Parameter Tweaking	no	no	yes	no
Scenario Import	yes	yes	yes	no
IEEE 802.11p Implemented	yes	yes	yes	no
Domain Specific Models in V2X Environment Implemented	yes	yes	yes	no
Active Development	yes	yes	no	yes

- **VEINS** - Vehicles in Network Simulation [34].
- **VSimRTI** - V2X Simulation Runtime Infrastructure [35].
- **VANETsim** - VANET Simulator [36].
- **Anylogic** - agent based simulation and modelling tool [37].

To gauge their ease of use and setup, we've installed and tried to find an existing vehicular example application for each simulator, documenting our findings. Additionally, we've studied the official documentation and quick start guides.

A compilation of our research on the mentioned simulators and their features is depicted in a comparative overview on Table 3.1.

3.3 Simulation Platform Stack

Taking into consideration the shortcomings of the different VANET simulators, the best choice for this project is VEINS as it is the only simulator able to fulfil all the simulation requirements. Thus, the chosen simulation platform can be decomposed into four parts:

OMNeT++ [38] , a network simulator which handles the sending and receiving of packets where INET provides realistic models of the radio medium via IEEE 802.11 physical and link layers. It is used to simulate ITS-G5 channels communication in a Vehicular Ad Hoc Network (VANET).

Vanetza [39] is an open-source implementation of the ETSI C-ITS protocol suite responsible for the routing between participants, featuring:

[ETSI Abstract Syntax Notation One (ASN.1)] message structure and definition (Facilities) such as CAM and DENM are implemented according to the ETSI standard's definition and allow for new alacarte message specification.

[Routing Algorithms], more specifically, topological and geographic C-ITS routing algorithms like the Single Hop Broadcast (SHB) and GeoBroadcast (GBC) are already implemented. To forward such messages, each C-ITS station in the network (represented by a Vanetza Router), based on the chosen routing algorithm, determines the next hop in the network and sends the packet (message) down to the physical layer, provided by INET.

Artery [8] is a VEINS framework that provides the application layer allowing for ETSI Day One applications to cooperatively communicate with each station via middleware facilities. It also implements:

[Storyboard] allows the definition of customisable scenario conditions as well as effects to provoke various traffic scenarios like accidents, weather conditions or traffic jams. Its effects can be specific to a vehicle, a sub-set of vehicles, an area or a time-frame.
As the Periodic Status Exchange (CAM) and Asynchronous Notifications (DENM) require specific triggering scenarios, the storyboard comes into play to provide them.

SUMO [40] provides the traffic simulation in a user-friendly GUI and was designed from scratch to handle large road networks. It seamlessly integrates with OpenStreetMaps and it can import realistic road networks from the world's road infrastructure which come bonded with individual road characteristics such as traffic-light timing, lane count, maximum speed, etc.

An holistic look over the whole architecture is depicted in figure 3.1

3.4 Simulation Platform Development Status

Taking into account the aim of this project is the realisation of a simulation platform for developing and testing C-ITS systems and applications, it is of the utmost importance to be working with an actively developed stack. Additionally, stable versions of each software were preferred as the goal isn't testing grounds but a solid foundation to build upon. The development state of each simulation framework as of

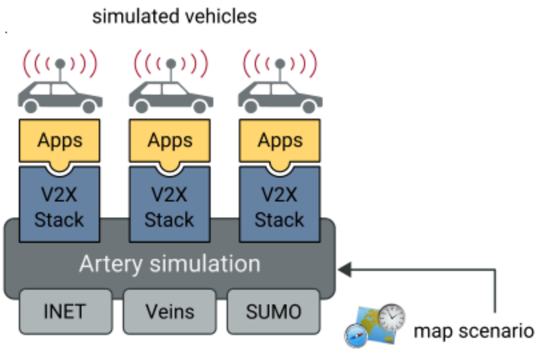

Figure 3.1: Architecture of the solution showcasing the interconnection between the different chosen frameworks, composing the simulation platform [5]

Table 3.2: Simulation Platform Development Software's version, development stage and usage license

SW Name	Used Version	Development Stage	Latest Release	Usage License
Ubuntu	16.04:LTS	Active	February 2019	Mostly GPL 2.0
OMNeT++	5.4.1:Stable	Active	June 2018	GPL 2.0
INET	3.6.5:Stable	Active	January 2019	GPL 2.0
Vanetza	1.0:Latest	Active	February 2019	LGPL 3.0
Artery	1.0:Latest	Active	April 2019	LGPL 3.0
SUMO	0.32.0:Stable	Active	April 2019	GPL 2.0

the writing of this document is compiled in table 3.2.

3.5 Programmes and Support to the community

It is worthy of notice that both OMNeT++'s and SUMO's development teams host annual Community Summits at the Hamburg University of Technology, Germany and at the German Aerospace Center in Berlin, Germany respectively.

Both Vanetza and Artery frameworks are part of ongoing research work at Technische Hochschule Ingolstadt in Ingolstadt, Germany and their respective github communities were very supportive in the realisation of this project.

3.6 Development Environment

All work was compiled and bundeled into a virtual appliance which is a pre-configured virtual machine image ready to run on a hypervisor. It was intended to eliminate the installation, configuration and maintenance costs associated with running complex stacks of software.

The appliance was offloaded onto the most commonly used format (Open Virtualization Format) to

Table 3.3: Laptop technical specifications: ThiccPad t420

Operating System (OS)	Ubuntu 18.04
Kernel	Linux 4.4.19
CPU	i5-2520m
Graphics Processing Unit (GPU)	intel sandybridge
RAM	8 GB
Solid State Drive (SSD)	120 GB Samsung 840

be used on further development. It is pre-configured to use 6 GB of Random Access Memory (RAM) and 100% of Central Processing Unit (CPU) execution cap & all available cores of the host machine it is running on.

The appliance features the complete development stack with a pre-configured Integrated Development Environment (IDE) and build structure.

All development and testing was made on the the appliance whose host was a ThiccPad t420 with technical specifications depicted in table 3.3:

4

Using the Simulation Platform

Contents

This section focuses on the implementation work flow using the chosen stack, specifically on how to deploy a real road map, custom vehicle routing, C-ITS application building and custom scenarios definition using the playbook.

4.1 Importing a Road Map and Traffic Flow

To increase simulation realism, maps of real roads around the world can be easily imported using a script that produces all the needed simulation files from a map that was downloaded from OpenStreetMap [41]. All their maps are licensed under the Open Data Commons Open Database License (ODbL) [42] by the OpenStreetMap Foundation (OSMF) [43] and can, therefore, be used for academic research purposes.

It is worthy of notice that the maps are built by a community effort that contribute and maintain data about roads, trails, railway stations, and much more, all over the world.

The process is divided into 3 steps:

I Select a map area and download the corresponding map, see figure 4.1.

II Run script that generates sample car routes for the road infrastructure and quickly tests the map, example in figure 4.2.

III Customise vehicle route parameters and RSU positions (optional)

4.2 Defining C-ITS Station's Services and Applications

Each station, represented by a Vanetza router, is pre-equipped with a facilities layer: Vehicle Middleware or Stationary Middleware if they are a vehicle or a RSU respectively. But, for stations to be able to run C-ITS applications that exchange messages, they require an ItsG5BaseService to be running in each which in turn utilise the available middleware. For this purpose, Extensible Markup Language (XML) files are used to specify what services and applications will be running in each station.

Already implemented in the Artery Framework are CAM and DENM services (based off of ItsG5BaseService) that support applications which, in turn, require synchronous and asynchronous message dissemination respectively. Due to the versatility of implementation, future services can be easily added. Depicted in listing 4.1 is an example featuring the definition of C-ITS station running a DENM Service on port 2002 and in listing 4.2, an application running within that'll be using the previously defined DENM service.

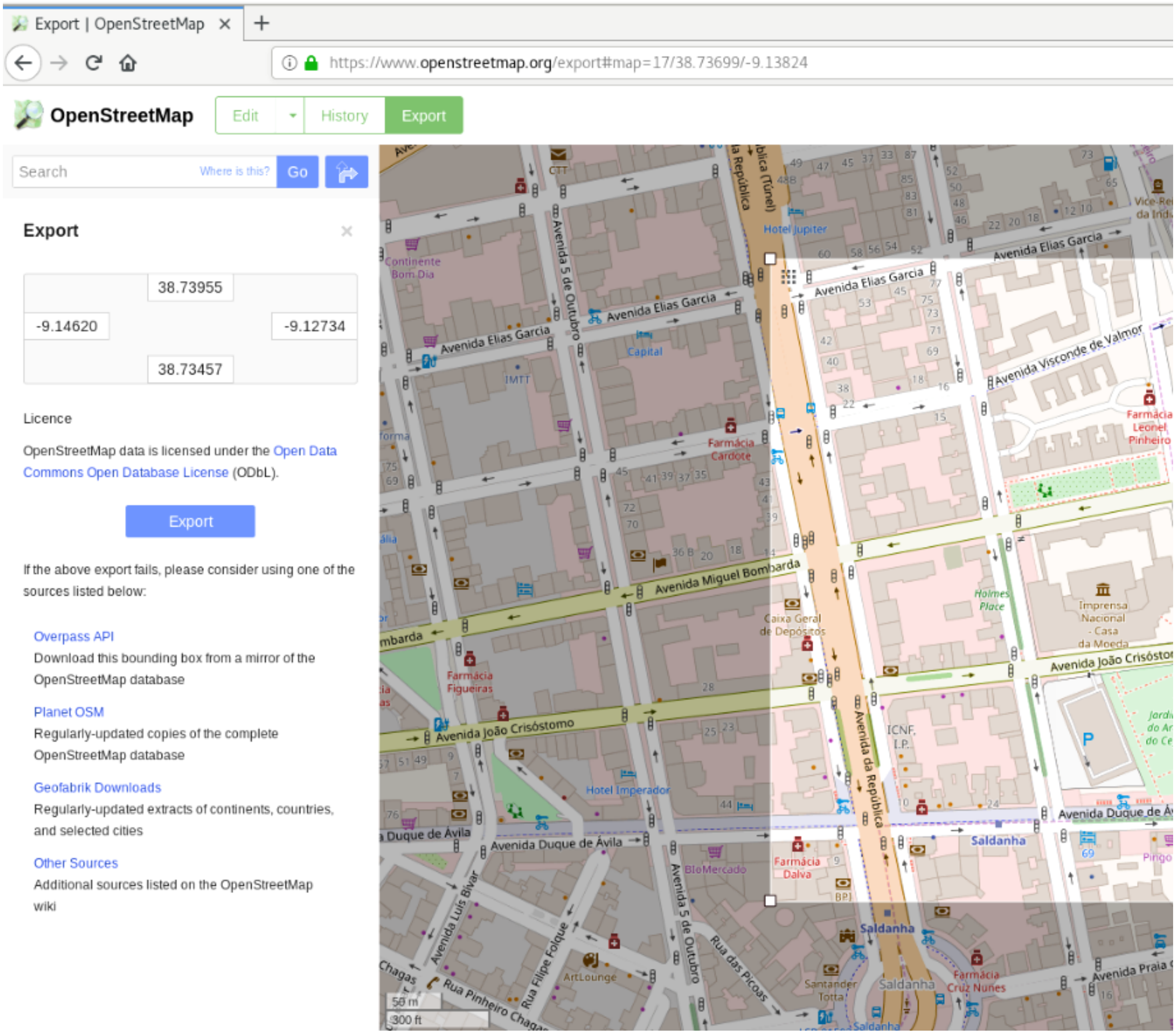

Figure 4.1: Selecting and downloading a map using OSM's website map import tool

Listing 4.1: Example of a C-ITS station XML configuration for running a Denm Service

```
<?xml version="1.0" encoding="UTF-8"?>
<services>
    <service type="artery.application.DenService" name="DEN">
        <listener port="2002" />
    </service>
</services>
```

Listing 4.2: Example of a C-ITS station XML configuration for running a Denm Application

```
<?xml version="1.0" encoding="UTF-8"?>
<usecases>
    <usecase type="artery.application.den.IcyRoad" />
</usecases>
```

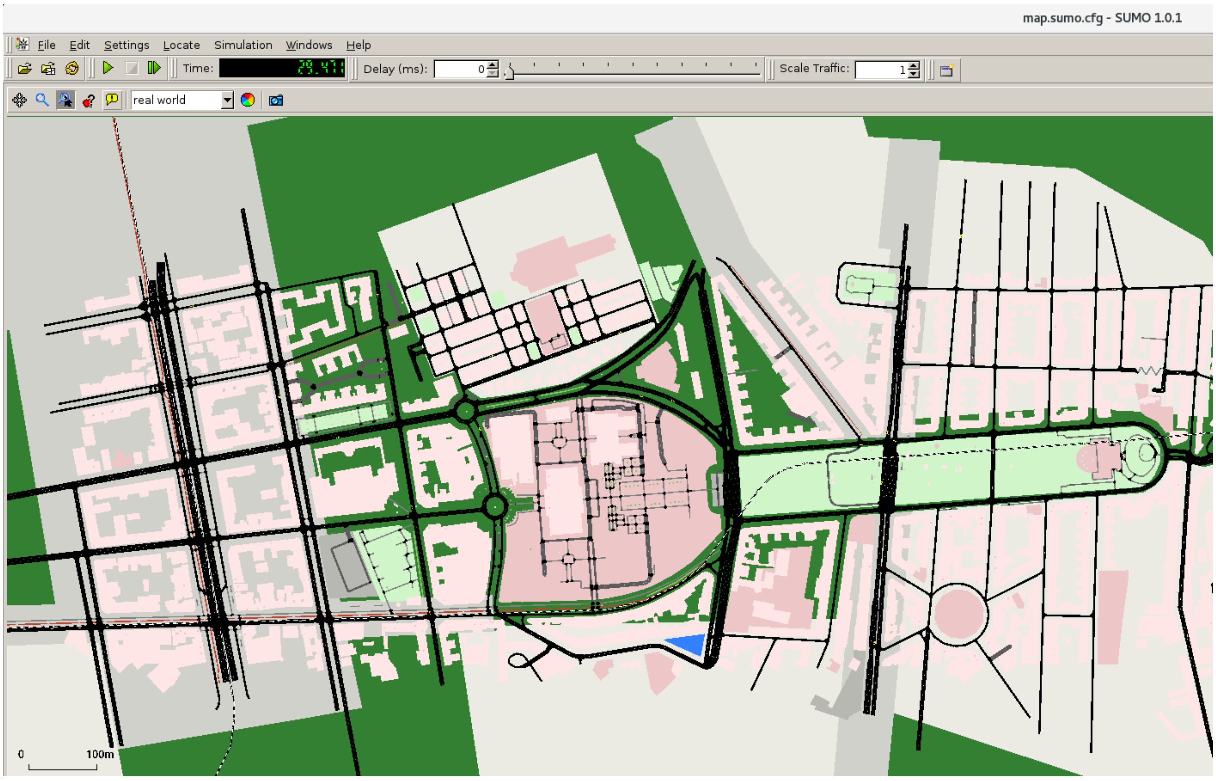

Figure 4.2: SUMO road simulator loaded with the IST Lisboa map and auto-generated vehicle routes

4.2.1 Traffic Control Interface

The TraCI Application Programming Interface (API) [44] gives access to a running road traffic simulation, it allows to retrieve information about the current state of each vehicle inside the simulation as well as changing vehicle parameters.

C-ITS Applications, therefore, have access to two types of methods [44], value retrieval and state changing, among which, for C-ITS research purposes, we emphasise on:

Value Retrieval

I Traffic Lights Value Retrieval: information about any traffic light's current phase (red, yellow or green), current phase duration, controlled lanes, and program (responsible for defining each stage's timing).

II Route Value Retrieval: information about each vehicle's detailed route that includes the source, the destination and all roads it plans to follow to get to its destination.

III Induction Loop Value Retrieval: information about induction loops, which are lane detectors that can be placed (see figure 4.3) in any road section (junction, road, lane, multiple-lane, etc) and retrieve metrics on vehicles which pass through the detector. Some of the available methods are compiled in table 4.1.

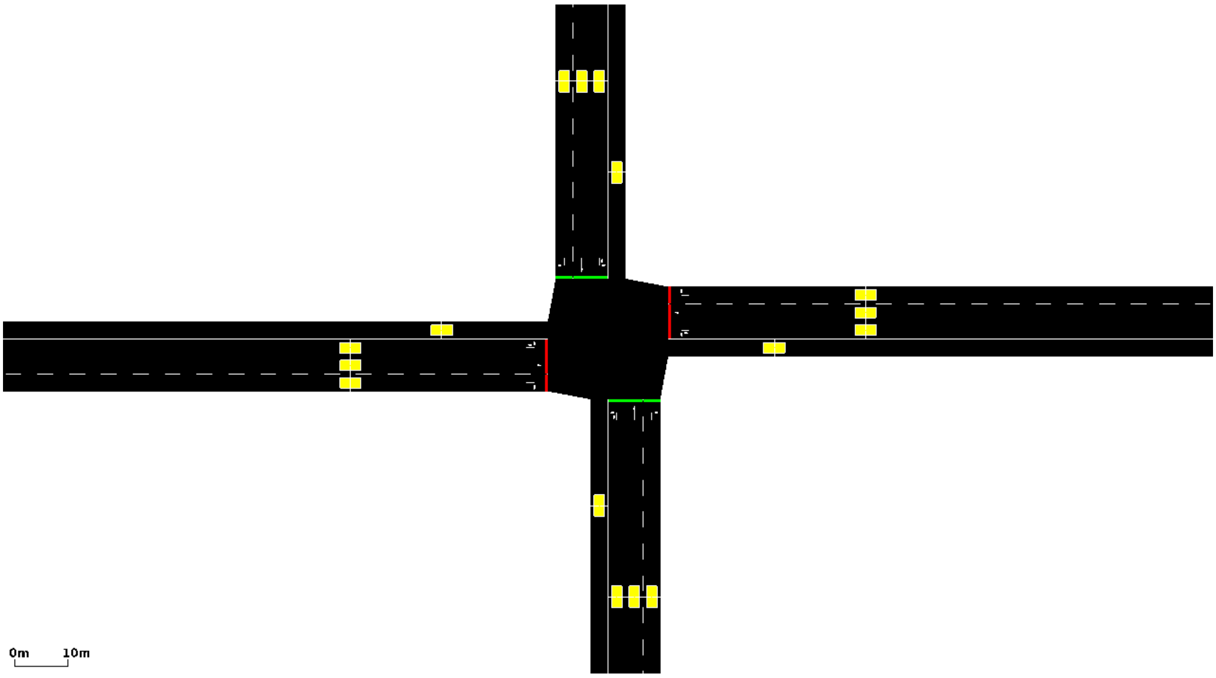

Figure 4.3: Example scenario with various rectangular induction loops (yellow rectangles) placed in all entrances/exits of a junction [6]

State Changing

I Change Traffic Light State: changes the state of a traffic light, like the duration of the current phase, the setting of a specific phase (e.g. force green on the next switch) or complete program definition where a completely new traffic automaton is loaded.

II Change Edge State: it allows for effort setting as well as travel time setting for a specific edge (road), where the weighted effort of that edge can be altered to influence routing algorithm's behaviour - Complex routing modelling. An example can be to temporarily set an edge's effort to a higher number to in representation of a traffic jam, so other vehicle's routing algorithms can avoid it (see figure 4.4 for A* algorithm example).

III Change Route State: change a vehicle's route which is composed by a list of edges (roads) that vehicle will follow.

4.2.2 Creating an Application

When running a simulation, Artery will automatically handle the connection to SUMO and OMNeT++, executing both simulators in parallel. As such, the C-ITS applications running in the network simulator (OMNeT++) can utilise the TraCI API, previously analysed in 4.2.1.

The applications, commonly refered as Use Cases in Artery, run within each C-ITS station and are

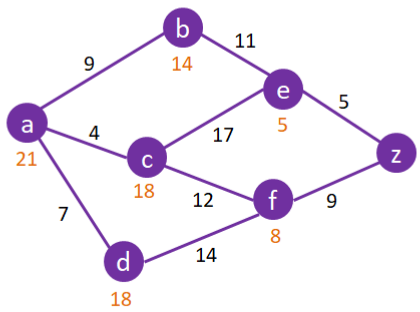

Figure 4.4: Example of an heuristically weighted graph where A* algorithm can be used to find a path between Z and A which can be viewed as the start point and destination, the edges as roads and the heuristic function as the average speed of traffic flow of that road

Method Name	Return Type	Description
nVehContrib	#vehicles	The number of vehicles that have completely passed the detector within the interval
nVehEntered	#vehicles	All vehicles that have touched the detector. Includes vehicles which have not passed the detector completely (and which do not contribute to collected values)
flow	#vehicles/hour	The number of contributing vehicles extrapolated to an hour
speed	m/s	The arithmetic mean of the velocities of all completely collected vehicles. This gives the time mean speed

Table 4.1: Subset of available methods on TraCI Induction Loop Value Retrieval

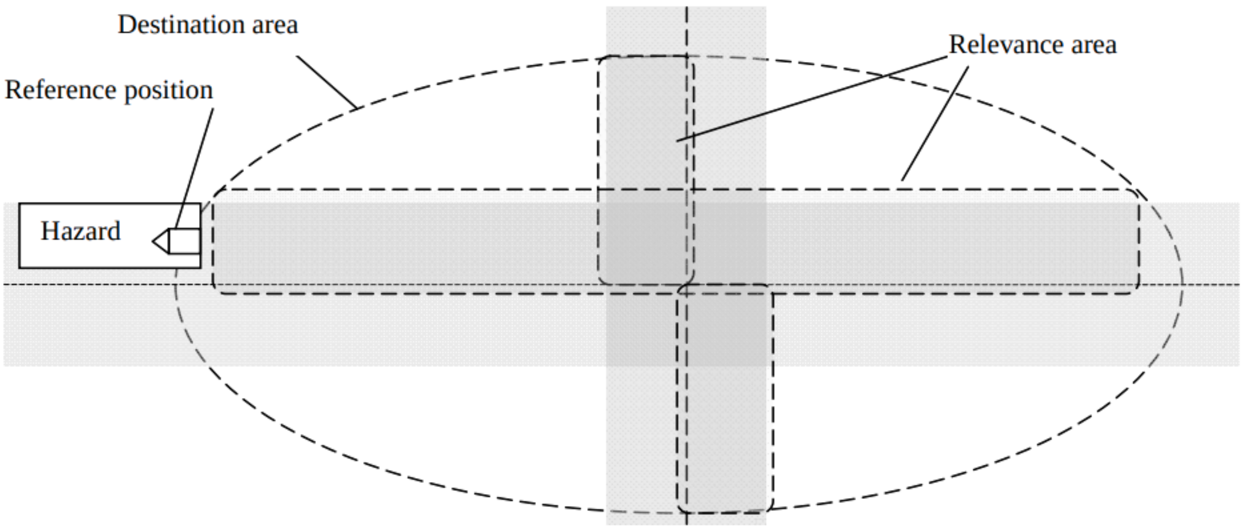

Figure 4.5: Example of the event reference position, the relevance area and the destination area [7]

described in C++ classes from which essential methods are relevant to showcase:

I UseCase::initialize() is called when the application is initialized, in this method, application's internal configuration parameters are loaded from a configuration file, e.g. sampling time, GBC message radius.

II UseCase::check() is called after each OMNeT++ simulation step, it shall be used to check if actuation condition of the application are met and acting accordingly.

III UseCase::indicate(packet) is called whenever a message sent from another C-ITS station and directed at that vehicle is received.

IV UseCase::createMessage() is where ETSI ASN.1 messages are created, they depict the relevant event in the case of DENM and the status report in case of CAM.

V UseCase::createRequest() is need for creating the message's accompanying request, also known as it's header. It depicts the transport protocol to be used as well as its specificities such as the destination in the case of GeoBroadcast which can be defined by a center point and a radius size, forming a geographic circle or oval shape within which all C-ITS stations shall receive the message, like in figure 4.5.

A subpart of an example IcyRoad application is depicted in listing 4.3, this application, when receiving a message whose cause code is Adverse Weather Condition Adhesion, meaning, for example, ice on the road, slows down the vehicle using the TraCI API.

Listing 4.3: Subpart of the IcyRoad application, depicting DEN message handling and TraCI API usage

```
void IcyRoad::indicate(const artery::DenmObject& denm)
{
    const vanetza::asn1::Denm& asn1 = denm.asn1();

    if (denm & CauseCode::AdverseWeatherCondition_Adhesion) {

        mVehicleController = &mService->getFacilities().get_mutable<traci::VehicleController>();

        auto vehID = mVehicleController->getVehicleId();
        printf("%d Received Icy Road\n", vehID);
        fflush(stdout);

        double speed = 3;
        auto vehicleID = mVehicleController->getVehicleId();
        double duration = 20000;

        mVehicleController->getLiteAPI().vehicle().slowDown(vehicleID, speed, duration);
    }
}
```

4.3 Creating a Scenario Storyboard

This module's goal is the realisation of test cases (stories) based on Test Case Document (TCD). TCDs describe the prerequisites for generating specific DENMs: For example, considering the Hazardous Location example scenario described in detail on section 2.7.1 and depicted in Figure 2.15, when the vehicle's ESP detects an oil spill on the road, a corresponding "Hazardous Location" DENM shall be sent. Instead of simulating an oil spill on the road, it is more efficient to inject an event directly onto a specific car's application, that event would have been generated by the ESP sensor in a real scenario, but the storyboard allows us to simulate it. Such behaviour is sufficiently realistic from a VANET perspective since the C-ITS vehicle's application will generate the appropriate messages and alter its vehicle behaviour accordingly.

To achieve the aforementioned behaviour, when executing a new simulation, Artery initialises the Storyboard module which is then used for executing a user provided Python script containing the scenario's stories. For this purpose, several C++ classes such as Condition, Effect and an interface to the Storyboard are exported to the embedded Python context, so the script can register its stories at Artery's Storyboard module. Given the Simulation Stack is Federated, meaning Artery has a socket connection to SUMO Traffic Command Interface (TraCI), these stories can apply effects to any vehicles via TraCI.

During the simulation, Artery requests updated vehicle data from SUMO at regular time steps. After each vehicle update step, the Storyboard becomes active as well: All previously registered stories are checked if associated effects have to be applied to or reverted from any vehicles. Changes regarding effects then influence the following simulation step, either by changing vehicle parameters directly such as speed or by reactions due to V2X message disseminations. Figure 4.6 depicts the interactions between the previously mentioned simulation components and the Storyboard module.

The dynamic aspects of a scenario are described by means of stories. Each story consists of at

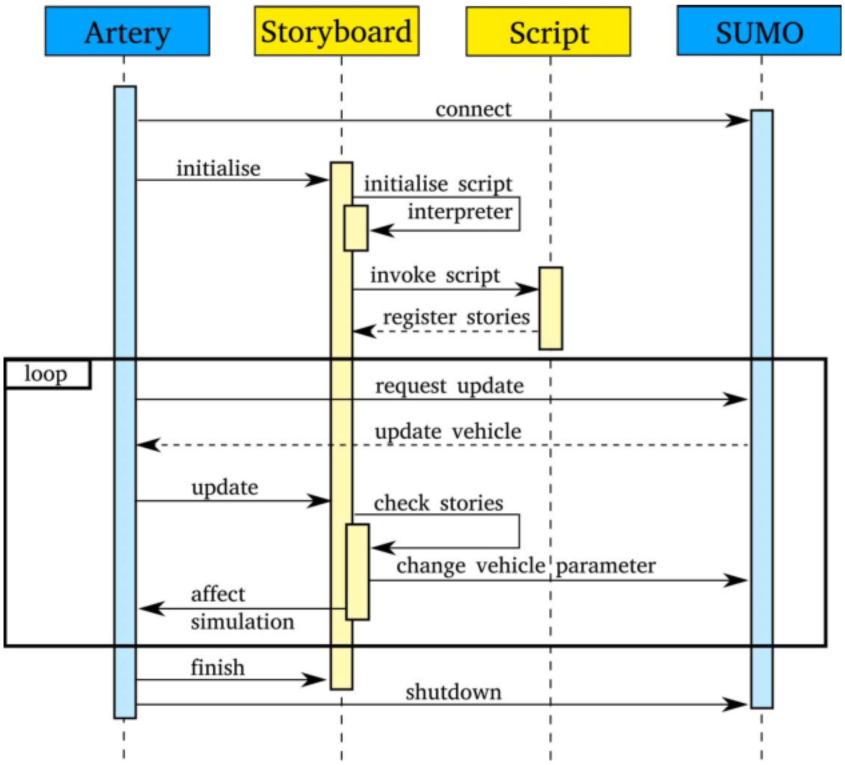

Figure 4.6: Runtime behaviour of Artery with Storyboard [8]

Table 4.2: Storyboard's Basic Set of Primitive Conditions

Condition	Type	Criteria
Time	Absolute	Simulation time is within given time window
SpeedLess		Vehicle's speed is below given limit
SpeedGreater		Vehicle's speed is above given limit
Identity		SUMO identifier matches given set
Polygon		Vehicle position is within polygon's boundary
Faster	Relative	Ego vehicle is faster than some other vehicles
Slower		Ego vehicle is slower than some other vehicles
TTC		Time-to-collision drops below threshold

Table 4.3: Storyboard's Basic Set of Effects

Effect	Type	Description
Speed	Persistent	Applies an upper speed limit to a vehicle
Stop		Stops a vehicle entirely
Signal	One-shot	Custom signalling in vehicle's application layer

least one condition and one accompanying effect, which shall be applied to each vehicle fulfilling the conditions of the story. In total three types of conditions can be differentiated:

i **Absolute conditions** restrict the applicability of a story for a particular vehicle on the basis of absolute parameters, e.g. the vehicle's own position, speed or the simulation time. These conditions return either true or false.

ii **Relative conditions** evaluate parameters relative to neighbouring vehicles, e.g. speed differences or collision risks. In addition to the boolean result, relative conditions also return the set of matching neighbouring vehicles.

iii **Logical conditions** combine any two other conditions according to a logical operation, e.g. AND or OR. If the sub-conditions return vehicle sets, these are combined in terms of intersections or unions, respectively.

An overview of the available logical conditions and effects is given in Table 4.2 and Table 4.3 respectively. With the aid of logical conditions, complex simulations can be translated into stories. An example of a storyboard script where the signalling effect for Hazardous Location is used is available in Listing 4.4, in this case for an icy road scenario.

Listing 4.4: Storyboard example featuring IcyRoad application signalling and a custom Jam Area condition

```
import storyboard
import timeline
from storyboard import Coord


def createStories(board):

        time_icy = storyboard.TimeCondition(timeline.seconds(1.5))
        vehicle_icy = storyboard.CarSetCondition("flowSouthNorth.0")
        time_and_vehicle_icy = storyboard.AndCondition(time_icy,
            vehicle_icy)
        signaleffect_icy = storyboard.SignalEffect("icy_road")

        story_icy = storyboard.Story(time_and_vehicle_icy, [
            signaleffect_icy])

        board.registerStory(story_icy)

        jamAreaCondition = storyboard.PolygonCondition([Coord(2920, 2673)
            ,
                                                        Coord(2903, 2955)
                                                            ,
                                                        Coord(2882, 3154)
                                                            ,
                                                        Coord(2901, 3154)
                                                            ,
                                                        Coord(2923, 2955)
                                                            ,
                                                        Coord(2943, 2673)
                                                            ])
        jamTimeCondition = storyboard.TimeCondition(timeline.seconds(60))
        jamTimeAndSpaceConditions = storyboard.AndCondition(
            jamTimeCondition, jamAreaCondition)
        jamConditions = storyboard.AndCondition(jamTimeAndSpaceConditions
            , storyboard.LimitCondition(10))
        jamStory = storyboard.Story(jamConditions, [stopEffect])
        board.registerStory(jamStory)

        print("Stories loaded!")
```

4.4 Running the Simulation

To run any OMNeT++ simulation, we need to create a configuration file (omnetpp.ini). This file tells the network simulation program which scenario to simulate: its network, values for model parameters, etc. An example for the Instituto Superior Tecnico (IST) scenario with the defined DENM Service running the IcyRoad application with the IcyRoad storyboard would look like listing 4.5.

Listing 4.5: Omnet.ini XML file containing the holistic definition of simulation parameters

```
[General]
sim-time-limit = 3600s
network = artery.inet.World
*.withStoryboard = true
*.storyboard.canvas = "World"

*.traci.core.version = -1
*.traci.launcher.typename = "PosixLauncher"

*.node[*].wlan[*].typename = "VanetNic"
*.node[*].wlan[*].radio.channelNumber = 180
*.node[*].wlan[*].radio.carrierFrequency = 5.9 GHz
*.node[*].wlan[*].radio.transmitter.power = 200 mW

*.node[*].middleware.updateInterval = 0.1s
*.node[*].middleware.datetime = "2017-06-26 12:00:00"
*.node[*].middleware.services = xmldoc("services.xml")
*.node[*].middleware.DEN.useCases = xmldoc("usecases.xml")
*.node[*].middleware.DEN.*.nonUrbanEnvironment = true

*.traci.launcher.sumo = "sumo-gui"

[Config rsu]
*.rsu[*].middleware.updateInterval = 0.1s
*.rsu[*].middleware.datetime = "2017-06-26 12:00:00"

*.rsu[*].middleware.services = xmldoc("servicesRSU.xml")
*.rsu[*].middleware.DEN.useCases = xmldoc("usecasesRSU.xml")
*.rsu[*].middleware.DEN.*.nonUrbanEnvironment = true

[Config highway_rsu]
extends = rsu
*.traci.launcher.sumocfg = "ist.sumocfg"
*.storyboard.python = "icyroad"
*.numRoadSideUnits = 1

*.rsu[*].mobility.initialZ = 10m
*.rsu[0].mobility.initialX = 900m
*.rsu[0].mobility.initialY = 0m
```

5

Evaluation

Contents

5.1 Evaluation Scenarios

To enrich the simulation realism, three different scenarios with vastly different road network morphologies were chosen for evaluation purposes, they are:

- **Highway Scenario** characterised by low-density traffic in long roads with occasional highway junctions. Due to this road topology, vehicle speed is higher and more consistent, ensuring an easier scenario for C-ITS communications. In this scenario, we aim to test the following Road Hazard Warning systems: electronic emergency brake light, weather conditions, slow or stationary vehicle, hazardous location and in-vehicle signage.

- **Rural Scenario** characterised by very low-density traffic in low-lightened roads where junctions and blind corners are frequent, slow agrarian vehicles are also a recurrent. Due to this road topology, different vehicle's speed difference can be abysmal and when added with bad visibility is a recipe for accidents. In this scenario, we aim to test the following Road Hazard Warning systems: slow or stationary vehicle, hazardous location, road works, signal violation intersection safety, additionally from Cooperative Navigation: probe vehicle data.

- **Urban Scenario** characterised by high-density traffic and filled with pedestrian crossings, traffic signals an road junctions. Due to this road topology, vehicle's speed is always changing and is not consistent, creating a challenging scenario for C-ITS communications. Given the complexity of this scenario, the entire stack of applications will be tested.

5.2 Implementing Evaluation Scenarios

In this section, the implementation process of the previously mentioned scenarios is presented. The chosen locations for the scenarios were specifically tailored to meet the requirements mentioned in section 5.1.

5.2.1 Highway Scenario

For this scenario, ***Ponte Vasco da Gama*** or Vasco da Gama Bridge (see figure 5.1 for details) was chosen as it represents an uninterrupted stretch of highway with 12.3 Km, the longest bridge in Western Europe [45]. It was originally built to avoid the incomming north-south transit from moving through Lisbon's interior.

Figure 5.1: Selected OSM map for Highway Scenario: Vasco da Gama Bridge

Figure 5.2: *Ponte Vasco da Gama* in a sunny day [9]

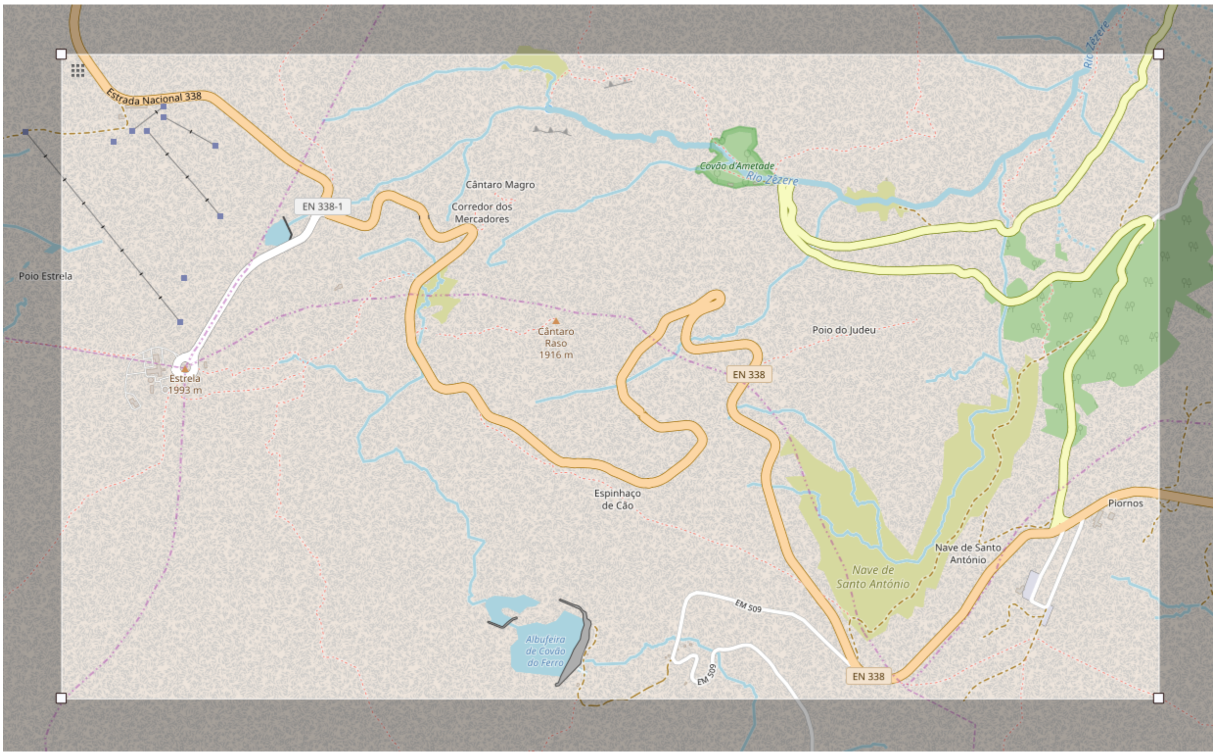

Figure 5.3: Selected OSM map for Rural Scenario: *Torre, Serra da Estrela*

5.2.2 Rural Scenario

For this scenario, ***Serra da Estrela*** (see figure 5.3 for details), namely the area around ***Torre*** was chosen as it represents a typical rural environment, with a population as low as 50 hab/km^2, compared to Oporto's 5 736,1 hab/km^2 [46]. It is also the highest mountain on Portuguese continental soil and, due to its altitude, it is covered in snow 2 to 3 months per year [47].

5.2.3 Urban Scenario

For this scenario, Downtown Lisbon was chosen; namely the area surrounding one of the most famous avenues, the ***Avenida da Liberdade*** (see figure 5.5 for details). The choice was based on Lisbon being the capital of Portugal, and the Downtown being a very popular tourist zone, flooded with traffic at any given hour of the day, resulting in one of the most polluted air zones in the whole country, registering in 2016 an average concentration of nitrogen dioxide per cubic meter of 57,3 mg/m3, superior to European Union's target of 40 mg/m3 [48].

Figure 5.4: *Serra da Estrela* in the winter [10]

5.3 Implementing Day 1 Services

Day 1 Services bundle, composed by Road Safety and Traffic Efficiency applications, have been fully implemented with the exception of Signal violation intersection safety, In-Vehicle Signage and Speed Limits, Shock-wave Damping, GLOSA and Traffic signal priority request by designated vehicles due to their dependence on RSU communicating with the traffic light infrastructure. Unfortunately, as of the writing of this document, RSU support is experimental and still under development, therefore the communication is only one-directional, meaning the RSUs can only receive messages and can't reply. Nevertheless, the application skeleton and storyboard bindings were created for all of them.

Depicted in table 5.1 is the implementation status and the scenarios referring to each Day 1 application. All applications were implemented according to the ETSI requirements depicted in Table 2.2 which include specific DENM container parameters tailored to each (see Figures 2.5 and 2.6). Additionally, custom storyboard bindings were made so they can be event signalled from storyboard's python scripts.

5.4 Test Case Documents

For each Evaluation Scenario detailed in section 5.1, a test case document was compiled whose goal was to target the appropriate Day 1 Applications as defined in Table 5.1 and the method of testing:

Figure 5.5: Selected OSM map for Urban Scenario: Downtown Lisbon

Figure 5.6: *Marquês de Pombal*, Lisbon [11]

Table 5.1: Day 1 Services Bundle Implementation Status and Target Environments

Application Name	Implementation Status
Electronic emergency brake light Weather conditions Slow or stationary vehicle(s) Traffic jam ahead Hazardous location Road works Emergency vehicle approaching	Fully implemented
Probe vehicle data Signal violation intersection safety Green light optimal speed advisory (GLOSA) In-vehicle signage Shock wave damping In-vehicle speed limits Traffic signal priority request by designated vehicles	Partially implemented due to RSU support still being in development

Application under Test	Method of Testing	Expected Result
Electronic Emergency Brake Light	Specific vehicle app signalling	Affected vehicles broadcast specific DENM message
Slow Or Stationary Vehicle		
Hazardous Location		
Weather Conditions		
Probe Vehicle Data	CAM service running on all vehicles	All vehicles broadcast CAM messages

Table 5.2: Highway Test case Document

Application under Test	Method of Testing	Expected Result
Electronic Emergency Brake Light	Specific vehicle app signalling	Affected vehicles broadcast specific DENM message
Hazardous Location		
Road Works		
Probe Vehicle Data	CAM service running on all vehicles	All vehicles broadcast CAM messages

Table 5.3: Rural Test case Document

Highway Scenario TCD in Table 5.2, Rural Scenario TCD in Table 5.3 and Urban Scenario TCD in Table 5.4.

5.5 TCD Testing and Results

To test the feasibility of the Simulation Stack as close to investigation workload as possible, we've conducted a series of experiments aiming to stress all the chosen scenarios, test case documents and the implemented Day 1 Services bundle. Therefore, three storyboards were created via python scripts where the application under test was subjected to a specific method of testing.

Oracle VM VirtualBox [49] was chosen to run the virtual appliance where the complete simulation stack is installed and pre-configured. The test results referring to each application were gathered via

Application under Test	Method of Testing	Expected Result
Electronic Emergency Brake Light	Specific vehicle app signalling	Affected vehicles broadcast specific DENM message
Weather Conditions		
Slow Or Stationary Vehicle		
Traffic Jam Ahead		
Hazardous Location		
Road Works		
Emergency Vehicle Approaching		
In Vehicle Signage		
In Vehicle Speed Limits		
Shock Wave Damping		
Probe Vehicle Data	CAM service running on all vehicles	All vehicles broadcast CAM messages

Table 5.4: Urban Test case Document

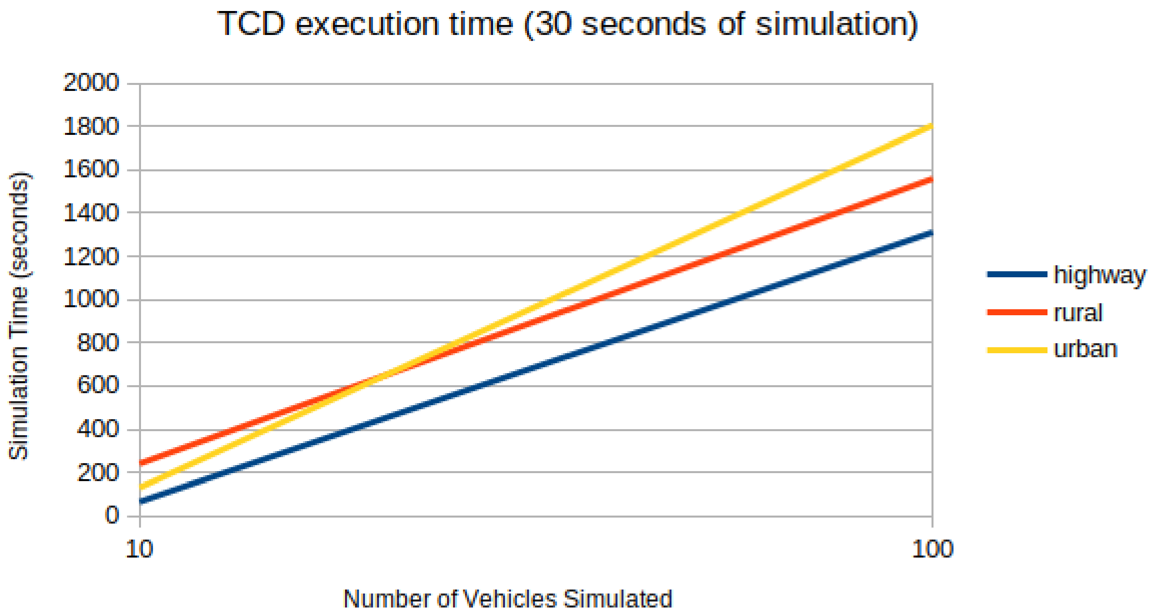

Figure 5.7: Linear Approximation of Execution times of each TCD for 30 seconds of simulation

statistics explorer in OMNet++ GUI interface and RAM usage was gathered using HTOP.

To ensure optimal results, the following measures were taken:

i All Test Case Documents were ran for 30 seconds of simulation time. This time differs from real time also known as CPU time, which refers to how long the simulation program has been running for.

ii All results were taken as an average of 3 runs; necessary to mitigate warm-up phase noise.

An analysis on Figure 5.7 showcases that the execution time of a 30 seconds simulation on any TDC is heavily influenced by the number of simulated vehicles, as expected. We can also infer that the small differences in execution time between any two TCDs when the number of vehicles simulated is the same might be attributed to the different C-ITS applications running in each TCD.

When memory utilisation is concerned, Figure 5.8 depicts its usage in each TCD scenario. An interesting finding was its usage was not significantly affected by the number of vehicles on simulation, deferring only in as much as 5%, therefore the figure does not differentiate between the number of vehicles simulated. We can also infer that the main culprit of the memory usage between the TCD were the scenarios themselves, the highway one, as expected, being the least resource intensive and the urban being the most, as expected.

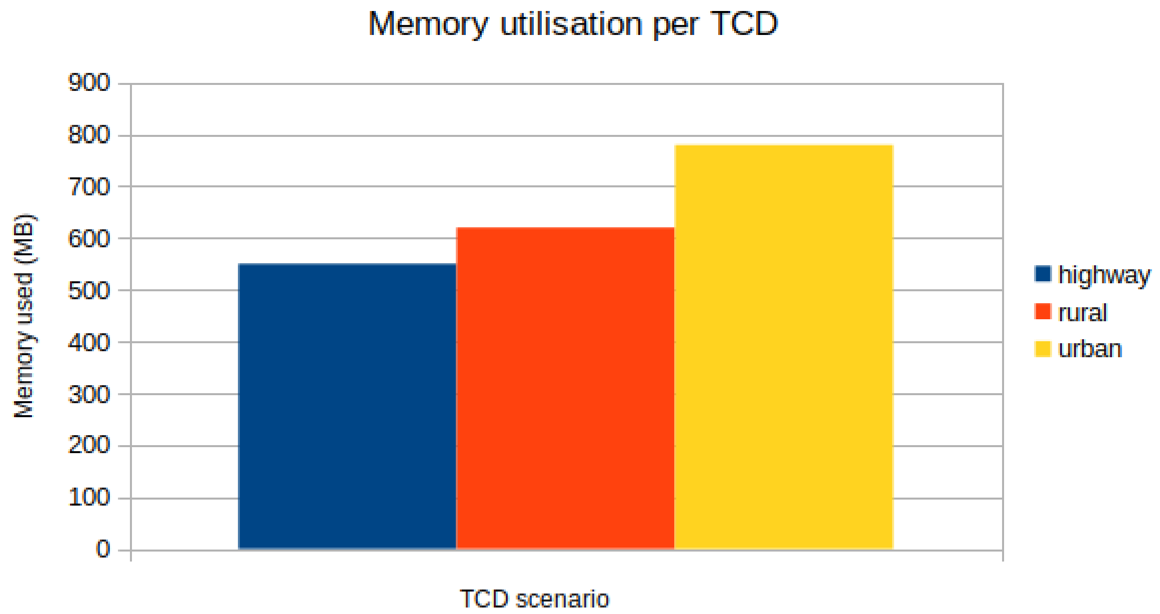

Figure 5.8: Memory utilisation of each scenario when running its TCD

5.6 Scalability

Simulation of complex scenarios featuring hundreds of C-ITS stations exchanging a multitude of messages each second require great architectural feats to ensure proper scalability. A scalable system should complete more work as size or load grows. Hence, the throughput should be an increasing function, or else the system is not scalable.

Linear scalability happens when the number of operations per second per node remains constant as the node count increases, meaning, as we increase the number or available processing cores, the time taken to simulate the same environment should decrease linearly as the increased workload is distributed across the available resources.

5.6.1 Analysis of the Simulation Stack Architecture

In this section, we analyse the both the network and the road simulators architecture, especially regarding their throughput on multi-core systems.

5.6.1.A OMNeT++

OMNeT++, the basis upon which all the used frameworks rely on, is a single threaded application by design. If multiple processing cores are available, Parallel simulation of two or more isolated simulations can be done. This feature can be very useful if several simulations with different parameters have to be executed, launching more than one simulation process at the same time, meaning each process can be assigned to a separate CPU.

5.6.1.B SUMO

SUMO, much similarly to OMNeT++, also runs on a single execution thread. However, vehicle routing in SUMO can be parallelized, meaning if especially complex routing algorithms are to be implemented on each vehicle, the simulation workload related to routing can be distributed to the available cores, effectively reducing contention and benefiting execution time (CPU time taken to execute a simulation).

5.6.1.C Simulation Stack

The throughput of the simulation stack is bottleneck by the least performant component due to being a federated case as previously depicted in Figure 2.26. The previously mentioned architectural characteristics negatively influence the stack's scalability especially when it comes to running a single simulation, compromising the throughput of the simulation stack when running in multi-core/multi-threaded computers and clusters. Nevertheless, given each simulator is running a single thread, two threaded hosts should have better performance than one. The theory is proven by Figure 5.9 that depicts the execution time differences when running on single and dual threaded hosts.

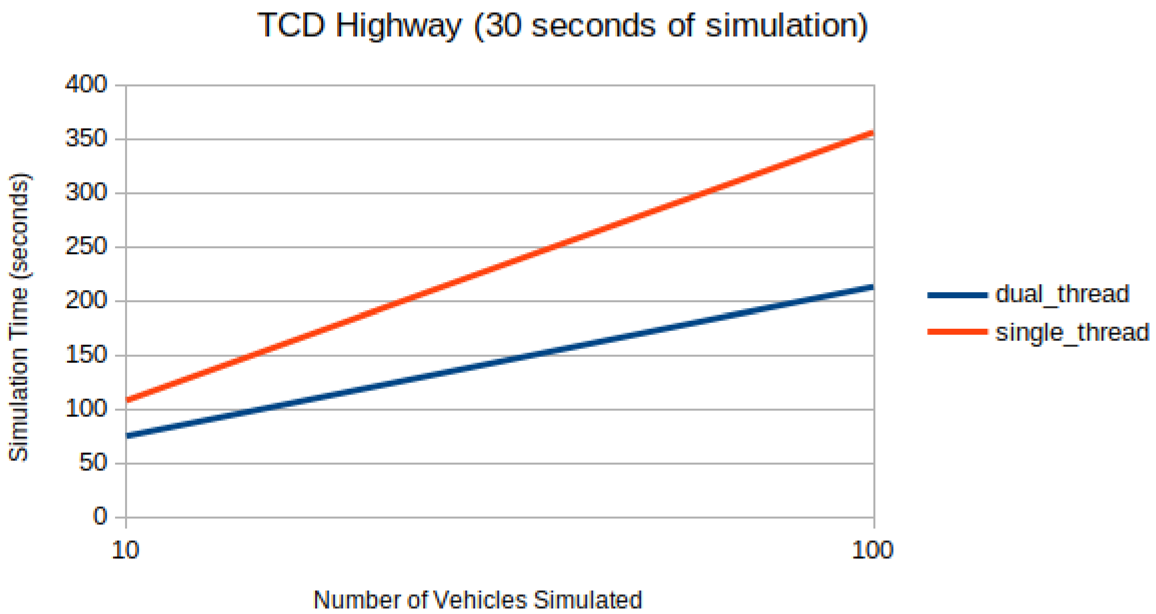

Figure 5.9: Linear Approximation of Execution times of Highway TCD for 30 seconds of simulation for single and dual threaded hosts

6

Conclusion & Future Work

Contents

6.1 Conclusions

This thesis work started with the study of the state of the art on Cooperative Intelligent Transport Systems. We've surveyed the European standards dedicated to the regulation of the communication aspects and application organisation and analysed the current development and testing methodologies for C-ITS systems. The packaged simulation framework was chosen amongst others as its realism is the foundation for meaningful research. It was thoroughly explored, birthing a whole chapter on how to best use it for C-ITS research purposes. We've then characterised the main real road scenarios, resulting in the definition of three test case documents, each one representative of future intelligent roads. Starting from these TCDs, we've chosen and imported real world locations that best fit each scenario requirements. Finally, the developed solution was evaluated on each TCD and the gathered results were enriched by a theoretical evaluation of the framework's scalability.

According to the evaluation results, the developed solution demonstrates itself as a suitable tool for C-ITS environment research, featuring a multitude of parameter tweaking for advanced users while maintaining a subtle learning curve thanks to the bundled frameworks. On the other side, its scalability is weak as both the network and the road simulator's execution is single threaded, culminating in long simulation times as seen on Figure 5.7 where a 30 seconds simulation of 100 vehicles took 30 minutes. Even though the stack's memory usage is very efficient given its built from the ground up with large networks in mind, simulation of such large networks on the thousands of C-ITS stations would be severely bottlenecked by the execution time. On a brighter note, given all bundled frameworks are still under active development, future advancements in C-ITS research are to be expected and considering the developed solution architecture's modularity and compartmentalisation, it can benefit from them. One major improvement would be to enable multi-threaded support for OMNet++ and SUMO.

At last, due to its elevated simulation realism, it presents itself as a meaningful alternative solution for the study, development and testing of complex C-ITS systems and applications without the necessity of having expensive test site facilities. Unfortunately, as it was possible to evince from the development work forces behind the whole simulation package, the development and testing of C-ITS systems is mainly a concern of car manufacturers and research institutes. Considering instead current universities curricula, it is noticeable that C-ITS related studies are not yet widely present as it is a multi-disciplinary field by nature, requiring Computer, Electronic and Civil Engineering knowledge at the same time; not to mention the social implications of cooperative applications. Mobility is a fundamental pillar of modern civilisation and the complexity of this area is enormous and only by joining forces of different experts can this endeavour be tackled, as such, our motivation was always to bridge the knowledge gaps and unite different people with the same goal, to build the future road system. For these reasons, our solution also aimed at creating a simulation platform meant as a tool able to support C-ITS related teaching activities.

In light of such high number of fatalities that afflict our roads, we hope our contribution can expedite the development of future technologies and obtain their promised benefits.

6.2 Future Work

The modularity of the simulation package allows future work can be made in different components autonomously, fomenting cooperation from different areas.

Due to framework shortcomings in RSU support, some Day 1 Services applications were not fully developed, thus Raphael Rielb, the main developer of Artery has been working alongside me to extend the framework with RSU support, it now supports the receiving of messages. As such, implementing the mentioned applications should be possible in the foreseeable future.

The main purpose of this work was never to analyse the benefits of deploying complex C-ITS applications into our roads as much as it was to lay the foundations for such studies. As such, more complex studies can be made using the packaged simulation tool. For instance, one could foresee a complex study utilising the developed urban scenario in *Avenida da Liberdade* where GLOSA can be fully implemented and tested in relation to gas emissions, travel time decrease and traffic flow efficiency.

Directly applicable to the developed simulation stack, from the current literature, we recommend the following projects:

Artery Hardware in the Loop testing is a 2017 study [50] by Christina Obermaier and Christian Facchi on using OMNeT++ and Artery software simulation coupled with physical hardware.

VANET Mobility and Routing Protocols is a 2011 survey [51] of mobility models and routing protocols that could be implemented onto SUMO and Vanetza respectively.

Mix-Zones for Location Privacy in VANETS is a study [52] by Julien Freudiger *et al.* that primarily focuses on the privacy impacts of periodic messages broadcast (CAMs) providing precise position information to nearby vehicles. In order to mitigate this threat, they've leveraged cryptography services to create a combination of mix-zones that ensure driver anonymity. They could be an extension to Vanetza framework.

For specific Day 1 Services C-ITS application development, we extol the following articles, some of which were developed and implemented using a very similar simulation stack:

A Decentralized Vehicle Re-routing Approach using Vehicular Ad-hoc Networks by S.T.Rakkesh *et al.* [53] proposes a rerouting strategy which can be implemented as part of a VANET solution, in order to reroute vehicles efficiently when an accident or roadblock occurs in the traffic region. It is implemented in OMNeT++ and SUMO as well.

VANET Communication on Reducing Traveling Time in Realistic Large Scale Urban Area by Hamed Noori and Mikko Valkama [54] also proposes a new method to find the fastest route from origin to destination by using the V2X communication, decreasing emissions, traffic congestion and fuel consumption.

Potentials and Limitations of Green Light Optimal Speed Advisory Systems is a study [55] by David Eckhoff *et al.* where the impact of GLOSA on the comfort for drivers by reducing waiting times and the number of stops, with promising findings that at low traffic densities these systems can meet all their promised goals.

Impact of GERTRUDE and SAEIP on Public Transportations in Lisbon is a study [56] by Manuel Augusto Vieira and Carlos Filipe Gomes Bispo, in 2004, where the real time co-operative feed-back between urban traffic control (GERTRUDE) and the Public transport vehicle location (SAEIP) systems was used to optimise the travel speed of a bus, in a fixed route (*Cais do Sodré/Campo Grande*) via controlling the in-car volume through traffic lights coming into *Avenida Joaquim António de Aguiar*.

In reality, it is tempting to acknowledge that the complexity of this simulation stack allied with its GPL license allows almost anything related to road and networks simulation to be done. As such, the amount of libraries and frameworks developed over the years is overwhelming [57]. We present some *avant-garde* biased examples of already developed and implemented frameworks that can be plugged onto the simulation stack for futuristic and very creative work:

SimuLTE is an innovative simulation tool enabling complex system level performance-evaluation of LTE and LTE Advanced networks. It is fully customisable with a pluggable interface. One can develop new modules implementing new routing algorithms and protocols and transfer all vehicle messaging to the cellular infrastructure.

MiXiM is a modeling framework created for mobile and fixed wireless networks. It offers detailed models of radio wave propagation, interference estimation, radio transceiver, power consumption and wireless MAC protocols, amongst which Zigbee [58] which is an IEEE 802.15.4-based specification targeted at low-power digital radios, such as for medical device data collection. An idea would be to sense the drivers health state via embedded sensors who communicate with this protocol.

OverSim is an open-source overlay and peer-to-peer network simulation framework, containing several models for structured and unstructured P2P systems and overlay protocols. It may be the basis for BlockChain implementation in the road environment, proving its usefulness when guilting a driver involved in an accident.

All the work developed will be available under a GPL v2.0 License in a ready to run pre-configured virtual image already setup with the whole simulation stack, an OMNet++ IDE with bundled compilation scripts and an OpenStreetMap custom import script. The resources will be available in my personal GitHub.

Proprietary software keeps users divided and helpless. Divided because each user is forbidden to redistribute it to others, and helpless because the users can't change it since they don't have the source code. They can't study what it really does. So the proprietary program is a system of unjust power.

Richard Stallman

Bibliography

[1] European Commission, "Gsm and gps combine for european ecall," http://www.mwee.com/content/gsm-and-gps-combine-european-ecall, Last accessed on 2019-03-20.

[2] European Telecommunications Standards Institute, "Cooperative intelligent transport system," https://ec.europa.eu/transport/themes/its/c-its_en, Last accessed on 2019-03-20.

[3] ETSI, "ETSI EN 302 665 Intelligent Transport Systems (ITS); Communications Architecture," *Etsi*, vol. 1, pp. 1–44, 2010.

[4] J. Härri, F. Filali, C. Bonnet, J. Harri, F. Filali, and C. Bonnet, "Mobility models for vehicular ad hoc networks: a survey and taxonomy," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 4, pp. 19–41, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=5343061{%}5Cnhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5343061

[5] Institute of Transportation Systems, "Christina obermaier and christian facchi presentation on observations on omnet++ at omnet++ summit 2017," https://summit.omnetpp.org/archive/2017/assets/pdf/OMNET-2017-Session_1-02-Presentation.pdf, Last accessed on 2019-03-15.

[6] ——, "Sumo induction loops," https://sumo.dlr.de/wiki/File:Induction_loops.svg, Last accessed on 2019-03-15.

[7] ETSI, "ETSI TR 102 638 V1.1.1 (2009-06): Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions," *ETSI, Sophia Antipolis Cedex, France*, vol. 1, pp. 1–81, 2009. [Online]. Available: http://scholar.google.com/scholar?hl=en{%}7B{&}{%}7DbtnG=Search{%}7B{&}{%}7Dq=intitle:Intelligent+Transport+Systems+(ITS);+Vehicular+Communicatons;+Basic+Set+of+Applications;+Definitions{%}7B{#}{%}7D1

[8] Institute of Transportation Systems, "Artery development page," https://github.com/riebl/artery, Last accessed on 2019-03-15.

[9] Osvaldo Gago, "Ponte vasco da gama, portugal," https://pt.wikipedia.org/wiki/Ponte_Vasco_da_Gama#/media/File:Vasco_da_Gama_Bridge_03.JPG, Last accessed on 2019-03-15.

[10] Luis Fonseca, "Serra da estrela coberta de neve, portugal," https://pt.wikipedia.org/wiki/Ficheiro:Serra_da_Estrela,_Portugal.jpg, Last accessed on 2019-03-15.

[11] SAPO, "Marquês de pombal, lisboa," https://thumbs.web.sapo.io/?H=547&epic=n/rEMRKiMY2qZDbpT1yRurqpSL3AoimYJReQUh9V50KKsu0sL7XiAfSwui6V352KN0z2ezBIP9UUdOZdbrPZgaZLh Last accessed on 2019-03-15.

[12] European Commission, "The interoperable eu-wide ecall," https://ec.europa.eu/transport/themes/its/road/action_plan/ecall_en, Last accessed on 2019-03-20.

[13] European Telecommunications Standards Institute, "Gsm and gps combine for european ecall," https://ec.europa.eu/transport/themes/its/c-its_en, Last accessed on 2019-03-20.

[14] European Commission, "C - ITS Platform: Final Report," no. January, p. 140, 2016.

[15] L. Figueiredo and I. Jesus, "Towards the development of intelligent transportation systems," *. . . Transportation . . .*, no. 81, pp. 1206–1211, 2001. [Online]. Available: http://www.universelle-automation.de/1984{_}Columbus.pdf

[16] ETSI, "ETSI EN 302 665 Intelligent Transport Systems (ITS); Communications Architecture," *Etsi*, vol. 1, pp. 1–44, 2010. [Online]. Available: http://www.etsi.org/deliver/etsi{_}en/302600{_}302699/302665/01.01.01{_}60/en{_}302665v010101p.pdf

[17] J. Santa, F. Pereniguez, A. Moragon, and A. F. Skarmeta, "Vehicle-to-infrastructure messaging proposal based on cam/denm specifications," *2013 IFIP Wireless Days (WD)*, 2013.

[18] V. Communications and A. B. Service, "Vehicular Communications ; Basic Set of Applications ; Part 2 : Specification of Cooperative," *History*, vol. 1, pp. 1–22, 2010.

[19] J. Santa, F. Pereñíguez, A. Moragón, and A. F. Skarmeta, "Vehicle-to-Infrastructure Messaging Proposal Based on CAM / DENM Specifications," 2013.

[20] ETSI, "ETSI TR 102 638 V1.1.1 (2009-06): Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions," *ETSI, Sophia Antipolis Cedex, France*, vol. 1, pp. 1–81, 2009.

[21] European Commission, "C - ITS Platform Specifics Portuguese," no. January, p. 94, 2016.

[22] D. Fintelman, H. Hemida, M. Sterling, and F. X. Li, "A numerical investigation of the flow around a motorbike when subjected to crosswinds," *Engineering Applications of Computational Fluid Mechanics*, vol. 9, no. 1, pp. 528–542, 2015.

[23] WAZE, “Waze navigation,” https://www.waze.com/pt-BR/, Last accessed on 2019-03-20.

[24] Google, “Google maps navigation,” https://www.google.com/maps, Last accessed on 2019-03-20.

[25] J. J. C. Aguilar, J. A. C. Carrillo, A. J. G. Fernndez, and E. C. Acosta, “Robust road condition detection system using in-vehicle standard sensors,” *Sensors (Switzerland)*, vol. 15, no. 12, pp. 32 056–32 078, 2015.

[26] K. Katsaros, “Performance study of a Green Light Optimal Speed Advisory (GLOSA) Application Using an Integrated Cooperative ITS Simulation Platform,” *Proceedings of Wireless Communications and Mobile Computing Conference (IWCMC)*, pp. 918–923, 2011.

[27] GNU Foundation, “What is free software,” https://www.gnu.org/philosophy/free-sw.html, Last accessed on 2019-04-20.

[28] O. Tran, T. Kim, V. Nguyen, and C. S. Hong, “Which network simulation tool is better for simulating Vehicular Ad-hoc network ?” *Winter Conference Proceedings*, pp. 2–4, 2014.

[29] V. N. Oanh Tran Thi Kim and C. S. Hong, “Which network simulation tool is better for simulating Vehicular Ad-hoc network?” 2014.

[30] D. Jiang and L. Delgrossi, “IEEE 802.11p: Towards an international standard for wireless access in vehicular environments,” *IEEE Vehicular Technology Conference*, no. June 2008, pp. 2036–2040, 2008.

[31] C. K. T. Francisco J. Martinez, “A survey and comparative study of simulators for vehicular ad hoc networks (VANETs),” *Wiley InterScience*, no. 12, pp. 32 056–32 078, 2009.

[32] A. Hassan, “VANET Simulation,” *Electrical Engineering*, no. May, p. 43, 2009. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs{_}all.jsp?arnumber=4746652

[33] A. Dahiya, A. Noonia, and B. Singh Jangra, “Vehicular Ad hoc Networks (VANETS): Simulation and Simulators,” *International Journal of Research in Management, Science & Technology*, vol. 2, no. 1, pp. 2321–3264, 2014. [Online]. Available: http://www.ijrmst.org/download/alpana.pdf

[34] “Veins - vehicles in network simulation,” http://veins.car2x.org, Last accessed on 2019-03-20.

[35] Daimler Center for Automotive Information Technology Innovations , “Vsimrti - smart mobility simulation,” https://www.dcaiti.tu-berlin.de/research/simulation/, Last accessed on 2019-04-20.

[36] A. Tomandl, D. Herrmann, K. P. Fuchs, H. Federrath, and F. Scheuer, “VANETsim: An open source simulator for security and privacy concepts in VANETs,” *Proceedings of the 2014 International Conference on High Performance Computing and Simulation, HPCS 2014*, pp. 543–550, 2014.

[37] Anylogic, “Anylogic simulation tool,” https://www.anylogic.com/, Last accessed on 2019-04-20.

[38] OMNeT++ Development Team, “Omnet++ development page,” https://omnetpp.org/, Last accessed on 2019-04-15.

[39] Institute of Transportation Systems, “Vanetza development page,” https://github.com/riebl/vanetza, Last accessed on 2019-03-15.

[40] ——, “Sumo,” https://sumo.dlr.de/, Last accessed on 2019-03-15.

[41] OpenStreetMap, “Openstreetmap,” https://www.openstreetmap.org/about, Last accessed on 2019-03-15.

[42] Open Data Commons, “Open data commons open database license,” https://opendatacommons.org/licenses/odbl/index.html, Last accessed on 2019-03-15.

[43] OpenStreetMap, “Openstreetmap foundation,” https://wiki.osmfoundation.org/wiki/Main_Page, Last accessed on 2019-03-15.

[44] Institute of Transportation Systems, “Introduction to traci,” https://sumo.dlr.de/wiki/TraCI#Introduction_to_TraCI, Last accessed on 2019-03-15.

[45] LusoPonte, “Principais características da ponte vasco da gama,” https://www.lusoponte.pt/vasco-da-gama/projecto/principais-caracteristicas, Last accessed on 2019-04-15.

[46] Instituto Nacional de Estatística, “Censos por zona,” https://censos.ine.pt/, Last accessed on 2019-04-15.

[47] Turismo do Centro, “Região da serra da estrela,” https://turismodocentro.pt/artigo-regiao/serra-da-estrela/, Last accessed on 2019-04-15.

[48] Diário de Notícias, “Lisboa, porto e braga. as zonas em que se ultrapassam limites de poluentes do ar.” https://www.dn.pt/sociedade/interior/zonas-de-lisboa-porto-e-braga-ultrapassam-limites-de-poluentes-do-ar-em-2016---zero-8834451.html, Last accessed on 2019-04-15.

[49] Oracle, “Welcome to virtualbox,” https://www.virtualbox.org/, Last accessed on 2019-04-15.

[50] C. F. Christina Obermaier, “Observations on OMNeT++ Real-Time Behaviour,” 2017.

[51] G. M. F. X. V. K. Evjola Spaho, Leonard Barolli, “VANET Simulators: A Survey on Mobility and Routing Protocols,” 2011.

[52] M. F. P. P. J.-P. H. Julien Freudiger, Maxim Raya, “Mix-Zones for Location Privacy in Vehicular Networks,” 2007.

[53] R. S.T.Rakkesh, A.R.Weerasinghe, "A Decentralized Vehicle Re-routing Approach using Vehicular Ad-hoc Networks," 2016.

[54] M. V. Hamed Noori, "Impact of VANET Based V2V/V2I Communication Using IEEE 802.11p on Reducing Traveling Time in Realistic Large Scale Urban Area," 2013.

[55] R. G. David Eckhoff, Bastian Halmos, "Potentials and Limitations of Green Light Optimal Speed Advisory Systems," 2013.

[56] C. F. G. B. Manuel Augusto Vieira, "Melhoria da Velocidade dos Transportes Públicos de Superfície em Lisboa por Regulação da Admissão de Trânsito," 2004.

[57] OMNeT++, "Simulation models and tools," https://omnetpp.org/download/models-and-tools, Last accessed on 2019-04-20.

[58] Y.-C. H. Jin-Shyan Lee, "ITRI ZBnode: A ZigBee/IEEE 802.15.4 Platform for Wireless Sensor Networks," *2006 IEEE International Conference on Systems, Man and Cybernetics*, 2006.