POLITECNICO DI TORINO

Master of Science in Computer Engineering

Master's Thesis

Implementation and evaluation of a simulation platform for C-ITS applications

Supervisors:
Prof. Massimo Violante
Prof. Alberto Manuel Ramos da Cunha

Candidate:
Alessio Ciavarella

April 2018

Preface

This thesis work has been developed inside the European Commission *ERASMUS+* programme that aims at supporting education, training, youth and sport in Europe. The involved university were the Politecnico di Torino in Italy and the Instituto Superior Técnico of Lisbon in Portugal. The studying period had a duration of six months, from September 2017 to March 2018.

The basis for this thesis work originally stemmed from my passion for cars and technology. It has been a result of a mix of technical knowledge achieved during my academic studies and the experience obtained in pursuing my passion. The importance of the new developing technologies related to the car world, especially the ones involving road safety, pushed me to do all the best, even during not easy situations. I hope that the developed work, even if it is just at the beginning, will be carried on and improved by whichever person wants to contribute.

I would really like to thank you my two supervisors. Professor Alberto Cunha, which whom I worked in close contact sharing ideas, not only on the thesis work, but also about Italian and Portuguese cultures. It has been a great pleasure to receive advices on places to visit and food to try. Professor Massimo Violante, whom, even from great distance, has been always available and supported me in the accomplishment of this work. Then I have inevitably to thank you my whole family which support, from the very beginning of this long adventure, has been always constant and fundamental. Finally, I would like also to thank you all the friends with whom I spent my university years, both close and far ones. I enjoyed all the fantastic adventures we had together. Among them a special thank you goes to all the friends that I met during my Erasmus in Portugal, both Italian and Portuguese, you have been the best Portuguese family that I could ever had. Obrigadissimo!!!

This document has been written with the LaTeX document preparation system.

The front cover image it is courtesy of the U.S. Department of Transportation (DOT).

Abstract

The development of Cooperative Intelligent Transport Systems (C-ITS) has the potential to play a significant role in responding to the current growing demand for new mobility services and the need of safer, more efficient and sustainable transport systems. Over the past decades there have been remarkable new developments in technologies that facilitate C-ITS.

Despite that, the realisation of C-ITS systems requires many resources in terms of time and investments because of their complex and widespread infrastructure. Moreover, since those systems are concerned with road safety and human lives, the need of performing adequate tests in controlled environments is of paramount importance. To overcome these issues a series of developing and testing methodologies have been defined, among which there is the use of well-suited simulation tools.

In this dissertation a new simulation platform able to simulate important future intelligent roads use cases is presented. The goal is to build a support tool for computer, transportation and mobility engineering students interested in deepening the study of C-ITS systems. The work was based on the SAE and ETSI standards for the message format and had a special concern with possible future type of road-side units (RSUs). The presented platform has the potential to allow the realisation of prototypes for future evolutions and enhancements for C-ITS systems. Our solution has been developed using the Anylogic simulation tool. A qualitative evaluation has been done in order to asses the simulation software features and the use cases implemented into the simulator. The goal has been reached since the developed tool has been successfully tested.

Keywords: C-ITS, transport systems, road safety, SAE, ETSI, Anylogic

Abstract

Lo sviluppo dei sistemi di trasporto intelligenti cooperativi (C-ITS) ha le potenzialità di svolgere un ruolo significativo nel rispondere alla crescente domanda di nuovi servizi di mobilità e alla necessità di sistemi di trasporto più sicuri, efficienti e sostenibili. Negli ultimi decenni ci sono stati notevoli sviluppi nelle tecnologie che facilitano la nascita di sistemi C-ITS.

Nonostante ciò, la realizzazione di tali sistemi richiede molte risorse in termini di tempo ed investimenti a causa della loro ampia e complessa infrastruttura. Inoltre, poiché tali sistemi riguardano la sicurezza stradale e le vite umane, la necessità di eseguire test adeguati in ambienti controllati è di fondamentale importanza. Per superare questi problemi sono state definite una serie di metodologie di sviluppo e testing, tra cui c'è l'uso di strumenti di simulazione dedicati.

In questa tesi viene presentata una nuova piattaforma di simulazione in grado di simulare importanti casi d'uso delle future strade intelligenti. L'obiettivo è quello di costruire uno strumento di supporto per studenti di ingegneria informatica, ingegneria dei trasporti e della mobilità interessati ad approfondire lo studio dei sistemi C-ITS. Il lavoro è stato basato sugli standard SAE e ETSI per il formato dei messaggi e ha avuto un particolare interesse riguardo possibili future tipologie di road-side units (RSUs). La piattaforma è in grado di consentire la realizzazione di prototipi per future evoluzioni e miglioramenti dei sistemi C-ITS. La nostra soluzione è stata sviluppata utilizzando lo strumento di simulazione Anylogic. Una valutazione qualitativa è stata fatta al fine di valutare le caratteristiche del software di simulazione e i casi d'uso implementati. L'obiettivo è stato raggiunto poichè il simulatore sviluppato è stato testato con successo.

Parole-chiave: C-ITS, sistemi di trasporto, sicurezza stradale, SAE, ETSI, Anylogic

Resumo

O desenvolvimento de Sistemas Inteligentes de Transporte Cooperativos (C-ITS) tem o potencial para desempenhar um papel relevante na resposta à crescente procura por novos serviços de mobilidade e para sistemas e redes de transportes mais seguros, eficientes e sustentáveis. Nas últimas décadas houve desenvolvimentos notáveis nas novas tecnologias que facilitam o C-ITS.

No entanto, a realização de sistemas C-ITS exige muitos recursos em termos de tempo e investimentos, devido à sua infraestrutura grande e complexa. Além disso, dado que esses sistemas são componentes da segurança rodoviária e deles dependem vidas humanas, a necessidade de realizar testes adequados em ambientes controlados é de suma importância. Para superar esses problemas, uma série de metodologias de desenvolvimento e teste foram definidas, entre as quais se inclui o uso de ferramentas de simulação bem adaptadas.

Nesta dissertação é apresentada uma nova plataforma de simulação capaz de representar importantes casos de uso para futuras estradas inteligentes. O objetivo é construir uma ferramenta de apoio a estudantes de engenharia informática, e engenharia de transportes e da mobilidade, interessados em aprofundar o estudo de sistemas C-ITS. O trabalho foi baseado no padrões SAE e ETSI para o formato de mensagem e tinha uma preocupação especial com possíveis tipos futuros de road-side units (RSUs). A plataforma apresentada tem potencial para permitir a realização de protótipos para futuras evoluções e aprimoramentos para sistemas C-ITS. A nossa solução foi desenvolvida usando a ferramenta de simulação Anylogic. Uma avaliação qualitativa foi feita para avaliar os recursos do software de simulação e os casos de uso implementados. O objetivo foi atingido, pois a ferramenta desenvolvida foi testada com sucesso.

Palavras-chave: C-ITS, sistema de transporte, segurança rodoviária, SAE, ETSI, Anylogic

Contents

List of Figures

Acronyms

ABS Anti-lock Breaking System

ADAS Advanced Driver Assistance Systems

C-ITS Cooperative Intelligent Transport System

CAM Cooperative Awareness Message

CU Control Unit

DENM Decentralized Environmental Notification Message

ESC Electronic Stability Control

ETSI European Telecommunications Standards Institute

FOT Field Operational Test

GHG Greenhouse Gases

GLOSA Green Light Optimal Speed Advisory

NHTSA National Highway Traffic Safety Administration

OBU On-Board Unit

RSU Roadside Unit

SAE Society of Automotive Engineers

V2I Vehicle-to-Infrastructure

V2P Vehicle-to-Pedestrian

V2V Vehicle-to-Vehicle

V2X Vehicle-to-Everything

Chapter 1

Introduction

In the last few years the transport sector is having remarkable changes. A wave of technological innovation and disruptive business models has led to a growing demand for new mobility services. At the same time, the sector is trying to respond to the pressing need of making transport safer, more efficient and sustainable.

According to one of the latest National Highway Traffic Safety Administration (NHTSA) reports, road accidents are still a relevant cause of losses of lives. In 2015, in the USA, there were an estimated 6,296,000 police-reported traffic crashes, in which 35,092 people were killed and an estimated 2,443,000 people were injured. [...] On average, 96 people died each day and one person was killed every 15 minutes in motor vehicle crashes [1]. Furthermore, analysing the economic impact of motor vehicle accidents happened in the 2010 in the United States, property damage costs for all crash types (fatal, injury, and property damage only) totalled $76.1 billion, representing 31% of all economic costs [2].

Considering instead, the environmental point of view, the transportation sector is also one of the main sources of pollution. Indeed, according to the Inventory of U.S. Greenhouse Gas Emissions and Sinks 1990-2015, this sector, with a share of 27%, is the second leading source of Greenhouse Gases (GHG) emissions in the United States, just behind electricity, and between 1990 and 2015, GHG emissions in the transportation sector increased more in absolute terms than any other sector [3].

In this scenario, digital technologies may drive those changes supporting the development of future mobility and reducing losses of lives, related costs and pollution.

1.1 Cooperative Intelligent Transport System

In many respects today's vehicles are already connected devices. However, in the very near future they will also interact directly with each other and with the road infrastructure.

A Cooperative Intelligent Transport System (C-ITS) connects vehicles (i.e. cars, buses, ambulances, etc.) and the infrastructure (roads, traffic lights, etc.) through the use of Information and Communication Technologies (ICT) in a way that road users and traffic managers share and use information to coordinate their actions (figure 1.1). This cooperative element is expected to significantly improve road safety, traffic

Figure 1.1: Cooperative Intelligent Transport System [5]

efficiency and comfort of driving, by helping the driver to take the right decisions and adapt to the traffic situation [4].

As a collateral benefit, C-ITS may guarantee a better usage of the current road network capacity, therefore, the construction of new and expensive road infrastructures to satisfy the increasing demand of mobility is no longer necessary. This could also relieves governments from collecting funds to finance those works.

The basic architecture of a C-ITS, depicted in figure 1.2, is made of multiple components, some installed into the road infrastructure and others on board of vehicles. Those components are:

- **Roadside-unit (RSU):** it is a device that is installed along the road that is able to communicate, within a certain distance, with vehicles passing nearby. It may also have a connection to a control unit to exchange information about traffic status and alerts
- **On-Board-unit (OBU):** device installed onto a vehicle that is able to put in communication the vehicle itself with the infrastructure
- **Control unit (CU):** a site where information gathered by RSUs is collected and analysed. It may also be a source of information (e.g. alerts) that is spread into a specific area to notify drivers about a particular event.

The interconnection of various C-ITS entities can be achieved through the usage of Vehicle-to-Everything (V2X) technologies, that by means of specific protocols enable the message exchange. According to the involved entities, those technologies may be classified as follows:

- **Vehicle-to-Vehicle (V2V):** interconnection used by 2 or more vehicles to communicate among themselves

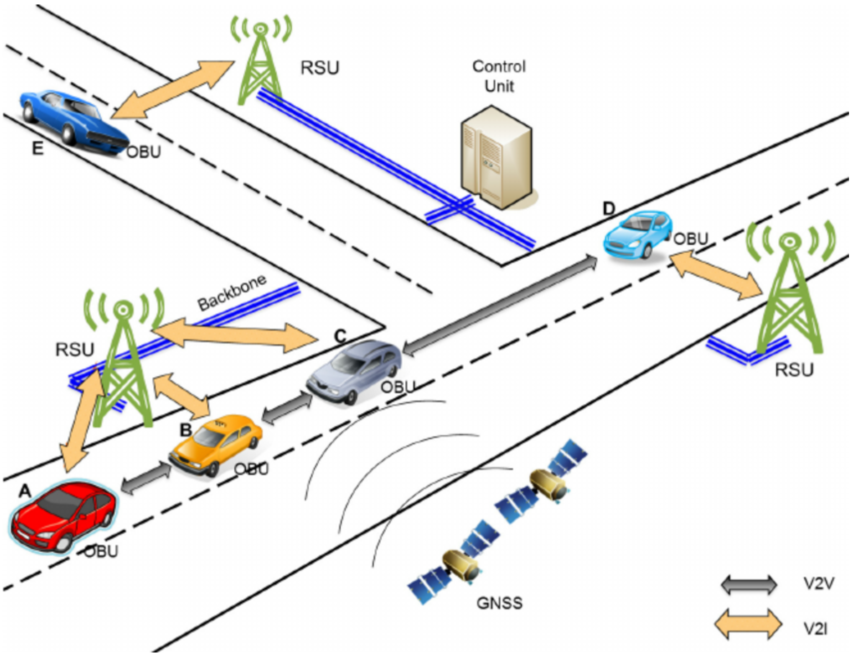


Figure 1.2: C-ITS infrastructure [6]

- **Vehicle-to-Infrastructure (V2I):** interconnection used by cars to communicate with the infrastructure. The opposite communication is defined as *Infrastructure-to-Vehicle* (I2V)
- **Vehicle-to-Pedestrian (V2P):** interconnection between vehicles and pedestrians and vice-versa

Possible C-ITS applications

Nowadays, an increasing number of modern vehicles have Advanced Driver Assistance Systems (ADAS) that alert the driver of hazardous road conditions, and in some cases, slow or stop the vehicle. Examples of those systems are applications like blind spot assist, lane departure warning, and forward collision warning. Those systems use the on board installed equipment like LiDAR[1], cameras and radars without any interaction with others vehicles or road infrastructure.

Systems like LiDAR and cameras have a reduced range of visibility of the surrounding environment due to the limited line-of-sight. This means that they have a 360-degree view only at a dozens of meters far from the vehicle. The vehicle's ability to "see" further down the road (at a blind curve or in bad weather conditions) may be improved thanks to V2X technologies. An example of possible applications is the road hazard warning where an alert is sent to vehicles in order to notify them, for instance, about accident or road works.

Another emerging technology that will change the transport sector is the autonomous driving where vehicles are able to drive from door to door without the driver intervention. Most autonomous vehicles currently in development are based on a perception subsystem consisting of on-board sensors, which build a map of the vehicle's environment, and a control subsystem that governs the longitudinal and lateral motion of the vehicle [8]. But, as traditional ADAS systems, also in this case the limited line-of-

[1] LiDAR - Light Detection And Ranging

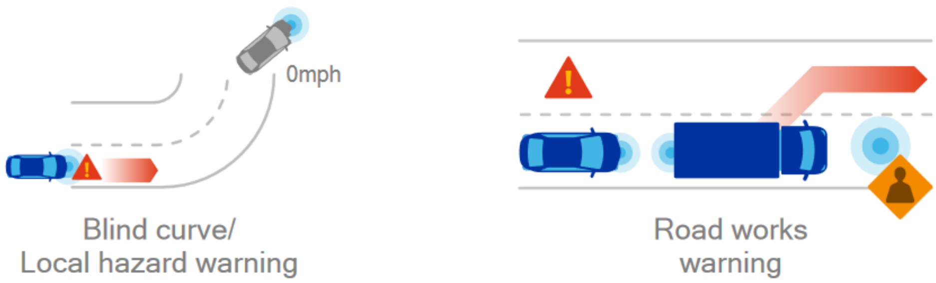


Figure 1.3: Road hazards warning examples [7]

sight of sensors allows the vehicle to detect only adjacent objects (cars, pedestrians, etc.). Secondly, due to the lack of vehicle cooperation, the management of complex manoeuvres is still hard to solve. An example could be a car entering a motorway using a slip road, in fact, in this situation it is necessary to have a sort of cooperation among drivers in order to smoothly join into the normal traffic flow. The necessary cooperative element of future autonomous vehicles may be enabled by C-ITS systems thanks to the usage of V2X communication, leading to the realisation of Connected Automated Vehicles (figure 1.4).

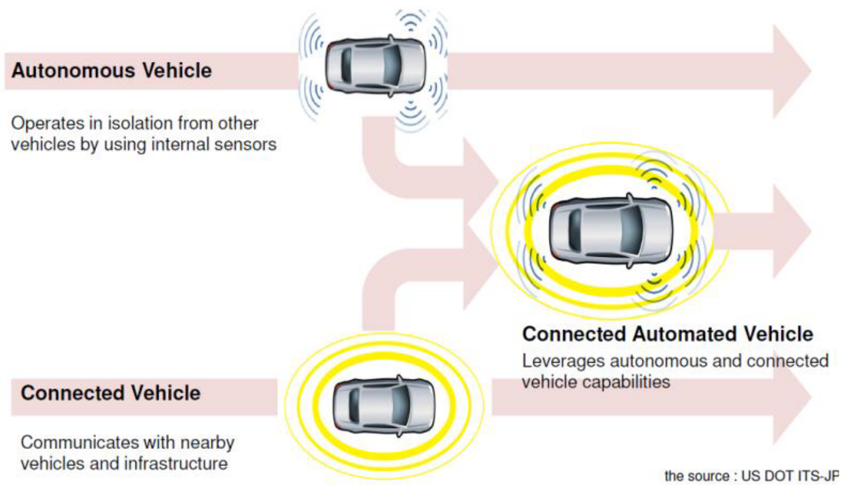


Figure 1.4: Connected Automated Vehicle [9]

1.2 C-ITS issues and challenges

Normally, any kind of application, before being delivered to customers in a real working environment, needs to pass a testing phase which goal is to ensure that it fulfils the starting requirements and that there are no bugs in the implemented features. This is even more important for C-ITS applications because many of them are safety-related, in the sense that possible system faults and misbehaviours may cause fatalities with dangerous effects for the involved people. For this reason, it is not possible to test such applications on real roads, at lest until they get a well defined level of reliability.

Another problem related to real road testing is that a C-ITS system has a relatively complex architecture involving not only the infrastructure of connected RSUs, but also vehicles equipped with proper on-board units, hence its realisation requires a remarkable amount of resources in terms of time and investments. All this leads to the necessity of implementing and testing applications in an environment that guarantees, at the same time, safety and reasonable costs.

1.3 Objectives

The Departamento de Engenharia Informática (DEI) of the Instituto Superior Técnico, in Lisbon, is aiming at building a reduced scale prototype of a basic C-ITS system able to provide an environment where it is possible to realise a first implementation, and relative testing, of those type of applications. Besides that, another goal is to build a possible teaching support tool able to help computer, transportation and mobility engineering students, interested in transports and mobility sector, to learn the organization of C-ITS systems. This thesis work is willing to pave the way for the realisation of this prototype and simulation environment.

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 organized in three phases: definition, implementation and evaluation.

The definition phase has two main objectives. The former is the characterisation of the main use cases taken from real road scenarios, considering both standard and hazardous situations. Possible scenarios are: intersections with or without traffic light, pedestrian crossing, hazardous road conditions and so on and so forth. The latter is the definition of a set of messages that applications use to exchange information. Those messages have been defined starting from the current ETSI (European Telecommunications Standard Institute) technical specifications regarding C-ITS applications.

The second phase is the implementation of the simulation platform. Starting from previously defined use cases, a basic implementation of some C-ITS applications will be done. Examples are: icy road warning, pedestrian crossing and Active Road Signs.

Lastly, the evaluation phase that has as its main scope, the analysis of the simulation software characteristics and the use cases implemented into the simulator.

1.4 Document overview

The present document is organized as follows:

Chapter 2 - State of the art description of the current methodologies, tools and standards adopted by the C-ITS sector

Chapter 3 - Future roads scenarios introduction to future scenarios in which C-ITS applications may be applied improving safety and efficiency

Chapter 4 - Simulation platform organization description of the implemented model architecture and message format definition

Chapter 5 - Set up of the simulation environment deep overview of all the implemented C-ITS applications highlighting all the relevant aspects and possible issues

Chapter 6 - Evaluation presents the analysis and the results of the implemented applications

Chapter 7 - Conclusions a summary of the thesis achievements and a brief description of what could be improved in future works

Appendix A - Road-sign container definition description of the Road-sign container format for some relevant road signs

Chapter 2

State of the art

The state-of-the-art analysis is focused mainly on the study of current C-ITS standards and the methodologies used by academic community and other stakeholders (e.g. car manufacturers and IT companies) to develop and test future C-ITS applications.

The first part is an analysis of the current standards that have been issued by two important American and European organizations, respectively the Society of Automotive Engineers (SAE) and the European Telecommunications Standards Institute (ETSI). In particular, since our work is mainly related to application development we will focus on protocols defining message formats and applications architecture.

The second part, instead, analyses what are the current methodologies, tool and facilities used by the academic community and other stakeholders to develop and test future C-ITS applications in a way that guarantee accuracy, reliability and safety. In particular there is a survey of current test site installations and simulation platforms used to validate the developed systems.

2.1 Current C-ITS standards

Due to their nature, C-ITS systems are characterized by the cooperative element. This means that all the components of those system, namely vehicles, RSUs and control unit, must be able to interact in an effective way. Besides that, due to the wide extension of cooperative ITS systems, it is also necessary to consider the fact that there are many competing stakeholders among which there are car makers, IT companies, telecom operators as well as road infrastructure managers and owners.

That said, it is evident that in order to get all the advantages of C-ITS systems and to make this process as fast as possible it is essential the definition of common standards and protocols. For this reason, the main active standardization organizations in the C-ITS field, namely SAE, ETSI and IEEE, have issued a series of standards.

The issued standards are not only related to the definition of a new communication protocol between vehicles and road infrastructure, but they define a whole protocol stack based on the ISO OSI reference model. The two main protocol architectures are the DSRC and C-ITS stacks, which are depicted in figure 2.1. Both standards cover messaging standardization, that is the definition of a well defined set of

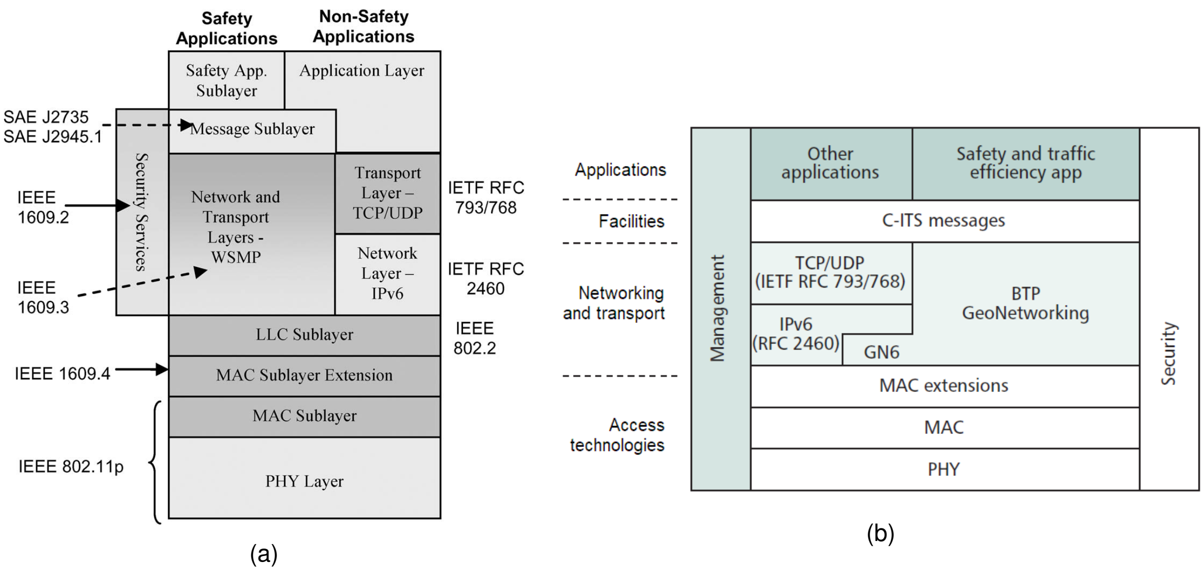


Figure 2.1: C-ITS communication stack developed in USA (a) and EU (b) [10], [11]

messages exchanged by applications.

As stated in [12], with the exception of some services, such as multimedia or common Web access, many ITS services have common communication requirements:

- **Periodic status exchange:** ITS services typically need to know about the status of vehicle or roadside terminals. This implies the periodic exchange of data packets with information about location, speed, identifier, etc.

- **Asynchronous notifications:** this kind of messages are used to inform about a specific service event. In contrast to the previous status messages, the reliable delivery of these messages to a single terminal or a group of them is usually a key requirement.

In order to support these two type of communication requirements, both the SAE and ETSI have defined some standards describing message format. In the next two sections there is an overview of those message sets, in particular with a deeper description of the European standards.

2.2 SAE message set

At the top of the protocol stack in figure 2.1a, the Application Layer includes application processes and additional protocols that provide direct support to applications. An example of the latter is the SAE J2735 DSRC Message Set Dictionary standard, which defines fifteen messages that collectively enable a core set of DSRC applications. In the figure 2.2 is shown the list of all the defined messages and their description.

This standard has not been deeply surveyed because it is not freely available on the Internet. Even though this limitation, a complete study of the current set of standards has been possible. In fact, both SAE and ETSI standards are perfectly compliant covering the standardization of the very same aspects,

Message Type	Purpose
A La Carte Message	Generic message with flexible content
Basic Safety Message	Conveys vehicle state information necessary to support V2V safety applications
Common Safety Request	A vehicle uses this to request specific state information from another vehicle
Emergency Vehicle Alert Message	Alerts drivers that an emergency vehicle is active in an area
Intersection Collision Avoidance	Provides vehicle location information relative to a specific intersection
Map Data	Sent by RSU to convey the geographic description of an intersection
NMEA Corrections	Encapsulates one style of GPS corrections – NMEA style 183
Probe Data Management	Sent by RSU to manage the collection of probe data from vehicles
Probe Vehicle Data	Vehicles report their status over a given section of road; aggregated to derive road conditions
Roadside Alert	Sent by RSU to alert passing vehicles to hazardous conditions
RTCM Corrections	Encapsulates a second style of GPS corrections – RTCM
Signal Phase and Timing Message	Sent by RSU at a signalized intersection to convey the signal's phase and timing state.
Signal Request Message	A vehicle uses this to request either a priority signal or a signal preemption.
Signal Status Message	Sent by RSU to convey the status of signal requests.
Traveler Information	Sent by RSU to convey advisory and road sign types of information

Figure 2.2: DSRC message set [10]

namely communication stack architecture, message set definition and so on and so forth. Therefore, the survey has been carried on analysing the ETSI documents that are completely open and available.

2.3 ETSI message set

2.3.1 Cooperative Awareness Message

Cooperative Awareness Messages (CAMs) are messages exchanged in the ITS network between ITS-Ss[1] to create and maintain awareness of each other and to support cooperative performance of vehicles using the road network. A CAM contains status and attribute information of the originating ITS-S [13]. The detail of the message structure is depicted in figure 2.3 and described as follows:

- **ITS PDU header:** contains protocol version, station id, message id, and the time-stamp
- **Basic container:** contains ITS station type (vehicle, RSU, etc.) and the geographic position
- **High frequency container:** contains fast-changing (dynamic) status information of the vehicle (speed, heading, etc.)

[1] ITS-S - ITS station

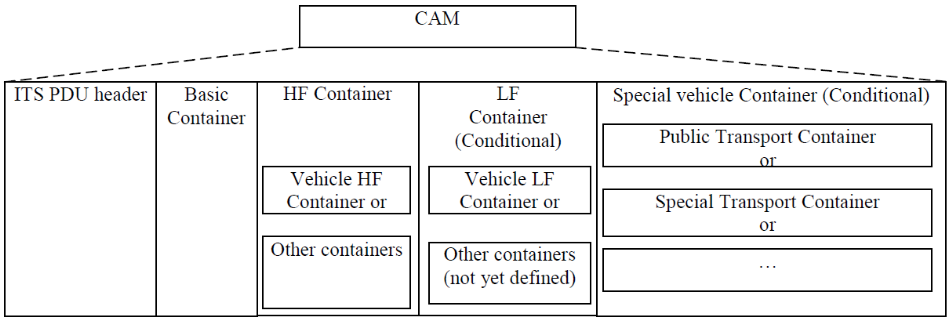


Figure 2.3: General structure of a CAM [13]

- **Low frequency container:** contains static or slow-changing vehicle data (e.g. status of exterior lights)
- **Special vehicle container:** provides further status information for special vehicles (e.g. rescue vehicle)

2.3.2 Decentralized Environmental Notification Message

A Decentralized Environmental Notification Message (DENM) is an event-triggered message used by ITS applications to alert road users of a hazardous event using ITS communication technologies. The DENM contains all the information related to the detected event [14].

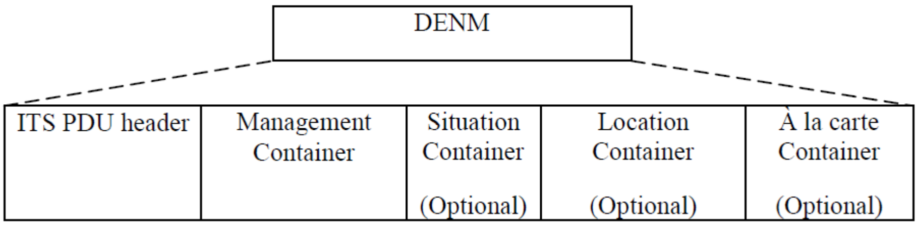


Figure 2.4: General structure of a DENM [14]

The detail of the message structure (figure 2.4) is as follows:

- **ITS PDU header:** contains protocol version, station id, message id, and the time-stamp
- **Management container:** contains basic information about the signalled event (position, time-stamp, etc.)
- **Situation container:** contains further information that describes the detected event (i.e. event type, cause code, etc.)
- **Location container:** contains the location of the detected event
- **À la carte container:** contains additional information that is not provided by other containers like application specific data

According to specific the situation, the message may be retransmitted multiple times for a certain time or till its cancellation.

2.4 Current implementation and testing methodologies

The aforementioned standards, in order to be effective and encompass all the aspects of Cooperative ITS systems must be validated, and if necessary reviewed, through proper analysis of real working applications. Furthermore, even though they define most of the aspects about the communication architecture, the real implementation is still left to manufacturers. For this reason, it is of paramount importance to perform an accurate testing phase not only to verify that the implementation adheres to the standard, but also to check the interoperability among different implementations performed by different vendors.

Surveying the current literature and some European funded projects it has been possible to detect that there are three main way of developing and testing applications. These methodologies are: real road testing, the use of dedicated testing facilities and simulations performed on well-suited tools.

2.4.1 Real road testing

One of the first European projects with the aim of testing C-ITS systems was Drive C2X[2]. It was an integrated project co-funded by the European commission involving many car makers such as Volvo, Ford and BMW, components suppliers (Bosch and Continental), communication equipment vendor like NEC and others research and government organizations.

This initiative had the goal of creating one of the first large-scale testing of intelligent vehicle safety systems with ordinary drivers and real-world conditions. The tests have been carried out in a total of seven national test sites located from the North to the South of Europe. Those test sites encompass both common used roads and specific testing site closed to normal traffic. An example is the Autostrada del Brennero (A22), an Italian motorway connecting the Austrian border on the Brennero pass with the city of Modena, equipped with a proper technological infrastructure including roadside units.

2.4.2 Dedicated testing facilities

The testing on dedicated facilities represents the safest way of testing C-ITS applications, especially the ones with human beings exposed to possible injuries. Indeed, they are specific built road infrastructure that are closed to normal traffic, where it is possible to simulate various real road scenarios as well as particular hazardous situations.

For instance, Anaya et al. [15] have performed some tests in order to demonstrate the operation of the Vulnerable Road Users (VRU) detection system. The main scope of this application is to avoid that larger and less vulnerable vehicles like cars and vans hit and run over more vulnerable road users, namely bikers and motorcyclists. Those tests were conducted on the Universidad Politécnica de Madrid (UPM) facilities of South Campus, under free traffic flow conditions.

[2]Drive C2X project - http://www.drive-c2x.eu/project

Instead, at the Swedish Test site in Gothenburg, Sweden, Stahlmann et al. [16] in collaboration with AUDI AG have performed Field Operational Test (FOT) for evaluate a Green Light Optimal Speed Advisory (GLOSA) system. This is one of the various C-ITS applications capable of introducing environmental benefits through lowering CO2 emissions and fuel consumption. To this end, information about traffic light signal phases is broadcast to vehicles approaching an intersection. Speed recommendations are then calculated by the vehicle to pass the traffic light avoiding unnecessary stops and acceleration manoeuvres.

In the following section there is a general description of an European test cite facility, highlighting its characteristics, features as well as road scenarios that is possible to reproduce.

AstaZero

AstaZero is a testing site placed in Hällered, Sweden. The name is a combination of ASTA - Active Safety Test Area - and Zero, which refers to the Swedish Parliament's vision for road safety with zero dead and seriously injured in traffic.

The test environments comprise a 5.7 kilometre rural road lane, a city area with four districts of buildings and streets, a multilane road for multi-lane traffic and a high speed area for high speed tests that makes it possible to test advanced safety systems and their functions for all kinds of traffic situations. It provides a complete roadside infrastructure for V2I and V2V communication, DGPS[3] system and a set of dummies like pedestrians, balloon cars and animals.

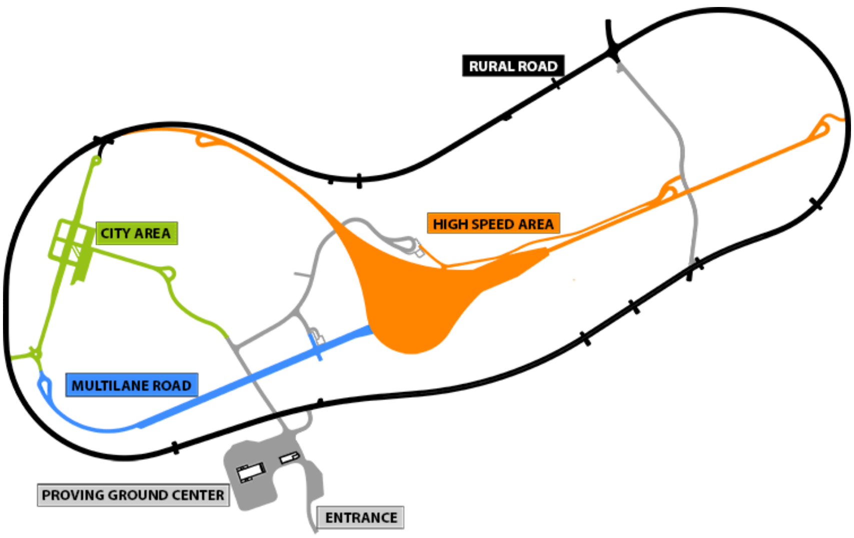


Figure 2.5: The AstaZero testing site [17]

Among all the conducted tests, this site was also used for the AutoNet2030 project[4] with the aim of testing a co-operative automated driving technology. The project results were evaluated through drive-testing and simulation tools and showcased in late 2016.

[3]DGPS - Differential Global Positioning System
[4]AutoNet2030 project - http://www.autonet2030.eu/

2.4.3 Simulation tools

As described in previous sections to evaluate the improvements that can be achieved by C-ITS applications, scenarios are defined and field tests are carried out. However, the realisation of field tests is rather complex and expensive. For this reason, simulations are essential to prepare the tests in real world situations and reduce their costs [18]. Furthermore, it is necessary that those simulation tools are enough accurate in order to model realistic traffic scenarios and communication amongst the participating entities.

Thanks to many academic initiatives and collaborations with car makers some simulation tools for C-ITS systems and V2X technologies were created. Those tools are:

- **VEINS** - Vehicles in Network Simulation [19]
- **VSimRTI** - V2X Simulation Runtime Infrastructure [20]
- **MSIECV** - Multiple Simulator Interlinking Environment for C2CC[5] in VANETs [21]
- **Anylogic simulator** - agent based modelling tool [22]

Each software is characterized by its own architecture, but analysing them it is possible to find some similarities. Indeed, the typical architecture is an aggregation of different simulators that may interact through properly designed interfaces. Examples of well-established simulators are: OMNeT++, an event-based network simulator, and SUMO, a road traffic simulator. Starting from these simulators it is possible to extend their functionalities to offer a comprehensive suite for modelling C-ITS systems.

Examples of research works based on simulator implementations were surveyed. For instance, Katsaros et al. [23] to monitor the impacts of GLOSA system on fuel and traffic efficiency used an integrated simulation tool based on the Fraunhofer VSimRTI. Instead, Szczurek et al. [24] through the MITSIM software, a microscopic traffic simulator, performed an investigation for analysing the relevance of the Emergency Electronic Brake Light (EEBL) application. The system works by disseminating reports about vehicles that perform emergency deceleration in an effort to warn drivers about the need to perform emergency braking. Vehicles that receive such reports have to decide whether the information contained in the report is relevant to the driver and warn him if that is the case.

Despite that, all those simulators mainly address vehicles and RSUs related aspects leaving aside other scenarios involving pedestrians or other means of transport. Therefore, a simulator able to model a comprehensive cooperative ITS system is still missing.

[5] Car to Car Communication (C2CC)

Chapter 3

Future roads scenarios

This chapter gives a description of future intelligent road scenarios. Firstly, it has a more in-depth description of what are the equipment needed to implement the road infrastructure, namely the roadside unit. Secondly, there is an overview of some scenarios where C-ITS technologies may be involved to improve safety and efficiency. Those scenarios have been taken from current literature and standards. The chapter includes also the definition of new use cases that have not been yet defined in the current literature as well as a specific type of roadside units.

All the presented scenarios have been chosen because they put in light the interactions that occur among different means of transports and also because it is possible to evaluate the cooperative aspect of C-ITS systems.

3.1 Roadside units

One of the main components of a C-ITS system is the RSU, a device installed along the road able to communicate with vehicles and the central unit. It may accomplish different functions according to the installed hardware and software, some applications are: tolling, emergency vehicle management, traffic control.

3.1.1 Milestone RSU

Vehicular networks are a challenging technology because they are characterized by rapidly changing topology, frequent fragmentation and limited redundancy. Indeed, since vehicles are entities continuously moving, the network topology connecting them and to RSUs varies rapidly causing fragmentation. In such situations messages addressed to cars or RSUs might not be successfully delivered due to the missing connection between car to car or car to RSU.

Studies show that in areas of low vehicle density, important safety broadcasts can take more than 100 seconds to reach all nearby cars. On the other hand, dense networks with too many vehicles can be overwhelmed with traffic and signalling, requiring careful coordination between the nodes to ensure proper operation [25].

Figure 3.1: Roman milestone in the city of Bisceglie (Puglia - Italy) [27]

One way to overcome these problems is to deploy roadside units along the road tracks. This solution recalls the ancient milestones used by Romans starting from the year 123 B.C. (figure 3.1). In the Roman provinces, their goal was to show the travel distance from the capital city or the main cities [26]. Instead, nowadays, their place has been taken by distance marker posts showing the distance value from the starting point of the road and, in some cases, also the road number.

The installation of these RSUs along the road at a constant distance can avoid the fragmentation problem guaranteeing a seamless connectivity along the whole road. This also permits to broadcast warning messages (e.g. road accident) further to reach cars that are still far from the interested area, for example, to advice them to change their path to the destination. The wide presence of those RSUs guarantees also full control of road status (e.g. ice presence, fog, etc.) and traffic congestion.

3.1.2 Active Road Sign

Nowadays, road signs are passive entities made of a pole and a signpost laying on the side of the road to either impose a prohibition (e.g. speed limit), notify a warning (e.g. dangerous curve) or change intersection priority (e.g. give way). It is up to the driver to pay attention to those signs, understand their meaning and then behave accordingly. Furthermore, very often, road signs are in such bad conditions that it is very difficult or even impossible to notice them: for example they could be damaged or covered by a tree or a bush.

On the contrary, the Active Road Sign, is an RSU that mimics a traditional signpost with the difference that it is able to communicate with the approaching vehicles and send to them the necessary information. For instance, in case of a speed limit, the signal will convey the speed limit value through a proper message. Another possible advantage of this system is the possibility of remote controlling and updating the sign without a physical intervention of human operators, for example for maintenance or road administration purposes (e.g. change speed limit, impose a road deviation or signal road works).

This application has not been defined in the current ETSI standard, indeed one of this thesis objective is to provide a first definition of this use case and provide a first implementation inside the simulator.

3.2 Intersections

In the following sections there is a detailed description of real road situations, mainly focusing on intersections and crossings. For each section is included a picture that illustrates the real scenario. Some of those pictures are referred to left-hand driving situations (e.g in the United Kingdom) but all the mentioned considerations can be applied to right-hand driving situations as well.

3.2.1 Pedestrian and bicyclist crossing

Pedestrian and bicycle crossing (figure 3.2) is the meeting point of different road users, namely pedestrians, cyclists and drivers. All of them have to interact and coordinate in order to safely pass the crossing. Unfortunately, due to the possible low level of attention of the involved people (provoked by many distractions), accidents may still happen. The pedestrian crossing can be equipped with a suitable set of devices that can detect and interact with road users in order to warn them about the possible hazards. This mainly includes the possibility to alert the cars that are approaching the crossing.

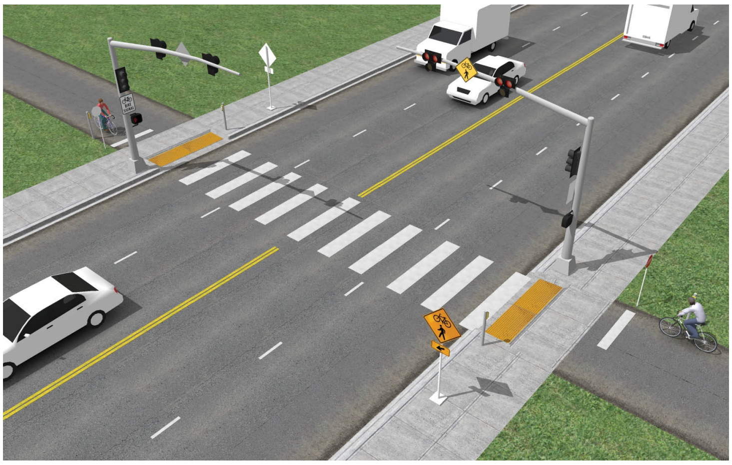


Figure 3.2: Pedestrian and bicyclist crossing [28]

This use case permits the evaluation of different methods that may be used to improve its safety as well as the implementation and evaluation of V2I and V2P message exchange.

3.2.2 Level crossing

The level crossing is where roads and railway meet together (figure 3.3). Since trains have priority over cars and travel at a relatively high speed, to make it safer and avoid collisions, the level crossing is provided with specific devices such as signs, lights, bells and often, but not always, barriers. Besides that, they are also equipped with standardized technologies allowing the communication between train and the level crossing in order to let it operate correctly even if in case of failures.

In a C-ITS system, the level crossing may be thought as an entity with its own working principles, able to interact with approaching cars informing them about the train arrival. In this way the driver can be early warned about a possible train passage rather than check it by himself when arrives at the crossing (especially the ones without barriers and scarce visibility).

Figure 3.3: Level crossing [29]

This use case has been considered as relevant for the following reasons:

- to define and analyse interactions between different transport entities, namely cars and trains
- improve the safety of the level crossing for further reduce the damages provoked by a collision

3.2.3 Intersection with or without traffic light

This is a crossing where the priority is simply established by either stop/give way signals or traffic light. Sometimes this kind of intersections may be characterized by low visibility and even the drivers might not pay the right attention to them causing car crashes.

In C-ITS scenarios, the traditional road signs and traffic light may be substituted by roadside units for improving the safety of the crossing. For instance it may be possible to inform drivers if there are vehicles coming from others directions, or in case of a traffic light, to send them the information about its current status and timing. All this can be achieved implementing a V2I communication between vehicles and road infrastructure.

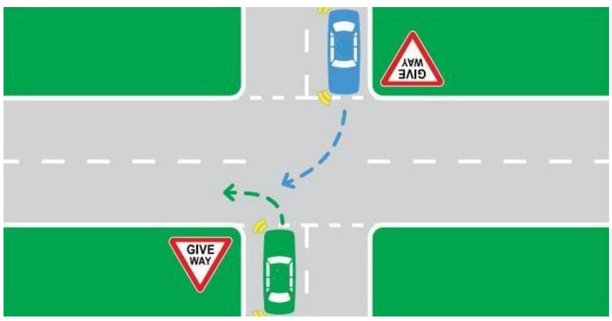


Figure 3.4: Intersection with modified priority [30]

This use case shall be considered for:

- evaluating different implementations of the algorithm to control the intersection
- evaluating V2V and V2I interactions defining the appropriate message exchange and formats
- experimenting the possibility of optimizing the crossing working principle, for instance using Green Light Optimal Speed Advisory (GLOSA) or other systems to dynamically extend/reduce a light phase duration according to the current traffic congestion
- put the basis for future autonomous driving technologies cars able to manage more easily crossing situations thanks to the use of V2X technologies instead of special cameras and complex image processing algorithms

3.2.4 Slip Road

The slip road[1] use case is a particular situation because it involves vehicles driving at high speed and slow vehicles that accelerate to merge into the normal traffic flow (figure 3.5). This scenario is characterized by the fact that there is a sort of coordination among drivers.

In other words, the driver that is going to join the main road has to take care of approaching vehicles, in particular their occupied lane and speed, and therefore decide whether he has to speed up, keep the speed or break. On the other hand, the vehicles driving along the main road may move, if possible, to the internal lanes to facilitate the merging of others vehicles.

All these decisions and actions made by motorists may be supported and improved by a specific C-ITS application executed on each vehicle. This application, thanks to vehicle-to-vehicle message exchange, can give advices to the driver about manoeuvres that he should do for merging the normal traffic flow in a safe and smooth way.

Figure 3.5: Slip road [31]

This scenario may be even more complicated in some particular situations, creating so called corner cases. A first corner case is a slip road made of multiple lanes: in this case drivers do not only coordinate with other drivers on the main road, but also with vehicles doing the same manoeuvres. Secondly,

[1] Slip Road - a road entering or leaving a motorway or dual carriageway

Figure 3.6: Slip road with multiple lanes and close separated roads [32]

another situation that is not typical to only this case but can be even found in other scenarios, is when there are totally separated roads but very close to one another bringing to different directions.

From an implementation point of view, both cases certainly cause new challenges about vehicle communication and coordination since there might be undesired effects causing potential accidents.

These use cases shall be considered for:

- implementing and evaluating the V2V message exchange
- evaluate complex situations (roads that are very close to one another, multiple lanes)

3.3 Hazardous situations

3.3.1 Generic road hazards

Many road hazards might be encountered while driving along roads, for instance obstacles (rocks, animals, etc.) or bad road surface conditions (ice, mud, oil). When these situations occur the vehicles that are involved can warn the others drivers that are moving towards the same place so that they can anticipate and change the style of driving in order to improve their own safety (e.g. slow down or change path). The evaluation of this use case permits to:

- implement and evaluate the vehicle to vehicle (V2V) message exchange
- emulate various road hazards
- evaluate the possible countermeasures that shall be taken by the vehicle

3.3.2 Stationary vehicle on the side of the road and road accident

This road situation is one of the most dangerous ones because it represents a sudden and not signaled obstacle that is present on the side of the road or even exactly on one of the road's lanes nearby which

Figure 3.7: Possible road hazards: animals crossing [33], obstacles on the road [34] and icy road [35]

there might be people waiting for assistance. This situation becomes even more dangerous during night hours and with low visibility conditions (road with many curves, rain, fog, etc.).

The accidents provoked by this condition may be mitigated installing appropriate on board devices that in case of problem may warn the vehicles that are driving towards the direction of the steady vehicle and interact with the road infrastructure to request assistance. This use case is useful to evaluate:

- message exchange among vehicles through the V2V technology
- minimum distance at which the alert message shall be sent according to the type of the road and road speed limit
- message exchange with the road infrastructure via V2I technology to send an assistance request

Figure 3.8: Stationary vehicle [36]

Some of the described scenarios have been implemented in the simulation platform. In the next chaper there is the description of the simulation platform and the set of which scenarios have been included. Then in chapters 5 and 6 there is the detail on how these scenarios have been implemented and evaluated.

Chapter 4

Simulation platform organization

As described in section 1.3 the thesis goal is to realise an initial version of a new simulation tool for C-ITS applications. This tool provides the possibility of reproducing a simplified city environment made of a road network with intersections, curves, pedestrian crossings and traffic lights. In addition, there is also a railway track that, in the points where it crosses the road network, is provided with level crossings. To give a better representation of the possible situations that can be encountered on real roads, the tool includes the also the possibility to simulate different dangerous situations like icy road or roadworks.

Besides the road and railway tracks, there is also a C-ITS infrastructure that includes a basic set of components, like roadside units and sign posts. Considering instead the road users, the simulator is provided with cars able to move throughout the city environment and already equipped with a basic set of functionalities enabling the communication with the C-ITS infrastructure. Finally, it allows also the representation of pedestrians moving throughout the city and crossing the roads at specific pedestrian crossings.

In addition to its basic features this tool allows also the definition of custom scenarios and applications running on cars or RSUs as well as the modification of existing ones.

4.1 The agent-based modelling and the Anylogic platform

The simulation platform has been developed through the creation of a model. Modelling is a way of representing real-world situations and problems in order to better understand and handle them. In many cases, it is not affordable to experiment with real objects to find the right solutions: building, destroying, and making changes may be too expensive, dangerous, or just impossible. In these situations it is possible to build a model that uses a modelling language to represent the real system. This process assumes abstraction: only the important detail are considered, all the rest is left aside. In this way the model is always less complex than the original system (figure 4.1).

In simulation modelling, there are three main methods to map a real world system to its model:

- System Dynamics
- Discrete Event Modelling

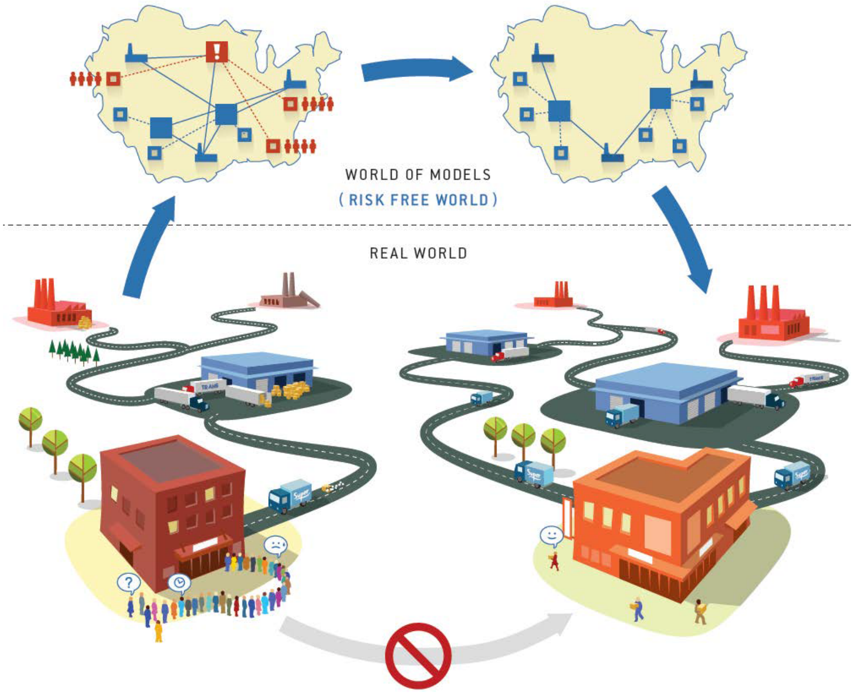


Figure 4.1: The modelling process [37]

- Agent Based Modelling

Each method serves a specific range of abstraction levels. System dynamics assumes very high abstraction, and it's typically used for strategic modelling. Discrete event modelling supports medium and medium-low abstraction. In the middle are agent based models, which can vary from very detailed models where agents represent physical objects to the highly abstract models where agents represent competing companies or governments.

The type of model that makes it natural to describe a C-ITS system is the Agent-Based Modeling (ABM). Indeed, the principle of this modelling method is to observe the whole system by characterizing the behaviour of the agents, of the system and the environment in which they interact and move. Then the system is modelled as a collection of autonomous decision-making entities called agents. An agent has a set of possible actions, rules of behaviour and variables that give an indication of its internal state. There can also be an event generator that will create events compatible with the reality of that agent.

Simulation modelling requires special software tools that use simulation-specific languages. Because of that the simulation platform has been developed thanks to the Anylogic technology. In particular, the Anylogic simulator is a software that allows the modelling and execution of a model that represents a complex system. Among all the included features it has also a set of built-in libraries such as:

- Rail Library for rail transportation, terminals, and yards.
- Pedestrian Library for pedestrian flows in airports, stadiums, stations, or shopping malls.
- Road Traffic Library for car, truck, and bus movement on roads, parking lots, and factory sites.

Pedestrian, Rail, and Road Traffic Libraries provide detailed physical-level simulation of objects' movement and interaction.

4.2 Simulation model architecture

The model organization reflects the C-ITS architecture defined in section 1.1 taking into account both components and their interconnections. Indeed, after an analysis of its organization it has been possible to identify various entities that can be modelled as agents. They are as follows:

- **Car** this agent models a car. It is able to move along the road and, through its connections, to interact with the other nearby agents.
- **RSU** it represents a fixed entities laying along the road track. According to its configuration it may carry out a different role. For instance, it may be a milestone RS or an active road sign.
- **Pedestrian crossing** it models an intelligent pedestrian crossing able to interact with both cars and pedestrians.
- **Level crossing** it represents an agent mimicking the behaviour of a level crossing. It can interact with both approaching cars and trains.
- **Icy road** it models an ice surface along the road. It interacts with the cars that pass over to let them know about the hazard

Considering instead the interconnections among the entities, for instance V2V and V2I, they are represented as inter-agent communications. Moreover, a set of messages that are exchanged by vehicles, roadside units and all the other systems have been defined. In the table below for each agent interaction there is a short description and type of exchanged messages.

Communication type	Description	Message exchanged
ARS2V	ActiveRoadSign-to-Vehicle	CAM
I2P	Infrastructure-to-Pedestrian	-
I2V	Infrastructure-to-Vehicle	CAM/DENM
LC2V	LevelCrossing-to-Vehicle	DENM
LC2T	LevelCrossing-to-Train	-
V2I	Vehicle-to-Infrastructure	DENM
V2PC	Vehicle-to-pedestrianCrossing	DENM
V2V	Vehicle-to-Vehicle	CAM/DENM
R2R	RSU-to-RSU	DENM

4.3 Inter-agent communication

A particular consideration has been given to the modelling of inter-agent communications in order to get the best representation of the real C-ITS communication system. For this reason has been done an analysis of all the possible interconnections established among the various C-ITS entities. From this analysis, besides the involved entities, it has been possible to highlight three common aspects, that are:

- the communication protocol
- the communication range
- the exchanged messages

About the communication protocol, since the objective was to have just a simple interconnection among agents, the modelling of the real communication channel has been done without implementing any specific protocol or standard. That is, agents establish connections among themselves and then exchange messages.

In real C-ITS systems the communication among different entities is limited to a certain distance. This is due to physical reasons (signal degradation) and also because it is not necessary to connect entities that are too far and hence do not need to share information. In order to represent this aspect, the inter-agent interconnection has been modelled in a way that it can be established only if two agents are within a certain distance. This has been achieved thanks to the possibility offered by the simulator that allows to perform a distance measurement between two agents.

The last aspects that was highlighted by the C-ITS interconnections analysis is the message exchange. Indeed, as described in section 2.1 there is a set of standards defining the message format. Since one of the main thesis objectives was focused on messages, their definition has been done starting from the current ETSI standards and which details are explained in the next section.

4.3.1 Custom message definition

The CAM and DENM message structure described in chapter 2 is fully compliant with the ETSI standards. In our specific application, the usage of this message format is not completely suitable for two main reasons. The first reason is that these messages are used in real applications, therefore contain a wide set of information that for our simulation model are either not necessary or not representable. Instead, in regard of the second aspect it has been necessary to add some other information to the CAM message in order to be able to represent information about active road signs. In fact, the standard does not provide any representation for this data, then it was necessary to define a new container and fields.

4.3.2 Custom CAM message

Considering the CAM message, in figure 4.2 is depicted the new message format derived from the ETSI standard definition, where the *LF* and *Special Vehicle* containers have been removed. On the other hand, in the *HF* container the *Road-sign* container has been added.

The *LF container* was removed because it contains some vehicle related information (e.g. exterior light status) that is neither relevant for our purposes nor possible to get it from the simulation environment

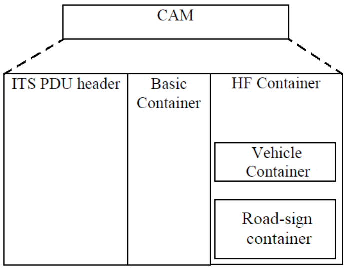


Figure 4.2: New structure of a custom CAM

and then manage it. The *Special vehicle* container was also removed because, in this phase, we are not interested in simulating special vehicles like ambulances or police cars in emergency state. In the *HF container* has been defined a new data structure, *Road-sign container*, that contains all the information generated by road-signals RSU. The following table describes the CAM's containers and their fields, as well as the data type of each field:

PDU part	**Content**	**Data type**
Header	protocolVersion	int
	messageID	int
	stationID	int
	timestamp	Date
Basic container	stationType	StationType
	geographicPosition	GeographicPositionXY
Vehicle container (opt.)	speed	double
	direction	TrafficDirection
	lanePosition	int
	vehicleWidth	double
	vehicleLength	double
Road-sign container (opt.)	type	RoadSign
	relevanceDirection	TrafficDirection
	...	...

Road-sign container

The road-sign container represents the data structure that contains information delivered by Active road sign RSU. The information generated by this kind of RSU is periodic, that is it is not generated only after a specific event occurred. For this reason the best message, between CAM and DENM, that shall be used for containing this data structure is the CAM message.

The message content varies according to the specific road sign type. Next follow two examples of messages broadcast by two corresponding specific active road signs:

1. **Dangerous curve**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	bendPositionXY	GeographicPositionXY

2. **Speed limit**

Road sign	Data	Type
	roadID	Integer
	speedLimit	Integer
	direction	TrafficDirection

In the two tables above are included two different data types: *TrafficDirection* type that can be either *forward* or *backward* and *GeographicPositionXY* a type containing the Cartesian coordinates, that are X and Y value. The full definition of the road-sign container and all the related details are included in the Appendix A.

4.3.3 Custom DENM message

Considering the DENM message, in figure 4.3 is depicted the new message format derived from the ETSI standard definition, where the *Location* and *À la carte* containers have been removed and the situation container has been completely redefined removing unnecessary fields. The two containers

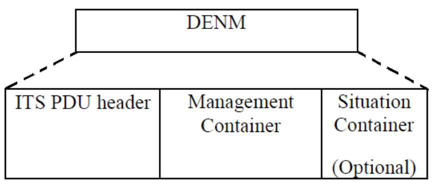


Figure 4.3: New structure of a custom DENM

were removed because they contain additional data that covers more specific situations in which we are not interested. Moreover, the *situation container* was completely simplified removing all the original fields and leaving just the *eventType* field that describes what type of event is broadcast by the originating entity (e.g. car, RSU). The following table describes the DENM containers and their fields, as well as the data type of each field:

PDU part	**Content**	**Data type**
Header	protocolVersion	int
	messageID	int
	stationID	int
	timestamp	Date
Management container	actionID	integer
	detectionTime	Date
	referenceTime	Date
	termination	int
	eventPosition	GeographicPositionXY
	relevanceDistance	double
	relevanceTrafficDirection	TrafficDirection
	validityDuration	Date
	transmissionInterval	Date
	stationType	StationType
Situation container (opt.)	eventType	eventType

Chapter 5

Set up of the simulation environment

After describing the characteristics of the simulation platform that is being developed and the definition of all the preliminary aspects, a deeper description of the real implementation work is presented. This description focuses on two different activities.

Firstly, the modelling of each agent presents what are their functionalities and how they have been implemented. Secondly, the description of all the steps dedicated to the implementation of the city environment, reporting some examples of deployed scenarios and the definition of the agents movements.

5.1 Use cases implementation

5.1.1 Car agent

This agent models a car able to move along a path and equipped with V2X technologies. Regarding the V2X connectivity it has 3 different connections: V2V, V2I and V2LC. Besides that, to simulate an ice sensor the agent has also a connection with the icy road agent (see section 5.1.5 for further details).

The logic of the agent is represented as a state chart (figure 5.1). The description of the various states is as follows:

- **startUp**: the initialization of the agent with a starting configuration
- **idle**: a state where the car waits for new messages from other agents
- **speedLimitMsgReceived**: the state where the message received from an Active Road Sign RSU (i.e. speed limit sign) is processed. In this case the car speed is set according to the speed limit value contained in the message.
- **curveWarningMsgReceived**: the state where the message received from a curve warning sign is received and processed reducing the speed of the car to the suggested value.
- **icyRoadMsgRcvd**: the state where the icy road warning message received from a car or an RSU is processed. The processing includes the evaluation of whether the message is addresses to the

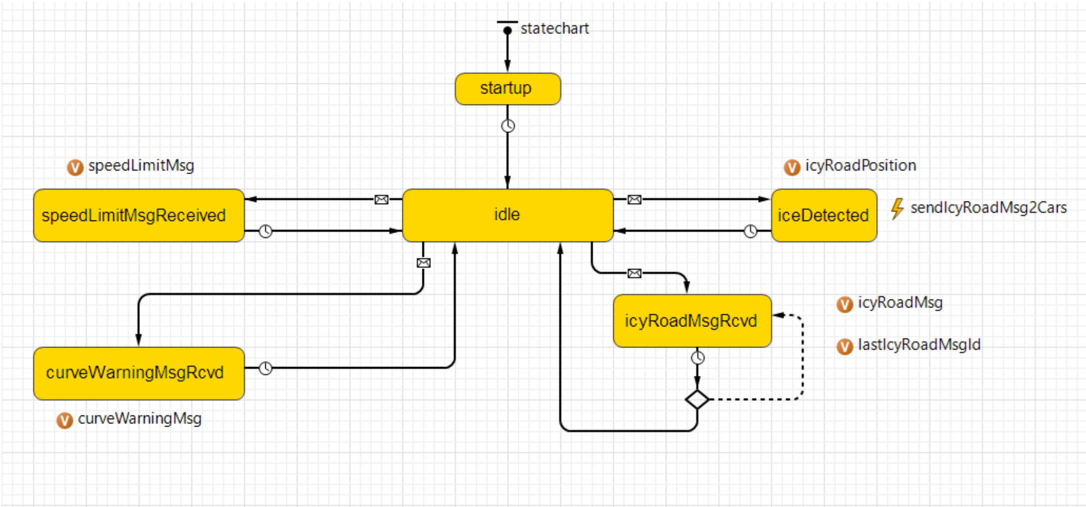


Figure 5.1: Car agent state chart

car or not and also the recognition of the overcoming of the danger with the consecutive set up of the speed at a standard value.

- **iceDetected**: it is the state that processes the situation when ice road is detected. The detection is done by receiving a specific message by the icy road agent. Then a DENM message is generated and broadcast to in-range vehicles and RSUs. Add the fact that the direction is added

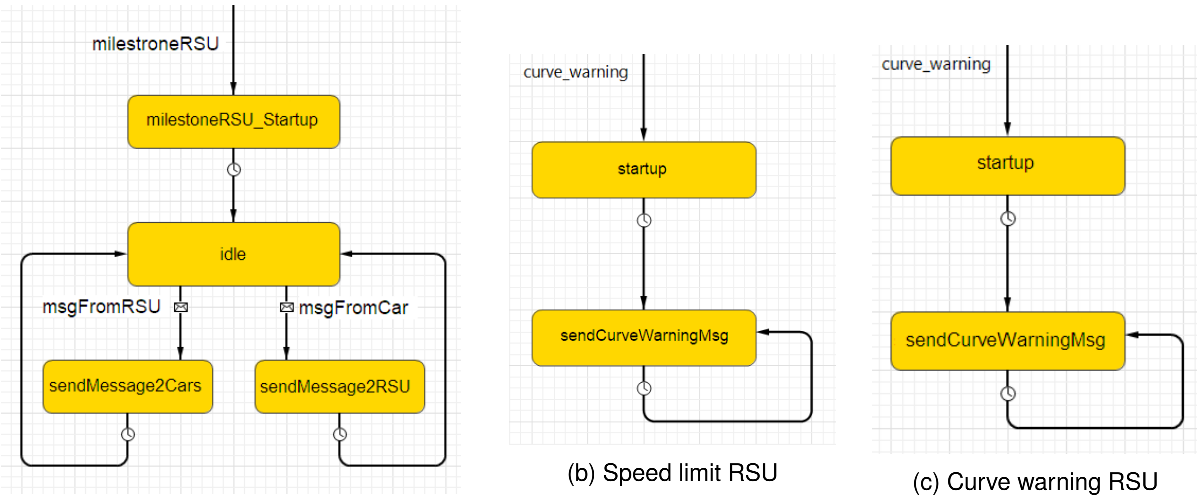
Besides those states, the agent has also two events that permit to execute actions independently of the state chart flow. These two events are related with the update of connections with the other agents: the *addNewConnections* event is responsible of detecting new agents (i.e. cars or RSUs) that lay in the communication area and to connect them, instead the *removeOldConnection* checks for the agents that are still connected but outside this area and disconnects them.

5.1.2 RSU agent

This agent represents the various RSUs that can be installed along the road track. This agent is simply characterized by a pole and a communication radius. It can carry on different functions according to the configured parameters. Indeed, the possible type of functions that it can do are: milestone RSU, and two type of active road signs, i.e. speed limit and curve warning. Further configurable parameters are speed limit, warning position, relevant direction, etc.

The agent logic has been defined in three different state chart one for each function (figure 5.2). For the milestone RSU state chart the description of the various states is as follows:

- **startUp**: the initialization of the agent with a starting configuration
- **idle**: a state where the RSU waits for new messages from either cars or other RSU
- **broadcastMsg2Cars**: the state where the message received from a RSU is received and forwarded to the nearby cars.



(a) Milestone RSU

(b) Speed limit RSU

(c) Curve warning RSU

Figure 5.2: RSU agent state chart

- **broadcastMsg2RSUs**: the state where the message received from a car is then forwarded to the connected RSU. This action is done in order to extend the coverage area of the message generated by the car.

Instead, for the two type of active road signs, since the logic is the same, just one description is given here and it is as follows:

- **startUp**: the initialization of the agent with a starting configuration
- **sendWarningMsg**: this state contains all the actions performed to broadcast a CAM message with the *road-sign container* containing all the information generated by the road sign. These operations are periodically executed with an interval of 100 ms.

5.1.3 Pedestrian crossing

This agent represents a generic pedestrian crossing along a road, and not a crossing at an intersection. It is made of two stop lines where cars stop in case there are pedestrians crossing. Moreover, it has a pedestrian entry and exit point. The first one creates pedestrians that are going to cross moving toward the exit point. The other point, the exit point, just collects the previously generated pedestrians. Finally, it has a sort of pedestrian detector able to discover pedestrians approaching the street and inform the vehicles. Since this agent interacts with both cars and pedestrians it has two interconnection with both of them and they are, respectively, PC2V and PC2P.

As usual the logic has been defined as a state chart, that is depicted in figure 5.3. The states description is as follows:

- **startUp**: the initialization of the agent with a starting configuration
- **detectPedestrians**: this state is responsible for detecting new pedestrians moving towards the road based on their distance and position

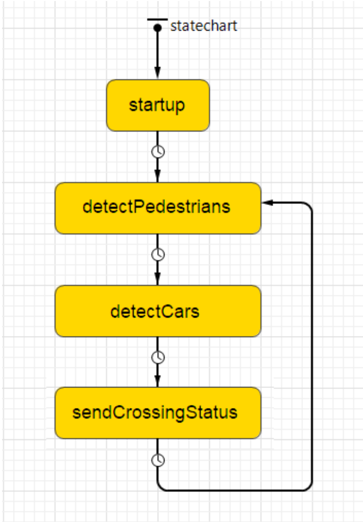


Figure 5.3: Pedestrian crossing agent state chart

- **detectCars**: this state is in charge of updating the connections with cars, in particular establishing connections with cars not yet connected but that are in the communication range and disconnect the cars that are out of this range.

- **updateCrossingStatus**: according to the pedestrian position this state decides to inform or not the cars about the danger (i.e. pedestrians on the road) through the delivery of a specific DENM message. In order to stop the cars it sets also the red status to the two stop lines.

5.1.4 Level crossing

This agent models a level crossing, a meeting point for cars and trains. It is made of two stop lines where cars have to stop in case a train passes and of a car/train detector. This entity is responsible of detecting the approaching trains and then notify the cars to let them stop.

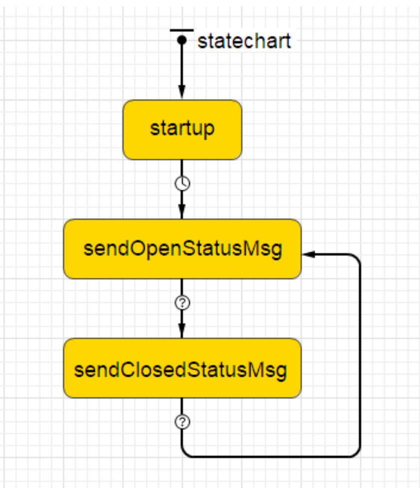


Figure 5.4: Level crossing agent state chart

The agent logic is depicted in figure 5.4 and described as follows:

- **startUp**: the initialization of the agent with a starting configuration
- **sendOpenStatusMsg**: this state represents the situation where there is no train and the barriers are upraised. In this case a DENM message is sent to cars with the information of free crossing.
- **sendClosedStatusMsg**: this state is the situation when the barriers are lowered while a train is passing. The cars approaching the crossing receive a DENM message with a warning.

Besides the state chart, the agent is also provided with four events: *updateCarConnections* responsible for establishing and dropping connections with cars taking into account their distance, *connectTrain* and *disconnectTrain* responsible for estalishing connections with trains and finally the *broadcastMsgToCars*. This last event is in charge of broadcast DENM messages and is activated only when there is a change in the status of the level crossing (i.e. open or closed).

5.1.5 Icy road

The general idea is that future cars will be equipped with special sensors able to analyse the surrounding vehicle environment for detecting possible hazardous situations. For instance, there will be sensors able to detect ice, rocks and even wild animals crossing the road. Once one of those risks has been detected, the vehicle on-board unit will broadcast a proper warning message to both cars and roadside units.

Some studies have demonstrated that the detection of road surface status may be done using Anti-lock Breaking System (ABS) and Electronic Stability Control (ESC) standard sensors installed in vehicles [38]. Aforementioned systems analysing specific interactions between the road surface and the vehicle wheel may provide an indication of the current road status. In order to simulate icy road conditions, a simple model that mimics this interaction has been designed.

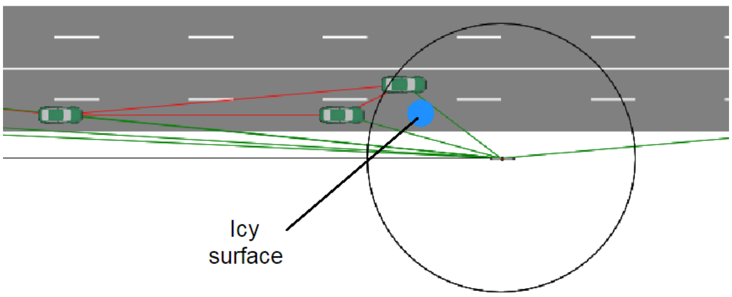


Figure 5.5: Icy road situation simulated in Anylogic

First of all, the icy road hazard has been modelled as an agent with its own logic and parameters. Instead, the icy road to vehicle interaction has been modelled as a unidirectional message exchange. More in detail, the agent is characterized by a periodic event that at each execution detects a car, sends it a message and then disconnects from it. The exchanged message is a Java object, called

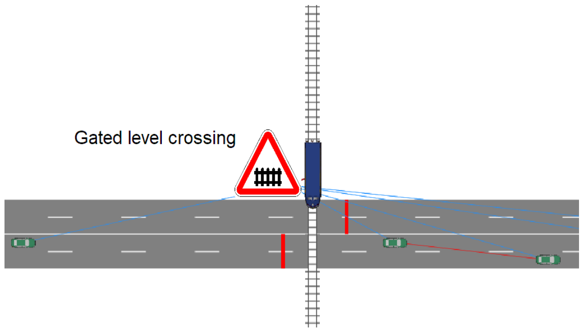
IcySurface2CarMsg, containing the information about the hazard geographic position (i.e. x and y coordinates). On the other side of the communication, the car passing over the icy road receives this message and thanks to the contained geographic coordinates detects the hazard position and then broadcasts a proper DENM message (refer to section 5.1.1 for further details).

It has not been possible to implement the icy road detection as standard vehicle sensors because of some simulator limitations. In fact, the realised communication mechanism is based on the fact that the icy road surface notifies its presence to the car and not that the car detects the hazard.

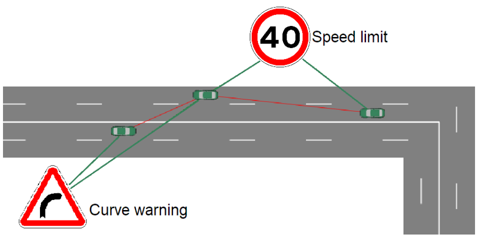
The problem arises from the fact that the hazard detection is a periodic action with a tight schedule and that the state chart does not allow multiple concurrent activities. Therefore, being the hazard detection an intensive activity, the agent running this action will be busy most of the time without the possibility of executing other operations. Because of that, and since the icy road agent has not other operations to execute, this activity has been moved from the car to the icy road agent changing the interaction paradigm.

5.2 Implementation of the city scenario

Besides the agent definition, the modelling phase included also the creation of a city scenario represented by a map made of roads, intersections, pedestrian and level crossings. In addition, throughout the represented track are placed a set of different RSUs and road hazards (e.g. icy road). Inside this city scenarios vehicles move along the roads dynamically establishing connections with other nearby agents. In the figure 5.6 are depicted some of the scenarios that have been implemented and simulated.



(a) Level crossing



(b) Speed limit and curve warning road signs

Figure 5.6: Examples of implemented scenarios

These scenarios have been implemented through a simple drag-and-drop operation plus a series of parameters configuration. For instance, regarding the speed limit signal, it has been placed along the road and then configured specifying: the speed limit value, the relevant direction, the road identifier and the communication radius. Similar actions have been done for others scenarios.

5.2.1 Car path definition

Once the car agent has been defined, it needs to be placed in the city scenario where it moves along roads and interacts with the other agents. In order to obtain this behaviour two main actions have been performed. The first one is the definition of the population of car agents. An agent population, in the Anylogic environment, is simply a set of agents of the same type. It collects all the agents that are present during the simulation execution. This activity corresponds to the creation of an Anylogic built-in object that has to be configured specifying the agent type and the initial population size (always empty in our cases).

The other action is the definition of the car path that cars have to follow. This path has been established by means of properly connected Anylogic functional blocks, namely *carSource*, *selectOutput*, *carMoveTo* and *carDispose* blocks. The description of each block function is as follows:

- **carSource**: it is a generator of agents (e.g. cars, trains, pedestrians, etc.). It may be configured specifying the creation rate, for instance 10 agents per hour, the road track entry point and some others parameters of the created agent (e.g initial speed). Each created agent is added to the corresponding agent population.

- **carMoveTo**: this block defines one part of the whole car path. In particular it is used for defining the next road on which the car has to go. The whole car path is then defined by a series of *carMoveTo* blocks, all connected in a waterfall fashion.

- **selectOutput**: this is a decisional control block used to differentiate the path of cars. According to a random number generation module it defines the path that the car has to take. It is used to create the random car path.

- **carDispose**: it is an agent destroyer. Once the agent reaches its destination point it has to exit from the model and disappear from the car agents population. This block is in charge of removing the agent from the simulation deleting it from its population.

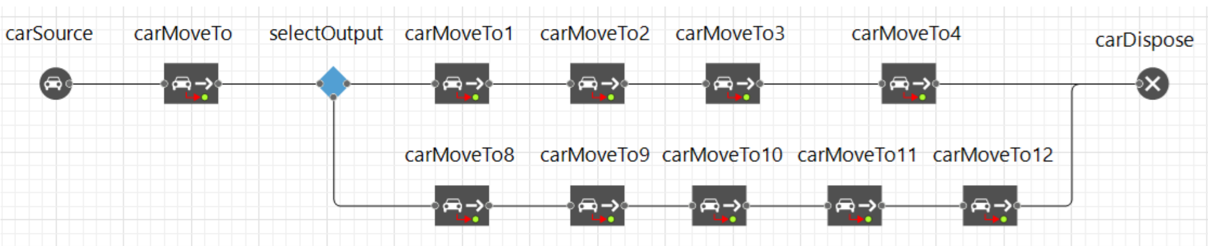


Figure 5.7: Car path definition

An example of car path definition is depicted in figure 5.7 where it is possible to see all the mentioned blocks and their interconnections.

5.2.2 Train path definition

Once that the train track has been deployed it was necessary to defined the trains movement. Also in this case the necessary steps are the train population creation and the train path specification. The train population creation has been done using the Anylogic *population* built-in object configured specifying the train agent type and empty initial population size.

On the other hand, the train path specification has been done connecting, in a waterfall fashion, three functional blocks belonging to the Railway Anylogic Library (figure 5.8). These blocks are as follows:

- **trainSource**: it is a generator of train agents. It may be configured specifying the creation rate and the railway track entry point. Each created train is added to the corresponding train population.
- **trainMoveTo**: this block defines one part of the whole car path. In particular it is used for defining the next road on which the car has to go. The whole car path is then defined by a series of *carMoveTo* blocks, all connected in a waterfall fashion.
- **trainDispose**: it is the train destroyer. Once the train reaches its destination point, that is the end of the railway track, it has to exit from the model and removed from the train agents population. This block is in charge of removing the train from the agent population.

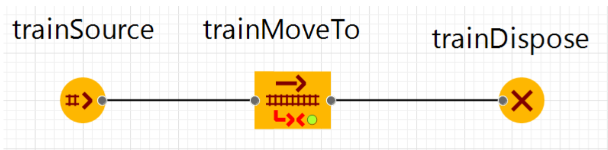


Figure 5.8: Train path definition

5.2.3 Pedestrian path definition

Once that the pedestrian crossings have been deployed along the road track it is necessary to define the pedestrians and their path. This objective has been obtained by means of pedestrians population creation and the pedestrians path specification.

The pedestrians population creation has been done using the Anylogic *population* built-in object configured specifying the pedestrian agent type and empty initial population size.

On the other had, the train path specification has been done connecting, in a waterfall fashion, three functional blocks belonging to the Pedestrian Anylogic Library. These blocks are as follows:

- **pedestrianSource**: it is a generator of pedestrian agents. It may be configured specifying the creation rate and the line entry point. Each created pedestrian is added to the corresponding population.
- **pedestrianGoTo**: this block defines the pedestrian path. In particular, it is used for defining the next point that the pedestrian has to reach. The whole pedestrian path is then defined by a series of *pedestrianGoTo* blocks, all connected in a waterfall fashion.

- **pedestrianDispose**: it is the pedestrian destroyer. Once the pedestrian reaches its destination point it exits from the model and is removed from the pedestrian agents population. This block is in charge of removing the pedestrian from his population.

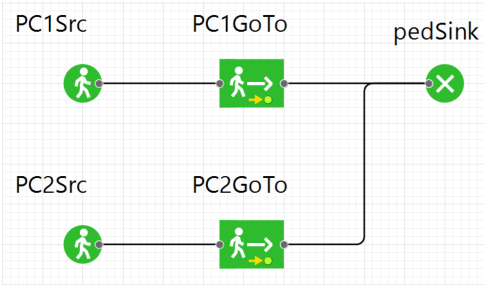


Figure 5.9: Pedestrian path definition

In the figure 5.9 is depicted the organization of Anylogic blocks to implement the pedestrian agents movement. As it is possible to notice there are two *pedestrianSource*, two *pedestrianGoTo* and a unique *pedestrianDispose*. This is due to the fact that in the created model there are two different pedestrian crossings.

Chapter 6

Evaluation

In this chapter are presented the evaluation results of the developed simulation platform with both the analysis of the Anylogic capabilities and the developed use cases as well as a focus on the performance aspect. The evaluation phase has been mainly a qualitative process that has been performed throughout the whole thesis work. The results have been obtained comparing the main functionalities and characteristics of future road scenarios with the functionalities that where possible to implement into the simulation platform. Instead, the performance evaluation has been done through the usage some monitoring tools.

In the second part of this chapter, taking into account the obtained results, there is the description of a possible solution able to overcome some of the encountered development problems and Anylogic limitations.

6.1 Evaluation results

In the following sections are presented the results obtained after the evaluation process. In each section is presented a table showing the list of identified requirements and their details, in particular, the requirements ID, the description and then the result of whether it has been achieved or not.

6.1.1 Anylogic simulator

This analysis is focused on the evaluation of the functionalities that were needed by the simulator. The first analysis focused on the possibility of creating a city scenarios with intersections, pedestrian crossing, etc. in order to have a simulation environment that mimics real road situations. Since it was of a paramount importance to have the possibility of creating a C-ITS infrastructure, the possibility of establishing V2X communications and proper message exchanging has been evaluated. They have been successfully obtained through inter-agent communication.

Finally, in order to have a sort of positioning system (e.g. GPS) it has been evaluated the Anylogic positioning mechanism and verified if it was compatible with our initial requirements. Also this evaluation has been positive, indeed it is possible through dedicated API, to get the current position of each agent moving along the road track.

Req. ID	Description	Achieved
R1.1	Possibility of deploying a custom road topology with cars and RSUs	yes
R1.2	Possibility of establishing V2X communications among agents	yes
R1.3	Possibility of defining custom messages	yes
R1.4	Integration of road library with pedestrian and rail library	yes
R1.5	Possibility of locating agents by means of Cartesian coordinates	yes

6.1.2 Car agent

The car agent analysis is focused on which are the accessible and modifiable parameters of the vehicle. Examples are the position, the speed, the acceleration and so on and so forth and the actions that were possible to execute. Firstly it has been considered the possibility of defining a track along which the car can move. Then the possibility of connecting the vehicle to other agents has been deployed and tested. About this aspect, in particular it has been proven that in case of dangerous situation cars can broadcast DENM messages in order to notify other cars and RSUs.

Besides that and due to the fact that the vehicle is a relative complex entity carrying on different operations, the possibility of executing parallel actions has been evaluated. From this analysis has been verified that this is not achievable because of the Anylogic limitations and in particular the fact that the agent logic can be defined only by means of a single flow state chart.

Finally, has been tried the possibility of changing the car path in consequence of a specific situation or event (e.g. icy road or roadworks). Unfortunately also in this case the expected result has not been obtained. The reason is due to the fact that the car path is defined by means of a series of predefined blocks. These blocks control the car movements giving only the possibility of specifying the start point and the end point, without the chance of changing them at run time after a specific event occurrence.

Req. ID	Description	Achieved
R2.1	Possibility of defining fixed or random car path	yes
R2.2	Possibility of exchanging messages with other vehicles or RSUs	yes
R2.3	Possibility of getting access to agent position through integrated API	yes
R2.4	Possibility of changing basic vehicle parameters (speed, acceleration)	yes
R2.5	Possibility of running different control flow in a concurrent fashion	no
R2.6	Ability to change car path on the fly after that a particular event occurred	no

6.1.3 RSU agent

The RSU agent analysis is focused on three main aspects: the possibility of placing it anywhere along the road track, the possibility of communicating with cars and connected RSUs and the possibility of implementing different type of applications.

The first aspects has been easily verified and successfully obtained thanks to the drag-and-drop functionalities provided by Anylogic. The second aspect evaluation aimed at testing two type of communications. The first one between cars and RSU with objective of getting possible warning notifications from cars about detected hazards. The second communication instead involves two different RSUs where warning messages generated by cars and delivered to the RSU ar then forwarded to another connected RSu in order to get a wider propagation. This result has been completely obtained establishing a new inter-agent communication between RSUs.

The latter possibility, instead has been partially obtained. In fact, besides the fact that it is possible to deploy both milestone RSU and Active Road sign, it has not possible to run these functions concurrently. That is, an RSU agent cannot manage the execution of two different applications. This is caused by the Anylogic limitation that allows the description of the agent logic only by means of a state chart characterized by a single execution flow.

Req. ID	Description	Achieved
R3.1	Possibility of deploying RSUs along the road track	yes
R3.2	Possibility of establishing V2X communications among agents	yes
R3.3	Possibility of behaving as milestone RSU	yes
R3.4	Possibility of behaving as Active Road sign	yes
R3.5	Possibility for a RSU to forward messages to another RSU	yes
R3.6	Possibility for a RSU to execute more functions at the same time	no

6.1.4 Pedestrian crossing

The pedestrian crossing analysis aimed at evaluating the possibility of creating an entity that could be put along a road track and where pedestrians could cross the road. The first evaluated aspect has been the possibility of interconnecting this entity with both cars and pedestrians, creating de facto V2I and I2P communications. Exploiting these interconnections it has been verified the possibility of detecting those agents ans takes appropriate actions.

Along with those aspects, it has been evaluated also the possibility of establish a proper message exchange between the pedestrian crossing and the cars and then stop them in case pedestrians are occupying the carriageway. All these aspects have been successfully evaluated and demonstrated.

Req. ID	Description	Achieved
R4.1	Possibility of deploying a pedestrian crossing on the road network	yes
R4.2	Possibility of establishing V2I and I2P communications with cars and pedestrians	yes
R4.3	Possibility of delivery warning messages to cars when there are pedestrian crossing the road	yes
R4.4	Possibility of detecting pedestrians approaching the road	yes
R4.5	Possibility of stopping cars at a given position before the pedestrian crossing	yes

6.1.5 Level crossing

The level crossing analysis aimed at evaluating the possibility of creating an entity that could be put along a road track where there is the intersection with a railway track. The first analysis focused on the capability of detecting approaching cars and trains and to establish V2I and T2I communication. Once that those aspects have been successfully proven, then has been tested the possibility of notify approaching cars about the level crossing status both in case there is or not a train passage.

Req. ID	Description	Achieved
R5.1	Possibility of establishing V2I and T2I communications with approaching cars trains	yes
R5.2	Possibility of detecting trains approaching the level crossing	yes
R5.3	Possibility of detecting cars approaching the level crossing	yes
R5.4	Possibility of stopping cars at a given position before the level crossing	yes
R5.5	Possibility of delivery warning messages to cars when there is an approaching train	yes

6.1.6 Icy road

The icy road agent analysis is focused on different aspects. First of all, like the RSU agent, it has been evaluated the possibility of placing the hazard on whatever part of the road track. This first result has been successfully achieved thanks to the drag-and-drop functionality provided by Anylogic.

The second phase of the evaluation is focused on the interaction between car and icy road agents since this is of paramount importance in order to simulate the real world scenario. This functionality has been completely achieved thanks to the inter-agent communication.

Req. ID	Description	Achieved
R6.1	Possibility of placing an hazard anywhere on the road	yes
R6.2	Possibility for cars to detect the hazard	yes

6.2 Performance evaluation

Besides te qualitative evaluation of the Anylogic platform and developed agents a performance evaluation has been performed as well. The computing platform used for running all the simulations is a commercial laptop, which characteristics are shown in the table below.

Laptop model	ASUS N552VW
CPU	Intel Core i7 6700HQ (3,50 GHz)
GPU	GeForce GTX 960M
RAM	16 GB
Storage	128 GB of SSD + 1TB of HDD
Operating System	Windows 10 Home Edition 64 bit

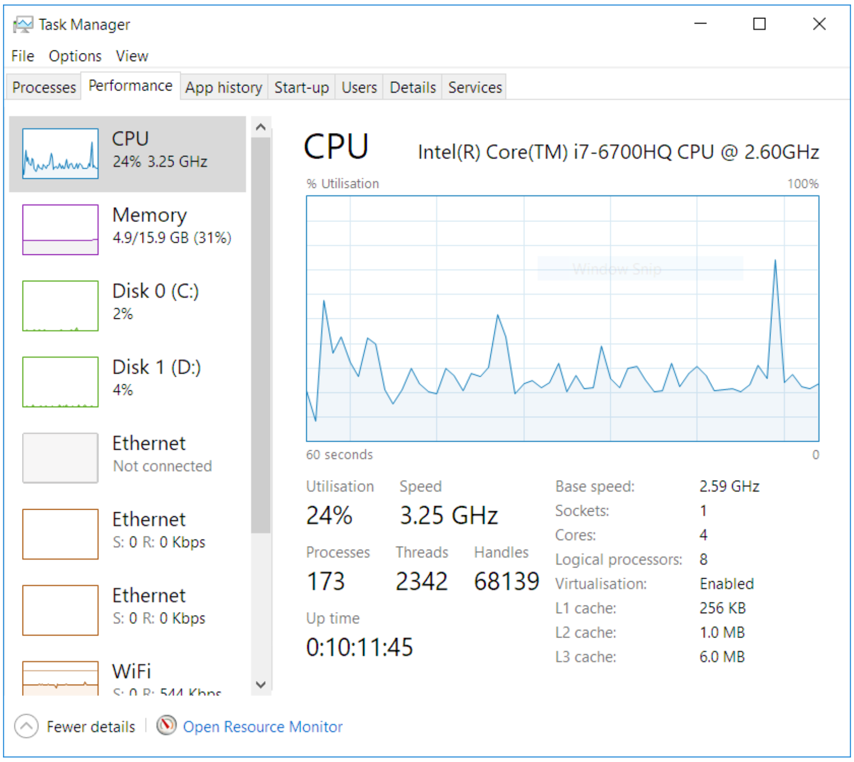


Figure 6.1: Screen-shot of the Windows Task Manager

The performance evaluation has been done using the built-in Anylogic tools and the Windows Task Manager. The tests have been conducted in an unloaded system running only the operating system and the Anylogic simulator.

The considered simulation has been executed with the full city road topology with the following characteristics:

- 13 roads
- 3 car entry points
- 7 intersections
- 4 curves
- 1 railway track
- average of 600 car agents per hour
- average of 2 trains per hour
- average of 80 pedestrians per hour
- 2 pedestrian crossing agents
- 13 RSU agents (including active road signs and milestone RSUs)
- 3 level crossings agents
- 2 icy road agents

In figure 6.1 is shown a screen-shot of the Task Manager. As it is possible to observe there is not heavy system usage both from a CPU and RAM viewpoint.

Using instead the built-in Anylogic memory usage indicator (figure 6.2) it is possible to notice that besides a model allocated memory of 2048 MB, only 132 MB are being used, meaning that it is used only a share of 6.45%.

Figure 6.2: Anylogic memory usage indicator

6.3 Architecture proposal for future applications

Observing the way in which the agents logic has been defined, it is possible to notice that it is based on a state chart definition. Those state charts are characterized by a set of states and transitions among them as well as a set of possible asynchronous events.

The main problem that arises with this kind of logic representation is that it is not possible to define different parallel actions that run concurrently. In fact, if an agents is stuck in a particular state it can only execute the actions declared in that state and not others declared in other part of the state chart. Moreover, an agent that has an intensive and repeated operation, for instance a continuous status checking like in the case of the icy road agent, it will continuously change its state just from the idle state to state that perform the action and vice versa. This causes the fact that other non-intensive operations will wait for a long time. Finally, the last problem is that the state chart does not allow a modular code organization, but all the code must be defined in each state without the possibility of reusing code.

For all this reasons a possible solution has been designed and proposed. The idea comes from the fact that Anylogic is a Java based tool, meaning that all the Java standard libraries are at programmer disposal.

The proposed solution consists in the realisation of a standalone Java application that is able to communicate with the Anylogic tool via a proper interconnection. The general system architecture is depicted in figure 6.3, where on the left side there is the Anylogic tool with the implemented model, on the right side there is the standalone application and finally, in the middle of these two entities, there is their interconnection represented by a TCP connection.

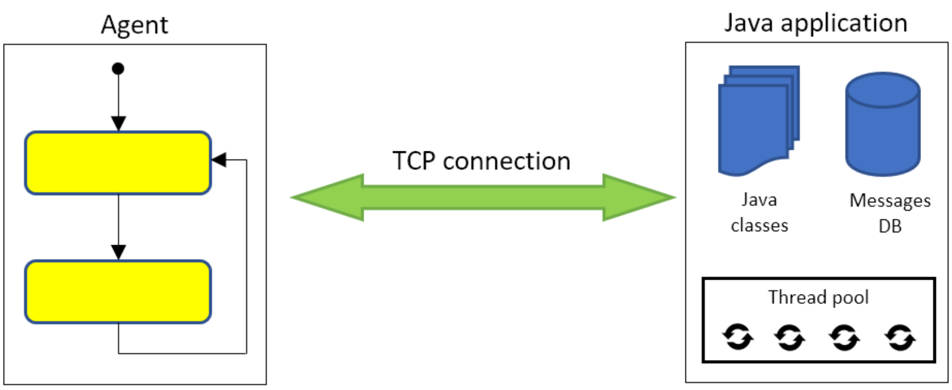


Figure 6.3: Architecture of the proposed solution

Concerning the Anylogic tool and its model, the proposed solution is based on the fact that the agents logic is characterized by a simple state chart with simple operations mainly devoted to the management

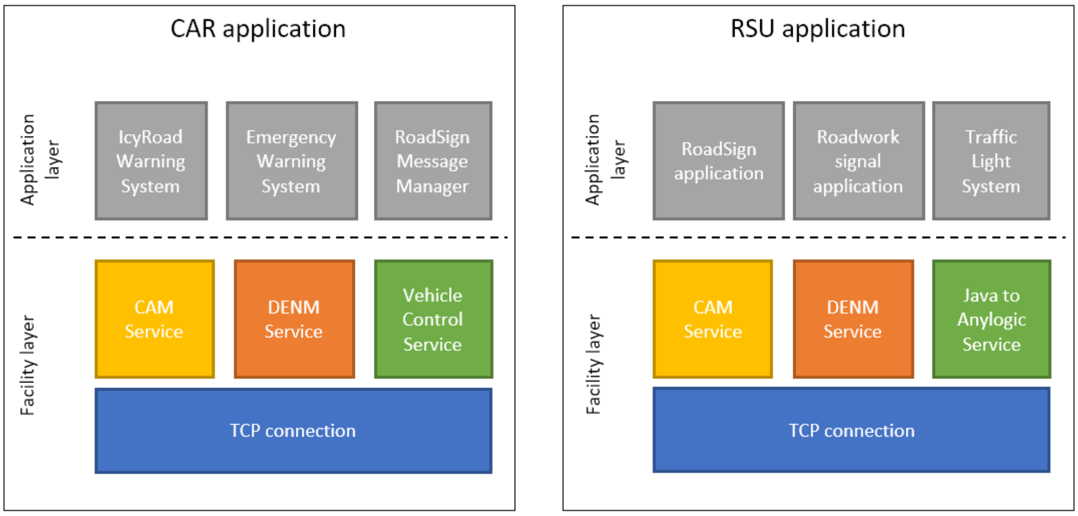


Figure 6.4: Possible architecture of the external application

of inter-agent communication and the interaction with the external Java application. This includes also the forwarding of received CAM and DENM messages to the external application without any further elaboration.

Considering instead the other side, the standalone application is characterized by a set of applications running on multiple threads and a database of messages. Each application corresponds to its respective counterpart that is represented by the agent inside the Anylogic simulator. Such application implements the real agent logic and, in order to reflect its behaviour and state to the respective agent that lays in the Anylogic simulator, sends specific messages through the TCP connection. Hence the Anylogic-to-standaloneApp interconnection is a set TCP connections established between one agent and its external application. The messages that pass over these TCP connections are not only generated by the application, but may be also generated by the Anylogic agent. Indeed, to have the Java application aware of what the agent is doing and to receive CAM and DENM messages, the agent must send to it appropriate notifications.

Regarding the database of messages, this can be considered as a possible support utility that may help in debugging and analysis phases. Indeed, it can be used to store all the messages exchanged by Anylogic and the standalone application, creating a sort of message log.

In figure 6.4 is depicted the possible organization of two type of applications running two different agents logic. The considered agents are the car and the RSU. The applications architecture has been defined in a layered fashion. The composing layers are the facility layer and the application layer.

The facility layer is characterized by different modules, that differs accordingly to the specific controlled agent:

- **TCP connection** represents the component in charge of establish and manage the TCP connection with the agent in the Anylogic simulator
- **CAM Service** represents the module in charge of managing CAM messages. It provides functionalities for both sending and receiving messages.

- **DENM Service** represents the module in charge of managing DENM messages. It provides functionalities for both sending and receiving messages.

- **Vehicle Control Service (only in car application)** contains the local representation of the current vehicle status (e.g. speed, position, acceleration, etc.). Moreover it provides functions to access and modify these parameters, for instance *getSpeed* or *setSpeed*. It is also in charge of keeping updated the local status information with the one contained in the Anylogic agent.

- **RSU Control Service (only in RSU application)** contains the local representation of the current RSU status (e.g. location, connected vehicles and their position, etc.). It is also in charge of keeping updated the local status information with the one contained in the Anylogic agent.

The application layer, instead, contains all the possible applications that may run on a specific agent. Examples are:

- **Icy Road Warning System** represents the on-board vehicle system able to detect ice on the road. It is based on a message exchange with the Anylogic agent about the road status.

- **Emergency Warning System** represents the module in charge of managing CAM messages. It provides functionalities for both sending and receiving messages.

- **RoadSign Communication System** represents the module in charge of managing CAM messages generated by Active Road Sign RSU.

- **Roadwork Signalling System** represents the system able to detect roadworks thanks to the management of received DENM messages.

- **Traffic Light System** it is related to the traffic light management. In particular, on the RSU application it runs the traffic light algorithm and communicate with vehicles through messages. Instead on the car application represents the system able to get and compute messages generated by traffic light system

The proposed solution, since it is a standard Java application, may benefit of all the Java features (e.g. inheritance, modularity, etc.) solving the problem encountered with the definition of agent logic through state charts. Furthermore, because of the full availability of Java standard libraries, it is possible to enable concurrency by means of the definition of multiple threads.

Besides all the advantages mentioned before it is still necessary to evaluate some aspects of this solution, for instance:

- the definition of a communication protocol between Anylogic and the Java application

- the analysis of possible race conditions among running threads

- the evaluation of the number of threads that will be running during the model execution

- the set up a storage system to keep a message logging

Chapter 7

Conclusions

Because of the growing demand for new mobility services and the pressing request of making transport safer and more efficient, the transport sector is rapidly evolving. Car manufactures along with IT companies are trying to respond to these needs by means of the development of Cooperative Intelligent Transport Systems.

The thesis work started with the study of Cooperative Intelligent Transport System state-of-the-art. This phase focused on two main aspects. The first is the survey of current international standards dedicated to the regulation of the communication aspects and application organization. The second aspect is the analysis of current development and testing methodologies for C-ITS systems.

Then the characterisation of the main real road scenarios has been carried out resulting in the definition of some future intelligent road scenarios. In addition to these scenarios have been also characterized some innovative aspects that are not yet present in the current literature, namely the milestone RSU and the Active Road Sign. Starting from these use cases, the implementation of the simulation environment has been done throughout the basic development of some C-ITS applications. Examples are: icy road warning, pedestrian crossing and Active Road Signs.

Finally, the developed solution have been evaluated performing, for each implemented use case, the analysis of all the features and functionalities that were possible to implement.

According to the evaluation results, the developed solution demonstrates that it is a suitable tool for developing future C-ITS applications that can be applied in different road scenarios, including also dangerous situations. Because of that, it may represent an alternative solution for the study, development and testing of initial prototypes of future C-ITS systems and applications without the necessity of having expensive testing site facilities. Some examples are the realisation of the milestone RSU and the Active Road Sign systems.

As it was possible to evince from the state-of-the-art survey, the development and testing of C-ITS systems is mainly a concern of research institutes, car makers and IT companies. Considering instead current universities curricula, it is noticeable that C-ITS related studies are not yet widely present. Therefore, there is a remarkable lack of courses dedicated to the introduction of these future technologies that could train fresh figures able to bring innovation. For these reasons, our solution also aimed at creating a simulation platform meant as a possible tool able to support C-ITS related teaching activities.

Besides all the practical obtained results, this work had generated other outcomes as well. For instance, it allowed a better understanding of the whole C-ITS context and complexity. In fact, at the beginning, the general knowledge about those systems was limited only to the possible benefits that could be achieved, without having any sort of awareness about the implementation details and related issues. Finally, in light of the still high number of fatalities that afflict our roads, the attention to those future technologies and their promised benefits has significantly grown.

Future works

Since this thesis work is the initial step for the realisation of the simulation platform for developing and testing C-ITS systems and applications, there is a series of future works that could introduce new functionalities. Moreover, since it has not been possible to implement all the intended functionalities because of some Anylogic limitations, possible future enhancements may help to improve the current solution and overcome those limitations.

The set of defined use cases is not yet complete. In fact, analysing the real environment there is a remarkable number of possible situations and scenarios that have not been covered yet. Therefore, a possible future work could be a deeper analysis of the literature related to the study of particular scenarios and then implement them into the simulation platform. This future work may help to have a more realistic simulation platform.

The set of defined functionalities stated at the beginning of the thesis work has no been completely implemented. In fact, from the evaluation phase it has been possible to notice that due to the missing of some Anylogic functional blocks, some desired features were not possible to implement. To overcame these limitations a future work may aim at exploring the possibility of defining custom blocks implementing custom functionalities. An examples is represented by the adaptive re-routing functionality, that is the capacity of cars to change their path to the destination in order to avoid a road hazard or road works. Since Anylogic allows the definition of new functional blocks, with their own logic and functionalities, then a future work may deeply study this solution and hopefully overcome the described issue.

The solution proposed in section 6.3 introduced a possible architecture for future application development. Indeed, currently, the agent logic definition has been done through a set of state charts characterized by states and transitions among states. This solution at the moment does not allow to run different control flow at the same time (i.e. multi-threading or multi-processing), but permits only a linear execution of a set of tasks. The solution proposal, that may be developed in the future, aims at exploring the possibility of moving the logic of agents (i.e. cars and RSUs) to an external Java application able to communicate with the Anylogic simulator. In this Way it would be possible to run concurrently different instructions flows as well as obtain a more flexibility for developing the desired applications.

Lastly, because of time restriction, a greater analysis of currents standards has not been possible. But since the ETSI standard is an important source of knowledge from which it is possible to obtain further details and ideas, a further and deeper analysis of those documents is suggested.

Bibliography

[1] *Summary of motor vehicle crashes (Final edition): 2015 data*. National Highway Traffic Safety Administration, Washington, DC, October 2017.

[2] *The Economic and Societal Impact Of Motor Vehicle Crashes (Revised)*. National Highway Traffic Safety Administration, 2010.

[3] *Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990-2015*. United States Environmental Protection Agency, 2017.

[4] European Commission. A european strategy on cooperative intelligent transport systems, a milestone towards cooperative, connected and automated mobility. https://ec.europa.eu/transport/themes/its/c-its_en. [Online; accessed March 26, 2018].

[5] European Telecommunications Standards Institute. Cooperative intelligent transport system. http://www.etsi.org/images/files/membership/ETSI_ITS_09_2012.jpg. [Online; accessed March 14, 2018].

[6] Researchgate. C-its infrastructure. http://www.mdpi.com/sensors/sensors-15-29889/article_deploy/html/images/sensors-15-29889-g001.png. [Online; accessed March 14, 2018].

[7] Qualcomm. Accelerating c-v2x commercialization. https://www.qualcomm.com/documents/path-5g-cellular-vehicle-everything-c-v2x. [Online; accessed March 26, 2018].

[8] L. Hobert, A. Festag, I. Llatser, L. Altomare, F. Visintainer, and A. Kovacs. Enhancements of v2x communication in support of cooperative autonomous driving. *IEEE Communications Magazine*, 53(12):64–70, 2015.

[9] AutoNet2030. Autonet2030 public final report. http://www.autonet2030.eu/wp-content/uploads/2016/11/D1_3_AutoNet2030_Final_report.pdf. [Online; accessed March 26, 2018].

[10] J. B. Kenney. Dedicated short-range communications (dsrc) standards in the united states. *Proceedings of the IEEE*, 99(7):1162–1182, July 2011. ISSN 0018-9219. doi: 10.1109/JPROC.2011.2132790.

[11] A. Festag. Cooperative intelligent transport systems standards in europe. *IEEE Communications Magazine*, 52(12):166–172, December 2014. ISSN 0163-6804. doi: 10.1109/MCOM.2014.6979970.

[12] J. Santa, F. Pereñíguez, A. Moragón, and A. F. Skarmeta. Vehicle-to-infrastructure messaging proposal based on cam/denm specifications. In *2013 IFIP Wireless Days (WD)*, pages 1–7, Nov 2013. doi: 10.1109/WD.2013.6686514.

[13] ETSI EN 302 637-2. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. Standard, European Telecommunications Standards Institute, Nov. 2014. v 1.3.2.

[14] ETSI EN 302 637-3. Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service. Standard, European Telecommunications Standards Institute, 2014. v 1.2.1.

[15] J. J. Anaya, E. Talavera, D. Giménez, N. Gómez, J. Felipe, and J. E. Naranjo. Vulnerable road users detection using v2x communications. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 107–112, Sept 2015. doi: 10.1109/ITSC.2015.26.

[16] R. Stahlmann, M. Möller, A. Brauer, R. German, and D. Eckhoff. Technical evaluation of glosa systems and results from the field. In *2016 IEEE Vehicular Networking Conference (VNC)*, pages 1–8, Dec 2016. doi: 10.1109/VNC.2016.7835967.

[17] AstaZero. The astazero testing site. `http://www.astazero.com/the-test-site/about/`. [Online; accessed March 14, 2018].

[18] T. Queck, B. Schünemann, I. Radusch, and C. Meinel. Realistic simulation of v2x communication scenarios. In *2008 IEEE Asia-Pacific Services Computing Conference*, pages 1623–1627, Dec 2008. doi: 10.1109/APSCC.2008.23.

[19] Veins - Vehicles in Network Simulation. `http://veins.car2x.org`, . [Online; accessed March 26, 2018].

[20] T. Queck, B. Schünemann, I. Radusch, and C. Meinel. Realistic simulation of v2x communication scenarios. In *2008 IEEE Asia-Pacific Services Computing Conference*, pages 1623–1627, Dec 2008. doi: 10.1109/APSCC.2008.23.

[21] MSIECV. `http://www.cn.hhu.de/en/our-research/msiecv.html`, . [Online; accessed March 26, 2018].

[22] Anylogic simulation tool. `https://www.anylogic.com/`. [Online; accessed April 02, 2018].

[23] K. Katsaros, R. Kernchen, M. Dianati, and D. Rieck. Performance study of a green light optimized speed advisory (glosa) application using an integrated cooperative its simulation platform. In *2011 7th International Wireless Communications and Mobile Computing Conference*, pages 918–923, July 2011. doi: 10.1109/IWCMC.2011.5982524.

[24] P. Szczurek, B. Xu, O. Wolfson, and J. Lin. Estimating relevance for the emergency electronic brake light application. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1638–1656, Dec 2012. ISSN 1524-9050. doi: 10.1109/TITS.2012.2201257.

[25] A. B. Reis, S. Sargento, and O. K. Tonguz. Parked cars are excellent roadside units. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2490–2502, Sept 2017.

[26] Wikipedia - the free encyclopaedia. Pietra miliare. https://it.wikipedia.org/wiki/Pietra_miliare. [Online; accessed March 01, 2018].

[27] S. Geppi. American center for mobility testing site. https://it.wikipedia.org/w/index.php?curid=4243519. [Online; accessed March 14, 2018].

[28] National Association of City Transportation Officials. Hybrid beacon for bike route crossing of major street. https://nacto.org/publication/urban-bikeway-design-guide/bicycle-signals/hybrid-beacon-for-bike-route-crossing-of-major-street/. [Online; accessed March 22, 2018].

[29] Variable Message Signs. Level crossing systems. http://www.vmslimited.co.uk/crossing.htm. [Online; accessed March 22, 2018].

[30] Roads and Maritime Services. Intersections and turning. http://www.rms.nsw.gov.au/roads/safety-rules/road-rules/intersections.html. [Online; accessed March 22, 2018].

[31] Fineline Driving Academy. Slip road. http://finelinedrivingacademy.co.uk/Driving-Tutorials/Slip-roads. [Online; accessed March 15, 2018].

[32] Bank09. Traffic road highway aerial view 2. https://www.shutterstock.com/it/video/clip-28080577-stock-footage-traffic-road-highway-aerial-view.html?src=rel/19859632:6/gg. [Online; accessed March 15, 2018].

[33] Flckr. Goat crossing. https://www.flickr.com/photos/oldrebel/5947892558. [Online; accessed March 23, 2018].

[34] Brad's blog. Wet season road hazards cooktown to wujal wujal. https://invivamus.files.wordpress.com/2012/03/wpid-2012-03-11-09-17-36-11.jpg. [Online; accessed March 23, 2018].

[35] Travelers. How to drive in icy conditions. https://www.travelers.com/resources/auto/safe-driving/how-to-drive-in-icy-conditions. [Online; accessed March 14, 2018].

[36] Publipunto. Cómo reaccionar ante una emergencia al volante. https://blog.publipunto.com/5668-2/. [Online; accessed March 23, 2018].

[37] Ilya Grigoryev. Modeling and simulation modeling. Anylogic in three days. pag. 7.

[38] C. Aguilar, C. Carrillo, G. Fernández, and C. Acosta. Robust road condition detection system using in-vehicle standard sensors. In *Sensors*, volume 15, pages 32056–32078, 2015.

Appendix A

Road-Sign container definition

This appendix contains the specification of the road-sign container. For each sign there is the definition of the information present in the container, in particular the list of parameters and for each parameter its type. The data type used are:

- **Integer:** an integer number
- **TrafficDirection:** a single value field that can be either *forward* or *backward*
- **GeographicPositionXY:** a multi-field data containing the Cartesian coordinates, that is X and Y value

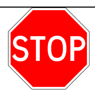
All the following images have been taken from the Wikipedia web page: *Comparison of European road signs* (available at `https://en.wikipedia.org/wiki/Comparison_of_European_road_signs` [Accessed on 2018-03-22]).

Priority signals

1. **Give way**

Road sign	Data	Type
	roadID direction intersectionPositionXY	Integer TrafficDirection GeographicPositionXY

2. **Stop**

Road sign	Data	Type
	roadID direction	Integer TrafficDirection

3. **Priority road**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

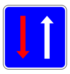
4. **End of priority road**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

5. **Give way to oncoming traffic**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

6. **Priority over oncoming vehicles**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

Warning signals

1. **Round about ahead**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

2. **Two way traffic**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

3. **Level crossing**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

4. **Gated level crossing**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

5. **Road narrow**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

6. **Dangerous crosswind**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	hazardPositionXY	GeographicPositionXY

7. **Steep ascent**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

8. **Steep descent**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

9. **Uneven road ahead**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	hazardPositionXY	GeographicPositionXY

10. **Dangerous curve**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	bendPositionXY	GeographicPositionXY

11. **Dangerous double curve**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	bendPositionXY	GeographicPositionXY

12. **Slippery road**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	dangerPositionXY	GeographicPositionXY

13. **Road accident**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	dangerPositionXY	GeographicPositionXY

14. **Road work**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	dangerPositionXY	GeographicPositionXY

15. **Other danger**

Road sign	Data	Type
	roadID	Integer
	suggestedSpeedLimit	Integer
	direction	TrafficDirection
	dangerPositionXY	GeographicPositionXY

Prohibition signals

1. **Speed limit**

Road sign	Data	Type
	roadID	Integer
	speedLimit	Integer
	direction	TrafficDirection

2. **Minimum safe distance**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection
	distanceValue	Integer

3. **No overtaking**

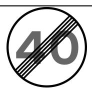
Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

4. **No entry**

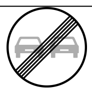
Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

End of prohibition signals

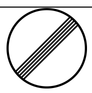
1. **End of speed limit**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

2. **End of overtaking prohibition**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection

3. **End of all previous prohibitions**

Road sign	Data	Type
	roadID	Integer
	direction	TrafficDirection