Dipartimento di
Ingegneria Civile, Edile e Ambientale

Ph.D. Program in Civil Systems Engineering
XXXVIII Cycle

Simulation and co-simulation environments for the Cooperative, Connected and Automated Mobility (CCAM)

by

Andrea Marchetta

Supervisor
Prof. Gennaro Nicola Bifulco
Co-Supervisor
Prof. Marcello Cinque

Ai miei nonni, tutti e quattro:
Ciccio, Italia, Maria, Nando.
Vi ho mentito,
a Dubai non ci sono mai andato..
Ma che importa?
I "petrodollari" possono aspettare..

Abstract

The future, if not the upcoming present, of traffic mobility is and will be *cooperative* and *connected*. Each and every road actor, from vehicles, to infrastructure sensors and vulnerable road users, will be able to share information between each other to simplify the driving task, smoothen traffic queues, and reduce accidents rate as well as their effects on traffic. This information sharing is granted by Vehicle-To-Everything (V2X) communication, which in turn allows for a plethora of services related to the paradigm of Cooperative, Connected and Automated Mobility (CCAM) to spread.
However, the deployment of such services is still hindered by the lack of definitive solutions allowing for efficient testing and validation: indeed, not only these strategies have to be deployed, but there also needs to be an extensive validation campaign to evaluate their impact of the whole traffic system. This statement is corroborated by a detailed literature analysis regarding testing and validation of Cooperative-Intelligent Transportation Systems (C-ITS), which also highlights the major gaps on the matter; a nonexhaustive list of these include the definitive absence of widespread and open-source Hardware-in-Loop (HiL) applications for testing and development of C-ITS, the absence of related software implementations of the aforementioned services, the inability to offer realistic simulations to shorten development and deployment time, and the lack of thorough testing of these applications from a cybersecurity point of view.

The literature analysis also sheds the light on the most promising co-simulation platforms, which are able to capture some of the heterogeneous aspects of the traffic environment, as well as to partially fill some of the described gaps, as well as possible open-source solutions which can be suitable for valuable extensions.

With this ample discussion in mind, the goal set for this thesis is to design, deploy and validate a co-simulation framework able of capturing all the heterogeneous aspects of traffic and cover the aforementioned gaps. Starting from a suitable open-source solution in the Eclipse MOSAIC software, we describe all the functional requirements of this platform and the necessary features, as well as the carried advancements to enable both C-ITS and HiL testing. Subsequently, the platform is challenged against different applications which exemplify various use cases: this includes the synthetization of real-world communication facilities on a real hardware board, HiL algorithm testing, and evaluating C-ITS against conventional detection strategies. Moreover, the framework also enabled cybersecurity testing of vehicular applications, including testing the application timeliness and ground truth testing for realism purposes.

Moreover, the co-simulation platform is also employed to evaluate the C-ITS in their intended environment with the full message exchange pipeline enabled by enabling a Public Key Infrastructure (PKI) platform. Finally, the advancement tested within the platform have been ported to a real-world use case related to a highway application in Italy.

Ultimately, the goal is to offer this platform as a valuable starting point to speed up rapid prototyping and deployment of C-ITS application in the near future, while also providing a scalable, portable and modular framework which can simulate various traffic environments in an integrated and realistic way.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Cooperative-Intelligent Transportation Systems

Over the last two decades, we have witnessed a drastic surge in the technological advancement of all sorts of road vehicles. Features which began as optionals, such as Lane Keeping Assistants (LKAs) or Intelligent Speed advisors (ISAs) are now mandatory in all newly manufactured vehicles in Europe [138]. Moreover, the advancements in communication technologies are pushing road vehicles to become more interconnected, to share all sorts of data between each other to contribute to a common goal: make roads safer, less polluted, and decrease accident rates. This goal is totally in line with the various declinations of the *Vision Zero* road safety project [137]. Such objectives call for the development of tailored measures in the form of **Cooperative-Intelligent Transportation Systems (C-ITSs)**, involving all the road actors, ranging from vehicles to infrastructure components.

In itself, the concept of Intelligent Transportation Systems (ITSs) is not new. The need for more complex traffic regulations traces back to the 80s, even though the technology was not yet ready

nor adequate for more advanced solutions. However, it was the advent of the smartphone which *"gave the broad public a means of envisioning technologies such as connected vehicles or automated vehicles becoming a reality [..]"*, effectively allowing for new services to be developed including actual *cooperative* features [65].
Clearly, the addition of the cooperative features is a big deal when referring to road transport: vehicles need to be more digitalized and able to continuously share real-time data; a road infrastructure needs to be put in place, further enhancing this cooperation through camera sensors, detectors, and other infrastructure-based utilities; the continuous data flow has to be analysed and further actuations deployed by a Traffic Management Centers (TMCs) structure. All in all, these interactions involve heterogeneous stakeholders, including car manufacturers, road operators, technical partners, and so on.

1.1.1 A brief example

To better envision the C-ITS landscape in the coming decades, the following example, visualized in figure 1.1 regarding traffic intersection is particularly exemplary:
traffic intersection are a particularly hazardous spot, especially in crowded cities with narrow roads; this is often the case for European cities, where the historical centres, clearly not built with cars in mind, further exacerbate this issue. As if it was not enough, double-parked cars, the presence of bus stops, the ever-increasing number of electric scooters and bikes along the road and the occasional inattentive user can pose a severe threat to road safety, especially linked to Vulnerable Road Users (VRUs) such as pedestrians or bikers.

Within this landscape, let us imagine that a pedestrian happens to cross the intersection while having the red light, and a vehicle is crossing at the same time. Since the driver's field of view is partly

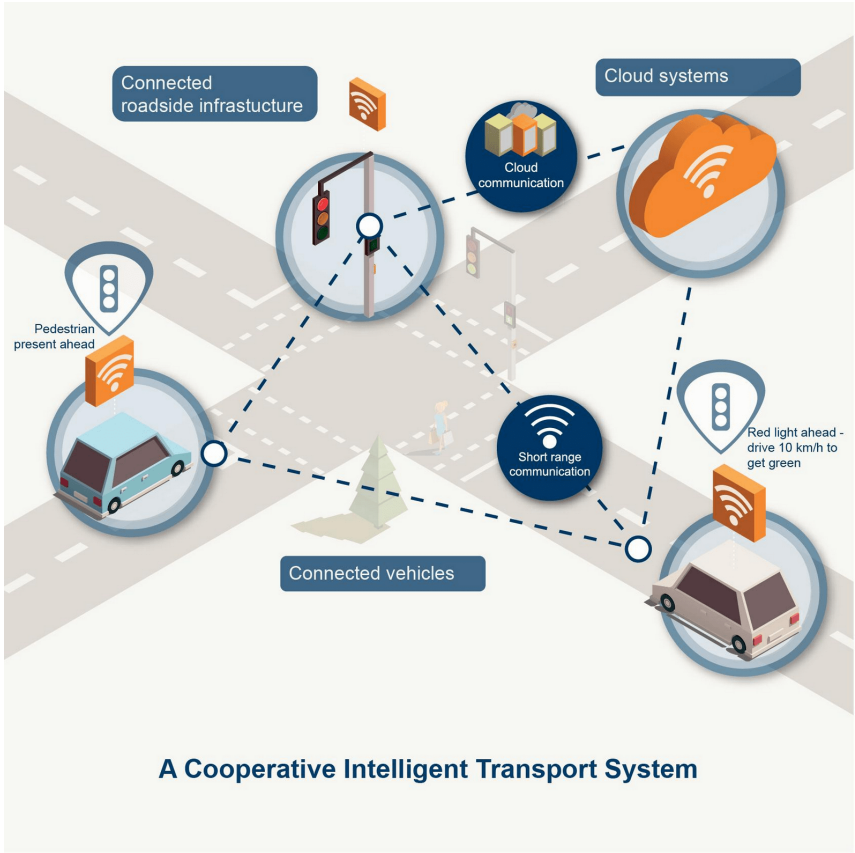


Figure 1.1: Depiction of hazardous traffic intersections

occluded, as explained before, this could lead to a potential collision, with more or less dangerous outcomes depending on the situation, from an emergency brake to a fatal accident. In the coming future, there is a lot that can be done by cooperative services to prevent such outcome.

For example, a camera-equipped Road-Side Unit (RSU) could share relevant information regarding the crossing pedestrian to the connected (and possibly automated) vehicle. Hence, despite the vision impairment, the latter is able to slow in time, perhaps even avoiding the slightest of emergency brakes. This is all delivered through one

of the latest devised C-ITS services, standardized by the European Telecommunications Standards Institute (ETSI), called Collective Perception Service (CPS) [55], which exactly serves the described use case. Thus, what seemed like a vision for a distant future is much closer to reality than what the reader might expect. Regardless, there are several other C-ITS services which are either in standardization phase or in deployment phase, and thus ready for actual road testing and usage.

1.2 The European agenda for C-ITS

As hinted before, the process of making vehicles and infrastructures cooperate for traffic services cannot happen without a solid common ground; indeed, road actors have to agree on communication standards, protocols, operations, and the likes. Thus, it is impossible to talk about C-ITS without describing all the efforts that the European Community (EC) is doing towards standardizing services, protocols, and cooperation between Union states.

1.2.1 Cooperative, Connected and Automated Mobility (CCAM)

One of the pillars of the EC is the definition of the paradigm of Cooperative, Connected and Automated Mobility (CCAM) [33], which is further explicited in an association with the same name. The goal of CCAM is *"to create a more user-centered and inclusive mobility system, increasing road safety while reducing congestion and environmental footprint"* [26]. The CCAM project is divided into seven clusters with dedicated working groups, as shown in figure 1.2

The importance of both the association and the paradigm lies on the focus on interoperation posed by the acronym itself. Not only the heterogeneity of the cooperative task is explicitly stated, but also there is an entire organization aimed at coordinating all of

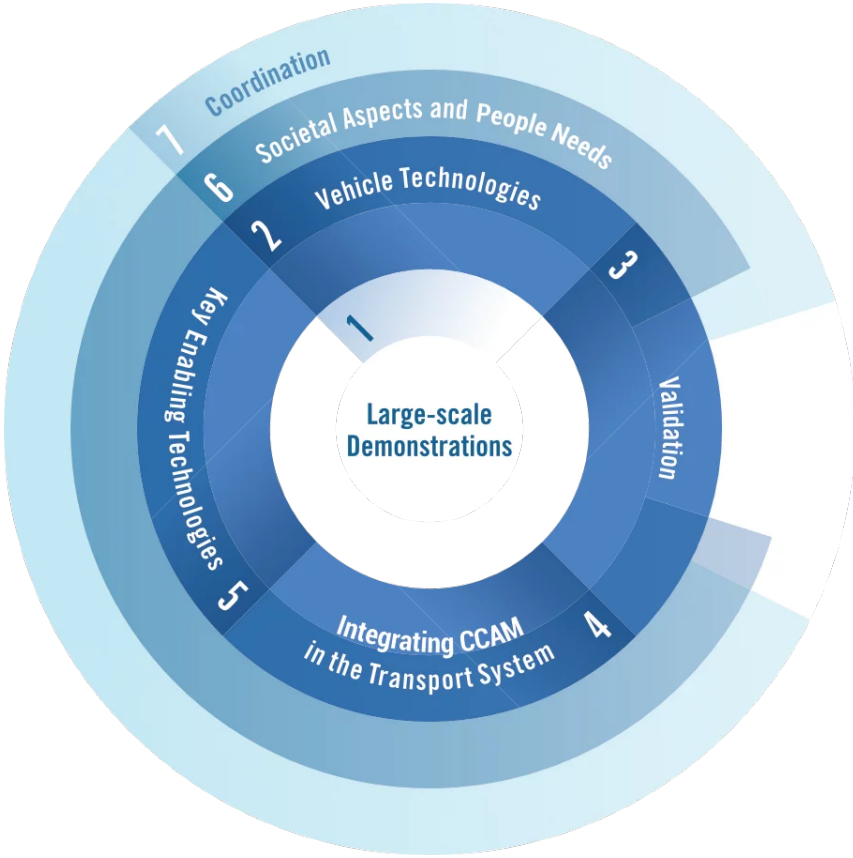


Figure 1.2: CCAM clusters of interest

them towards the common goal.

Since the inception of CCAM, the European Union has founded several related projects through the Horizon Europe program 2021-2027. Each of the projects has a reference cluster, and thus different lesson learned or outputs depending on it. For example, the Synergies project [59], funded in 2024, is part of the validation cluster, thus has set its goal on creating a European platform to share and make interoperable scenario databases. Plus, the platform will implement the Safety Assurance Framework which was instead the goal of a previous project called Sunrise [58], which has recently

ended in August 2025. Sunrise set its goal in creating the aforementioned framework, as well as a method and toolchain for evaluation and a safety assurance handbook for all the interested partners. Another example is the soon ending AI4CCAM project [56]: as part of the key enabling technologies cluster, this project has investigated the trustworthiness of AI-driven systems in Urban scenarios, stepping forward in complex matters such as scenario explaining and understanding or VRU trajectory prediction, with the ultimate goal of making roads safer through CCAM. Finally, the coordination project called FAME has greatly contributed to the creation of a common, shared knowledge base [57] regarding research and innovation projects, regulations, standards, evaluation guidelines, data sharing and even a map of all the CCAM test sites spread all across Europe.

1.2.2 The C-ITS act and C-ROADS

Given the increasing relevance of C-ITS services, the European Commission decided on November 30, 2016 to adopt an European Strategy aimed at normalizing and investing in such services with the goal of supporting and speeding up their availability in Europe. The idea was to create an attractive platform on both a legal and economic point of view [35].
The main step the European Commission took toward C-ITS services was the C-Roads Platform, which consists of two major phases: [101]

- Phase I (2014-2016): during this phase the participating nations of the commission started this platform and agreed on a shared development of C-ITS services within the European Union borders. The idea was that the different C-ITS services, developed by each member state, could interact seamlessly between each other.

- Phase II (2016-2017): during the second phase the aim of this project shifted to the deployment of these services. It was based on taking into account essential information such as road safety and privacy.

The C-Roads platform adopts a holistic approach, aiming to cover all areas involved in the development of C-ITS services. It includes sharing the acquired knowledge and the occurred problems, but also it promotes the exchange of the ideas and solutions among all the users. Moreover, this platform incentivises a bottom-up approach, starting with national developments and solutions and scaling up to cover the entire European landscape. Given that this projects starts with national solutions, testing becomes of primary importance to ensure that these services can be utilized across Europe and comply with agreed European principles [20].

1.2.3 C-ITS taxonomy

C-ITS services are categorized depending on various aspects, such as the degree of technological maturity as well as their market readiness. Starting from 2016, the European Union has given a first taxonomy of C-ITS services, as published in [35]:

- Day 1 services, which are considered technologically mature for end-users. Plus, they're already being deployed in some degree or form (e.g. in-vehicle speed limit, infotainment services, traffic jam ahead warning, weather conditions)
- Day 1.5 services, considered technologically mature but whose full specifications or standards are not production-grade (e.g. fueling & charging stations information, on street parking information, traffic and smart routing)

Some of the already established use-cases are showcased in a report made by the C-Roads platform, showcased in figure 1.3.

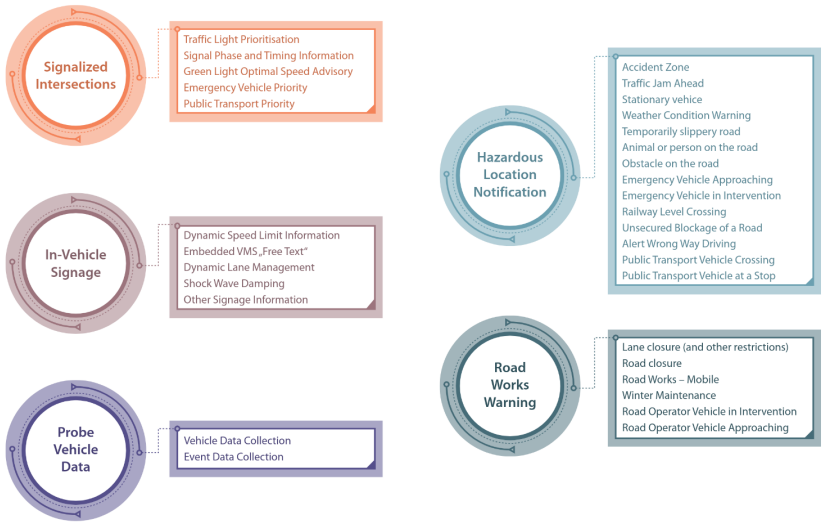


Figure 1.3: All verified Day 1, Day1.5/2 C-ITS services, as per [21]

However, a further vision is provided by organizations such as the Car2Car consortium, one of the European initiatives on road technologies with a specific focus on road safety [23]. The consortium, including vehicles manufacturers, equipment suppliers, engineering companies, road operators and research institutions, also defines Day 3 services, which will *"add further sophisticated services like sharing intentions, supporting negotiation and cooperation that paves the way towards cooperative accident free automated driving"*. For this reason, they distinguish three deployment phases, as described in figure 1.4:

- Awareness driving: *"The exchange of status data via cooperative V2X communication, e. g. the position, speed, driving direction or special incidents like a vehicle defect, enables a set of information and warning services"*, such as emergency vehicle, traffic jam or stationary vehicle warnings.

- Sensing driving: *"On top of status data, cooperative V2X ca-*

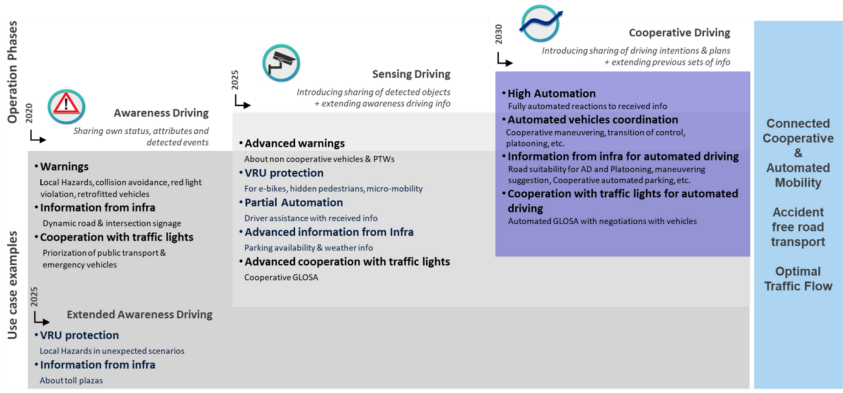


Figure 1.4: ca2car day 3 services deployment towards Vision Zero

pable road users can share observations gained by sensors, and advanced environmental information". Examples of these services are overtaking driving or cooperative adaptive cruise control.

- Cooperative driving: finally, *"in addition to status and sensor data, cooperative V2X road users can also provide intention data, allowing them to interact intelligently and to coordinate their behaviour even in complex traffic situations.".* This category befalls to platooning or cooperative merging services, for example.

Further information on the most known standardized C-ITS services, as well as on the related C-ITS messages, can be found in Appendixes B and A, respectively

1.2.4 Italian Directives for C-ITS

In recent years, the Italian government has been paying great attention to the issue of Smart Roads, promoting a policy aimed at standardizing and unifying practices and being first and foremost the guarantor of the creation and optimal management of the Ital-

ian road network. For this reason, 2018 saw the release of the so-called Smart Road Decree [63] by the Italian Ministry of Infrastructure and Transport (MIT); this decree, aimed at fostering the digital transformation process, introduced the concepts of traffic observation and monitoring platforms as well as data and information processing models; moreover, it provides advanced services to infrastructure managers, public administration and road users, so to create a technological ecosystem able to guarantee the interoperability between new infrastructures and vehicles generation. In summary, this digital transformation process is finalized to enhance traffic management systems by exploiting real-time traffic congestion conditions; enhance road safety and security by introducing innovative services enabled by new technologies; and ensure interoperability with new generation vehicles and the commissioning of C-ITS services, starting with the day-1 services identified by the EC.

1.3 C-ITS deployment challenges

Bringing these systems to actual road usage is a rather complicated task. First of all, the heterogeneity of the involved stakeholders is a double-edged sword: while on one hand creates a heavily interconnected environment, on the other hand it is difficult to merge all the heterogeneous necessities. As a basic example, the needs of a road operator are completely different from a car manufacturer ones. In turn, this leads to impactful integration costs and major development challenges. Despite these difficulties, it is mandatory to assess the impacts of any cooperative solution on the road environment taken as a whole: can a truck platoon formation form in conditions of heavy traffic ahead? Will a slow down policy enforcement due to a subsequent traffic jam impact nearby entrances on the studied highway? Plus, this issue is further exacerbated by the

further spreading of automated vehicles in the coming years.
Another major obstacle towards the diffusion of C-ITS is the security and privacy concerns. All in all, connecting thousands of devices together means sharing an enormous amount of data, with a clear threat to security. In cybersecurity terms, it is mandatory to also assess the impacts on this extended attack surface, evaluating the privacy risks, such as position and identity spoofing, and ensuring that that the Public Key Infrastructure (PKI) is working correctly.
Therefore, the combination of standard compliance, security risks, and heterogeneous stakeholders are still open challenges towards C-ITS design and deployment. In order to sort these problems and address them correctly, it is important to understand what are the current strategies, as well as the limitations, regarding the whole process.

Chapter 2

State of the art: C-ITS development and deployment

Following the briefly described challenges towards C-ITS development and deployment, this chapter serves as an overview of the major trends on the subject.

2.1 The role of Testing and Validation

One of the most debated aspects of C-ITS and automated vehicles is related to testing and validation strategies. After all, the growing sophistication of Cooperative Intelligent Transport Systems (C-ITS) calls for robust and efficient evaluation approaches. These methods must ensure safety, quantify the benefits, and identify potential challenges. Moreover, the validation processes should demonstrate the soundness of the operations of automated and autonomous vehicles, including routine performance, crash avoidance, and fallback mechanisms. Among the most common testing strategies we can include *Field Operational Testing* (including *Real-World and Closed*

Area testing) as well as various degrees of *Virtual Testing* [111].

2.1.1 Field Operational Testing

Field Operational Tests (FOTs) represent the most authentic validation approach, as they involve prototype vehicles in public or closed environments. While highly realistic, FOTs demand significant effort and resources. Historically, manufacturers have sought to minimize FOT usage due to the high costs. FOTs are further categorized into real-world and closed-area testing.

Real-World Traffic Testing

Real-world traffic tests are conducted on public roads to assess the safety and functionality of automated or autonomous vehicles [85]. Since this method offers the advantage of natural, real-world conditions, it offers several benefits:

- **High environmental validity:**, enabling the assessment of the vehicle within its intended Operational Design Domain (ODD) and diverse conditions.
- **Scenario diversity:** allows testing of elements such as weather and infrastructure (e.g., bridges, tunnels) that are rather impossible to replicate on test tracks.
- **Validation of simulations and track tests:** facilitates comparison of Advanced Driver Assistance Systems (ADASs) performance in simulated, track, and real-world environments.
- **Interaction assessment:** evaluates ADAS performance in relation to other road users, such as maintaining traffic flow and correct behaviour.
- **Comprehensive validation:** supports the validation of models, individual software components, and toolchains.

However, this approach also has notable limitations:

- **Limited controllability:** Public roads offer minimal control over ODD conditions.
- **Low reproducibility:** Scenarios are difficult to replicate precisely in different locations.
- **Restricted repeatability:** Iterative testing of identical scenarios is challenging.
- **Limited scalability:** Public road scenarios may not scale effectively.
- **Resource-intensive:** Requires substantial resources and time, though less costly than track testing.
- **Safety risks:** Poses potential hazards to test personnel and the public.

Closed Area Testing

Closed area testing involves conducting tests in specially designed, controlled environments [149]. This method allows for the evaluation of physical vehicles through a defined set of realistic scenarios, with controllable external inputs and conditions. While safer than real-world testing, it remains resource-intensive and is typically used for critical, pre-selected scenarios. Advantages include:

- **Controllability:** Enables management of many test elements, including certain ODD aspects.
- **Fidelity:** Involves functional, physical vehicles, real obstacles, and environmental conditions.
- **Reproducibility:** Scenarios can be replicated across different locations and testing entities.

- **Repeatability:** Allows for multiple test iterations under identical conditions.
- **Efficiency:** Accelerates exposure to rare or safety-critical events compared to real-world testing.
- **Simulation validation:** Supports comparison of ADAS performance in simulation and test track environments.

Disadvantages are:

- **Time and cost:** Requires significant setup time and specialized equipment.
- **Limited variability:** Infrastructure and conditions may be difficult to modify for diverse test elements.
- **Safety risks:** Physical vehicles and obstacles can create hazardous environments.
- **Representativeness:** Cannot fully replicate the complexity of real-world environments.

Artificial cities Within this landscape, an interesting use case is brought by Artificial cities, which are environments built for the exact purpos of testing Autonomous Vehicles (AVs) for testing autonomous driving (AD) systems. Examples include the K-city in South Korea and the Mcity in the USA. For example, Mcity features building facades, a tunnel, a bridge, a four-lane highway, mechanical pedestrians, road markings, and traffic lights. Testing in such environments offers several benefits:

- **Safety:** Minimizes risk to other road users.
- **Reproducibility:** Test conditions are highly reproducible, except for weather.

- **Environmental configuration:** Allows for flexible adjustments, such as changing signs or traffic lights.

However, drawbacks include:

- **Danger to drivers:** Some risk remains for test drivers.
- **High cost and time:** Construction and maintenance are expensive and time-consuming.
- **Vehicle functionality:** The vehicle must be fully operational for testing.

Test track testing Test track testing is a cost-effective alternative, often used as an initial phase before real-world trials. Individual obstacles are placed on the track to validate specific safety systems, such as emergency braking. This method is popular due to its affordability and rapid implementation; however, it still retains all the drawbacks related to closed area tests, plus the possible complications due to the low degree of realism offered by the track layout.

Semi-Virtual Tests Semi-virtual tests combine virtual and real elements. Initially, a real track is mapped into a simulation. A test vehicle equipped with high-accuracy Differential GPS (DGPS) sends its location to the simulator, which returns environmental data. This data is fed to the vehicle's Electronic Control Unit (ECU), triggering the active safety system. The test is conducted on an empty track, but the vehicle perceives virtual traffic. Drivers may use augmented reality goggles to enhance the experience. Another approach involves driving on an empty track while wearing VR goggles that display a fully virtual environment. The main advantage is the driver's experience of relevant forces during a virtual test. However, the presence of a driver in a prototype vehicle can be hazardous, a risk mitigated by using remotely controlled actuators.

2.1.2 Virtual Testing

Virtual testing is a powerful method for evaluating automated/autonomous systems under conditions that are impractical for physical testing. It involves replacing physical elements with simulation models to replicate real-world interactions. Virtual testing expands the scope of physical tests, enabling cost-effective assessment across a wide range of variables and scenarios. It also allows for the safe verification of system requirements in uncertain conditions, without the risk of real accidents [82, 4, 135].
Virtual testing is particularly useful for evaluating C-ITS services in safety-critical scenarios that are difficult or unsafe to reproduce on test tracks or public roads. This approach allows evaluating the system performance through agile, controllable, repeatable, and efficient validation. The simulation toolchain for virtual testing can be categorized based on the interaction between the system under test and the environment:

- **Open-loop virtual tests:** Virtual object actions are data-driven and not self-corrected based on system feedback.

- **Closed-loop virtual tests:** A feedback loop continuously sends information from the controller to the C-ITS service, allowing digital objects to react dynamically.

Model-in-the-Loop (MiL) In the Model-in-Loop (MIL) approach, the system is modeled using tools such as MATLAB/Simulink. This high-level model allows for system design without delving into implementation details. Testing involves providing simulated inputs and analyzing the model's response.

Software-in-the-Loop (SiL) Software-In-Loop (SIL) testing evaluates the software implementation on general-purpose computing

systems. It uses executable code, such as algorithms or entire controller strategies, within a modeling environment. Risks include potential differences in machine code due to compiler variations and behavioral discrepancies when executed on final hardware.

Hardware-in-the-Loop (HiL) Hardare-In-Loop (HIL) testing involves the final hardware of a vehicle subsystem running the final software, with inputs and outputs connected to a simulation environment. This method replicates sensors, actuators, and mechanical components, testing Electronic Control Units (ECUs) before final integration. The main limitation is the lack of real-world interaction with other ECUs and power supplies, which may affect real-world performance.

Vehicle-in-the-Loop Vehicle-In-Loop (VIL) combines a real vehicle with virtual objects, reflecting real-world vehicle dynamics. It can be conducted on a vehicle test bed or track and supports both open and closed-loop testing.

Driver-in-the-Loop (DiL) Driver-in-Loop (DIL) testing, typically conducted in driving simulators, evaluates human-automation interaction. It uses physical hardware and must operate in real-time, which can limit scalability.
All in all, the advantages introduced by virtual testing can be listed as:

- **Controllability:** offers unparalleled control over test aspects.
- **Agility:** allows for immediate reevaluation of system changes.
- **Efficiency:** enables concurrent testing in a short timeframe.
- **Cost-effectiveness:** lower running costs compared to physical testing.

- **Scenario coverage:** facilitates exploration of a wide range of safety-critical scenarios.
- **Data analysis:** provides a convenient platform for data gathering and performance analysis.
- **Repeatability:** ensures identical re-execution of virtual tests, aiding fault identification.

However, the clear disadvantages are:

- **Lower environmental fidelity:** models may not fully replicate real-world environments and behaviors.
- **Risk of over-reliance:** overemphasis on virtual results without sufficient physical validation.
- **Expensive software lifecycle:** simulation models may require ongoing, costly maintenance.

Despite the recognized drawbacks of virtual testing [134], the continuous development of new pieces of software makes its usage much more common and widespread. Thus, over time, more and more simulation platforms for C-ITS and AV have been deployed and showcased to the public.

2.2 Simulation - and Co-simulation - platforms

2.2.1 Traffic environment components

Given the importance and usefulness of virtual testing approaches, over the years we saw an increasing number of platforms, focused on one or more aspects of the traffic environment components. Following the schema shown in figure 2.1, we can identify the following components:

- **Road Infrastructure**: consist in the entirety of the road layout, including number of lanes, crossroads, emergency lanes, and the likes.
- Traffic, of any kind of road user, from passenger vehicles, to trucks, to bicycles and pedestrians
- **Communication System**, including realistic protocols as well as standardized messages
- **Monitoring System**, capable of detecting all kinds of road actors
- **Traffic Management Center**: also called Traffic Control Room in a real scenario, it is necessary to validate any actuation proposed by the system. For example, it is the driving factor behind traffic policies activation, such as a dynamic speed limit
- **Actuation System**: the set of enforced road policies and actuations made on the road, both by the operator and the involved vehicles
- **Controlled Vehicles**: differently from all the involved traffic, it comprises the connected and/or automated vehicles which are part of the simulation, and actively studied during it

Figure 2.1: Depiction of all the possible traffic environment components

depending on the different focus of the specific developer, many are the efforts towards representing one or more of the underlined components.

2.2.2 Most employed standalone simulators in AV

Given the strategic context and traction towards connected vehicles, it should not come as a surprise that the majority of standalone simulators are heavily focused on vehicle dynamics. A non exhaustive list include Webots, Prescan, CARLA, DYNA4, rFpro,

or dSPACE simulators. Not only most of these are commercial products, but they are also often employed by car manufacturers in production, other than in various third-party applications. Nonetheless, several of them have already been proven a good fit for connected and automated applications [128]. For example, a study concerning autonomous vehicles interactions with bicycles is tackled using DYNA4 [116]. Or, there are various applications of Prescan towards simulating traffic accidents [143] or cooperative driving applications [83]. Multiple are the examples of employing the open-source Webots simulator in vehicular applications [103, 113, 61]. Probably, the most notable and well known software is the open-source CARLA simulator.

Ever since its inception in 2017 [43], CARLA, acronym for CAR Learn to Act, has rapidly become one of the most prominent tools in autonomous driving testing. At the time, it was one of the first photorealistic tools for autonomous vehicles training and testing [133, 44]. Later on, it became a versatile and widespread tool for AV development, employed among other usages for generating scenarios datasets [40], testing against physical adversarial attacks [148], modelling non linear control strategies [25] or evaluating more advanced deep learning methods [69, 105]. As previously hinted, this simula-

Name	vehicle dynamics	sensors	v2x	traffic modeling	graphics engine	3d	License
Veins	-	-	Omnet++	SUMO	-	-	GPL
Webots	X	X	-	SUMO	OpenGL	-	GPL
CARLA	X	X	-	SUMO	Unreal	X	GPL
Prescan	CarSim	X	-	Vissim	Unreal	X	Commercial
DYNA4	Simulink	X	-	SUMO	X	X	Commercial
dSpace	ASM	X	-	X	X	X	Commercial
rFpro	X	X	-	SUMO	X	X	Commercial
IPG Carmaker	X	X	-	X	X	X	Commercial

Table 2.1: Non-exhaustive list of commonly employed standalone simulators. the "-" symbol stands for a missing feature, while "X" means the resource is present, albeit either proprietary or unspecified

tor has also been extended to enable Hardware-In-Loop strategies [18], including applications for teleoperated driving research [74] or innovative ways to overcome the potential pitfalls in timeliness and real-time dependability [37, 110].
Crucially, all of these tools share a decisive weakness: none of them is directly designed to include C-ITS development and deployment within their core loop. This lack basically remarks the heterogeneity underlined in chapter 1, prompting the development of newfound solutions which can incorporate the different aspects of the road environment.

2.2.3 Co-simulation platforms

The increasing need of testing and validation of C-ITS has pushed the research efforts towards many different tools, each one trying to incorporate one or more aspects of the connected mobility landscape [132]. These are commonly referred as *Co-simulation platforms* as they usually integrate one or more tools together, or extend pre-existing ones; nonetheless, despite the heavy focus on AVs, there already are established tools for microtraffic simulation, such as the open-source Simulation of Urban MObility (SUMO) software, as well as network simulators, OMNeT++ and ns3[1] to name a couple. As a first example of co-simulation software, the decisive need of testing V2X technologies has already pushed the development of two tools which couple the aforementioned software, namely Artery [119] and Veins [130]; they were created as an extension of one another, and served the purpose of allowing simulated traffic to enable V2X communication. Since they have been used in a multitude of tests and simulations [71, 77, 107, 87], they stand as a notable example of how a co-simulation platform can, albeit only for just two of the identified traffic environment components, being Traffic and Communication System (see Table 2.1), enable realistic testing of

[1] see OMNeT++ [108] and ns3 [106] website for further info

V2X based communications.
Other than that, there are many different efforts towards co-simulation strategies, which are summarized in table 2.2 and discussed below. Cui et al. develop a universal Cooperative Adaptive Cruise Control (CACC) evaluation platform, which targets compatibility with multiple controllers. Moreover, the controller is tested towards stability and crash severity. The simulation platform design is not disclosed, thus kept Ad-Hoc for the task, and crucially missing wireless network simulation as a crucial feature. Jia et al. Instead focus on two open-source tools, being SUMO and Webots, to model automated driving behaviours in a traffic environment, also integrating a network simulator in Omnet++. Plus, the paper shows an example of traffic flow optimization as proof of concept. The work by Zhang et al. also tackles the task of creating a simulated traffic environment where to test Connected Automated Vehicles (CAVs) with realistic scenarios, also testing various algorithm procedures, such as cooperative driving at signal-free intersections.
When referring to the concept of *Digital Twin*, Wagner et al. incorporated a Green Light Optimal Speed Advisor (GLOSA) algorithm into their hardware Programmable Logic Controller (PLC), which is used to communicate with a real connected vehicle. The simulation, which instead takes care of the surrounding traffic, is obtained through coupling Sumo and Omnet++, while using Matlab as an interface for the PLC. The authors themselves admit their willingness to extend their platform to accommodate for a 3D simulator, such as Unity 3D.
Finally, a notable example of a multi-stack framework is represented by Raviglione et Al. In this case, the authors couple the SUMO TRAffic Control Interface (TRACI) with the well-known network simulator ns-3; this allows to reproduce via software the full network stack of C-V2X messages, which are realistically represented within the simulation. Basing the platform on a network simula-

tor also means that all the communication methodologies, such as Wireless via 802.11p and LTE, are fully reproducible. Moreover, the authors also allow the possibility to make HiL tests via the such called *"emulation mode"*. While this platform sounds extremely promising for the foreseeable future, as a further extension including CARLA has already been envisioned [24], the full depiction of all the traffic components is still crucial towards C-ITS; nevertheless, only including vehicles inside the simulation omits the presence of a vehicle-agnostic traffic management service, infrastructure sensors, and the likes.
As we can see, despite many of these solutions manage to capture some of the envisioned traffic environment components, they fall flat at creating comprehensive solutions that encapsulate them all:

1. First of all, similarly flawed to the tools described in section 2.2.2, some co-simulation platforms are solely dedicated to Autonomous Driving testing. While this was the clear intent of the authors, it is indicative of the fact that C-ITSs are still not tackled thoroughly in their evaluation and distribution

2. Moreover, even when the platforms do include V2X and/or C-ITS strategies, the evaluation is still end-to-end; thus, the traffic environment is not taken as a whole, but rather as a collection of use-cases, such as signalized intersection or merging strategies. As much as safety-critical scenarios are an established technique to validate AV behaviour [42, 153], this practice takes away the overview on the full traffic scenario. Hence, it becomes nearly impossible to evaluate ITS management strategies

3. Since tests on C-ITS and cooperative driving together are shallow, it is likewise difficult to evaluate the impacts of such technologies into a realistic V2X scenario, especially in terms of cybersecurity. This includes issues regarding misbehaviour

detection, platooning, and different categories of attacks on C-ITS, as we will further discuss in section 2.4

Comparison Analysis

Taking a more formal approach to our analysis of the discussed C-ITS platforms, we performed a compatibility assessment between the latter and the traffic environment components identified in figure 2.1; to goal here is to check if there are one or more platform which comply with our theorization of the components, or at least for the most part. The analysis is summarized in table 2.3. On the columns, we have the platforms reviewed in the previous subsection. On the rows, the traffic components as well as additional informa-

Paper	Deployed co-simulation strategy	Use cases
Riebl et al. [119]	Sumo + Omnet++	ITS & V2X
Sommer et al. [130]	Sumo + Omnet++	ITS & V2X
Cui et al. [36]	Ad-Hoc	Adaptive Cruise Control
Jia et al. [76]	Sumo + Omnet++ + Webots	Automated Driving
Zhang et al. [152]	Ad-Hoc	Connected and Automated Vehicles
Wagner et al. [142]	Sumo + Omnet++ + Matlab	ITS Digital Twin
Raviglione et al. [118]	ms-van3t (SUMO + ns3)	C-ITS and ITS management
Schrab et al. [122]	Eclipse Mosaic platform	Realistic V2X and C-ITS

Table 2.2: Literature review on published works regarding co-simulation frameworks and their applications

co-sim platform / traffic env. component	Riebl [119]	Sommer [130]	Cui [36]	Jia [76]	Zhang [152]	Wagner [142]	Raviglione [118]	Schrab [122]
Traffic	X	X	-	X	X	X	X	X
Communication System	X	X	X	X	X	X	X	X
Monitoring System	-	-	-	-	X	X	-	X
Traffic Management Center	-	-	-	-	-	-	-	impl
Actuation System	-	-	veh	veh	veh	veh	impl	impl
Controlled Vehicles	-	-	X	X	X	-	impl	impl
Road Infrastructure	X	X	-	X	X	X	X	X
Year	2015	2019	2018	2021	2023	2023	2024	2022
Code availability	X	X	-	-	X	-	X	X
Actively maintained?	X	X	?	?	N	?	X	X

Table 2.3: Compatibility assessment between the co-simulation platforms found in literature and the traffic environment components seen in figure 2.1. *Impl* means that the feature is implementable, on a software point of view, without major extensions

tion regarding the year of inception, whether the code is available and whether the platform is actively maintained. As we can see, almost any of them has traffic, road infrastructure (mostly through SUMO) and communication systems at disposal, largely due to the scope of these platforms. As we also underlined before, the traffic monitoring system as well as the management center are often neglected, mostly since the platforms focus on the single, automated vehicle instead of the traffic flow as a whole. This also reflects the inability of reproducing actuation systems which are not solely tied to the tested vehicle. Of course, these classes of platforms have ample support for controlled vehicles, as it is their primary focus. Crucially, only five out of the eight platforms have code available, and only half are still actively maintained. Moreover, it has to be noted that this analysis does not take X-in-Loop (XIL) evaluation into account.

2.2.4 Focus: Eclipse Mosaic

One notable example of a valid solution for C-ITS testing and development is represented by Eclipse Mosaic [122][2]. Written in Java, it was originally a standalone project developed by the Fraunhofer Institute for Open Communication Systems (FOKUS), the DCAITI (Daimler Center for Automotive IT Innovations), and the openMobility Working Group; afterwards, it became part of the Eclipse Foundation and it is now an open-source project under the Eclipse Public License.

At first glance, Mosaic offers a lot of desirable features for the aforementioned goal:

[2]While the work by Raviglione at al. [118] might also seem a valid alternative, it was crucially released in the second half of 2024, which is when this thesis work was more than on the way. This does not take away from the fact that this work might be re-evaluated for future considerations

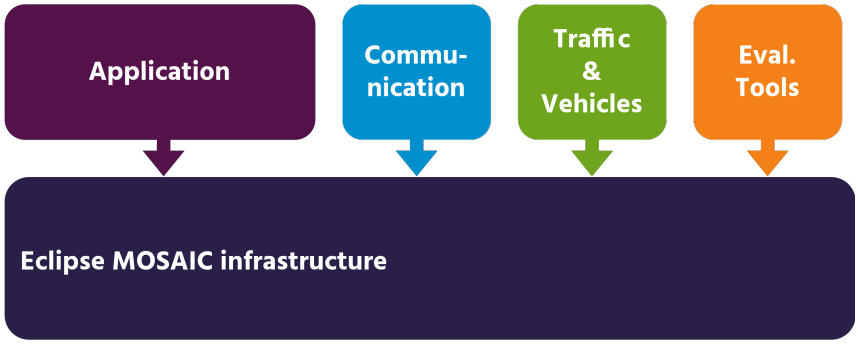


Figure 2.2: Eclipse Mosaic Software Components Overview

- **Integration of pre-existing simulators**: without reinventing the wheel, Mosaic already couples heterogeneous simulators; this includes SUMO for micro-traffic simulation and a custom network simulator, called Simple Network Simulator (SNS), written by the Mosaic team in Java. Moreover, this communication simulator can be swapped for other established ones, like Omnet++ or Ns3.

- **The addition of the application simulator**: a crucial aspect which makes Mosaic immediately ready to use is the addition of an application simulator, also developed in Java by the creators. This simulator allows users to create a control logic for connected vehicles, directly impacting the SUMO simulation. Plus, applications can also be written for servers or RSUs, effectively offering a full stack to characterize various traffic environment components. The tight coupling with SUMO is even capable of writing application logic depending on active road elements such as detectors or traffic lights.

- **Scalability**: as a notable mention, Mosaic is built to provide standardized interfaces, so that new simulators could easily be coupled, at least in theory. As an example, the authors underline the possibility of also having a 3D simulator in the

loop as CARLA, which has been recently integrated [36].

- **Other benefits**: the Mosaic project includes several evaluation tools, such as logs for the simulators and the road actors applications, as well as more specific tools which are part of an *extended*, commercial license.

Regarding its software architecture, MOSAIC offers standardized interfaces, and it is based on a so-called *Run-Time infrastructure (RTI)* that is the building block for the simulators to be coupled with through a Federate-Ambassador schema. Basically, for a simulator to work with MOSAIC, a standardized interface has to be provided to offer both a Federate Ambassador, which will handle the communication from MOSAIC to the simulator, and a MOSAIC Ambassador for the vice versa. This is all managed by the Federation Management in the RTI, while the Time Management and Interaction Management take care of synchronization and data transfer between all the coupled simulators.
As we can see, Eclipse Mosaic looks promising on paper to fulfil a lot of the necessities underlined in this section. Crucially, Mosaic is built as a software-only product, without the possibility of enhancing the realism of the simulation with further tools or testing. Indeed, at the time of writing, almost every research work which employed Mosaic [114, 123, 102, 124, 6, 125] as a simulation tool does not include any of the aforementioned XIL strategies.

2.3 Notable C-ITS T&V examples

Despite all the mentioned criticalities regarding C-ITS development and deployment, and regardless of which simulation strategy employed, there are several examples in literature of different tests of these intricate systems, some of which are reported in the section below.

Software and driver-in-loop applications Aramrattana et al. [8] deploy yet another framework for C-ITS, which is built in-house, and it is comprised of SUMO and Omnet++ plus their own driving simulator. The addition of this simulator inside the loop is beneficial to test some peculiar C-ITS strategies, albeit some not involving the collaboration of the infrastructure in any way, such as platooning and platooning merging, and their interaction with actual human drivers. This is crucial, as the cohabitation on the road of both human and computer agents will definitely happen in the coming future. The interaction between human drivers and automated vehicles is also investigated by Wang et al. [145], whose aim is to evaluate the impact of C-V2X communications on traffic safety and efficiency. More specifically, by using a newfound car-following model, called *Acceleration-Impairment Driver Model (AIDM)* and extended from the conventional IDM, they showcase how the higher the penetration rate of AVs, the lower both the number of conflicts and the travel time in minutes become. Another driving simulator test is shown in [150], where the authors leverage a combination of Vissim and Unity to understand driving behaviours at *"dilemma zones"*, such as intersection scenarios where the traffic lights are transitioning from green to yellow. The study shows that the addition of C-V2X, including traffic signal information and proposed speed limit to approach the intersection, changes the way drivers tackle the dilemma zone, showcasing various behaviours depending on connection quality and packet loss.

Hardware-In-Loop and Digital Twins Rapelli et al. [117] propose OScar, short for Open Stack car. Their goal is to reproduce a full ETSI-compliant, C-ITS stack for embedded boards which has to be lightweight and fit even low-end hardware, motivated by the lack of open-source implementations available to the public. Within this spirit, they implemented the full stack on a low-end board which

could transmit at the V2X frequency, being 5.8/5.9 GHz, together with a GNSS antenna connected to another external board. Their evidence through CAM dissemination show the validity of the developed software stack for V2X and digital twins applications. Different is the work by Geller et al. [64], which instead focuses on extending the previously mentioned CARLA platform by creating CARLOS. This framework prioritizes on containerized building blocks, which arranged in a microservice architecture greatly speed up the process development and deployment stages through the concepts of data-driven development and automated testing. Their architecture is composed of the CARLA server, a ROS bridge as a communication actor and a control actor being the Scenario Runner software. Their evaluation shows the scalability and test capabilities of the framework, opening up on valuable digital twin realizations.
Finally, the work by Capdevila et al. [22] presents a Mixed Reality Environment (MRE) tailored for ADAS and HIL testing. The environment is composed of a fully equipped mobility laboratory, including a real dissected car placed in front of a 220°angled curved screen and all the necessary sensors including LiDAR, RADAR and 4k cameras. This effort is conducted with the ultimate goal of allowing for comprehensive hardware and software testing, especially in scenarios where the AV shares the road with potential hazards or VRUs.

Real world applications There are also some reported use cases of real world applications based on C-ITS applications. For example, Kang et al. [81] analyze the impact of C-ITS services on driving behaviour in the Daejeon-Sejong test bed in South Korea. More specifically, this road section was equipped with RSUs which could communicate with voluntary drivers offering their location data in exchange for OBU capabilities through a smartphone app, eventually enabling C-ITS warning services. Therefore, this year-long

experimentation was tackled to check the effects of various C-ITS services, as well as their impact on data streams. Their analysis show how the involved drivers suffered much less from hard braking and deceleration, underscoring the potential benefits of widespread deployment of C-ITS services.

Another example of real world testing is the work by Marquez-Barja et al. [98], which described the Smart Highway test bed for V2X built nearby the city of Antwerp, in Belgium. This smart highway is equipped with standard-compliant RSUs as well as GNSS receivers, with a cloud-based backend for more intense computations, posing this work as a crucial application for the foreseeable future of C-ITS validation and deployment.

2.4 C-ITS cybersecurity

In their essence, C-ITS services undermine non other than standard wireless (and cellular) communications. Thus, it is impossible to discuss about them without mentioning all the possible shortcomings caused by the very nature of these types of applications. It could indeed be said that wireless communications have been the staple of internet connections for decades, and C-ITS are "just" one of their possible variations. However, there are definitely some issues to immediately pinpoint, whether they might or not only relate to C-ITS and V2X [144, 126], which are separately tackled below.

2.4.1 Common Cyberattacks applied to C-ITS

The most known Cyberattacks in literature are reported in table 2.4. For each category, when found, there are some literature examples on the matter. As we will soon see in the subsections below, the most critical attacks in C-ITS are mostly related to authentication issues or to availability due to malicious users disrupting safety critical applications.

Category	Attacks
Authentication	Sybil attack, GPS spoofing/position faking attack, Node impersonation attack, etc. [9, 141]
Availability	DoS attack, DDoS attack, Jamming attack [97, 3], black hole attack, etc. [155, 94]
Data Integrity	Masquerading attack, Replay attack, etc.
Confidentiality	Eavesdropping attack, Traffic analysis attack, etc. [86, 80, 28, 41]
Non-repudiation	Loss of events traceability, etc.
Real-time constraints	Timing attack, etc.

Table 2.4: Security Threats by Category

2.4.2 The issue of multiple connected devices

The extremely large number of connected devices on the road have two major drawbacks:

- The heterogeneity of both the hardware devices and the C-ITS software implementations leads to a larger attack surface for the malicious user to exploit. In hindsight, this issue is similar to the one caused by the multiplication of connected, low-power boards on software defined vehicles [93]: more connected users on the road means that the possibility for an attacker to find a vulnerable entry point gets higher the more connected vehicles spread on the road. Moreover, if a new Common Vulnerabilities and Exposures (CVE) is found, as it often happens in the field of Information Technology (IT), it is highly unlikely that all users download the new update patching the cybersecurity risk, and so on. Since these issues are highly correlated to standard Internet of Things (IoT) approaches, similar solutions have been tested to conduct cyber threat analysis on the matter [62] as well as other kinds of security assessments [120, 72].

- Message Congestion is another major issue in C-ITS. Using Cooperative Awareness Message (CAM) messages as an example, their frequency ranges from 1 to 10Hz; thus, the higher the penetration rate of connected vehicles, the higher the risk of channel congestion due to the high influx of messages. And this issue is already established without taking into account other kinds of messages, including more critical ones such as platooning messages or cooperative perception ones. To solve this issue, European Telecommunications Standards Institute (ETSI) has disseminated a document regarding Decentralized Congestion Control (DCC) strategies, which aim at reducing

the number of messages in case of high channel pressure[3]. However, as the ETSI document does not keep newer, more complex services into consideration, several other approaches on DCC have been explored in literature [11], both evaluating the effectiveness of the ETSI standard [95] or exploring alternative routes [121, 38], especially for more complex services [68, 10]

2.4.3 The role of the PKI

As for any other application depending on internet communications, C-ITS services also need the CIA triad of information security: *confidentiality, integrity and availability.* Two of them, confidentiality and integrity, are directly solved by the PKI. We are referring to the security framework that enables the secure exchange of information through the use of public key cryptography, digital certificates, and trusted third-party entities. PKI establishes a hierarchical trust model, where Certificate Authoritys (CAs) issue, manage, and revoke digital certificates that bind public keys to the identities of users, devices, or services. Applied to C-ITS, the PKI provides the necessary authentication and message integrity, preventing spoofing, tampering, and unauthorized access, which are critical for the reliability and safety of automated driving and traffic management systems [73].

Some literature works have already tried to assess the performance of the PKI protocol in C-ITS [70, 127] with a look to the future, while others have taken a slightly different approach including certificate-less methods for signature verification [131].
As a final note, the European Commission has established a reference platform, called *C-ITS Point of Contact* regarding *[...] the legal and technical requirements for the management of public key certificates for C-ITS applications by issuing entities and their usage*

[3]see the ETSI official document on the matter [47] for further info

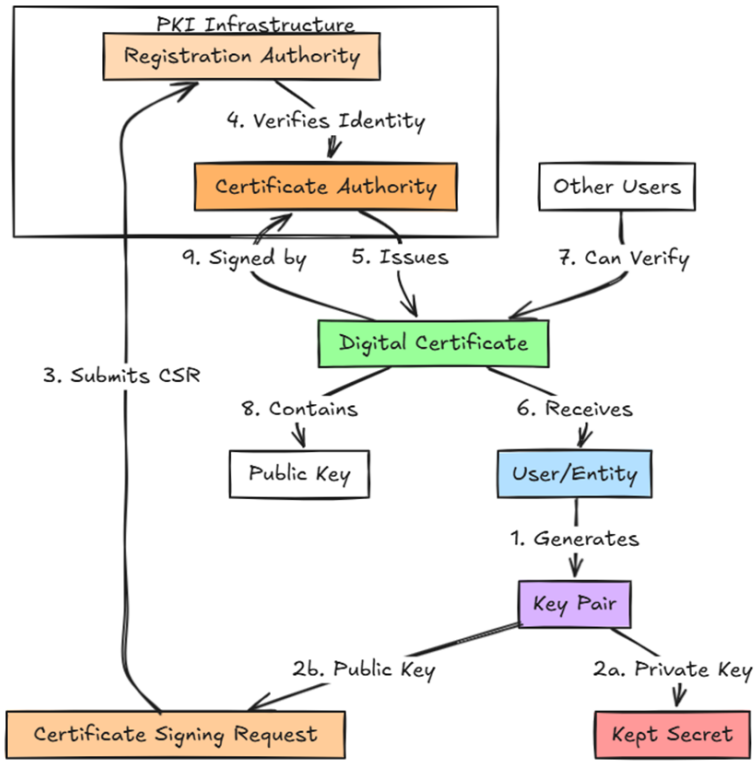


Figure 2.3: A very basic understanding of the PKI architecture. Source: thecyphere blog [15]

by end-entities in Europe.[4] certificate authority. More specifically, the platform provides C-ITS relevant documents and guidelines for the deployment within the European Union C-ITS Security Credential Management System (EU CCMS)[5].

[4]from Certificate Policy for Deployment and Operation of European Cooperative Intelligent Transport Systems (C-ITS) [32]

[5]See the EC website [31] for further details

2.4.4 C-ITS related cyberattacks

User tracking and anonymity

CAM messages, as defined in the ETSI document, broadcast essential details about nearby vehicles. These details cover each vehicle's location, velocity, heading, acceleration, dimensions, and classification, among other parameters. To safeguard privacy and avoid tracking attempts, ETSI has also established a mechanism of *pseudonyms*, which is specified in the related pre-standardization document. Pseudonym usage is crucial, as having the same identity over time would massively increase tracking capabilities by malicious users. Indeed, the pseudonym mechanism not only asks for the usage of different names during the same road trip, changing at set times, but also asks the update of the MAC address, and even the IPv6 address. However, even with these privacy measures, the information shared in CAMs can still allow a malicious user to identify the sender. For instance, if only a few vehicles occupy unique positions relative to the receiver, such as one directly ahead and another directly behind, the receiver can match CAM data to specific vehicles by comparing reported positions with its own. Similarly, if a CAM indicates a transmitting vehicle is 10 meters long and only one large truck is nearby, the correlation becomes straightforward. Studies have found [46] that additional attributes, like vehicle size, significantly enhance the ability to link pseudonyms, potentially allowing up to 80% of vehicles to be tracked. Moreover, the density of vehicles in the area plays a critical role, with the study concluding that location privacy diminishes even when pseudonyms are changed.

Certificate based message verification

Since each message is paired with a digital certificate via PKI, the assumption should be that the receiving vehicle can always validate

the certificate chain back to the root CA to ensure authenticity. However, this assumption does not always hold true in practice, especially when dealing with moving stations such as in vehicular networks.
For instance, Cooperative Awareness Messages (CAMs) do not always include a full certificate [151]. While CAMs transmit a complete certificate at least once per second, more frequent messages use the last 8 octets of the certificate's hash rather than the full certificate. This approach reduces message size and bandwidth usage. According to the message specification, if a vehicle encounters an unknown digest or a certificate signed by an unfamiliar Authorization Authority (AA), it must request the missing certificate from nearby vehicles and await a response . Only after receiving the necessary information can the vehicle authenticate the message. This additional exchange introduces a verification delay of at least several hundred milliseconds, which has a huge impact on safety-critical V2X communications, such as platooning or cooperative maneuvers.

Misbehaviour detection

An interesting topic in the field of C-ITS is the issue regarding *misbehaviour detection*. While this sounds like a very broad term, within this scope has a more restricted definition in *"any abnormal occurrence in the C-ITS network, caused by one or more participating nodes, that disrupts the normal functioning of C-ITS services"* [7, 139]. Generally speaking, literature has found two main reasons as misbehaviour causes [16]: hardware (sensors) failure or malicious attacks, including the common literature attacks shown in table 2.4, but declined in the field of C-ITS.
Misbehaviour can come in various ways, ranging from ghost vehicle attacks, where the message sender places himself on a different geoposition, causing potential issues to the receivers. A traffic analysis attack, where a malicious user sends hundreds of messages com-

ing from ghost vehicles, could cause the detection of a non-existent traffic queue. Or worse, cars driving in a platoon could include a malicious user disrupting the fleet in dangerous ways [13].
The importance of misbehaviour detection lies in its mitigation strategies. For starters, there are realistic ways to find out whether, for example, a ghost vehicle attack is being carried on[129]. Crucially, if the attacker's false position does not match the carriage direction, or a real road position, or the sequence of positions is not physically plausible, the attack is relatively easy to detect, with more sophisticated position falsifications being potentially more effective. Similarly, a traffic analysis attack could be discovered by just comparing historical data, or verifying whether the vehicles'positions are consistent.
As for cooperative vehicles fleets, there are more complex challengers. First, platooning applications run at much higher frequencies than standard C-ITS messages for safety reasons. Thus, a malicious user could create many hazardous disruptions in such test cases. Secondly, it is difficult in a chain of trust such as a platooning application to find the malicious user: if the attack is very sophisticated, one could question whether the malicious user is wrong or the other vehicles in the fleet have some sensors malfunctioning. These issues are all linked to the macro-area of *cooperative misbehaviour*. Indeed, different works in literature [92, 1, 91] have tried to address the matter, from voting systems to fault-tolerant systems to more sophisticated machine learning based solutions.
All in all, this discussion only scratches the surfaces on the various issues surrounding the cybersecurity topic in C-ITS. The mixture of safety issues for road users, unreliability of wireless communications, and dire necessity of user privacy and anonymity generates many different questions which are still up for answers from the scientific and technical community.

Chapter 3

Research questions

The in depth analysis of the current state of C-ITS has ultimately underlined a set of challenges that have to be overcome in the near future. In order to do so, we set as the thesis goal to build a co-simulation platform able to address them. More specifically, we pointed out the following Research Questions (RQs) listed below

- **RQ1: given the cooperative nature of C-ITS, is there a software platform which can accurately reproduce all the heterogeneous traffic environment components** seen in figure 2.1? Only by addressing this heterogeneity, it is possible to really improve simulation realism and accuracy.

- **RQ2: is there a way to test the impact of C-ITSs on the whole road environment?** More often than not, new automated vehicle features are only tested for the single vehicle only. Therefore, any propagating effect on the other road actors is yet to be evaluated. A non-exhaustive list might include understanding the effect of ISA policies on traffic congestions, or evaluating a possible reduction in (check dangerousness) of a specific hazardous traffic intersection when installing collective perception cameras, and so on.

- **RQ3: how can we seamlessly enable the rapid prototyping of cooperative services within their intended environment?** Crucially, the ever-changing landscape of wireless communication and connected devices requires agile methods of development and deployment. Plus, this is further exacerbated by the fact that some of these services are still exiting the pre-standardization phase, hence changes could happen at any time. Being able to quickly respond to these changes is mandatory to ensure new C-ITS services are promptly evaluated and tested.

- **RQ4: when C-ITS services are developed, are all the risks related to user security taken into account?** Such evaluation should happen simultaneously to the development and deployment of C-ITS. Plus, as already underlined in chapter 2, there are several cybersecurity risks linked to cooperative services, such as GPS and identity spoofing, man-in-the-middle attacks, misbehaviour detection attacks and so on. Thus, a thorough analysis would also point out these issues, helping to increase the overall system robustness.

- **RQ5: are C-ITS tested in their intended environment, and with the full message exchange pipeline enabled?** Ensuring this is crucial to allow faster development and deployment times. Moreover, it is important to design new functionalities keeping current standardizations in account, such as including the full PKI into the co-simulation loop.

Taking matter in our hands, we can get a clear thesis statement, which is:

The thesis goal is to **design, develop, validate and deploy a co-simulation platform** capable of performing testing and validation of Cooperative-Intelligent Transportation Systems. The platform needs to represent **all the different road environment components** in a **realistic** and integrated way, covering all the aspects of the Cooperative, Connected and Automated Mobility, thus trying to solve all the underlined Research Questions.

Thesis Goal

More specifically, the rest of this thesis is organized as follows: chapter 4 will describe the design and development process of the co-simulation platform. Then, chapter 5 will showcase some notable applications which are enabled by the newfound platform. Following, chapter 6 will showcase how this platform is also enabled for testing vehicular attacks in C-ITS as well as appropriate software mitigations. Among them, the enhancements made to the platform to allow for PKI infrastructure testing, further increasing the realism of the simulation. As a final note, a real world use-case implementation shaped around the A56 Tangenziale di Napoli Italian highway is described in chapter 7. Finally, we draw some conclusions around the ecosystem and subsequent developments in chapter 8.

Chapter 4

The co-simulation platform

This section defines and motivates the design choices behind the co-simulation platform, envisioned as one of the main thesis goals.

4.1 Platform foundations

As we saw during the literature review in 2, there is a crucial lack in co-simulation tools which can capture all the heterogeneous traffic environment components. This, in turn, leads to a defective analysis, and subsequent evaluation, of the impacts that the C-ITS could have in the foreseeable future when in actual deployment stages. This is the reason why we set one of the thesis goals to create a co-simulation platform which can fulfil the aforementioned necessities. Besides, this goal is completely aligned with RQ1:

> Given the cooperative nature of C-ITS, is there a software platform which can accurately reproduce all the heterogeneous traffic environment components?
>
> **RQ1**

The coming sections will describe the inception of the platform while giving some insights regarding the designed components.

4.1.1 Design Goals

Given the complexity of the task, following standard software engineering practices, the first step was to pursue a requirements analysis which could guide us through all the subsequent software development stages.

Functional Requirements

The two, declared, mandatory functional requirements identified are the following:

- **Holistic approach to the traffic environment:** as thoroughly explained in the previous sections, the inclusion of the entire traffic road environment is a necessary condition to make realistic and reliable C-ITS testing. More specifically, including infrastructural actors such as actuators, monitoring, and traffic management systems is of primary importance; indeed, these components are often neglected during testing, especially regarding V2X-enabled applications. This is a definitive issue, as the role of the infrastructure ranges from highly beneficial (such as in long-range communication applications) to absolutely mandatory (in cases where the monitoring system is crucial, such as in Cooperative Perception (CP) applications.
- **Enable HiL applications:** the realism of the whole simulation depends on the employment of both hardware and software components, which are seamlessly transposed into real-testing in subsequent phases. This is why HIL testing is so important, and why it is equally important to enable it into the co-simulation platform.

Non-functional Requirements

The platform also has to comply with several non-functional requirements, which are briefly detailed below:

- **Scalability**: the framework needs to be scalable to allow for both small, scenario critical applications, and bigger, more complex scenarios such as highways or big urban contexts
- **Modularity**: we want to be able to enable or disable the various components at will, both for performance reasons and to vary the kind of tests depending on the final objective
- **Reproducibility**: the co-simulation platform must deliver deterministic results, net of some wanted randomness injected on purpose. This is very important, as evidences on the lack of absolute determinism have already been noticed, for example, on 3D engines [27]
- **Portability**: if possible, we want to be able to migrate the platform without major encumbrances, again to speed-up any specific test in given environments

High-Level Architecture

the depiction of the functional, and non-functional requirements leads us to the definition of an high-level architecture, depicted in figure 4.1. The identified high-level components are the following:

- **Simulation manager:** this component holds the full simulation stack, as well as all the involved third-party components. The simulation manager is capable of starting and stopping the simulation, as well as gathering useful log data from the latter for evaluation purposes. Plus, the simulation stack has to contain the traffic environment components in itself.

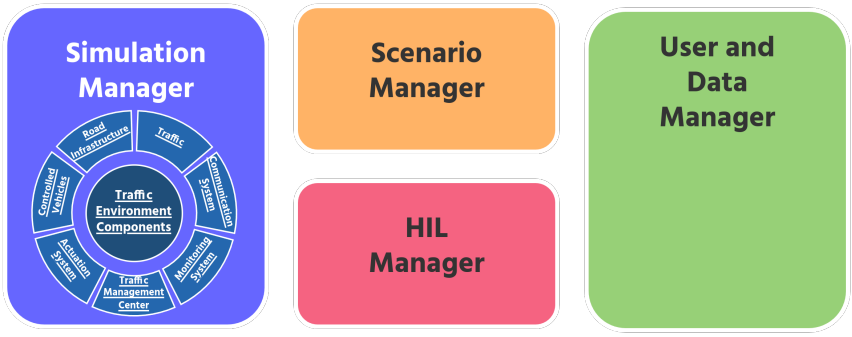


Figure 4.1: High-Level architecture for the co-simulation platform

- **Scenario manager:** no realistic simulation can be obtained without a scenario manager, whose job is to create and define the test scenarios, as well as creating a suitable file format, accessible to all the other platform components.
- **HiL manager:** given our requirement of enabling HiL applications, the software stack has to communicate in some way to selected hardware platforms, chosen for given testing purposes. Hence, this crucial communication role is offered, at high level, by the HiL manager.
- **User and Data manager:** this component, albeit not mandatory, needs to hold historical simulation data, as well as prepare a user access system for future purposes. Thus, this component will hold any user-specific data in case the platform becomes a multi-user one.

4.1.2 The role of Eclipse Mosaic

During section 2.2.4, we discussed how a notable and open-source co-simulation platform was found in Eclipse MOSAIC. This software has distinguished itself for several features, including modularity, portability and scalability, which are all part of our non-functional

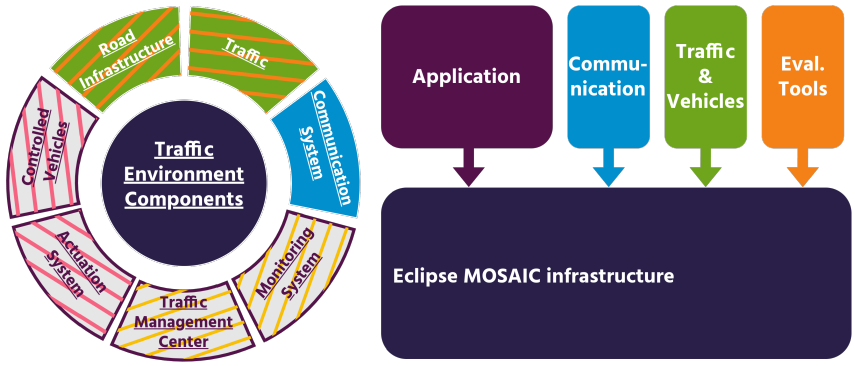


Figure 4.2: Side-by-side comparison between the already established Eclipse MOSAIC software and how its components are mapped to the traffic environments' one. In matching colours, the MOSAIC features which are able to replicate the given traffic component

requirements. Plus, MOSAIC already offered co-simulation between some established simulators, hence it already includes some parts of the traffic environment components in its design. For this reason, we decided to evaluate the usage of this tool as a platform foundation, conducting an informal gap analysis on the matter. Besides, it is not only beneficial to shorten development times, but also code reuse is an established and promoted practice in software engineering processes.

Gap analysis

We can use figure 4.2 to have a side-by-side comparison between the tools offered by eclipse MOSAIC and a possible mapping with respect to the traffic environment components wheel. For further clarity:

- Items in the traffic environment components wheel with the same colours of the Eclipse MOSAIC boxes are already reproduced inside the latter.
- Items with yellow stripes indicate that the component is in

part already found in Eclipse MOSAIC, but a further extension is mandatory to reach our goals.

- Items with red stripes indicate that not only the component does not exist, but also that its testing depends on the presence of dedicated hardware boards for realistic HIL testing .

The gap analysis gave us the following insights:

- The depiction of **Traffic**, in MOSAIC, is done through Eclipse SUMO, which coupled with the MOSAIC evaluation tools (indicated by the orange stripes in the figure) grants a realistic reproduction of the vehicles on the road, as well as the **Road Infrastructure**.
- Likewise, the **Communication System**, which happens through either the Simple Network Simulator (SNS) or one of the coupled wireless communication simulators, is already well represented, albeit without any HIL capabilities.
- When we deal with everything regarding applications development and deployment, the first thing to notice is that through the application simulator, MOSAIC offers the possibility to write such software for each road actor. This includes all kinds of vehicles as well as infrastructural actors such as sensors or detectors. However, the possibility to implement these capabilities does not mean that they are already available to the public. This is definitely true for both the **monitoring system** and the **traffic management center**. Indeed, all the actuation logic pertained to these components has to be written from scratch, as it is not part of the available MOSAIC package; this is also true for any of the vehicle-side applications, despite the fact that there are some examples regarding their usage.

- Regarding the **actuation system**, if we refer to the traffic policies, it leads to the application logic of the traffic management center, hence it is a component which can be developed, but not yet in place. However, if we also take into account the actuation logics of both conventional and connected vehicles, the problem becomes twofold, as not only there needs to be an actuation logic, but also a behavioural logic. This, in turn, takes us to the last point of discussion regarding **controlled vehicles**.

- Finally, the **controlled vehicles** need a more complex discussion. As we just said, their actuation logic has to be written within the MOSAIC application facility, and thus no extra extension is necessary. However, in order to give more realism to the simulation, a controlled vehicle as well as a connected and automated one could be controlled by an external entity, it being either a third-party software or an entire different hardware board. Plus, this could also be true for the actuation logic of infrastructure-side applications for realism purposes. This leads to a twofold problem: on one hand, a controlled vehicle could be managed through the application interface which communicates directly to SUMO, and thus using the car-following models found within the latter. On the other hand, it is a desirable feature to not only be able to write the actuation logic outside the application simulator, but also to have an entirely different Automated Driving System (ADS) completely submerged inside the simulation. This is a decisive missing feature of MOSAIC that has to be greatly taken into account.

When we compare Eclipse MOSAIC with respect to how the High-Level architecture was envisioned, we can see that:

- The high-level *Simulation Manager* and *Scenario Manager* are

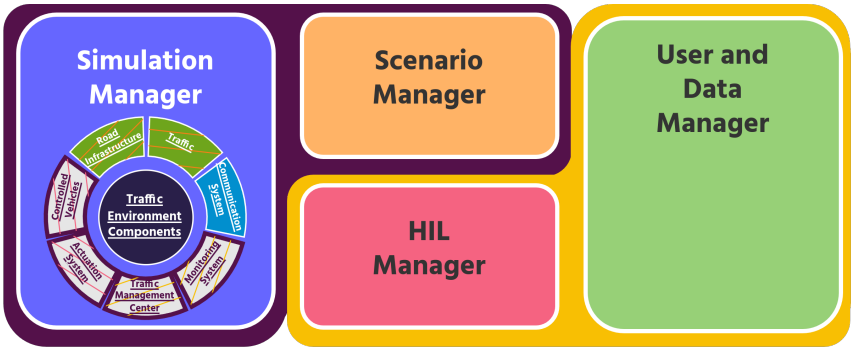


Figure 4.3: Depiction of how the High-Level architecture is impacted by the usage of Eclipse MOSAIC. In Purple, the parts which can be synthesized by MOSAIC alone; in yellow, the ones which are not included by the software

well established inside the software. Indeed, MOSAIC offers a complete set of tools to create new simulation scenario, decide how and which applications logics are found inside it and run the simulation while gathering data through logs.

- On the other hand, the *HIL manager* and the *User and Data manager* are completely missing from MOSAIC, as they were not part of the original goals envisioned by the creators.

This further analysis is depicted in figure 4.3.
After careful considerations, we decided that the strengths of the tools outweighed its weaknesses and lacking features, so we decided to extend the simulation tool instead of starting from scratch.

4.2 Enabling HIL testing

Given the assumptions stated in the previous section, the proposed extended co-simulation framework builds up on the MOSAIC existing platform, extending its functionalities and adding new functionalities, aimed at supporting HIL testing above all. The component diagram describing the architecture is shown in figure 4.4. As we can see, there are a couple of components built upon Eclipse MOSAIC, which are described below:

- The first notable component is the addition of a so-called **co-simulation backend**, which is the yellow rectangle in the component diagram. This component, written in Java, needs to wrap the simulation starting logic for user access purposes, as well as storing the data regarding the various simulation runs. Thus, this component realizes the missing *HIL manager* and *User and Data manager* depicted in the high-level architecture. Moreover, it holds all the unique names of the hardware boards which take part into the simulation. This management choice is done to ensure a much smoother flow of operations, since the actual hardware boards are never directly coupled with the MOSAIC infrastructure and real-time interface. Finally, this component exposes a simple API interface, which can be contacted with standard HTTP requests coming from any specialized testing or development tool.[1] These design choices highly keep modularity and portability in mind, while fulfilling the main goal of enabling HIL simulations.

- At this point, there needs to be a methodology to link the Eclipse MOSAIC platform and any of the possible hardware boards inside the simulation. Given the heterogeneity of the commercially available hardware boards, the choice fell on an

[1] such as Postman [112], Bruno [19] or Hoppscotch [14]

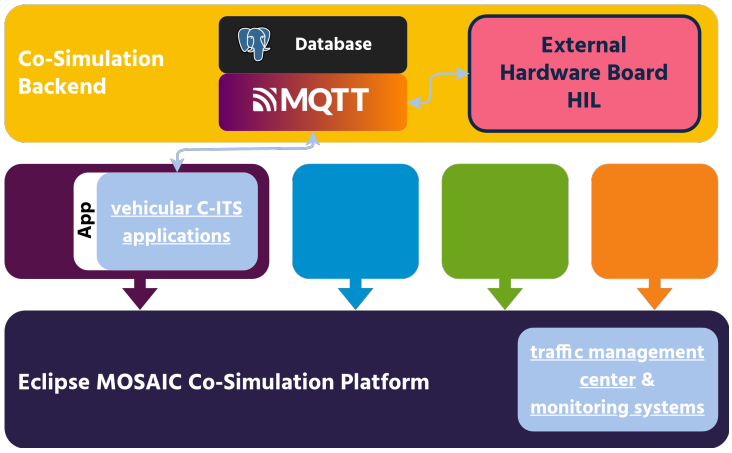


Figure 4.4: Component diagram of the developed co-simulation framework. In blue, green and orange, the communication, traffic, and evaluation modules respectively, which are already part of the Eclipse MOSAIC base package

MQTT Broker ready to listen to simulation messages and forward them to the actual hardware and vice versa. This choice is motivated by the extreme popularity of MQTT as a lightweight and scalable protocol, and thus available in a very wide range of devices, ranging from APIs in several high-level languages (Java included) to libraries for 16-bit microcontrollers. Finally, MQTT is also a standardized protocol, and possibly applicable even in more time-constrained applications, as it could be the case for HIL applications.

- Finally, the co-simulation backend is linked to a PostgreSQL database which is used to store both user, simulation, and hardware data, as previously described. Plus, every time a new simulation is started via HTTP request to the backend, the system automatically knows which boards are connected to the MQTT server and therefore can be coupled with one of the virtual vehicles.

4.3 Software extension

A non-negligible detail is the software additions made inside the Eclipse MOSAIC application logic to allow C-ITS testing. Indeed, to seamlessly include these services inside the software structure, a *mosaic-cits* library was created and put together with the other utility libraries that MOSAIC uses. This addition does not impact the non-functional requirements of the platform, and allows us to enrich the platform with more and more services depending on the use case. As there were different deployed services, their description is left to the matching sections in future chapters (see chapters 5 and 6). Moreover, the deployed C-ITS services are detailed in appendix B for further reading.

4.4 RQs mapping

Recalling chapter 3, the creation of the co-simulation platform was done with several goals in mind, which were synthesized by the detailed research questions. Having a look at these aforementioned questions, we can inspect them one by one and check if they match with the course of actions which led to the aforementioned co-simulation framework. Hence, we can see that the following RQs are answered here:

> Is there a way to test the impact of C-ITSs on the whole road environment?
>
> **RQ2**

The development of the *mosaic-cits* library effectively covers the missing, *"to be developed"* traffic environment components we identified during the gap analysis (see figure 4.2), more specifically the monitoring system and the traffic management center. This is because the library effectively mimics a virtual control room, which could take decisions given the received C-ITS messages. Depending on the developed use case, we could have different actuation logics, as we will see in chapter 5.

> How can we seamlessly enable the rapid prototyping of cooperative services within their intended environment?
>
> **RQ3**

As the application logic is mainly written in Java through MOSAIC, SIL testing is already granted by the latter. Moreover, our co-simulation extension enables several use cases where the application (or communication) logic is written directly on the deployed hardware board, enabling HIL testing as well. This feature greatly expands the rapid prototyping capabilities of the platform, ensur-

ing that any tested C-ITS service can be done with the appropriate, intended hardware which could be immediately deployed in its intended environment. Some of the developed use cases supporting this RQ are reported in chapters 5 and 7.

When C-ITS services are developed, are all the risks related to user security taken into account?

RQ4

The sole presence of a communication simulator is a good start to assess the matter, especially given that the MOSAIC platform can already be coupled with more sophisticated network simulators such as *ns3* or *Omnet++* which could heavily enhance the determinism and realism of the V2X interactions happening during C-ITS testing. Differently to other RQs, assessing the security risk as well as the platform suitability requires further platform validation, hence why some of these tests were included within the platform use cases, as we will see in chapter 6.

Are C-ITS tested in their intended environment, and with the full message exchange pipeline enabled?

RQ5

While Eclipse MOSAIC does not offer this feature out-of-the-box, the addition of the co-simulation platform allows coupling not only with external hardware, but also with external servers and/or application with which the communication packets, as well as the protocol (being MQTT), are agreed upon. Thus, the full message exchange pipeline could be enabled by further extensions, such as the one proposed in the use case detailed in section 6.4, offering PKI capabilities to the platform.

Chapter 5

Platform use cases: C-ITS applications

In the previous chapter, we detailed the design and development of the co-simulation platform envisioned for C-ITS testing. After that, it was employed for a series of use cases related to the study of the impacts of C-ITS services on the whole road environment. This is fully aligned with the research questions outlined in chapter 3, more specifically to the RQ2 and RQ3 discussed in section 4.4. Thus, the following section will explain in detail which were the identified use cases, and how they were tested thanks to the platform.
As a final note, each use case will be structured as follows: first, the motivations behind the use case. Secondly, the case study where it is applied, including the testing scenarios. Finally, the test results and any further conclusion.

5.1 UC1: Real-world communication facilities

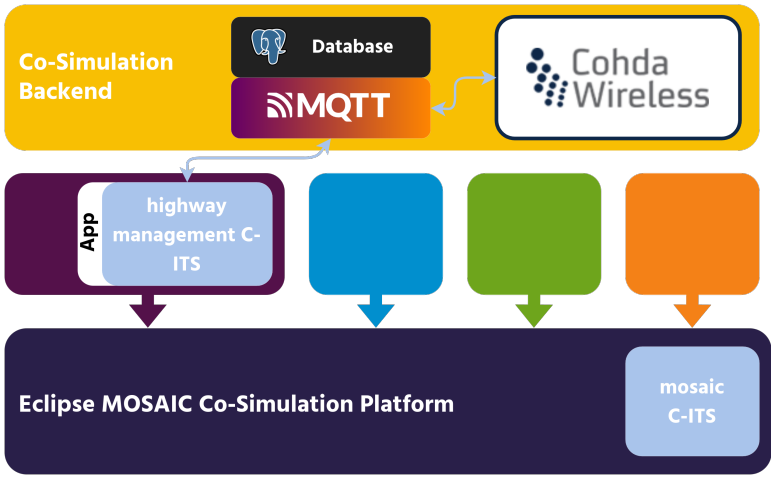
One of the two goals of the platform on the topic of HIL testing is allowing actual, real-world messages to be sent from the platform

itself through an actual, standard-compliant hardware board. This leads to two achievements: for starters, the HIL board is tested inside a realistic traffic environment, including geolocalization and surrounding vehicles, which is great to validate the board capabilities. Secondly, the ETSI communication stack is already implemented into the hardware board, in our case a commercial Cohda MK5 obu device.[1] In turn, this means that any external board could, in theory, receive the Cohda device messages and be tested in a way which is completely agnostic to the simulation. Finally, recording the Cohda messages enables message validation and inspection with respect to security standards or software replication. The architectural schema and test setup for this use case are shown in figure 5.1a.

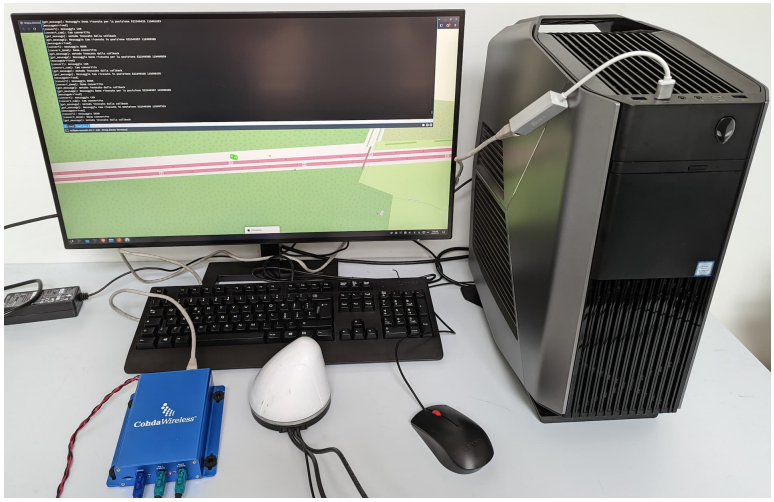
5.1.1 Case study

This case study includes the development of the Traffic Jam Ahead service, one of the day1 services as per the Car2Car consortium taxonomy [23]. This service includes various activation conditions indicating that a traffic jam is forming in a specific point of the road infrastructure. For the sake of simplicity, the activation conditions are described in table 5.1, while further details on this service are left to the reader in Appendix B, table B.1. It is worth noting that TRCO_3 is not implemented as the simulation does not include radio communications, while TRCO_5 is not implemented as the vehicles are assumed to be not equipped with on-board sensors. Finally, in the future references of this subsection, we will refer to the specific CAV equipped with the Cohda MK5 as the *Ego Vehicle*.

[1] see the MK5 OBU website [147] for further details



(a) Use Case 1 architecture overview



(b) HIL test setup and physical connections. In the bottom half of the picture, in blue, the Cohda MK5 OBU.

Figure 5.1: Use Case 1 architecture and physical setup

5.1.2 Testing Scenarios

Baseline Scenarios

The case study includes three *simple scenarios*, which were devised to prompt the activation of the service-specific conditions.

Within the simple scenarios, all vehicles are considered to be Connected Conventional Vehicless (CCVs), operated by human drivers and equipped with V2X communication devices sharing CAM and DENM messages exchange.

Simple Scenario 1, covering TRCO_0, consists of the Ego Vehicle traveling at low speed along an empty road stretch of the A2 Highway;

Simple Scenario 2, covering TRCO_1 AND TRCO_2, consists of a CCV and the Ego Vehicle traveling along an empty road stretch of the A2 Highway, as leader and follower. At some instant in time, the first vehicle stops for the necessary amount of time, thus activating the condition and sending the DENM message signalling the traffic jam ahead.

Simple Scenario 3, covering TRCO_1 AND TRCO_4, consists of five CCVs and the Ego Vehicle traveling along an empty road stretch of the A2 Highway. The five CCVs travel at a very low speed, in this case 5 m/s, which is lower than the service treshold and thus activates the stressed condition.

Count	Triggering condition
TRCO_0	The ego vehicle is moving with an average velocity of 30 km/h or less and more than 0 km/h over a period of 120s (stop & go traffic)
TRCO_1	The ego vehicle velocity is equal to 0 km/h for at least 30 s.
TRCO_2	At least one DENM corresponding to the *traffic jam - traffic jam ahead* C-ITS service with the same driving direction has been received.
TRCO_4	CAMs indicate a velocity of 30 km/h or less of at least five other vehicles within 100 m and with the same driving direction.

Table 5.1: Triggering Conditions of the Traffic Jam Ahead developed on the described use case

(a) Overview of the A56, highlighted in yellow.

(b) Detail of the Camaldoli entrance where traffic jam occurs. In yellow, the considered driving direction (from east to west).

Figure 5.2: A56 highway modelled in the realistic traffic scenario.

Realistic Traffic Scenario

The study area is the A56 Highway (Italy), also known as the Tangenziale di Napoli, connecting the SS7 near Arco Felice/Pozzuoli from the west to the Autostrada A1 to the east. The A56 is a three-lane highway, with a total length of 20.2 km, a maximum legal speed of 80 km/h, and 14 entries/exits. For our case study, we consider mainly the east-to-west direction from Autostrada A1 to the Arco Felice/Pozzuoli exit as depicted in Figure 5.2.

The traffic demand is synthetically generated. Since congestion is a requirement to verify the considered C-ITS service more thoroughly, we have generated a very high, unrealistic traffic demand so that stop-and-go phenomena occur in the proximity of one of the

highway entrances. More specifically, we assume a traffic demand of more than 5000 veh/h along the mainline and 1500 veh/h entering the highway at the Camaldoli entry.
The vehicle fleet consists of Conventional Vehicles (CVs) and CCVs. To verify the operation of the implemented C-ITS service under different mixed traffic conditions, a penetration rate of 5-10-20-30 % of CCVs with respect to the total traffic demand is considered.
Moreover, to increase the realism of the traffic scenario, slow and aggressive drivers are modelled. The microscopic parameters used for characterizing the behaviour of each type of driver, listed in Table 5.2, are the ones defined in [122] Here, the first five parameters are related to the general behaviour of the drivers, while the second four are related to the lane-changing model. Regarding these latter, *lcKeepRight* is the eagerness for following the obligation to keep right, *lcAssertive* is the likeliness to accept lower gaps for lane changes in dense traffic situations, *lcSpeedGain* is the likeliness to change lanes to gain speed and, finally, *lcCooperative* is the willingness for performing cooperative lane changing, Further details are shown in [45]. With specific regard to the desired speed followed by both types of drivers, it is distributed as a truncated normal distribution to represent a large spectrum of driving behaviour. Specifically, the

Parameter	**Slow Driver**	**Aggressive Driver**
Length	4.5 $[m]$	4.5 $[m]$
Headway	1.2 $[s]$	1.0 $[s]$
Max Acceleration	2.5 $[m/s^2]$	3.5 $[m/s^2]$
Max Deceleration	6.5 $[m/s^2]$	6.5 $[m/s^2]$
Speed Distribution	N(1, 0.1, 0.2, 2)	N(1, 0.1, 0.2, 2)
lcKeepRight	6	1.5
lcAssertive	1.8	3.0
lcSpeedGain	0.8	1.6
lcCooperative	1	0.8

Table 5.2: Drivers parameters

distribution is such that 95% of the vehicles drive between 80% and 120% of the legal speed limit. In our scenarios, we assume that 25% of drivers are aggressive and 75% are slow.

5.1.3 Test results

This Section reports the results of the case studies. Each time the simulation starts, the Cohda MK5 is connected via MQTT to a particular CCV (the Ego Vehicle), while the other CCVs are equipped with application OBUs alone.
It is worth noting that *baseline scenarios* were merely used for the functional verification and validation of the extended co-simulation framework. Moreover, each of them represents a simplified synthetic traffic condition that should trigger one of the service-specific activation conditions described in table 5.1. For this reason, the results have been omitted here as the positive outcome of these tests simply indicates the correct functioning of what has been implemented.

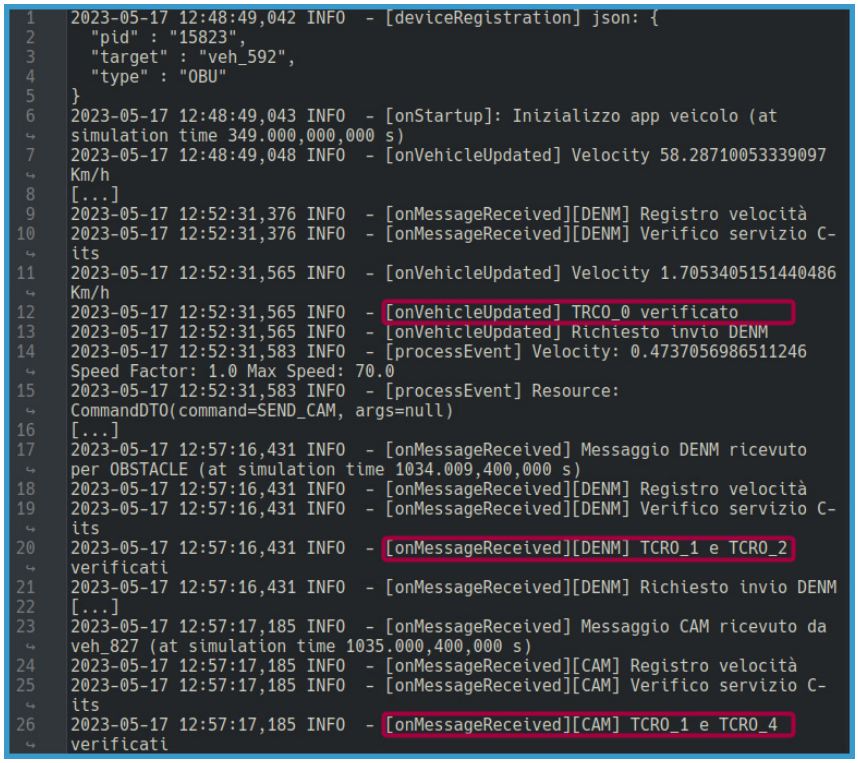
Realistic Traffic Scenario Results

This test is aimed at understanding if and how the service is activated in realistic traffic conditions. It is worth noting that in

Penetration Rate	Activated Condition	Activation Time (s)	Stressed Conditions
30%	TRCO_0	729	TRCO_0,1,2,4
20%	TRCO_0	745	TRCO_0,1,2,4
10%	TRCO_0	758	TRCO_0,1,2,4
5%	TRCO_0	762	TRCO_0,1,2

Table 5.3: Activation conditions observed during the test cases under different penetration rate of connected vehicles.

this scenario, the Ego Vehicle is inserted in the simulation at time 500s in order to encounter the congested traffic conditions near the Camaldoli entry.
Testing results, summarized in Table 5.3, provide information about the service-specific condition that activates the C-ITS service first, the activation time, and the possible other service-specific conditions observed, given a certain penetration rate and the traffic demand.
As expected, the TRCO_0 condition is always the activating condition for the service, as it covers the case of stop-and-go traffic conditions. Moreover, the lower the penetration rate, the longer before the service activates since the first CCV spawning in the simulation is found later into the traffic queue, thus showing approximately a 30s difference between the 30% and 5% penetration rate cases.
Given that we generated a high traffic demand to observe congested traffic conditions, the conditions for the TRCO_1 and TRCO_2/ TRCO_4 are correctly met and printed into the application log files. We provided a screenshot of the filtered log file for one of the CCVs in Table 5.3 for further clarity. We also noticed that the TRCO_1 condition is met more or less at the same simulation time, which was 1035s for the 30%, 20% and 10% penetration rates and 1060s for the 5% one. The first time this condition is met, it is in conjunction with TRCO_2, which is again correct as the CCVs in the area will communicate the traffic jam via DENM messages. Shortly after, TRCO_4 is activated as CAM messages are also received with slow speed indications for both the 30% and 20% penetration rates, while the 10% rate activated such condition about 60s later than the first two, which is reasonable considering that such a low penetration rate makes finding 5 other CCVs within the 100 m range more difficult to happen. Finally, the 5% penetration rate test did not show any TRCO_4 condition activation, as the CCV penetra-



```
2023-05-17 12:48:49,042 INFO  - [deviceRegistration] json: {
  "pid" : "15823",
  "target" : "veh_592",
  "type" : "OBU"
}
2023-05-17 12:48:49,043 INFO  - [onStartup]: Inizializzo app veicolo (at
simulation time 349.000,000,000 s)
2023-05-17 12:48:49,048 INFO  - [onVehicleUpdated] Velocity 58.28710053339097
Km/h
[...]
2023-05-17 12:52:31,376 INFO  - [onMessageReceived][DENM] Registro velocità
2023-05-17 12:52:31,376 INFO  - [onMessageReceived][DENM] Verifico servizio C-
its
2023-05-17 12:52:31,565 INFO  - [onVehicleUpdated] Velocity 1.7053405151440486
Km/h
2023-05-17 12:52:31,565 INFO  - [onVehicleUpdated] TRCO_0 verificato
2023-05-17 12:52:31,565 INFO  - [onVehicleUpdated] Richiesto invio DENM
2023-05-17 12:52:31,583 INFO  - [processEvent] Velocity: 0.4737056986511246
Speed Factor: 1.0 Max Speed: 70.0
2023-05-17 12:52:31,583 INFO  - [processEvent] Resource:
CommandDTO(command=SEND_CAM, args=null)
[...]
2023-05-17 12:57:16,431 INFO  - [onMessageReceived] Messaggio DENM ricevuto
per OBSTACLE (at simulation time 1034.009,400,000 s)
2023-05-17 12:57:16,431 INFO  - [onMessageReceived][DENM] Registro velocità
2023-05-17 12:57:16,431 INFO  - [onMessageReceived][DENM] Verifico servizio C-
its
2023-05-17 12:57:16,431 INFO  - [onMessageReceived][DENM] TCRO_1 e TCRO_2
verificati
2023-05-17 12:57:16,431 INFO  - [onMessageReceived][DENM] Richiesto invio DENM
[...]
2023-05-17 12:57:17,185 INFO  - [onMessageReceived] Messaggio CAM ricevuto da
veh_827 (at simulation time 1035.000,400,000 s)
2023-05-17 12:57:17,185 INFO  - [onMessageReceived][CAM] Registro velocità
2023-05-17 12:57:17,185 INFO  - [onMessageReceived][CAM] Verifico servizio C-
its
2023-05-17 12:57:17,185 INFO  - [onMessageReceived][CAM] TCRO_1 e TCRO_4
verificati
```

Figure 5.3: Filtered log file for a CCV achieving the traffic jam ahead service conditions

tion rate was so shallow that this condition could not be fulfilled.

5.1.4 Conclusions

This use cased showed how the integration of the Cohda MK5 OBU commercial hardware, which is provided with a real V2X hardware/-software stack, inside the co-simulation framework allowed to test a complex and impactful C-ITS service such as the Traffic Jam Ahead One. For the evaluation phase, we created three simple scenarios aimed at verifying the correctness of the extended architecture, as well as the proper functioning of the implemented C-ITS service.

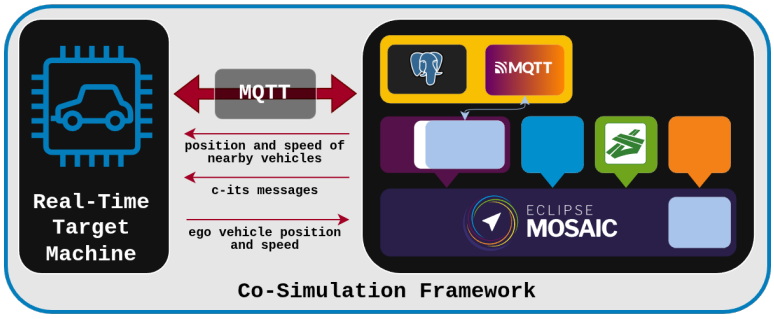
Moreover, a detailed scenario was formulated to conduct a thorough verification of both the robustness of the architecture and the functionality of the service under realistic traffic mixed-traffic conditions. Overall, we deemed our test successful to assess both the co-simulation framework capabilities and the HIL correct functioning.

5.2 UC2: HiL algorithm testing

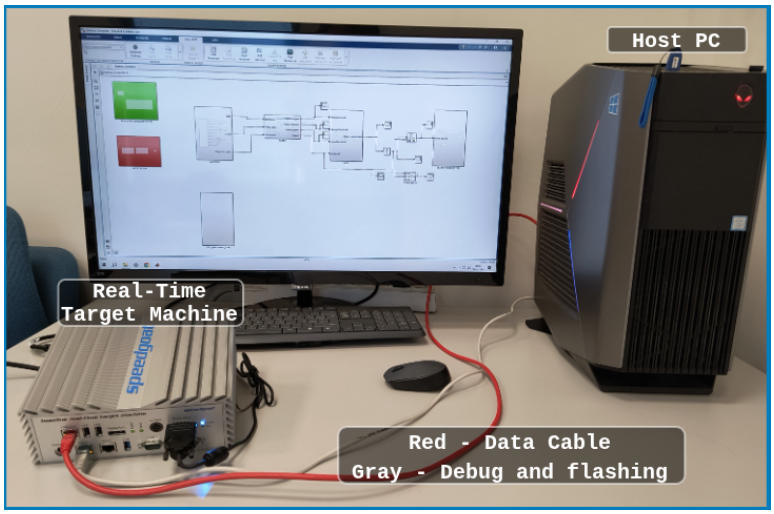
This second use case focuses on the evaluation of ADS algorithms inside the co-simulation framework, again deploying HIL strategies. More specifically, the hardware target machine gets nearby vehicle data such as their speed and location as input, as well as any C-ITS message sent to the ego vehicle corresponding clone in the simulation framework. The node then elaborates such information pieces, sending back to the application the updated Ego-Vehicle position and speed. By doing so, we allow the ego vehicle to be tested inside a realistic scenario, thus avoiding too simplistic validation scenarios. An overview of this extension, including an image of the hardware setup, is shown in figure 5.4a.

5.2.1 Real-Time Target Hardware

The real-time target hardware allows for a cost-effective testing and validation of an ADS without having the complete system hardware. In this way, it is possible to rely on a real-time simulator that acts as a digital twin of the whole system or parts of the system. Here, we used a Speedgoat baseline as target hardware, and Matlab/Simulink as development environment for the different components.
As shown in Figure 5.5, the real-time target hardware consists of different components (or subsystems): the *Filter*, which collects, organizes and eventually merge traffic data; the *ADS* subsystem, containing the set of automated driving systems that control the motion of the Ego-Vehicle; and finally, the *Vehicle Dynamics* subsystem, containing the model that emulated the motion of a real vehicle. Moreover, there is a pair of MQTT client blocks which manage data exchange between the connected MQTT server (host PC) and the application running on the real-time target computer. Precisely, the Speedgoat receives messages from the MQTT server about traffic data. Similarly, as the ADS outputs the control strat-



(a) Extended Co-Simulation Framework overview



(b) HIL test setup and physical connections. In the bottom-left, the Speedgoat Baseline real-time target machine.

Figure 5.4: Use Case 2 architecture and physical setup

egy after further elaboration via the vehicle dynamics, the Ego-Vehicle speed and position data is sent via MQTT back to the server. Finally, the Ego vehicle MOSAIC application is also connected to the latter, closing the communication loop between the HIL and the co-simulation framework.

Filter

The data filter is responsible for data collection and organization, scraping unnecessary pieces of information and eventually fusing different sources together (e.g. vehicle ahead speed and road speed limit) to properly feed the ADS subsystem. It is thus a mandatory component for safe navigation and decision-making. For example, if the vehicle is equipped with multiple ADSs, the filter could send to each of them the information strictly necessary for their operation. In our application case, as our system is mainly composed of an ACC, as described below, the filtered data consists of: speed of an ahead vehicle and the relative distance $\Delta d(t)\,[m]$ to the Ego-Vehicle; desired target speed $v_{des}(t)\,[m/s]$, which corresponds to the legal road speed limit.

Automated Driving System

The longitudinal motion of the Ego-Vehicle is regulated by an Adaptive Cruise Control (ACC) system that adapts the speed in response to upstream traffic conditions. The ACC control algorithm implemented is the one proposed by [100], computing a desired longitudinal acceleration $u(t)$ as follows:

$$u(t) = \begin{cases} min\{K_v e_v(t); [K_p e_d(t) + K_w \Delta v(t)]\} & \text{if } e_d \leq 0, \\ K_w \Delta v(t) - K_p e_d(t) & \text{if } e_d > 0, \end{cases} \tag{5.1}$$

where $e_v(t) = v_{des}(t) - v(t)$ and $e_d(t) = d_s(t) - \Delta d(t)$; $K_v = 0.5$, $K_p = 0.2$ and $K_w = 0.4$ are the control gains; $d_s(t) = d_{st} + h_g \cdot v(t)$ is the safety inter-vehicle distance in $[m]$, being $d_{st}[m]$ and $h_g[s]$ the standstill distance and the minimum time gap, respectively. The ACC has two operating modes:

- if $\Delta d(t) \geq d_s(t)$, the control goal is to track the desired speed $v_{des}(t)$ (*speed control operating mode*);

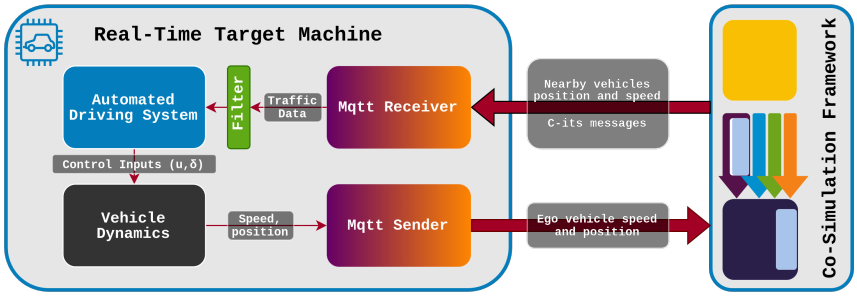


Figure 5.5: Detail on the Real-Time Target machine

- if $\Delta d(t) < d_s(t)$, the control goal is to maintain the safe distance $d_s(t)$ (*spacing control operating mode*).

It is important to note that, since we consider a solely longitudinal controller, we assume a control steering angle $\delta = 0$, thus delegating the control of the vehicle's lateral motion to SUMO.

Vehicle Dynamics

The vehicle dynamics is emulated via a rigid single-axle 3 Degree-of-Freedom (DOF) vehicle body model, considering the motion in the longitudinal, lateral and yaw directions. The model accounts for body mass, aerodynamic drag, and weight distribution between the axles due to acceleration and steering. The specific equations underlying the model are omitted here for space reasons, but more details can be found in [66, 99]. Ego-Vehicle parameters are summarized in Table 5.4.

5.2.2 Case study

Again, the case study includes the verification of the correct functioning of the ADS control system in a realistic traffic scenario. Moreover, on the software-side, the In-Vehicle Signage (IVS) C-ITS service is implemented, feeding the ADS with information regarding

Figure 5.6: Detail of the road segment of the Asse Mediano motorway used as a testing scenario. In green, the considered driving direction (east to west)

not only the preceding vehicle, but also the possible variation in the road speed limit. Clearly, the usage of the IVS service also underlies the usage of a Variable Speed Limit (VSL) service, as this information is directly sent to the human (or automated driver). Further details on these day 1 C-ITS services are shown in appendix B.

5.2.3 Testing Scenarios

Simple Scenario

The *simple scenario* is characterized by the presence of only the Ego-Vehicle proceeding along a straight road, where RSUs are positioned every $500\,m$. The legal speed limit is assumed to be $90\,[Km/h]$. At second 30, the speed limit changes to $70\,[Km/h]$ for 30 seconds.

Param.	**Value**	**Param.**	**Value**
I_{zz}	$4000\,[kg \cdot m^2]$	a	$1.25\,[m]$
b	$1.35\,[m]$	C_{yf}	$12000\,[N/rad]$
C_{yr}	$11000\,[N/rad]$	μ_f	1
μ_r	1	F_{znom}	$5000\,[N]$
h	$0.35\,[m]$	R	$28.96[Jkg^{-1}K^{-1}]$
T	$273\,[K]$	P_{abs}	$101325\,[Pa]$
g	$9.81\,[m/s^2]$	ρ_{air}	$1.21\,[kg/m^3]$

Table 5.4: Ego-Vehicle parameters.

After that, the limit goes back to 90 $[Km/h]$. The test is mainly used to understand the correct functioning of the architecture in the first place, and then whether the vehicle receives and processes the data received from the IVS correctly.

Realistic Scenario

The realistic scenario considers as a study area the road segment of the *Strada statale 162 NC Asse Mediano*, a motorway connecting all the suburbs in the northern area of the city of Naples, with a total length of 34 $[Km]$ and a legal speed varying from 50 $[Km/h]$ up to 70 $[Km/h]$ depending on the line-of-sight and road conditions. It is made up of two carriageways separated by a traffic island, with each carriageway consisting of two lanes. For our case study, we considered the Ego-Vehicle travelling from the Lago Patria entry to the Qualiano exit, for a total length of about 10 $[Km]$ (Figure 5.6). This section of the Asse Mediano has a legal speed limit of 50 $[Km/h]$. For the purpose of the experiment, virtual RSUs are placed along the road 500 meters away from each other, assuming a communication radius of about 300 $[m]$. Moreover, we suppose that after a certain point the speed limit goes up at 60 $[Km/h]$.

The traffic demand is synthetically generated, and consist of a constant flow of 2000 $[veh/h]$ moving in the same direction of

Parameter	**Driver**	**Parameter**	**Driver**
Length	4.5 $[m]$	lcKeepRight	6
Headway	1.2 $[s]$	lcAssertive	1.8
Max Acceleration	2.5 $[m/s^2]$	lcSpeedGain	0.8
Max Deceleration	6.5 $[m/s^2]$	lcCooperative	1
Speed Distribution	N(1, 0.1, 0.2, 2)		

Table 5.5: Drivers parameters

.

the Ego-Vehicle. All the vehicles are assumed to be equipped with OBUs to share C-ITS messages, but manually driven. The drivers are modelled after the microscopic parameters described in Table 5.5, which are defined in [122]. Note that the first five parameters refer to general driver behaviour, while the latter four describe the lane-changing model. The driver distribution is designed so that the vehicles drive between 80% and 120% of the legal speed limit. The traffic demand does not resemble actual traffic conditions within the area, however it is used to stimulate the IVS service. The idea is that, due to unforeseen events linked to portions of the road external to the simulation test bed, a speed limit change is advised in order to make up for this event (e.g. the presence of a stopped vehicle or road works). With the vehicles having a reduced speed limit, the ACC has to deal with both surrounding vehicles and the new speed limit. Thus, the scenario is characterized as follows:

- At the start of the simulation, no IVS has been sent to the vehicles, thus the speed limit is $60\,[Km/h]$ as established per the actual road structure. The Ego-Vehicle gets the vehicle ahead relative distance and speed, returning a speed profile to the simulation. However, during the first 15 seconds of simulation, a slower vehicle is placed ahead of the Ego one running at $50\,[Km/h]$ and possibly slowing down the Ego-Vehicle

- After the 15s mark, the slower vehicle moves away allowing the Ego-Vehicle to increase its speed.

- At simulation time 26s, the speed limit is reduced to $50\,[Km/h]$ for safety reasons. This information is sent by RSU to the connected vehicles via IVS. The ACC of the Ego-Vehicle, taking this into account, uses this data to change the desired speed.

- At simulation time 130s, the speed limit is further lowered

at 30 $[Km/h]$, in order to take into account a broader road hazard which prompts a major speed limit reduction. The Ego-vehicle correctly slows down.

- Finally, the speed limit goes back to 60 $[Km/h]$ at simulation time 148s back to the standard speed limit.

Note that, in this example, the Ego-Vehicle adapts its motion as soon as the IVS is received. This scenario makes use of the Mosaic perception module [114], which actually enables the use of the ACC by perceiving the relative distance and speed of the surrounding vehicles.

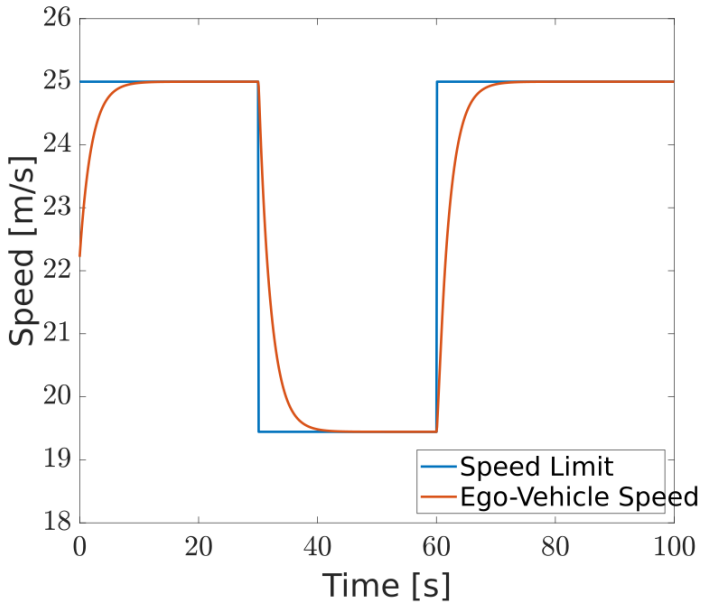
5.2.4 Test results

5.2.5 Simple Scenario Results

The result of the *simple scenario*, depicted in Figure 5.7 is straightforward, but is reported for completeness as it represents the preliminary test to then stress the envisioned architecture. The speed profiles in Figure 5.7a shows how the ACC reacts and modifies the Ego-Vehicle speed as the speed limit changes according to the IVS. We provided a screenshot of the filtered log file when the Ego-Vehicle in 5.7b for further clarity.

5.2.6 Realistic Traffic Scenario Results

The results of the *realistic scenario* are reported in Figure 5.8. Similarly to the simple scenario test, it reports the speed profile of the Ego-Vehicle, both the real one and the virtual clone, versus the speed limit imposed by the connected road infrastructure. As we can see, the Ego-Vehicle is impaired by the slower vehicle in the first 15 seconds of the simulation, having a constant speed of 50 $[Km/h]$. As soon as it moves away, the ACC system tries to accelerate the vehicle up until the 60 $[Km/h]$ target. However, the IVS message



(a)

```
INFO — [onVehicleUpdated] current speed: 25 m/s (at simulation time
    ↪ 29.800 s)
INFO — [messageArrived] message arrived from topic: simulation/
    ↪ idHardware/1047m4sp91f5dr392c51f1c709137116
INFO — [processHardwareMessage] received: {
  "messageType" : "SPEED_ACTUATION",
  "body" : {
    "controlSpeed" : 19.444444
  }
} (at simulation time 30.100,000,000 s)
INFO — [processHardwareMessage] set speed after receiving
    ↪ control_speed: 19.444444 m/s (at simulation time
    ↪ 30.300,000,000 s)
INFO — [onVehicleUpdated] current speed: 19,435629 m/s (at simulation
    ↪  time 45.300 s)
```

(b)

Figure 5.7: *Simple Scenario* results. **a**) Ego-Vehicle speed versus speed limit; **b**) Filtered log of simulation: the Ego-Vehicle receives the IVS message from the RSU.

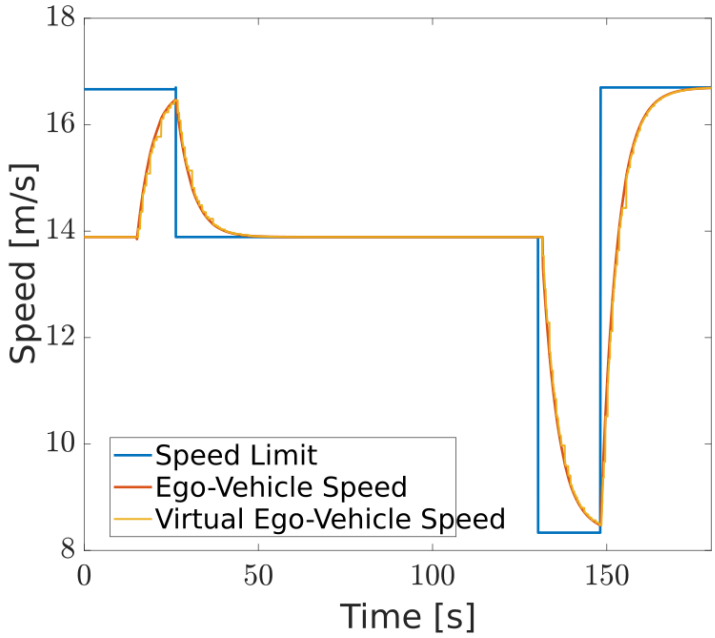


Figure 5.8: *Realistic Scenario* results: Ego-Vehicle speed (real and virtual) versus speed limit.

sent at the 26s mark signalling a new speed limit changes its behaviour, slowing the vehicle again. The same thing happens both when the limit goes down to 30 $[Km/h]$ and back to 60 $[Km/h]$. The filtered log file found in Figure 5.9 further clarifies these interactions, indicating the timestamps and the Ego-Vehicle control actions.

5.2.7 Conclusions

In this use case, we tried to couple the co-simulation framework with a real-time target machine (a Speedgoat baseline) equipped with an ADS, effectively creating a *controlled vehicle* inside the scenario. In addition, we also developed and implemented the IVS C-ITS service following related standard. To test this, two simu-

```
INFO — [onVehicleUpdated] vehicle ahead id: veh_0 edgeId: 188334050
    ↪ _1223336706_18120285 laneIndex: 0
INFO — [onVehicleUpdated] vehicle ahead speed: 13,88881368718734 distance
    ↪ : 81.15356736235637 airDistance: 81.4913422941845 (at simulation
    ↪ time 12.700,000,000 s)
INFO — [onVehicleUpdated] current speed: 13,8889 m/s (at simulation time
    ↪ 12.700,000,000 s)
INFO — [onVehicleUpdated] current speed: 15,7903 m/s (at simulation time
    ↪ 25.000,000,000 s)
INFO — [messageArrived] message arrived from topic: simulation/idHardware
    ↪ /1047m4sp91f5dr392c51f1c709137116
INFO — [processHardwareMessage] received: {
  "messageType" : "SPEED_ACTUATION",
  "body" : {
    "controlSpeed" : 13,888888
  }
} (at simulation time 26.000,000,000 s)
INFO — [processHardwareMessage] set speed after receiving control_speed:
    ↪ 13,888888 m/s (at simulation time 26.100,000,000 s)
INFO — [onVehicleUpdated] current speed: 13,889 m/s (at simulation time
    ↪ 70.500,000,000 s)
INFO — [messageArrived] message arrived from topic: simulation/idHardware
    ↪ /1047m4sp91f5dr392c51f1c709137116
INFO — [processHardwareMessage] received: {
  "messageType" : "SPEED_ACTUATION",
  "body" : {
    "controlSpeed" : 8,333333
  }
} (at simulation time 130.000,000,000 s)
INFO — [processHardwareMessage] set speed after receiving control_speed:
    ↪ 8,333333 m/s (at simulation time 130.100,000,000 s)
INFO — [onVehicleUpdated] current speed: 8,984215 m/s (at simulation time
    ↪ 145.600,000,000 s)
INFO — [onVehicleUpdated] current speed: 16,666666 m/s (at simulation
    ↪ time 178.300,000,000 s)
```

Figure 5.9: *Realistic Scenario* Results: Filtered log of simulation, where the Ego-Vehicle receives the IVS message from the RSU.

lation scenarios have been considered to verify the correctness of the co-simulation framework, as well as the proper functioning of both the implemented C-ITS service and the ACC. Importantly, the devised realistic scenario proved successful to assess both the robustness of the architecture and the functionality of the C-ITS service and ADS system under realistic traffic conditions.

5.3 UC3: C-ITS vs conventional detectors

This third use case leverages the software extension of the co-simulation platform to try to answer to a simple, yet interesting dilemma: can the usage of C-ITS and V2X replace conventional traffic monitoring facilities? Indeed, it is well known that currently traffic disruption are mainly signalled through traffic detectors (magnetic coils), cameras supervised by road operators or user alerts. However, the future dissemination of connected vehicles will automatically offer some categories of data, especially regarding vehicle direction, position and speed (the so-called *Floating Car Data (FCD)*. So, what happens when C-ITS services such as the Traffic Jam Ahead one are enabled? Are vehicles alone enough to measure a traffic jam condition?
To answer this question, we conducted a thorough analysis using the co-simulation framework. This included the calibration of a *Fundamental Diagram (FD)* on a custom simulation scenario and a further discussion regarding both the results and the pros and cons of this solution.

5.3.1 Case Study

Simulation Scenario

The road network consists of a three-lane straight, flat road with a lane-drop bottleneck where the number of lanes is decreased to two. This allows us to emulate a bottleneck activation, which may be caused by roadworks or an incident, among other reasons. In the literature, this kind of simulation scenario has been shown to exhibit representative characteristics of an incident in SUMO [67]. The three-lane segment is 7.5 km long, while the two-lane one is 1.5 km; this results in a total length of 9.0 km. In Eclipse SUMO, the bottleneck is modelled as a zipper where both the middle and left lanes must be used until the merging point and vehicles interleave.

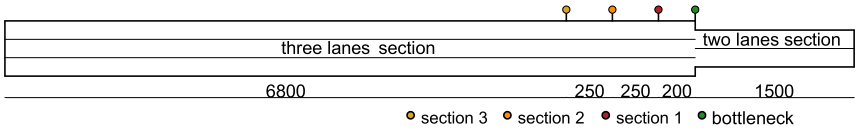


Figure 5.10: Road stretch layout, also indicating the location of induction loops.

The maximum allowed speed on the road is assumed to be 120 km/h. These characteristics ensure that the segment qualifies as a non-urban road, satisfying the context requirements for the *Traffic Jam Ahead* service.

Three cross-section detectors are located along the road at approximately 6800, 7050, and 7300 meters downstream of the starting point. In each counting section, one detector per lane is considered. Each cross-section detector collects data about passing vehicles and their individual speeds; the data are aggregated using three different time intervals (1 min, 3 min, and 5 min). Figure 5.10 shows an illustration of the simulated road stretch.

The traffic demand is synthetically generated as shown in Figure 5.11, and the simulation lasts for 175 minutes. The input flow is held constant at $1500\,[veh/h]$ for 10 minutes, then it is increased to $4500\,[veh/h]$ for 25 minutes to activate the bottleneck. The flow is then decreased to $1500\,[veh/h]$ for 25 minutes and further down to $1260\,[veh/h]$ for 10 minutes. The flow increases again to $3420\,[veh/h]$ for 25 minutes to activate another bottleneck, and is subsequently decreased to $1260\,[veh/h]$ for the congestion to clear. Finally, the flow increases once more to $2340\,[veh/h]$ for 15 minutes, and then it drops to $720\,[veh/h]$. This pattern creates three distinct peaks in demand, stressing the road network under both congested and uncongested conditions and including multiple phases of congestion formation and dissolution.

The fidelity of the simulation heavily relies on a realistic parametrization of individual driver behaviours. For this study, all microscopic

parameters were drawn from the comprehensive calibration and validation work presented in Berrazouane et al. [12], which established these values against real-world traffic data. Fundamental vehicle characteristics were standardized, with all vehicles having a length of $4.5\,[m]$ and being capable of a maximum acceleration of $2.78\,[m/s^2]$ and a maximum deceleration of $-7.43\,[m/s^2]$. Longitudinal movement (car-following) is governed by the widely-used Krauss model [17], coupled with a Constant-Time-Gap (CTG) spacing policy set at a time gap of $1.2\,[s]$ and a standstill distance of $2.33\,[m]$. To avoid an overly uniform traffic stream and better replicate real-world conditions, driver heterogeneity was introduced. Desired speed factors were sampled from a normal distribution with a mean of 1.193 and a standard deviation of 0.091, allowing for natural variations in preferred speeds. Driver imperfection, which accounts for minor deviations in following behaviour, was set with the parameter $\sigma = 0.292$. Finally, and crucial for modelling the bottleneck dynamics, lateral movements are characterized by the lane-changing model parameters: *lcAssertive* (1.616), *lcSpeedGain* (0.887), and *lcKeepRight* (0.835). The adoption of these specific values is explicitly justified by their validation in the work of Berrazouane et al. [12]. In particular, the *lcAssertive* value captures the more aggressive merging behaviour and acceptance of smaller longitudinal gaps typically observed as drivers approach a congested lane-drop bottleneck, a behaviour that is essential for realistically simulating queue formation.

FD Calibration

The calibration of the Fundamental Diagram (FD) ensures that the model reflects real-world traffic behaviour, improving the accuracy of traffic management and prediction systems. For calibration purposes, we perform 100 simulations with different random seeds, randomly sampled from the set $[1, 1000000]$. Flow and speed data are

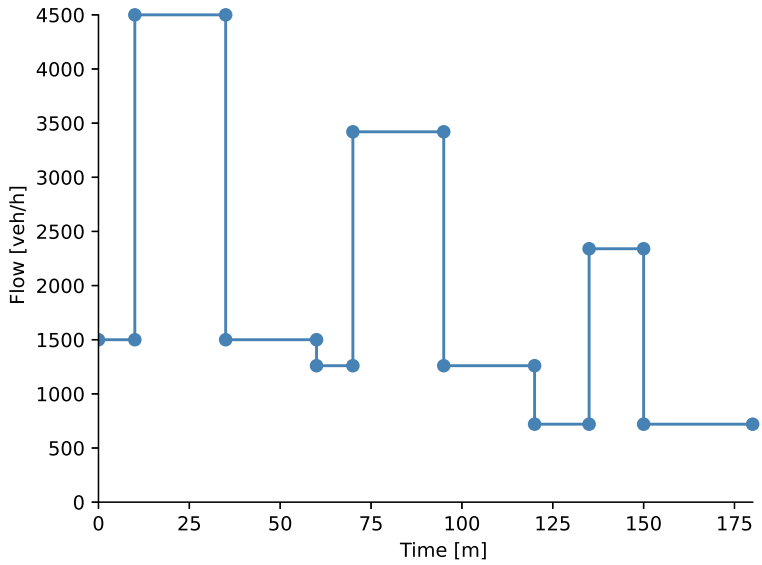


Figure 5.11: Traffic flow profile for the considered scenario.

collected using cross-section detectors, considering the three aggregation periods $1min$, $3min$ and $5min$. The density is then estimated using the fundamental equation of the traffic flow being $q = k \times v$. The chosen traffic flow model is the Newell Model, whose equation

Sec.	**Agg.** $[min]$	v_0 $[km/h]$	k_j $[veh/km]$	λ $[1/s]$	$\mathbf{R^2}$	**Adj.** $\mathbf{R^2}$	**RMSE**	q_m $[veh/h]$	k_c $[veh/km]$	v_c $[km/h]$
1	1	136.82	150.00	0.85	0.99	0.99	3.79	1851	34.24	54.97
2	1	136.81	144.17	1.02	0.98	0.98	3.63	2115	36.16	58.49
3	1	136.75	90.96	1.59	0.98	0.98	2.75	2500	32.56	76.79
1	3	136.69	150.00	0.78	0.99	0.99	3.52	1762	33.48	52.63
2	3	136.88	143.30	1.01	0.96	0.96	3.80	1916	34.98	54.78
3	3	136.60	76.62	1.72	0.96	0.96	2.58	2237	30.58	73.15
1	5	136.74	150.00	0.77	0.99	0.99	3.45	1723	32.74	52.64
2	5	136.95	141.03	1.03	0.95	0.95	3.60	1834	33.86	54.17
3	5	136.54	58.02	1.87	0.94	0.94	2.53	2033	27.26	74.59

Table 5.6: Calibrated Parameters of the Newell Model for each set of cross-section detectors and aggregation period.

of the macroscopic model is the following [104]:

$$v(t) = V_f \left(1 - e^{-\frac{\lambda}{V_f}\left(\frac{1}{k} - \frac{1}{k_j}\right)}\right)$$

where λ (in s^{-1}) is the slope of the speed-spacing curve. The Newell model was chosen for this study because, despite its parametric simplicity, it has been shown to accurately describe vehicle behaviour specifically in congested conditions and during queue discharge processes, which are central to the formation and dissipation of traffic jams [88, 2].
The model calibration is done via Least Squares Regression: first, we fit the observed data to the model's mathematical form using the least squares to minimize the difference between observed and predicted values. Then, a Goodness-of-Fit (GoF) metric is used to assess the accuracy of the model. Here we use Root Mean Squared Error (RMSE) as GoF metric, but for the sake of completeness we report also the R-Square and the Adjusted R-Square. The results of the calibration are summarized and listed in Table 5.6. In the same table, the last three columns also report the critical values q_m, k_c and v_c that can be deduced after the calibration.
As expected, the results of the calibration (and of the critical values of the quantities) depend not only on the distance to the bottleneck, but also on the aggregation period. Since the variation in the results are not negligible, but significant instead, this shows that the first condition of uncertainty regarding the FD estimation is actually represented by the choice of the location of the sensors and the aggregation period.

Simulation Analysis Setup

A real traffic environment is stochastic. Thus, in order to obtain reliable results, it is crucial to obtain as much realism as possible

in a simulation scenario. To this end, we perform 1000 simulations for each penetration rate by using random seeds, resulting in a truly random behaviour. Different seeds will result, for instance, in randomness when loading vehicles (e.g., speed deviation) and probabilistic insertion of vehicles as well as different vehicles being equipped with the vehicular application and messaging capabilities. To ensure statistical independence between runs, the 1000 seeds for the analysis were sampled uniformly at random and without replacement from the integer set $[1, 1000000]$. It is worth noting that the seeds used for the analysis are different from the seeds used for the calibration of the FD.

To verify the behaviour of the implemented C-ITS service under different mixed traffic conditions, a penetration rate of 0-5-10-20-30-40 % of connected vehicles (equipped with the C-ITS service) within the total traffic demand is considered. The test case with no connected vehicles overlaps with the analysis conducted exclusively using the traffic detectors.

For each penetration rate, each simulation and each CCV, data is gathered about the (first) time instant of activation of the C-ITS service (if any), as well as the latitude and longitude positions at the time of activation and the condition of the C-ITS service activated. Through data analysis, a log filtering step was deemed necessary for various reasons: first, the latitudinal position is neglected since the road is a straight segment and only the longitude varies, and no assumptions on the various lanes are inferred. Secondly, as different simulation yielded similar results in both position and time of activation, an aggregation step helped us to frame the occurrence of the C-ITS service through the various penetration rates. Finally, for each non-zero penetration rate only the TRCO_0 condition was activated; this was validated by a thorough SUMO scenario inspection as the bottleneck only produced slow vehicles, but crucially no completely stationary ones, hence never validating the TRCO_1

condition. As for the simulation with no CCVs, through the offline evaluation of the traffic flow, we gathered data about all the three aggregation times in which the traffic flow had surpassed the critical density threshold.

5.3.2 Results

By inspecting the gathered results, we found that there are two distinct time windows during which a traffic queue is formed, consistent with the traffic flow profile plotted in Figure 5.11: the first wave causing the bottleneck can be observed from minute 10 to 40, and the second one from minute 70 to 100. For this reason, we separate our analysis accordingly into an *Hour 1* and an *Hour 2* period. Table 5.7 reports the percentage of simulation runs (out of 1000) in which a congested traffic state is detected for each combination of aggregation period and penetration rate. The first three blocks of rows correspond to the detector-based method (with threshold $q \geq q_c$), while the last block corresponds to the C-ITS *Traffic Jam Ahead* service.

Test Case	Hour 1	Hour 2	Test Case	Hour 1	Hour 2
Sec-1-1min	100%	61,7%	Sec-2-1min	100%	0%
Sec-1-3min	100%	28,7%	Sec-2-3min	100%	0%
Sec-1-5min	100%	23,3%	Sec-2-5min	100%	0%
Sec-3-1min	100%	0%	Pen-5%	100%	97,6%
Sec-3-3min	99,4%	0%	Pen-10%	100%	100%
Sec-3-5min	99,8%	0%	Pen-30%	100%	100%
			Pen-40%	100%	100%

Table 5.7: Percentage of occurrences, out of the total of 1000 simulations, in which a congested traffic state is detected. The first three row blocks are related to the detectors, while the last one is related to the C-ITS service *Traffic Jam Ahead*.

Hour 1

As expected, higher penetration rates of connected vehicles lead to more frequent activations of the *Traffic Jam Ahead* service. Notably, even a low penetration rate of 5% is sufficient to trigger the service at least once in this scenario. Some activations occur just downstream of the bottleneck; this is because the TRCO_0 condition uses a 120-second average speed window, so a vehicle can pass the bottleneck and still have a low enough average speed (over the previous 120 seconds) to trigger a warning.

Here we are discussing the results of the bottleneck happening during the first simulated hour, specifically between minutes 10 to 40. The distribution of the traffic service activations for each penetration rate can be appreciated in Figure 5.12, noting with a deeper blue colour the number of occurrences for each pair Longitude/Time, each one rounded at the third and second digit, respectively. As we can expect, the higher the penetration rate, the more vehicles trigger the activating condition for the traffic jam ahead service. Crucially, even a very low penetration rate of 5% is more than enough to trigger the activating condition and send the related DENM.

It might seem out of place that some vehicles trigger the traffic service after surpassing the bottleneck. However, it is worth noting that the TRCO_0 activating condition specifically expresses an average speed during the last 120 seconds lower than $30km/h$. Hence, the bins with a longitude greater than the bottleneck are vehicles which have surpassed the bottleneck, but their increase in speed in the subsequent seconds is not reflected in an average speed higher than $30km/h$.

Looking at the distribution of the first instant of traffic jam detection for the first hour in Figure 5.13, we can see that the traffic detectors only manage to outperform the detection via C-ITS service with detectors 1 and 2 and consistently only with an aggrega-

tion period of 1 minute. Penetration rates as low as 20% already manage to detect the traffic jam at the 17 : 30 mark, while an aggregation period of 3 minutes forces the detection to take place at the 18 minute mark, extended to the 20 minute mark for the 5 minutes one. However, while the comparison is close in some test cases, there are two major considerations to take into account:

1. Despite the good results, the detector analysis is completely offline due to the required traffic flow analysis. While a previous analysis of similar conditions or inference through historical data could grant the possibility to do the traffic jam detection online, the C-ITS one is completely online and easily repeatable, also enabling the detection of stationary traffic through TRCO_1

2. Even though the vehicular detection is obtained at an average longitude higher than the one of the detectors, the communication range of the vehicular wireless technology would more than suffice for this deficit. For instance, even a wireless range of 300 meters would forward the traffic jam ahead DENM message of a vehicle found right under detector 1 50 meters beyond detector 2.

Hour 2

During the second hour of simulation (minutes 70 to 100), a traffic jam does form, but it is more sparse and less severe than the one in the first hour (as shown in Figure 5.14). As before, higher penetration rates lead to more vehicles detecting the jam, with several activations only occurring after vehicles have passed the bottleneck. The reduced severity of this congestion is evident from the fact that no vehicle triggers the C-ITS service before reaching detector 1.
Figure 5.15 provides further insight. First, detectors 2 and 3 failed to detect the congestion, as they were too far from the bottleneck.

However, a traffic queue still formed, as evidenced by vehicles registering low average speeds during the second hour. This finding highlights that detector placement is crucial: in a general road network, whether a traffic jam is detected can depend heavily on how close detectors are to the bottleneck (in our controlled scenario, detector locations were chosen to facilitate analysis), In this case, the C-ITS-based traffic jam detection is far superior. Only at the lowest penetration rate (5%) did it ever fail to detect the congestion, and even then in just 24 out of 1000 simulations. Secondly, in this less severe traffic jam, the detector-based approach with a 1-minute aggregation window detected the jam roughly 3 minutes earlier on average than the C-ITS service. The 3-minute aggregation detector performed worse on average (detecting the jam later) but its detection timing was still generally within the timeframe of the vehicles' experience of congestion. By contrast, the 5-minute aggregation detector failed to detect the jam in a timely manner (similar to what was observed in the first hour). Ironically, the detector with a 5-minute aggregation was the only one that detected the traffic jam in all simulation runs, whereas the 1-minute and 3-minute detectors missed the jam in some cases (see Table 5.7).

These findings from the second-hour simulation illustrate a fundamental trade-off between the two detection paradigms, a dichotomy widely discussed in traffic management literature. The performance of the FD-based method is inherently constrained by its static, infrastructure-bound nature. Its success is critically dependent on the optimal physical placement of sensors relative to a bottleneck's location, a significant challenge given that deploying sensors densely across an entire network is often impractical due to budget constraints [146]. As our results show, if congestion does not reach a sensor's precise location, the event goes undetected. This is because fixed detectors provide only localized, point-based measurements, capturing a "singular snapshot" of a vehicle as it passes, thereby

lacking the continuous spatial information needed to map the full extent and dynamics of a queue [75, 79].
In contrast, the C-ITS approach leverages vehicles as a distributed network of mobile sensors, offering an intrinsic spatial flexibility that fixed infrastructure cannot match [5]. This paradigm shift from Eulerian (fixed-point) to Lagrangian (moving-point) data collection allows for detection to occur wherever a connected vehicle encounters congestion, providing a much richer, high-resolution, and network-wide picture of traffic conditions [89]. However, this flexibility introduces its own critical dependency: the effectiveness and reliability of C-ITS services are directly tied to the market penetration rate and the spatial distribution of connected vehicles [115]. While our study demonstrates remarkable reliability even at low penetration rates, the system's performance is fundamentally a dynamic challenge of achieving a critical mass of equipped vehicles, as opposed to the static, capital-intensive challenge of optimizing a fixed sensor layout [146].

5.3.3 Discussion

The study was guided by two key questions, and the results provide clear answers. The first question asked how the real-time C-ITS approach compares with the model-based FD approach in detecting traffic jams. Our findings show that the C-ITS service is a fundamentally more reliable and spatially flexible detection method. Its key advantage is its ability to detect congestion regardless of where it forms relative to fixed infrastructure. This was most evident in the second-hour scenario, where the C-ITS service successfully detected the less severe congestion in over 97% of simulations, an event largely missed by detectors located further from the bottleneck. The primary disadvantage of the FD method is its rigid dependency on sensor placement; its success is highly contingent on the queue forming at or near a sensor's physical location, making it

inherently unreliable for a dynamic road network.
The second question asked under what specific conditions each method excels or falters. Here, the trade-off become explicit. The FD-based method's advantage is its potential for slightly faster detection under ideal conditions: when a sensor is placed optimally near a known bottleneck and a short data aggregation period (e.g., 1 minute) is used for severe jams. However, its disadvantages are significant: it falters dramatically with non-optimal placement or longer aggregation periods, and it can completely fail to detect less severe congestion. Conversely, the C-ITS approach primary advantage is its consistent performance across different locations and congestion severities. Its main limitation is its dependency on market penetration, as it falters at very low penetration rates if a connected vehicle is not present at the onset of congestion. These key characteristics are systematically summarized in Table 5.8.

The detailed comparison in Table 5.8 reveals that the two methods are not merely competitors but are, in fact, highly complementary. This complementary nature points directly towards the most promising direction for future traffic management: intelligent integration, as during the (long) transitional period toward full C-ITS deployment, a hybrid system can leverage the strengths of each to mitigate the weaknesses of the other.

5.3.4 Conclusion and future directions

This work showed offered a comprehensive simulation campaign clear insights regarding the potential of C-ITS for enhanced traffic management solutions. More specifically, the results show that the C-ITS-driven *Traffic Jam Ahead* service offers superior reliability and spatial flexibility (only constrained by market penetration rates) than the detector-based approach, which is strictly dependent on the detectors' position within the traffic scenario and yields

inconsistent results when dealing with less severe traffic jam conditions.

We contend that the fundamental principles revealed by our com-

Feature	FD-Based Method (Infrastructure-Centric)	C-ITS-Based Method (Vehicle-Centric)
Detection Principle	Macroscopic flow theory; identifies state transition when density or flow crosses a critical threshold at a fixed point [136].	Microscopic vehicle kinematics; event-triggered based on individual vehicle speed and behaviour over a time window [96, 60].
Detection Speed & Temporal Resolution	Potentially faster for severe jams with optimal sensor placement. Highly sensitive to aggregation period (1-5 min), which introduces detection lag [29].	Consistently fast and near real-time. Event-triggered messages (DENMs) are generated and disseminated instantly upon condition fulfilment [96].
Reliability & Robustness	Low. Highly dependent on sensor proximity to the bottleneck. Can miss less severe events. Prone to measurement errors [146].	High. Reliable even at low penetration rates ($\geq$5%). Performance degrades only if no connected vehicle is present at the onset of congestion [115].
Spatial Flexibility & Resolution	Very low (point-based). Provides data only at discrete sensor locations. Cannot detect events occurring between sensors [146].	Very high (network-wide, contingent on penetration). Detection occurs wherever a CCV encounters congestion, offering continuous coverage [5].
Data Granularity	Macroscopic and aggregated (flow, density, average speed). Loses individual vehicle detail [136].	Microscopic and granular (individual vehicle position, speed, heading). Enables high-resolution event data and analysis [84].
Deployment & Scalability	High capital cost for new sensor installation and ongoing maintenance. Poor scalability, as expanding coverage requires new physical installations [146].	Lower physical infrastructure cost over time (primarily RSUs). Excellent scalability, as coverage expands organically with vehicle market penetration [78, 115].
Integration Potential	Provides ground-truth data for calibrating and validating the physics-based components of hybrid systems [90, 109].	Provides rich, real-time data streams ideal for training and validating the data-driven/ML components of hybrid systems [90, 154].

Table 5.8: Systematic Comparison of Congestion Detection Paradigms

parison —specifically, the trade-off between the FD method's spatial dependency and the C-ITS method's penetration dependency— remain valid for other extra-urban and motorway contexts. Furthermore, it has to be noted that not all activation conditions of the C-ITS service were implemented, making our C-ITS performance estimate a conservative one.

Thus, future developments should address these limitations and explore new avenues. A crucial next step is to extend the comparative analysis to more complex and realistic road networks, including urban corridors and multiple-bottleneck scenarios, likely requiring different modelling paradigms.

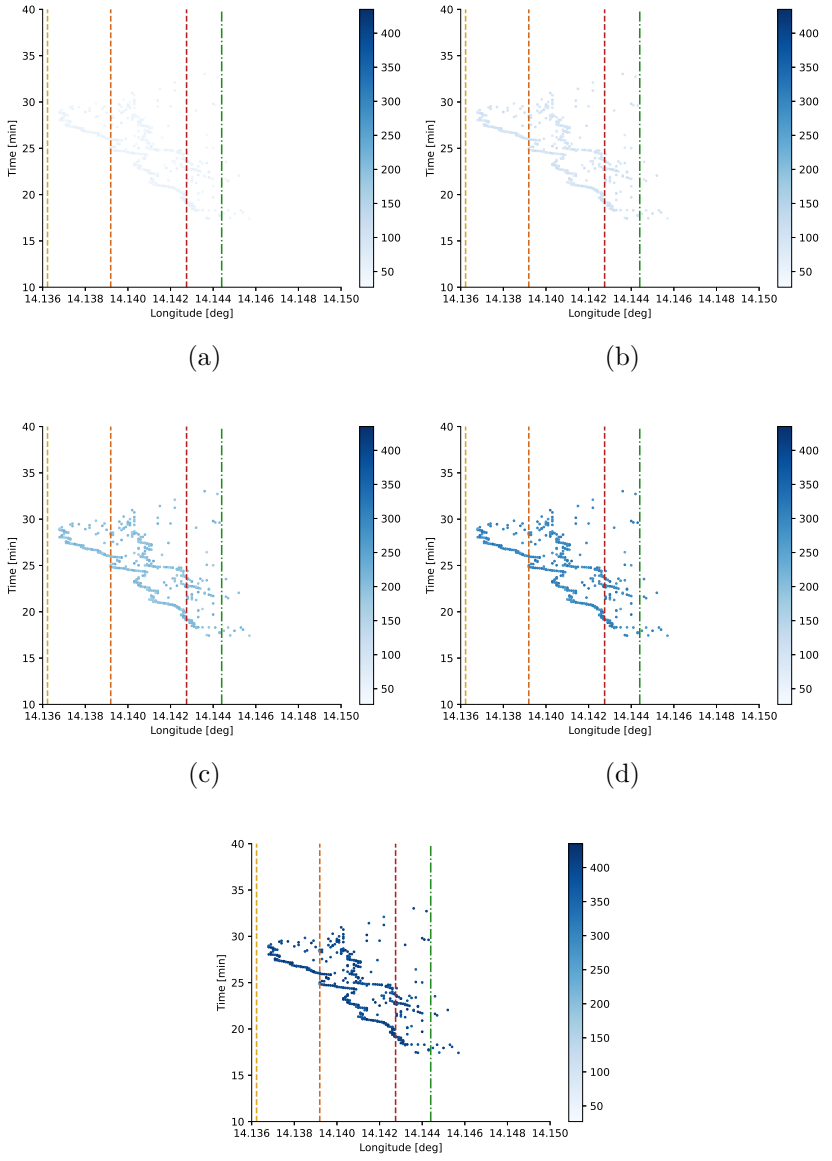
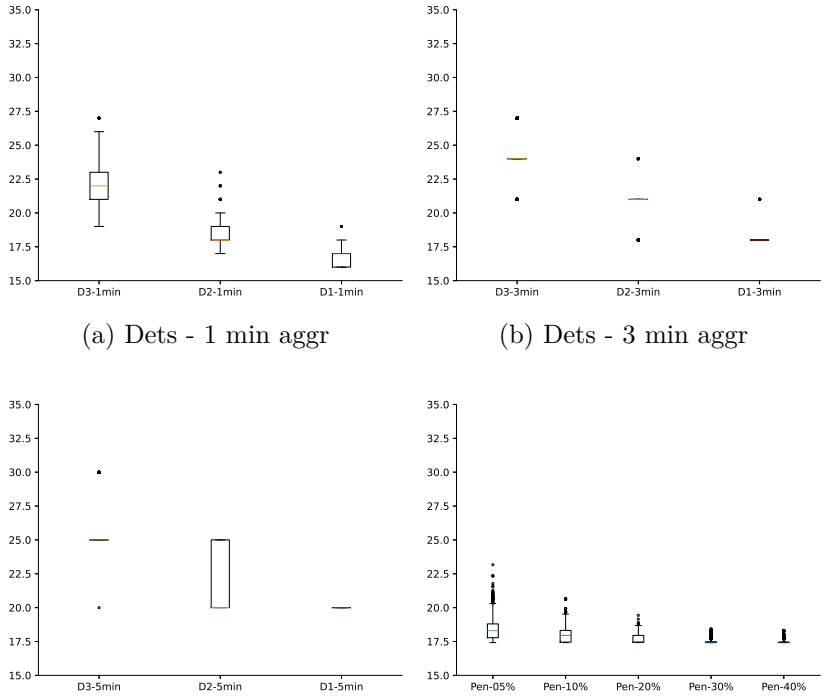


Figure 5.12: Distribution of the first activations for each CCV in the first hour of simulation, divided by penetration rate. In red, orange and yellow: the longitudinal positions of detectors 1, 2 and 3, respectively. In green, the bottleneck position. (**a**) 5% penetration; (**b**) 10% penetration; (**c**) 20% penetration; (**d**) 30% penetration; (**e**) 40% penetration.



(a) Dets - 1 min aggr

(b) Dets - 3 min aggr

(c) Dets - 5 min aggr

(d) Penetration rates

Figure 5.13: Boxplots of the first detection time in the first hour of simulation: (**a**) Detectors with 1-min aggregation; (**b**) Detectors with 3-min aggregation; (**a**) Detectors with 5-min aggregation; (**d**) *Traffic Jam Ahead* service (varying penetration rates).

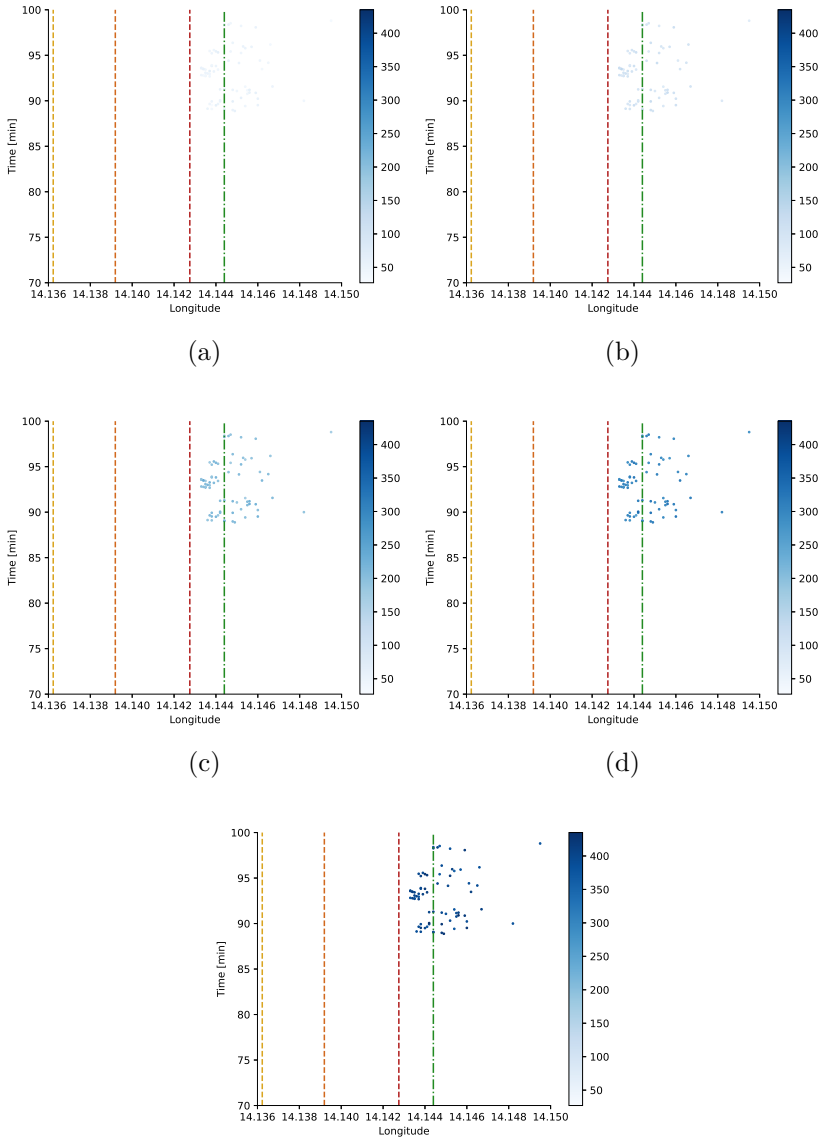
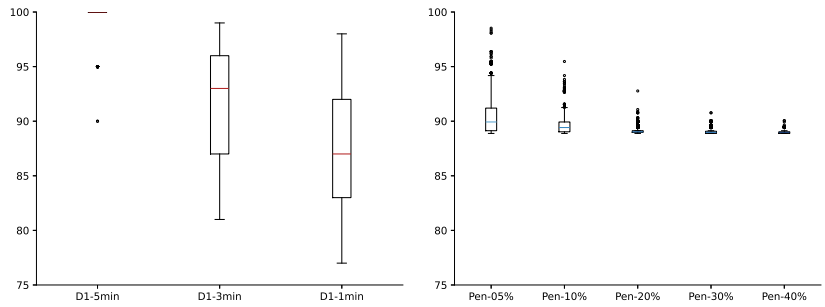


Figure 5.14: Distribution of the first activations for each CCV in the second hour of simulation, divided by penetration rate. In red, orange and yellow: the positions of detectors 1, 2 and 3, respectively. In green, the bottleneck position. (**a**) 5%; (**b**) 10%; (**c**) 20%; (**d**) 30%; (**e**) 40%.



(a) Dets - 1 min aggr (b) Penetration rates

Figure 5.15: Boxplots of the first detection time in the second hour of simulation. Only detector 1 is shown, since detectors 2 and 3 recorded no detections (see Table 5.7): (**a**) Detector 1 with 1-min and 3-min aggregation; (**b**) *Traffic Jam Ahead* service (varying penetration rates).

Chapter 6

Vehicular attacks in C-ITS

One of the crucial shortcomings when evaluating C-ITS services, and V2X communication in general, is the lack of proper tools dealing with cybersecurity issues. As we saw in section 2.4, this is crucial to ensure that C-ITS deployment goes hand-in-hand with user data privacy, cyberattacks mitigations and attack scenarios' coverage. For this reason, the platform has undergone even more testing and validation to cover these use cases. As explained before in section 4.4, this is in line with one of the identified research questions, which is:

> when C-ITS services are developed, are all the risks related to user security taken into account?
>
> **RQ4**

As a final note, this part of the work was done in collaboration with the ***Budapest University of Technology and Economics (BME)*** as part of the abroad period as a visiting PhD student.[1]

[1] Thank you for everything Budapest, köszönöm szépen 🇭🇺

6.1 Parallel Simulation for V2X

Since the goal here is to ensure that the platform is suitable for cyberattacks testing, a further step in conceptualization is required to lean towards the concept of *parallel simulation*, which is effectively described in [144]. Thus, the goal is to reproduce a digital twin of both real vehicles and the network communication, which is of primary importance in these test classes

6.1.1 Conceptualization

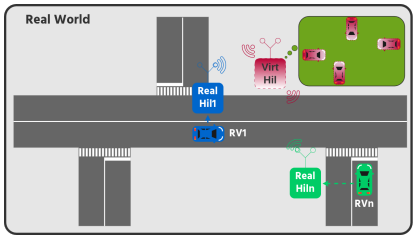
Therefore, we want the real vehicle to be fully submerged inside the simulated environment, seamlessly and transparently to the vehicle itself. To accomplish that, we refer to the extended architectural schema, which is presented in figure 6.1. The components described are the following:

Real world

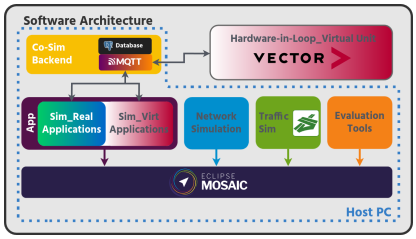
The real world setup, shown in Fig. 6.1a consists of one or more real vehicles, equipped with an OBU, freely moving on a stretch of road which is also tracked inside the simulation. The vehicles can send V2X messages, which are then received by another hardware unit, which we will refer to as HiL Virtual board (HiLV) in direct contact with the simulation, thus representing all the other involved simulated vehicles in the real world. Likewise, messages from the simulated vehicles are sent through the aforementioned HiLV, thus reaching the real vehicle. Hence, this allows to create actual VIL test cases with the same exact software architecture.

Software architecture

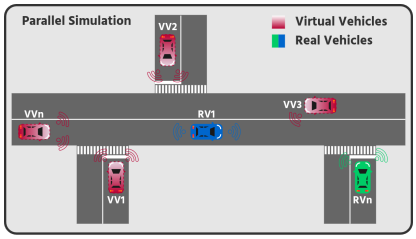
Indeed, the co-simulation platform remains the same, with MQTT as the primary communication protocol between software and hard-



(a) Real-world depiction: the RSU connected to the simulation communicates with one or more real vehicles



(b) Depiction of the software architecture stack. In this particular instance, it is linked to Vector hardware boards



(c) Parallel simulation, describing what the real vehicle "sees" through V2X messages

Figure 6.1: System architecture overview

ware components. Again, its flexibility, ease of deployment and widespread support on a wide range of devices is key within this use case, as the employed hardware boards, the Vector VN4610[2], also have native support for this protocol.

For the sake of simplicity, within the scope of these tests, our architecture distinguishes two types of applications: *simulated* applications are the ones found within the co-simulation framework, and are part of the MOSAIC software stack. Instead, HiL applications represent the control and processing logic written directly into HiL devices, such as the HiLV unit connected to the software architecture. The purpose of this unit is to forward the V2X messages from

[2]further details on the vector support page [140]

real units inside the simulation and vice versa.
On the other hand, the application layer distinguishes two types of *simulated* vehicular applications: *simulated virtual* ones contain the full control and messaging logic for all the fully simulated vehicle; on the other hand, *simulated real* applications are only used to keep track of the real vehicle position to correctly locate it when sending the V2X message through the network simulator, ultimately fullfilling the goal of correct delay representation. The complete software architecture can be viewed in Fig. 6.1

Parallel Simulation

Combining the two views together, we obtain the parallel simulation shown in Fig. 6.1c; thanks to the role of the network simulator in the software architecture is key here, actual network delays are correctly represented, despite the placement of the various hardware units in the real world.
It is also important to highlight both the flexibility and transparency of this architectural schema: indeed, every hardware unit with the ETSI ITS-G5 stack deployed can work within this parallel simulation approach. This also means that any Vehicle under Test (VuT) is almost completely transparent to the whole simulation. As mentioned before, the HiL units of choice need a mechanism to send and receive MQTT topic messages, although it is a feasible requirement given the popularity of this protocol. Finally, the MQTT broker can be completely decoupled from the host PC and deployed in another machine, such as a cloud-based one, further increasing the flexibility of this architecture. The finalized full desk setup can be observed in Fig. 6.2

6.2 UC4: Ground truth testing

6.2.1 Message flow example

The flow of a V2X message in our framework can be effectively visualized in the flowchart provided in Fig. 6.3, which represents the message flow from a generic virtual vehicle and one of the real vehicles. The message flow starts inside the $simulated\_virtual_a$ application inside the co-simulation framework, when the unit generates and dispatches a V2X message through the network simulator. Given that the vehicles are close enough to communicate between each other and there are no blocking obstacles between them, as established by the network simulator, the message arrives to the $simulated\_real_1$ application, which anchors the real vehicle inside the simulation. The message is then sent to the MQTT broker, more specifically it is published on the topic named *IV/simtohil*. Since the *hil_virtual* application, which is the hardware counterpart of all the simulated vehicles, is connected to the recipient MQTT broker and subscribed to the topic, it will receive the message on a callback. Finally, the content of this message is used to construct a real, ETSI-G5 compliant CAM message, which is then forwarded through a physical interface and it can be finally be received by the real vehicle.

Host PC specifications	
CPU	Ryzen 7 3800x
GPU	Nvidia GTX 1650 Super
RAM	16GB 3000MHz 2x16
Storage	500Gb SSD NVMe + 3Tb Hard Drive
OS	Windows 11 Enterprise

Table 6.1: Host PC specifications.

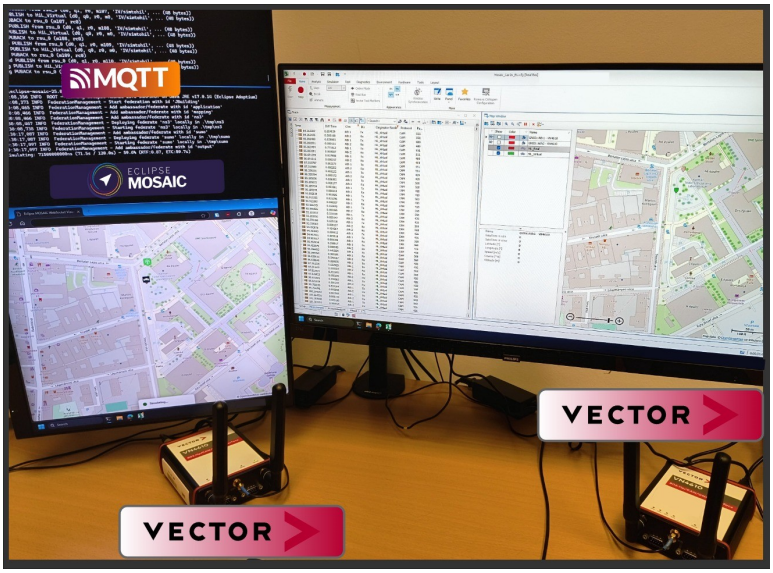


Figure 6.2: Desk setup showcasing the system architecture in operation. On the left screen, open terminals display the MQTT broker and the MOSAIC framework, along with its visualization at the bottom. The right screen illustrates the proper functioning of the two Vector units through the CANoe software.

6.2.2 Scenario description

The hardware connection is validated through a proof of concept test, which shows the capability of reproducing realistic real world conditions between a real unit and the virtual ones by leveraging the capabilities of the tools at our disposal. To do so, we developed a simple scenario of a vehicle travelling around a block while crossing another parked vehicle; the visual representation inside the simulator of this scenario can be appreciated in Fig. 6.4.

The goal of this scenario is to replay an already previously recorded V2X test involving the two real vehicles represented in the scenario, a parked one and a moving one, exchanging CAM messages at a varying frequency. To do so, inside the ns-3 network simulator there are some buildings placed in correspondence to the main wire-

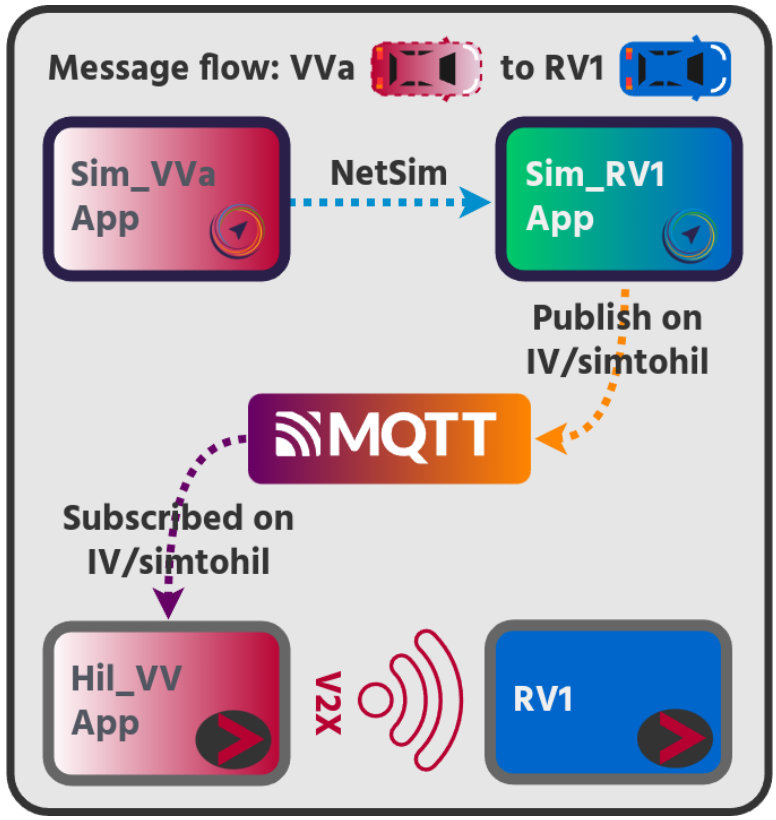


Figure 6.3: Flowchart of the V2X message route from a virtual vehicle to one of the real vehicles

less blocking paths to get as close as possible to the real working conditions. It is worth noting how currently ns-3 only allows the possibility to place buildings which are aligned with the X and Y axes, hence there is a discrepancy with respect to the real buildings placement. To reproduce the path and message sending of the moving vehicle, the real world logs were analyzed, processed and fed to a modified *simulated_virtual* vehicle application; this module is now capable of following the precise vehicle trajectory and sending CAM messages with the same frequency of the real one, effectively fully replaying the movement of the real vehicle. On the

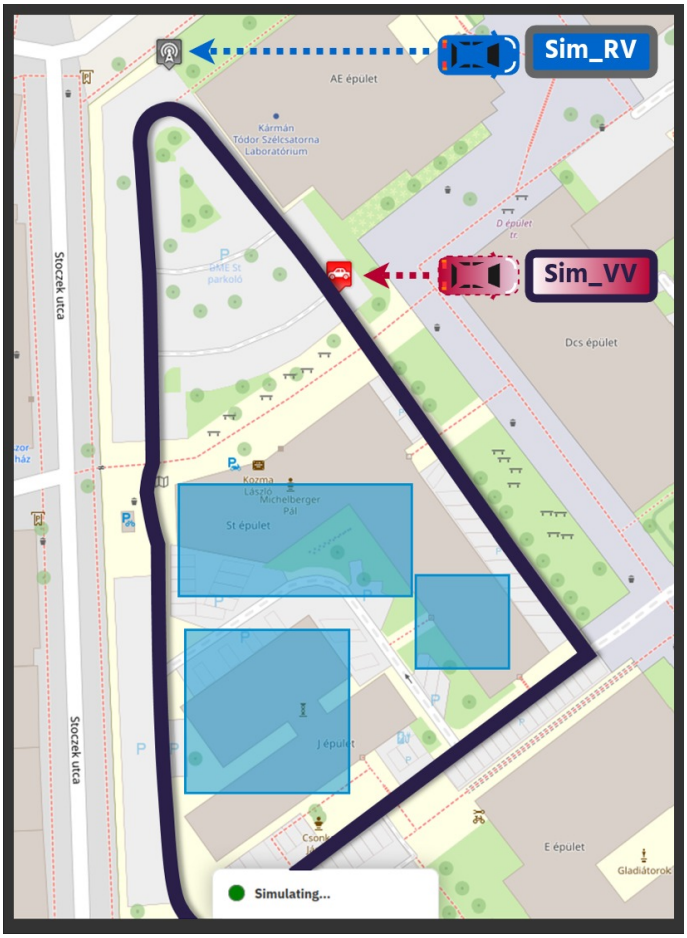


Figure 6.4: Screenshot of an ongoing simulation test run in MOSAIC. Highlighted here are the vehicle route and the buildings placed in ns-3, as well as the two vehicles placed in the simulation

other hand, the parked vehicle application was set to behave like a *simulated_real* application in our software architecture; thus, it publishes MQTT messages on reception, while its OBU counterpart only receives messages through real V2X message sending provided by the *HIL_Virtual* unit.

6.2.3 Results discussion

The main objective of this proof of concept scenario is to test whether there is a correct correspondence between the real life sent messages and the ones inside the simulation. To check that out, we analyzed the received messages on the *Real_HIL* unit through multiple runs and compared it with the real life results. The obtained plot is shown in Fig. 6.5. The results show a correspondence between real world and simulation of around 75%, with the remaining 25% shared between false positives and negatives.

By analyzing the plot, we can deduct that the majority of them happen for two fundamental causes: the first discrepancy can be easily registered in some random message skips which are not correspondent between the two tests. This randomness contributes to both false positives and negatives. On the other hand, we can see that the majority of false positives come from the fact that the simulation still manages to send messages way after the real world test has surpassed its line of sight. This is most likely caused by two factors: firstly, the geometry and placement of the buildings, which are not perfectly placed with respect to the real world, cause an imperfect reproduction of the line of sight for the two wireless devices. In hindsight, a more refined work is needed on the ns-3 part, given the limitations imposed by this software. Secondly, the SUMO simulator will not necessarily place the vehicle on the specified coordinates, but would rather follow the road positions, thus giving some partial displacement which might further modify the line of sight. More specifically, most of the false positives happen when the real vehicle turns left on the bottom right of Fig. 6.4, which is a private road in the real world. However, SUMO still handles it as a two lanes road, hence the virtual vehicle is placed slightly more on the left with respect to the real one, further justifying the false positives.

Overall, the important thing to note is that the software is easily

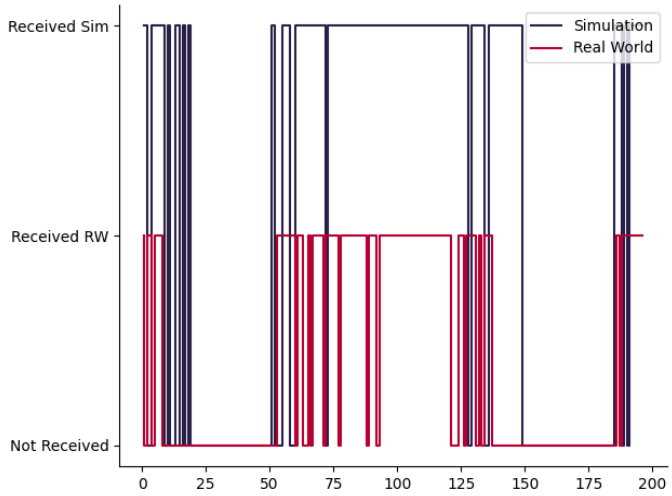


Figure 6.5: Plot of messages received for both the logged ground truth (real world) and simulated ground truth

capable of reproducing the majority of the real world conditions, thus offering a realistic network reproduction for co-simulation purposes.

In conclusion, this test has again underlined not only the flexibility and the extensibility of the co-simulation framework, but also the possibility to include validation campaigns to assess the most dangerous cyberattacks, both physical and software based.

6.3 UC5: Platform timing analysis

Afterwards, a later focus regarding the co-simulation platform was assessing its ability to reliably communicate *in real-time* with the other hardware sources. To do so, we proceeded with the creation of a *timing analysis server* which was used to spoof the actual communication delays and check out whether there were significant deltas which could be impactful towards the full assessment.

6.3.1 Information flow

The information flow including the timing analysis server is illustrated in Fig. 6.6, which shows the data transfer process from a Virtual Vehicle (VV) to a Real Vehicle (RV). In this case, the communication process starts at the virtual vehicle component that gathers vehicle state parameters from the traffic simulator module. The vehicle state information is attached as tags to a placeholder CAM packet and then transmitted through the network simulator. Using tags ensures minimal processing time and no precision loss. Based on the digitalized environment of the virtual vehicle, the tagged packets are transmitted according to the shadowing and fading characteristics of the environment. If the packet reaches the real vehicle in the simulation, the co-simulation framework forwards the vehicle state through MQTT to the HiL framework. Upon reception of the MQTT message, the HiL module uses the vehicle state to construct an ETSI-compliant CAM packet according to ASN1 and broadcasts it into the air.
The standalone timing analysis server is used to keep track of the total delta time it takes for a message to be sent from a virtual vehicle to the real vehicle OBU. The choice of a separated server is leveraged by two main factors: first, the server builds upon the exact communication mechanisms that are part of the simulation architecture, thus creating no additional overheads other than the

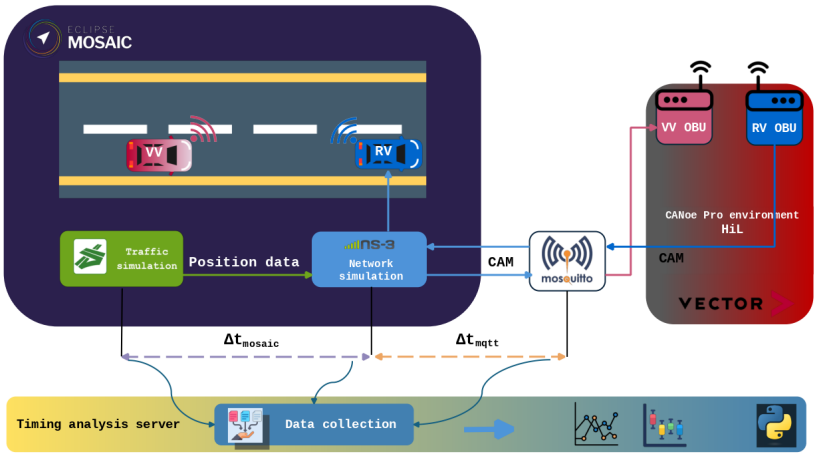


Figure 6.6: Information flow diagram

actual message reception and elaboration. Secondly, despite OBUs utilizing the manufacturer's Software Sync technology to synchronize their clocks to the simulation PC, we found that the deployed OBUs repeatedly suffered from clock skew issues. Eventually, we opted for a standalone solution to perform reliable timing analysis.

6.3.2 Scenario Description

The test scenario devised to validate the platform through the timing analysis server consists of a sample simulated scenario with two vehicles, a parked one and another running around a block. Both vehicles are equipped with OBUs so that they can exchange CAM messages between each other, with various sending frequencies depending on the current speed of the moving vehicle. Again, the moving vehicle follows the trajectory of a previously made experiment in real life, registering the sending of CAM messages between this car and the parked one. During the simulation, we want to keep track of how much time each simulated message takes to get from the Eclipse MOSAIC framework to actually be received by the

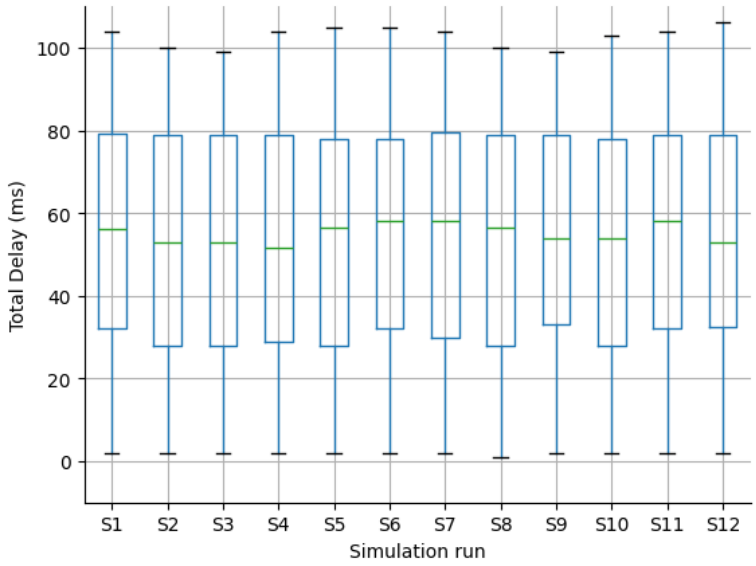


Figure 6.7: Boxplot of the timing analysis for each simulation run

receiving OBU. This is done via the timing analysis server, as previously mentioned. We collected multiple simulation runs to avoid aleatory fluctuations caused by standard computer functioning, for a total of twelve test runs.

6.3.3 Results and discussion

For each simulation run, we first plotted their data distribution, as shown in Fig. 6.7. As we can see, every simulation has an average sending delay just below 60 ms, with the majority of sending delays never exceeding 80 ms. This is the correct and expected behaviour, since the simulation works at a fixed step of 100 ms, hence every simulated vehicular application activates at a frequency of 10 Hz. Thus, the main goal is to keep the communication delay below 100 ms, which is almost always guaranteed barred isolated occurrences in some of the simulation runs. Hence, we can testify how the simulation framework is correctly working towards obtaining the parallel

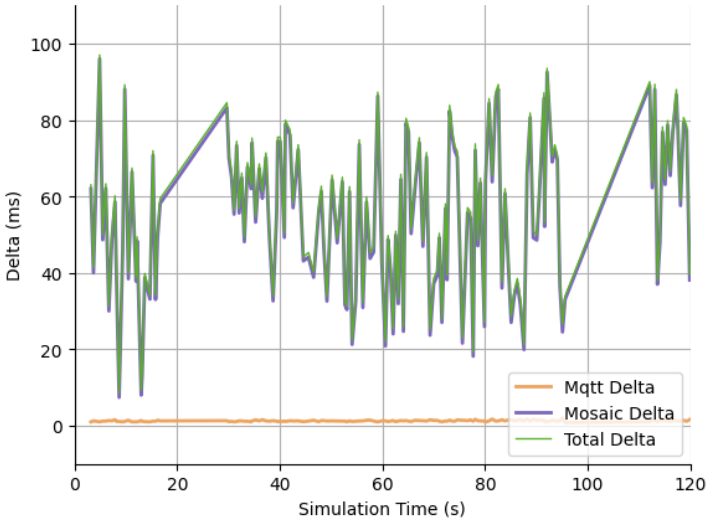


Figure 6.8: Average time delay for the sent messages during the entire simulation span.

simulation features.

Afterwards, we wanted to investigate how the communication delays affected the simulation, on average, during the entire 120 seconds of simulation run. To do so, we took all the simulation delays for every sent message during this span, and averaged it with all the simulation runs; the results can be observed in Fig. 6.8. As we can see, most of the delay is caused internally by the co-simulation framework, while the Mosquitto broker only adds a few milliseconds more. It is worth noting how the delay is heavily dependent on the simulation framework and on the evaluated scenario: as we can see, there are two regions, around the 30 and 100 seconds mark respectively, where no messages are received. This is due to the fact that, within the scenario, the vehicle is driving around some buildings, hence no messages arrive to the other simulated unit. Also, the delay time depends on when Eclipse Mosaic wakes up the receiver

application and the latter further forwards it to the OBUs. As explained before, given the simulation step of 100 ms, this can happen whenever within this time window. Hence, the fluctuations can be explained by the standard, aleatory behaviour of an operating system, as well as by aleatory wake-ups of the receiver application. It is still important to note how the MQTT delay is consistently very low, thus ensuring no further delays after the message exits the simulation framework.

This test finally confirms the complete suitability of the whole simulation platform with respect to real-time testing, with future works being in development towards assessing the impact of cyber-attacks using the platform.

6.4 UC6: PKI platform integration

One of the possible mitigations that greatly reduces the cybersecurity risk is the addition of a PKI platform. Besides, section 2.4 already shed the light on various pros and cons related to the actual usage of the PKI inside vehicular transmissions. Moreover, one of the goals of the co-simulation framework is to verify its testing capabilities in an increasingly realistic context. This includes, among others, the availability of a full communication stack, including certification, encryption and decryption of the messages sent within the framework.
As we said, despite its usage in modern internet, PKI still has some complications to tackle regarding vehicular communications, which can be summarized as the following:

- Moving ITS stations, such as vehicles, might have a harder time connecting to a Certificate Authority to have their certificates refreshed throughout the vehicle's journey. While LTE effectively mitigates this issue, there will be scenarios in which these stations are dependent on RSUs to check the validity of the certificates

- The high volume of traffic might interfere with the correct functioning of the certification process. Not only the network pressure could create unfair scenarios, where some of the vehicles are excluded from the C-ITS network due to not receiving a certificate, but this delay could significantly impact the correct functioning of the cooperative services themselves. Nevertheless, messages from uncertified users are discarded by default, thus resulting in possibly wrong computation, for example, for traffic detection services.

Again, these issues lead to a common, unfulfilled necessity: testing the complete infrastructure, avoiding mockups or simplifica-

tions, and doing realistic, holistic tests which can ensure a smooth deployment in the coming years. For this reason, this use case explores this issue within the context of the collaboration with the company Almaviva S.P.A., an Italian company based in Rome[3]. Regardless, this use case is in line with RQ5, which stated:

> Are C-ITS tested in their intended environment, and with the full message exchange pipeline enabled?
>
> **RQ5**

6.4.1 Architectural Schema

The goal of this application is very simple: forward each and every C-ITS message to the PKI platform, so that the latter can encrypt the messages and evaluate the performance metrics in cases of a high message rate. Within the context of the PKI, the role of the company encryption platform is depicted in figure 6.9. In short, the platform acts as a root certificate authority, accepting enrolment requests from and to enrolment and authentication authorities, respectively.

The full architectural schema is shown in figure 6.10. As we can see, the Co-simulation framework is still key here, as it allows the messages coming "from" the simulated environment to travel all the way to the company component, which is called the *PKI manager*.

Platform messages vs ETSI compliance

An important step in this work consisted in analyzing the differences between the vehicular messages sent within the co-simulation framework and the actual implementation of such messages in an ETSI-compliant standard. More specifically, for the purposes of the

[3]Almaviva is also a co-financial supporter of the PhD scolarship, under the Italian decree DM 352

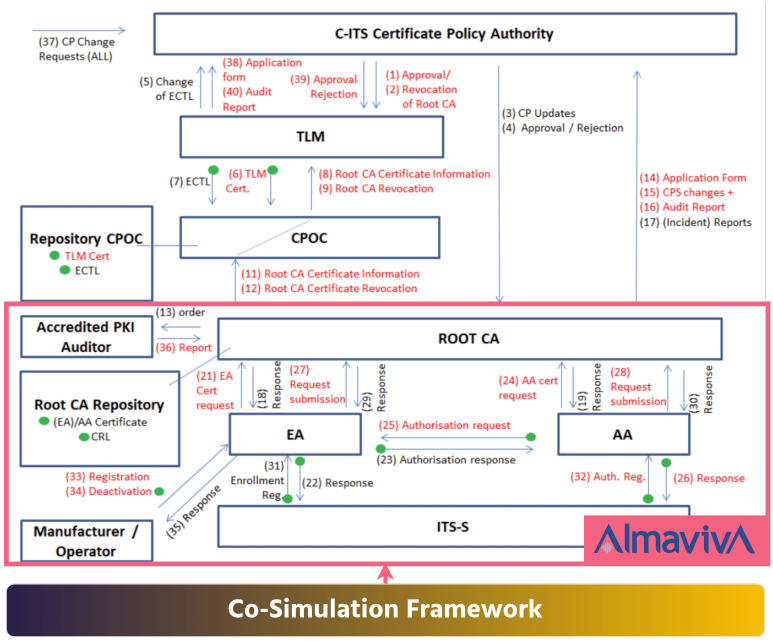


Figure 6.9: Depiction of the role of the company software with respect to the PKI platform

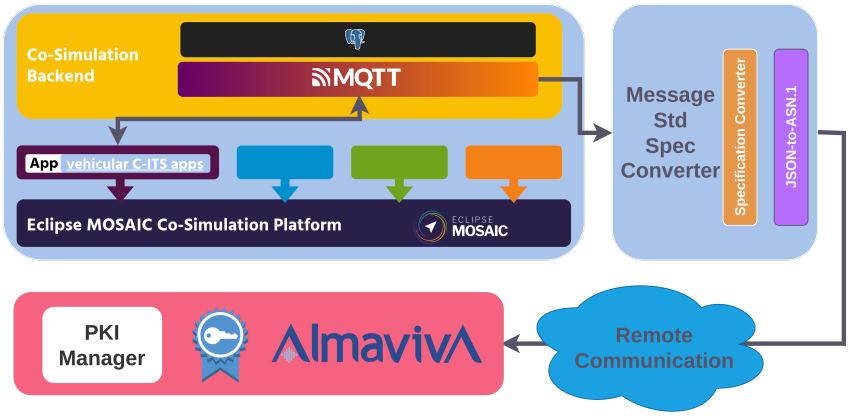


Figure 6.10: Co-simulation framework - PKI manager message flow

test, we focused on CAM messages, which are detailed, among oth-

ers, in ASN.1 notation[4]; the analysis immediately underlined major discrepancies with the messages exchanged within the co-simulation framework. These differences are explained by several factors:

- First, since the co-simulation framework CAM messages are generated using data coming from the SUMO simulator, many fields in the various containers described by the specification are left unused, as SUMO does not contain this information[5]. The implementation within the framework is kept simple, albeit non-compliant with the standard

- Since many of the fields are unused, the only necessity within the framework (and Eclipse MOSAIC) is to obtain a realistic payload size for the purpose of message exchange and decoding. The only relevant part of the message content are geolocation data, driving direction and speed of the vehicle who is sending the CAM message

For this reason, the use case has required the addition of a message specification converter, from the "MOSAIC" internal message to the actual, ETSI-compliant one. Hence, the module intercepts messages coming from the co-simulation framework and actively forwards them in the correct format to the PKI manager.

6.4.2 Proof of Concept

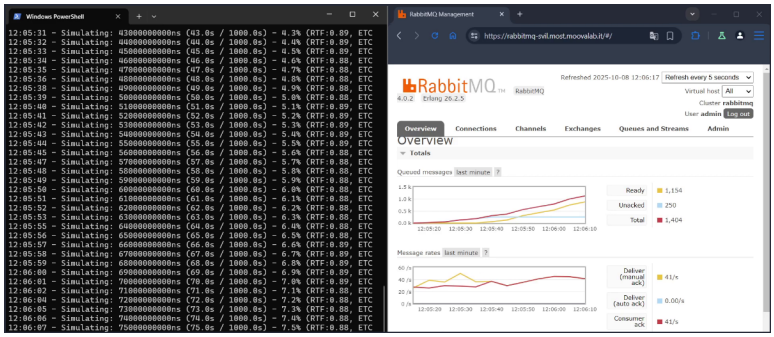
The message flow has been successfully tested in a sample scenario consisting on vehicle travelling along the A56 Tangenziale di Napoli, which was previously shown in figure 5.2a. For this use case, we just considered vehicles travelling in the east-to-west direction, starting at the entrance of *Corso Malta* and ending their travel at the exit of *Fuorigrotta*. It has to be noted that each and every vehicle within

[4]available in the official specifications document [52]

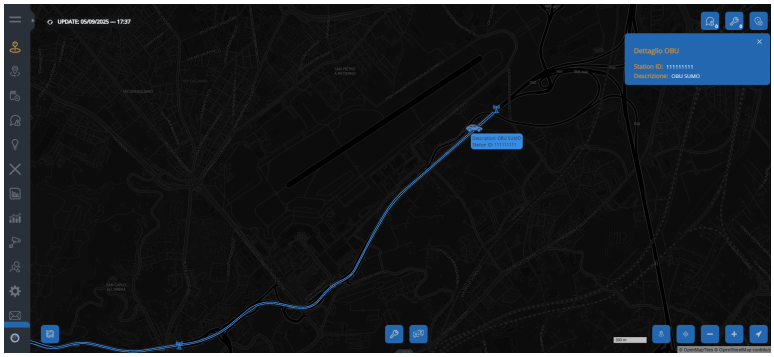
[5]this is a known issue in MOSAIC. See the issue on github for further info

the simulation, consisting of a total of around 3000 vehicles per hour, are connected vehicles for the purpose of the experimentation. During the simulation, the messages are sent to the PKI manager, found on a cloud server, which elaborates the message. If the message is correct with respect to the agreed syntax, it is considered digested by the platform. The scenario is made so that the CAM messages are sent with a rate of 1Hz. As we can see in figure 6.11, the messages are correctly digested by the platform, and the vehicles tracked on the road as if they were real vehicles.

In conclusion, this is only one of the first steps related to PKI platform testing. Nevertheless, these kinds of performance evaluations are of primary importance for the coming future, as more speculations regarding the placement and distribution of the PKI need to be made to ensure that the network pressure is kept under control, especially in dire circumstances such as heavy traffic phenomena.



(a) Side-by-side screenshot of the simulation platform in action on the left and PKI manager dashboard on the right, indicating the rate of incoming CAM messages



(b) Screenshot of the alternate view of the PKI manager dashboard showing one of the vehicles sendind the CAM messages to the platform. The messages trace a route followed by the vehicle, which is exactly the segment of the A56 where the simulation is taking place

Figure 6.11: Screenshots taken of the devised platform including a) the PKI manager and b) the tracking dashboard.

Chapter 7

Real-world applications: the A56 case study

When devising a testing and validation platform, the first thing to always keep in mind is the ultimate goal of shortening development and deployment times, effectively reducing the time-to-market of the proposed solutions. Besides, this is one of the outcomes of our RQ3, which stated:

> How can we seamlessly enable the rapid prototyping of cooperative services within their intended environment?
>
> **RQ3**

Taking this assumption a step further, after testing and validating the various C-ITS services with a realistic platform such as our project, the assumption is that such services are ready to be applied to a *real-world use case*. While this might seem the case of an early adoption, the previously reported UC3, detailed in section 5.3, suggests that the cohabitation of conventional traffic management strategies and C-ITS is certainly the way to go, with a gradual shift to connected services simultaneous with the increase

in the penetration rate of autonomous vehicles. These are the main motivations which have guided the collaboration with *Tangenziale di Napoli S.p.A.*, the road operator responsible for the A56 Tangenziale di Napoli, which was already chosen as the location for some of the previously described use cases, more specifically for UC1 in section 5.1[1].

The strategic role of the *Tangenziale* Before continuing, it is important to frame the strategic role that the A56, which will be called *"Tangenziale"* moving forward, has with respect to C-ITS testing and experimentation:

1. It is a stretch of highway of only 22km, managed by a single road authority who focuses on the Tangenziale only. This means that even devising an equipment strategy for the full road would be self-comprised in a relatively short section. Moreover, the full length of the Tangenziale is comprised within the same administrative authority, being the province of Naples.

2. In turn, this makes the possibility of doing tests on the Tangenziale tempting for geographical and economic reasons. Besides, the Tangenziale is already equipped with several cameras pointing on various sections of the road infrastructure for the manual traffic management center to check on possible traffic fluctuations.

3. Another strategical advantange lies on the fact that this highway crosses the entire city of Naples. Thus, it is extremely important and frequently used by the inhabitants, who depend on it to reach key city areas, such as the hospital district (*zona ospedaliera*) or the economic district (*centro direzionale*).

[1] Also chosen for the *real-life experience* of the author regarding the *nerve-wracking, never-ending, borderline infuriating* traffic jams at some of the entrances

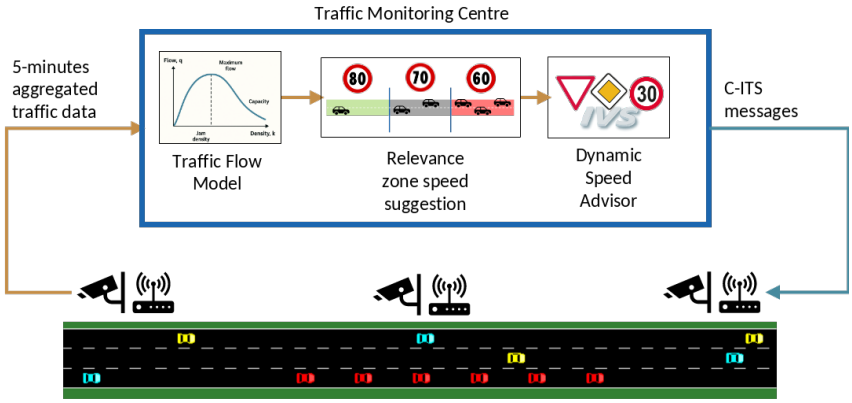


Figure 7.1: System architecture of the Tangenziale project collaboration

4. Consequently, some of the areas of the Tangenziale have important traffic spikes in peak hours, issue exacerbated by the occasional (albeit pretty common) minor traffic accident which essentially causes a lane closure at a critical moment.

Finally, the experimentation is rather important as it revolves around a broader perspective of **allowing both connected and automated vehicles to be tested within the Tangenziale jurisdiction without needing traffic closures or impediments**. This is a rather decisive step towards the adoption of C-ITS and automated vehicles on the road, and fully supported by the current Italian legislation.

7.1 Architecture and data flow

The project architecture is based on a modular design, which can be appreciated in figure 7.1.

As we can see, the various components can be summarized as follows:

1. The **Ingestion Module** is responsible for interfacing with

traffic estimates/forecasts, which serve as external input. Its sole function is to subscribe to, receive, and make traffic estimates/forecasts available to the C-ITS information generation module. More specifically, the traffic flow model estimates and forecasts for each 100-meter elementary segment of the road network; network state forecasts are projected one hour ahead, and information is provided with a temporal disaggregation of one minute. This decoupling allows data sources to be modified or added without impacting service logic. The module receives traffic data in 5-minutes aggregates, divided by detection camera.

2. The **Processing Module** contains a collection of submodules, each implementing the business logic of a specific C-ITS service. Each service makes some assumption based on the traffic forecasts made available by the Ingestion Module, processes them according to its own algorithms, and produces information compliant with the C-ITS paradigm. Adding C-ITS services involves adding further business logic and producing additional informative content.

3. Finally, the **Release Module** makes the information corresponding to the various C-ITS services available to the overall solution, after appropriately packaging it into an output compliant with C-ITS standards. The release occurs via REST APIs exposed by the cloud server.

Currently, the solution is implemented and validated to process and manage information useful for two services, which are the VSL and the Traffic Jam Ahead, both of which are described in appendix B.

7.2 On-field Test

The conducted test included the presence of an autonomous vehicle, a Maserati GranCabrio Folgore, which ran in fully autonomous mode while escorted by a road operator vehicle on a specific stretch of the Tangenziale, going from the *Arenella* entrance to the *Fuorigrotta* exit[2]. During the experimentation, the traffic management center was actively doing traffic estimations, and thus suggesting a speed limit to follow via VSL service. Thus, the vehicle adapted its speed accordingly during the test. On the other hand, the on-site traffic management center offered a view of the speed limits on the various road sections of the Tangenziale, showing the speed the autonomous vehicle had to follow.
The whole experimentation proved successful across multiple test runs. This, combined with the infrastructural effort carried on by the other involved project partners, certified the potential of the Tangenziale, as well as the one of C-ITS services, within a real-world use case.

[2]This was also reported by national media. See The Italian Ministry of Transport article [39] on the matter

Chapter 8

Conclusion

This thesis work tried to convene a simple, almost obvious, yet powerful idea: that the future of transportation is definitely going to be **digital**. From autonomous vehicles to smart infrastructures to intelligent transportation systems, the innovation in this sector will bring ample benefits to everyday passengers, reducing traffic jams, offering smoother driving experiences and also reducing the environmental carbon footprint.
However, the thesis also stated another important thing, which is that no matter the evolution of autonomous driving, no digitalization of the traffic environment would be complete without the **cooperation** of the involved road actors, from vehicles, to pedestrians, to the traffic infrastructure.
The role of this cooperation is often neglected, mostly due to the heterogeneity of the involved actors. While steps are going into the right direction, with respect to the European landscape, including collaborations between administrative authorities and car manufacturers, the aforementioned disconnection causes a general lack of **testing and validation strategies** for C-ITS; with a specific focus on the available tools, no thorough platform has been found solving this issue: either the focus was completely devolved to autonomous

driving, or the traffic environment representation was lacking crucial components.

8.1 Thesis Contributions

Given all these aspects revolving around this matter, we set the following thesis goal:

> The thesis goal is to **design, develop, validate and deploy a co-simulation platform** capable of performing testing and validation of Cooperative-Intelligent Transportation Systems. The platform needs to represent **all the different road environment components** in a **realistic** and integrated way, covering all the aspects of the Cooperative, Connected and Automated Mobility, thus trying to solve all the underlined Research Questions.

Thesis Goal

Thus, the first major contribution of the thesis, given a thorough analysis of the state of the art around C-ITS, was to develop the cited co-simulation platform. This process was guided by a standard process of software engineering development in order to guarantee that the required functional and non-functional requirements where correctly met and tracked inside the platform. Plus, following the welcomed practice of code reuse, the literature analysis offered us a valuable starting point for the platform in the Eclipse MOSAIC framework, which already included several of the traffic environment components identified. Crucially, the extension of the tool allows for Hardware-in-Loop testing to be performed, which are extremely important to increase the realism of the simulation and enable rapid prototyping features.

The completed platform was later challenged against different applications, which tried to fill several research gaps:

Hardware-in-Loop applications

- **[UC1]** Hardware-in-Loop testing of dedicated communication hardware allows for more realistic testing, as the device is tested in its intended environment, with realistic messages incoming.
- **[UC2]** Hardware-in-Loop testing of the Automated-Driving-Systems allowed to focus on the impact that the autonomous vehicle has on general traffic.
- **[UC4-5]** Finally, the platform also proved beneficial to test the cybersecurity features of the vehicular communications facilities, also including real devices into the loop. This was done both through timing analysis and ground truth testing

C-ITS software implementations During the platform deployment, we developed and tested different C-ITS services while performing HiL testing. This includes Traffic Jam Ahead [UC1], In-Vehicle Signage [UC2] and Variable Speed Limit [UC2]. Moreover, the Traffic Jam Ahead service was also used to test its competitiveness against standard detection strategies and traffic theory estimations **[UC3]**. Finally, the cybersecurity risks that the platform was able to quantify were partially mitigated by the applicative goal of securing the message flow through a Public Key Infrastructure platform **[UC6]**.

Real-world correlation Finally, as proof of the platform's quality features, a collaboration with the road operator of the *A56 Tangenziale di Napoli* allowed us to correlate the aforementioned test with real world usage of an Intelligent Speed Advisory service, used

to modulate the speed of an autonomous vehicle running in open traffic conditions during the experimentation.

8.2 Future directions

While this work sets the base for an employable co-simulation platform in the foreseeable future, there are several available directions to enhance the platform and the tests incidence:

- A valid direction is represented by the testing and development of more sophisticated C-ITS services. Nevertheless, the platform rapid prototyping capabilities allow to quickly test new regulations or standards devised in the foreseeable future, and their impact on the traffic environment.

- Moreover, there are still open questions regarding how autonomous vehicles impact the standard road traffic in critical scenarios, for example in cases of cooperative manoeuvres like merging or lane changing. Taking more and more autonomous vehicles into account, following the recent developments in automotive, is crucial to ensure their deployment is in harmony with the road traffic and the operators.

- Also, one key direction of the co-simulation platform is to use it for cybersecurity testing; this could include threat modelling tailored on the C-ITS case, and also the verification of compliancy with respect to standard security models such as STRIDE.

- Finally, to further enhance the quality and realism of the simulation, a further extension could include the addition of a 3D simulator. Indeed, its usage could increase the realism of the simulation, especially when considering the usage of object detection algorithms for cameras. Moreover, a 3D simulator

underlines a physics engine, which could further towards with the realism goal. Finally, the simulator could also be employed for demonstration purposes, which a nice addition in this context.

Appendix A

C-ITS messages taxonomy

This appendix contains basic info of the most known standardized C-ITS messages, included the ones actually employed and cited throughout the thesis work. For every message, a datasheet will be produced containing:

- The document source for the message
- Key details regarding structure and/or actual usage
- Main applications

A.1 Cooperative Awareness Message (CAM)

Source: ETSI TS 103 900 version 2.1.1. Last update: 11/2023 [54]

Description: from the ETSI document - *Cooperative Awareness Messages (CAMs) are messages exchanged in the ITS network between ITS-Ss 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. The content varies depending on the type of the ITS-S. For vehicle ITS-Ss the status information includes time, position, motion state, activated systems, etc. and the attribute information includes data about the dimensions, vehicle type and role in the road traffic, etc. On reception of a CAM the receiving ITS-S becomes aware of the presence, type, and status of the originating ITS-S. The received information can be used by the receiving ITS-S to support several ITS applications. [...].*
Basically, CAM messages are used to signal the presence of either a moving vehicle or a stationary RSU on the road. CAMs generation, management and dissemination is obtained through the *Cooperative Awareness* service, which is also found in the deployed ITS-S.
The CAM structure can be seen in figure A.1. A key highlight is the presence of the so called *Basic Container*, which includes the type of ITS-S as well as its latest position. Then, a vehicle ITS-S is also required to fill a *High-Frequency Vehicle Container*, which includes *"all fast-changing (dynamic) status information of the vehicle ITS-S such as heading or speed"*. Finally, a special vehicle might fill the optional *Special vehicle Container*.
Cam dissemination has to comply to security requirements (including a valid PKI certificate to transmit). Moreover, there are rules for generation frequency management, according to wireless

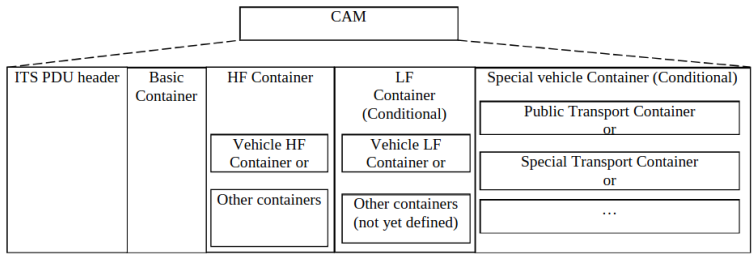


Figure A.1: General CAM structure from the ETSI document

channel usage, as detailed by the requirements of *Decentralized Congestion Control*[1]. In any case, CAM generation frequency is never lower than 1Hz neither higher than 10Hz.

Applications: since CAM messages are used to notify the presence and status of an ITS-S station, their usage is considered the basic layer of V2X communications. Plus, they are also employed in several monitoring and management applications, such as the Traffic Jam Ahead service or Cooperative Variable Speed Limit services.

[1] detailed in ETSI TS 102 724 [49]

A.2 Decentralized Environmental Notification Message

Source: ETSI TS 103 831 version 2.1.1. Last update: 11/2022 [50]

Description: from the ETSI document - *The Decentralized Environmental Notification basic service (DEN) is an application support functionality operating atthe facilities layer. The DEN basic service is responsible for the generation of DEN Messages (DENMs) based onreceived triggering information from applications. It is also responsible for the processing of received DENMs fromother C-ITS stations. The DEN service is especially suited for the exchange of event based safety related information.The DENM information is mainly used by ITS applications in order to alert road users of a detected event using ITS communication technologies [...]*
The detected event can be of different kinds, such as a notification of a traffic queue, accidents, roadworks, or an emergency vehicle approaching.
The DENM structure, shown in figure A.2, is composed of two mandatory containers, the Protocol Data Unit (PDU) and the management container, and then three optional containers which depend on the notified event. More specifically, the *situation container* describes the detected event, including type, linked cause and event zone if present; the *location container* describes the location of the detected event, including event speed and heading if it is a moving vehicle. Finally, the *à la carte container* contains additional information that is not provided by the other containers.
A DENM is only disseminated after a trigger, and each specific application has a repetition interval and duration which regulate the sending of a DENM message.

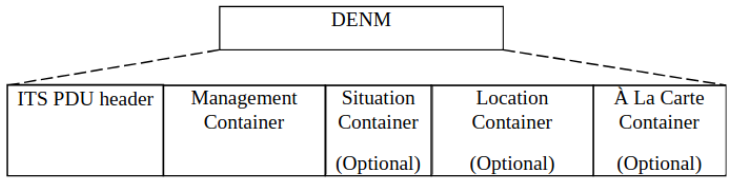


Figure A.2: General DENM structure from the ETSI document

Applications: the DENM messages are the standard for every event happening on the road, like hazards, traffic jams or emergency vehicles incoming. For this reason, any application requiring an event notification message creates a DENM message as a result.

A.3 Collective Perception Messages

Source: ETSI TS 103 324 version 2.1.1. Last update: 06/2023 [48]

Description: from the ETSI document - *Collective Perception Messages (CPMs) are transmitted by ITS-Ss in order to share information about perceived objects(such as vehicles, pedestrians, animals and other collision relevant objects) and perception regions (road regions that allow receiving ITS-Ss to determine unoccupied regions) in the local environment. This enhances the environmental perception of CPS-enabled ITS-Ss by providing information about non-V2X-equipped road users, other collision relevant objects, unoccupied regions and also increases the number of information sources for V2X-equipped road users. [...]*
A CPM contains a set of perceived objects and regions, including their observed status and attributes. Clearly, the content of a CPM varies depending on the detection capabilities of the ITS-S.
The CPM structure is found in figure A.3. As we can see, after the header containing the CPM PDU, the payload is highly variable, and depends on the detection capabilities of the ITS-S. Interestingly, the standard allows including up to eight *cpmContainers* for *"simplified future extensibility of the CPM"*. Regardless, each cpmContainers includes information regarding either the originating vehicle or RSU; then, there is one *Sensor Information Container* for each sensory capability of the ITS-S. Afterwards, one *Perception Region Container* for each perception region identified by the sensor, with special rulings for overlapping sensors. Finally, each identified object is placed inside a *Perceived Object Container*.

CPM generation is periodic and depends on various factors, including number of perceived objects and channel available resources.

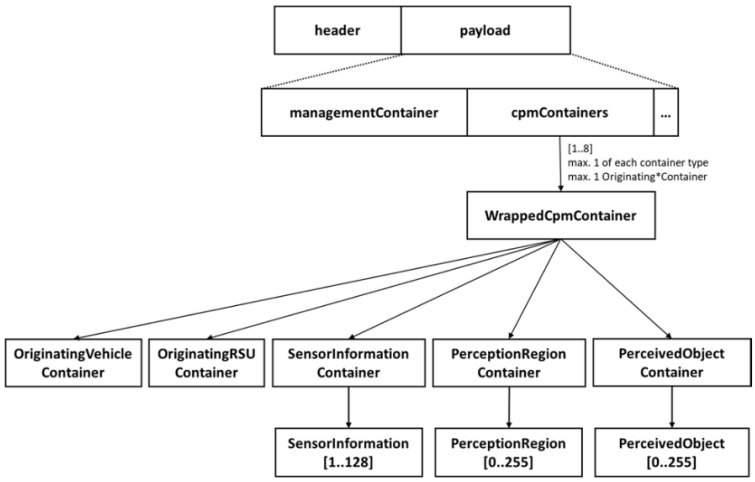


Figure A.3: General CPM structure taken from the ETSI document

Applications: the dissemination of CPMs are subject to the related Collective Perception Service (CPS), whose goal is to share cooperative data about perceived objects to enhance traffic safety and bolster cooperative awareness driving applications.

Appendix B

Relevant C-ITS services

This appendix contains a non-exhaustive list of C-ITS services which are already standardized by either the European Commission or the ETSI. Following the reported documents structure, most of the described services are divided in specific categories. As a final note, more details are given on the services deployed as part of the thesis use cases.

B.1 Priority C-ITS services

Source: European Union, Annex of Document C(2019)1789 *supplementing Directive 2010/40/EU of the European Parliament and of the Council with regard to the deployment and operational use of cooperative intelligent transport systems* [30]

Description: this annex details several services classified as *priority services*, which include the following categories: Within the *vehicle-to-vehicle services*:

- Traffic jam
- Stationary vehicle warning

- Special vehicle warning
- Exchange of Impact Reduction Containers (IRC) (detection of highly likely or unavoidable crash)
- Dangerous situation
- Adverse weather conditions

Within the *infrastructure-to-vehicle services*:

- In-Vehicle Signage
- Hazardous locations notification
- Road works warning
- Signalised intersections

These are all service categories which require prioritization, in the form of sending alarms through DENMs, due to the special occasion or hazard occurring at that moment.

Traffic Jam Ahead With a specific focus on the Traffic Jam Ahead, this C-ITS service *traffic jam ahead* is categorized as a Day 1 service according to the Car2Car consortium, thus it is considered a production-ready and standardized service. This service triggers the sending of DENM messages in situations where the related ego-vehicle detects a condition of traffic jam.
More specifically, before the service triggering, there are the following preconditions:

- no *Stationary Vehicle Warning* or *Special Vehicle Warning* are detected
- the ego vehicle is located in a non-urban environment. The location is determined in one of the following ways:

  - the vehicle speed is higher than 80 km/h for at least 30 s in the 180 s prior to each detection and the absolute value of the steering wheel angle is less than 90° for at least 30 s in the 60 s prior to each detection;
  - via on-board camera sensors;
  - via an on-board digital map.

- The speed and angle values are measured continuously.

If the pre-conditions are satisfied, at least one of the following service-specific conditions needs to be fulfilled:

- TRCO_0;
- TRCO_1 AND (TRCO_2 OR TRCO_3 OR TRCO_4 OR TRCO_5)

A summary and description of these triggering conditions is offered in table B.1.

Special Vehicle Warning This priority service has triggering clauses depending on the different use case, whether the emergency vehicle is moving to operation, it is safeguarding a hazard area or if it is a recovery service vehicle supporting a broken-down vehicle. The first use case, also known as the *Emergency vehicle approach* service, triggers the transmission of a DENM message from the emergency vehicle C-ITS station the moment the an operation is required.
More specifically, given the simple preconditions that the ITS-S is associated to an actual special vehicle and the latter is not already safeguarding a hazard area, the DENM transmission is triggered if the emergency vehicle light bar or siren is in use and the vehicle is not stationary. While the conditions are satisfied, the generated DENM is updated every 250ms.

In-Vehicle Signage The In-Vehicle Signage (IVS) is a Day-1 service in the C-ITS taxonomy which aims at reproducing both static and dynamic information of road signs inside the vehicle. The IVS service is employed to inform road users about static and dynamic road signs via I2V communication, like Message Signs, Text Panels

Count	Triggering condition	Status
TRCO_0	The ego vehicle is moving with an average velocity of 30 km/h or less and more than 0 km/h (this threshold is introduced to avoid overlap and to distinguish TRCO_0 and TRCO_1). The average velocity shall be calculated over a period of 120 s (the duration condition excludes frequently changing traffic states from triggering). Note: This TRCO covers the scenario where the ego vehicle is surrounded by stop-and-go traffic.	vehicle dynamics
TRCO_1	The ego vehicle velocity is equal to 0 km/h for at least 30 s. Note: This TRCO covers a scenario in which the ego vehicle is stationary and surrounded by other road users.	vehicle dynamics
TRCO_2	At least one DENM corresponding to the *traffic jam - traffic jam ahead* C-ITS service with the same driving direction has been received.	environment
TRCO_3	At least one traffic jam notification with the same driving direction has been received by means of mobile radio.	environment
TRCO_4	CAMs indicate a velocity of 30 km/h or less of at least five other vehicles within 100 m and with the same driving direction.	environment
TRCO_5	On-board sensors indicate a velocity 30 km/h or less of at least five other vehicles within 100 m and with the same driving direction.	on-board sensor

Table B.1: Triggering Conditions of the Traffic Jam Ahead, taken from the EC document

and Direction Signs among others. The ultimate goal is to increase the driver awareness by displaying the Dynamic Sign directly on the vehicle infotainment system throughout the whole validity period, thus reducing issues regarding limited line of sight.
There are various applications of IVS, among which there is the *Dynamic speed limit information*, which transmits Infrastructure-to-Vehicle (I2V) information using the Infrastructure to Vehicle Information (IVI) service, which is listed among the infrastructure services.

B.2 Infrastructure Services

Source: ETSI TS 103 301 version 1.3.1. Last update: 02/2020[1] [51]

Infrastructure to Vehicle Information Message (IVIM)

Description: this ETSI document describes and classifies infrastructure related ITS services, which support the connected traffic participants with additional info on any traffic occurrence. Thus, these services share the communication mode, as it is exquisitely I2V. Moreover, they share common protocol requirements, including security mechanisms and playload encapsulation. The services in question are the following:

- Traffic Light Maneuver (TLM)
- Road and Lane Topology (RTL)
- Infrastructure to Vehicle Information (IVI)
- Traffic Light Control (TLC)
- GNSS positioning correction (GPC)

[1] A newer version 2.1.1 published in 2021 is still in early draft phase and only contains the index

each service also has access to its related messages. For example, the TLC service manages the generation of Signal Request Extended Messages (SREMs) or Signal request Status Extended Messages (SSEMs).

Infrastructure to Vehicle Information Focusing on the IVI, this service exists to provide information to connected vehicles of different kinds, such as contextual speeds and road works warning. More specifically, IVI uses IVIM messages, which can provide both static and variable road signs as well as virtual signs. The generation and contextualization of an IVIM is determined by the service provider. Within this context, the standardization offered by ETSI is in armonious overlap with the specifications devised by the European Commission described in B.1.

B.3 Cooperative Services

Source: ETSI TR 102 638 version 2.1.1. Last update: 04/2024 [53]

Description: this standardization document provided by the ETSI describes all the ITS services, as well as the related use cases, which are *"intended to be the baseline for the development of the set of ITS Release 2"*. Hence, we are referring to *day3* services as per the taxonomy described by the car2car consortium [23], including *awareness driving* and *sensing driving* use cases.
More specifically, the document details several services and use cases. Some are briefly described below:

- Partial and high automation services: this category *"makes use of ITS service to trigger and control automated reactions at vehicles with low (i.e. L1 - L2 vehicles where the driver is still in charge of driving and monitoring tasks) as well as*

high automation capabilities (L3 + vehicles where the driver is released partially or totally from its monitoring and driving responsibilities). In this context, vehicles V2X communication and ITS-S services extend the capabilities of traditional ADASs and automated functions providing longer and non-line of sight detection ranges, as well as explicit communication of information between senders and receivers.". Examples include *hazardous location notification*, *Cooperative Adaptive Cruise Control*, *Cooperative Adaptive Emergency Brake System*, *Advanced Pre-Crash sensing*, *Cooperative Active Lane Keeping* or *Cooperative Intelligent Speed Adaptation*.

- CCAM augmented perception: includes all classes of services which try to augment the autonomous perception of non-connected dynamic objects through *Collaborative Perception Services*. Use cases of this category might include perceiving a non-connected vehicle at an intersection, a stationary vehicle, or non-connected *Vulnerable Road User*.

- Vehicles' coordination: under this category there are ITS applications that *"use ITS-S services to coordinate vehicle movements in terms of manoeuvres or trajectories."*. This includes situations where a vehicle ITS-S allows others to follow its trajectory, or where vehicles exchange messages to notify the intention to implement specific manoeuvres, or enable cooperative ones. Use cases exemplifying this category are *Cooperative Lane Merging, Lane Change, or Platooning* applications

- Advanced warning and information for Vulnerable Road Users (VRUs) protection: this category includes all the use cases related to ITS applications made to enhance VRUs safety, such as cyclists, standing scooter drivers or motorcyclists. Related use cases include an *overtaking motorcycle*, *VRU presence awareness* or *collision warning*, or *Interactive VRU crossing*.

B.4 Green Light Optimal Speed Advisory (GLOSA)

Source: Study on the Deployment of C-ITS in Europe: Final Report [34]

Description: the GLOSA is a day 1 I2V service which provides speed advice to drivers approaching traffic lights, with the goal of reducing the likelihood that they will have to stop at a red light. In turn, this reduces the potential sudden acceleration (or braking) phenomena, decreasing the accidents rate and enhancing traffic efficiency, vehicle operation (fuel saving) and environmental benefits by reducing unnecessary accelerations.

Glossary

ACC Adaptive Cruise Control

ADAS Advanced Driver Assistance System

ADS Automated Driving System

AV Autonomous Vehicle

CA Certificate Authority

CACC Cooperative Adaptive Cruise Control

CAM Cooperative Awareness Message

CAV Connected Automated Vehicle

CCAM Cooperative, Connected and Automated Mobility

CCV Connected Conventional Vehicles

C-ITS Cooperative-Intelligent Transportation System

CP Cooperative Perception

CPM Collective Perception Message

CPS Collective Perception Service

CVE Common Vulnerabilities and Exposures

DCC Decentralized Congestion Control

DIL Driver-in-Loop

ETSI European Telecommunications Standards Institute

FCD Floating Car Data

FD Fundamental Diagram

FOT Field Operational Test

GLOSA Green Light Optimal Speed Advisor

HIL Hardare-In-Loop

I2V Infrastructure-to-Vehicle

IoT Internet of Things

ISA Intelligent Speed advisor

ITS Intelligent Transportation System

IVI Infrastructure to Vehicle Information

IVS In-Vehicle Signage

LKA Lane Keeping Assistant

MIL Model-in-Loop

MRE Mixed Reality Environment

ODD Operational Design Domain

PDU Protocol Data Unit

PKI Public Key Infrastructure

PLC Programmable Logic Controller

RMSE Root Mean Squared Error

RSU Road-Side Unit

SIL Software-In-Loop

SUMO Simulation of Urban MObility

TMC Traffic Management Center

VIL Vehicle-In-Loop

VRU Vulnerable Road User

VSL Variable Speed Limit

XIL X-in-Loop

Author's Publications

Publications in inverse chronological order

- **Andrea Marchetta**, Tamás Márton Kazár, Angelo Coppola, Zsombor Pethő, Zsolt Szalay, Árpád Török. 2025. Performance evaluation of a parallel simulation framework for V2X testing. **9th IEEE Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)**.
  DOI: 10.1109/MT-ITS68460.2025.11223561

- Stefano Balirano, **Andrea Marchetta**, Angelo Coppola. 2025. Mitigating Merge Congestion with CCAM services: a Comparative Study on A56 Italian motorway. **9th IEEE Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)**.
  DOI: 10.1109/MT-ITS68460.2025.11223549

- Angelo Coppola, Luca Di Costanzo, **Andrea Marchetta**. 2025. Enhancing Sustainable Mobility: A Comparative Analysis of C-ITS and Fundamental Diagram-Based Traffic Jam Detection. **Sustainability**.
  DOI: https://doi.org/10.3390/su17188217

- **Andrea Marchetta**, Tamás Márton Kazár, Marcello Cinque, Angelo Coppola, Zsombor Pethő, Zsolt Szalay, Árpád Török. 2025. Network and Vehicle Digital Twins for Hardware in the

Loop Simulation of C-ITS Applications. **55th Annual IEEE International Conference on Dependable System Networks Workshops (DSN-W)**.
DOI: 10.1109/DSN-W65791.2025.00054

- Mario Fiorentino, Michele Caggiano, Alessandro Magliacane, Angelo Coppola and **Andrea Marchetta**. 2025. Bridging the Safety Gap: A C-ITS Solution for Protecting Vulnerable Road Users. **55th Annual IEEE International Conference on Dependable System Networks-Supplemental Volume (DSN-S)**.
DOI: 10.1109/DSN-S65789.2025.00043

- **Andrea Marchetta**, Martina Togna, Angelo Coppola, Marcello Cinque. 2025. HIL Robustness Testing and Validation of Computer Vision and C-ITS for Vehicle Fault Mitigation. **28th International Symposium on Real-Time Distributed Computing (ISORC)**.
DOI: 10.1109/ISORC65339.2025.00035

- **Andrea Marchetta**, Angelo Coppola, Marcello Cinque and Gennaro Nicola Bifulco. 2024. Co-Simulation Framework for Hardware-in-the-Loop testing of Integrated CCAM services. **27th IEEE International Conference on Intelligent Transportation Systems (ITSC)**.
DOI: 10.1109/ITSC58415.2024.10919518

- Angelo Coppola, Luca Di Costanzo, **Andrea Marchetta**. 2024. Comparative Energy Assessment of Automated Electric Vehicles Equipped With Acc Systems. **7th edition of the International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles and International Transportation Electrification Conference (ESARS-ITEC)**.

DOI: 10.1109/ESARS-ITEC60450.2024.10819768

- Angelo Coppola, **Andrea Marchetta**. 2024. Safety and Energy Assessment of Electric Vehicles equipped with Adaptive Cruise Control system. **24th International Conference on Environment and Electrical Engineering (EEEIC)**. **DOI**: 10.1109/EEEIC/ICPSEurope61470.2024.10751520

- **Andrea Marchetta**, Angelo Coppola, Marcello Cinque, Mario Fiorentino, Gennaro Nicola Bifulco. 2023. An Eclipse MOSAIC-based Hardware-in-Loop V2X Co-Simulation Framework for CCAM services. **26th IEEE International Conference on Intelligent Transportation Systems (ITSC)**. **DOI**: 10.1109/ITSC57777.2023.10422599

- Vincenzo M Arricale, Michele Caggiano, Marcello Cinque, Angelo Coppola, Flavio Farroni, Mario Fiorentino, Andrea Garofalo, **Andrea Marchetta**, Antimo Perfetto, Aleksandr Sakhnevych. 2023. EMER-GO: real-time grip enhanced speed advisory for emergency intelligent transportation systems. **53nd Annual IEEE International Conference on Dependable Systems and Networks Workshop (DSN-W)**. **DOI**: 10.1109/DSN-W58399.2023.00028

- Marcello Cinque, Luigi De Simone, **Andrea Marchetta**. 2022. Certify the uncertified: Towards assessment of virtualization for mixed-criticality in the automotive domain. **52nd Annual IEEE International Conference on Dependable Systems and Networks Workshops (DSN-W)**. **DOI**: 10.1109/DSN-W54100.2022.00012

Bibliography

[1] Abdullahi Modibbo Abdullahi, Wassila Lalouani, and Messaoud Rahim. A robust misbehavior detection system for cooperative driving network. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6. IEEE, 2024.

[2] Soyoung Ahn, Michael J. Cassidy, and Jorge Laval. Verification of a simplified car-following theory. *Transportation Research Part B: Methodological*, 38(5):431–440, 2004.

[3] Md Shah Alam, Abiral Acharya, and Jared Oluoch. A novel technique for mapping jammed areas in connected and autonomous vehicles (cavs). In *2021 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, pages 111–117, 2021.

[4] Hesham Alghodhaifi and Sridhar Lakshmanan. Autonomous vehicle evaluation: A comprehensive survey on modeling and simulation approaches. *Ieee Access*, 9:151531–151566, 2021.

[5] M. Aljamal, M. Abdel-Aty, and M.H. Zaki. Real-time traffic state measurement using autonomous vehicles' open data. *IEEE Transactions on Intelligent Transportation Systems*, 24:1–12, 2023.

[6] Sebastian-Ioan Alupoaei and Constantin-Florin Caruntu. Optimizing urban traffic efficiency and safety via v2x: A simulation study using the mosaic platform. *Sensors*, 25(17):5418, 2025.

[7] Mohamed Ahzam Amanullah, Seng W Loke, Mohan Baruwal Chhetri, and Robin Doss. A taxonomy and analysis of misbehaviour detection in cooperative intelligent transport systems: A systematic review. *ACM Computing Surveys*, 56(1):1–38, 2023.

[8] Maytheewat Aramrattana, Tony Larsson, Jonas Jansson, and Arne Nåbo. A simulation framework for cooperative intelligent transport systems testing and evaluation. *Transportation research part F: traffic psychology and behaviour*, 61:268–280, 2019.

[9] Marwane Ayaida, Nadhir Messai, Geoffrey Wilhelm, and Sameh Najeh. A novel sybil attack detection mechanism for c-its. In *2019 15th international wireless communications & mobile computing conference (IWCMC)*, pages 913–918. IEEE, 2019.

[10] Ali Balador, Chumeng Bai, and Foroogh Sedighi. A comparison of decentralized congestion control algorithms for multi-platooning communications. In *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 674–680. IEEE, 2019.

[11] Ali Balador, Elena Cinque, Marco Pratesi, Francesco Valentini, Chumeng Bai, Arrate Alonso Gómez, and Mahboubeh Mohammadi. Survey on decentralized congestion control methods for vehicular communication. *Vehicular Communications*, 33:100394, 2022.

[12] Mohamed Berrazouane, Kailin Tong, Selim Solmaz, Martijn Kiers, and Jacqueline Erhart. Analysis and initial observations on varying penetration rates of automated vehicles in mixed traffic flow utilizing sumo. In *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pages 1–7. IEEE, 2019.

[13] Roghieh A Biroon, Zoleikha Abdollahi Biron, and Pierluigi Pisu. False data injection attack in a platoon of cacc: Real-time detection and isolation with a pde approach. *IEEE transactions on intelligent transportation systems*, 23(7):8692–8703, 2021.

[14] Thecyphere blog. Hoppscotch - open source api development ecosystem. `https://hoppscotch.io/`. [Accessed 09-12-2025].

[15] Thecyphere blog. What is pki (public key infrastructure) in cyber security? `https://thecyphere.com/blog/pki-explained/`. [Accessed 09-12-2025].

[16] Mohammed Lamine Bouchouia, Houda Labiod, Ons Jelassi, Jean-Philippe Monteuuis, Wafa Ben Jaballah, Jonathan Petit, and Zonghua Zhang. A survey on misbehavior detection for connected and autonomous vehicles. *Vehicular Communications*, 41:100586, 2023.

[17] Mark Brackstone and Mike McDonald. Car-following: a historical review. *Transportation Research Part F: Traffic Psychology and Behaviour*, 2(4):181–196, 1999.

[18] Craig Brogle, Chao Zhang, Kai Li Lim, and Thomas Bräunl. Hardware-in-the-loop autonomous driving simulation without real-time constraints. *IEEE Transactions on Intelligent Vehicles*, 4(3):375–384, 2019.

[19] Bruno. Bruno - reinventing the api client. https://www.usebruno.com/. [Accessed 09-12-2025].

[20] C-Roads. Objectives: C-roads. https://www.c-roads.eu/platform/objectives.html, 2016. [Accessed 19-08-2025].

[21] C-Roads. The c-roads platform - an overview of harmonised c-its deployment in europe. https://www.c-roads.eu/fileadmin/user_upload/media/Dokumente/C-Roads_Brochure_2021_final_2.pdf, 2021. [Accessed 19-08-2025].

[22] Marc Gonzàlez Capdevila, Natanael Vitorino, Yuri Poledna, Bruno Malena, Maikol Funk Drechsler, Gustavo G Albuquerque, Tales Furlan, and Roberto S Netto. Smart&safe mobility lab: Mixed reality environment with hil, cv2x for vru detection. In *2024 IEEE 13th International Conference on Cloud Networking (CloudNet)*, pages 1–5. IEEE, 2024.

[23] Car2Car. Car2car communication consortium. https://www.car-2-car.org/, 2019. [Accessed 19-08-2025].

[24] Carlos Mateo Risma Carletti, Claudio Casetti, Jérôme Härri, and Fulvio Risso. ms-van3t-carla: an open-source co-simulation framework for cooperative perception evaluation. In *2024 19th Wireless On-Demand Network Systems and Services Conference (WONS)*, pages 93–96. IEEE, 2024.

[25] Stefano Catozzi. *Design of a NMPC System for Automated Driving and Integration into the CARLA Simulation Environment.* PhD thesis, Politecnico di Torino, 2024.

[26] CCAM. Ccam homepage — ccam.eu. https://www.ccam.eu/, 2021. [Accessed 19-08-2025].

[27] Greg Chance, Abanoub Ghobrial, Kevin McAreavey, Séverin Lemaignan, Tony Pipe, and Kerstin Eder. On determinism of game engines used for simulation-based autonomous vehicle verification. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):20538–20552, 2022.

[28] Fadlallah Chbib, Sherali Zeadally, Ahmad Fadlallah, Rida Khatoun, and Ali El Attar. Tracking vehicles in cooperative intelligent transportation systems: Attacks, defense solutions, and future directions. *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[29] Nicolas Chiabaut, Christine Buisson, and Ludovic Leclercq. Fundamental diagram estimation through passing rate measurements in congestion. *IEEE Transactions on Intelligent Transportation Systems*, 10(2):355–359, 2009.

[30] European Commission. Annex to the commission delegated regulation supplementing directive 2010/40/eu of the european parliament and of the council with regard to the deployment and operational use of cooperative intelligent transport systems. `https://eur-lex.europa.eu/resource.html?uri=cellar:9a2fe08f-4580-11e9-a8ed-01aa75ed71a1.0014.02/DOC_2&format=PDF`. [Accessed 09-12-2025].

[31] European Commission. C-its point of contact. `https://cpoc.jrc.ec.europa.eu/Documentation.html`. [Accessed 09-12-2025].

[32] European Commission. Certificate policy for deployment and operation of european cooperative intelligent transport systems (c-its), release 1.1. `https://transport.ec.europa.eu/system/files/2018-05/c-its_certificate_policy-v1.1-track_changes.pdf`. [Accessed 09-12-2025].

[33] European Commission. Cooperative, connected and automated mobility (ccam) — transport.ec.europa.eu. https://transport.ec.europa.eu/transport-themes/smart-mobility/cooperative-connected-and-automated-mobility-ccam_en. [Accessed 19-08-2025].

[34] European Commission. Study on the deployment of c-its in europe: Final report. https://transport.ec.europa.eu/system/files/2016-10/2016-c-its-deployment-study-final-report.pdf. [Accessed 09-12-2025].

[35] European Commission. A european strategy on cooperative intelligent transport systems, a milestone towards cooperative, connected and automated mobility. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52016DC0766, 2016. [Accessed 20-08-2025].

[36] Lian Cui, Jia Hu, B Brian Park, and Pavle Bujanovic. Development of a simulation platform for safety impact analysis considering vehicle dynamics, sensor errors, and communication latencies: Assessing cooperative adaptive cruise control under cyber attack. *Transportation research part C: emerging technologies*, 97:1–22, 2018.

[37] Gabriel Toffanetto França Da Rocha, Rodrigo Moreira Bacurau, and Janito Vaqueiro Ferreira. microautoware: An autoware vehicle interface designed for real-time embedded systems with hardware-in-the-loop (hil) support. In *2025 IEEE Intelligent Vehicles Symposium (IV)*, pages 1546–1551. IEEE, 2025.

[38] M Deeksha, Ashish Patil, Muralidhar Kulkarni, N Shekar V Shet, and P Muthuchidambaranathan. Multistate active com-

bined power and message/data rate adaptive decentralized congestion control mechanisms for vehicular ad hoc networks. In *Journal of Physics: Conference Series*, volume 2161, page 012018. IOP Publishing, 2022.

[39] Ministero delle infrastrutture e dei trasporti. Mit: primo test auto a guida autonoma a traffico aperto su tangenziale di napoli. `www.mit.gov.it/comunicazione/news/mit-primo-test-auto-guida-autonoma-traffico-aperto`. [Accessed 09-12-2025].

[40] Jean-Emmanuel Deschaud. Kitti-carla: a kitti-like dataset generated by carla simulator. *arXiv preprint arXiv:2109.00892*, 2021.

[41] EM Roopa Devi, R Shanthakumari, T Harini, K Lokesh, and VS Anusuyaa. Detection of position falsification attacks in vanets using ensemble learning. In *2024 3rd International Conference on Artificial Intelligence For Internet of Things (AIIoT)*, pages 1–6. IEEE, 2024.

[42] Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A survey on safety-critical driving scenario generation—a methodological perspective. *IEEE Transactions on Intelligent Transportation Systems*, 24(7):6971–6988, 2023.

[43] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[44] Daniel Dworak, Filip Ciepiela, Jakub Derbisz, Izzat Izzat, Mateusz Komorkiewicz, and Mateusz Wójcik. Performance of lidar object detection deep learning architectures based on

artificially generated point cloud data from carla simulator. In *2019 24th International Conference on Methods and Models in Automation and Robotics (MMAR)*, pages 600–605. IEEE, 2019.

[45] Jakob Erdmann. Sumo's lane-changing model. `https://elib.dlr.de/102254/1/Springer-SUMOs_Lane_changing_model.pdf`, 2015. [Online; Accessed 27 Sep 2025].

[46] Stephan Escher, Markus Sontowski, Knut Berling, Stefan Köpsell, and Thorsten Strufe. How well can your car be tracked: Analysis of the european c-its pseudonym scheme. In *2021 IEEE 93rd vehicular technology conference (VTC2021-Spring)*, pages 1–6. IEEE, 2021.

[47] ETSI. Decentralized congestion control mechanisms for intelligent transport systems operating in the 5 ghz range. `https://www.etsi.org/deliver/etsi_ts/102600_102699/102687/01.02.01_60/ts_102687v010201p.pdf`. [Accessed 09-12-2025].

[48] ETSI. Intelligent transport system (its); vehicular communications; basic set of applications; collective perception service; release 2. `https://www.etsi.org/deliver/etsi_ts/103300_103399/103324/02.01.01_60/ts_103324v020101p.pdf`. [Accessed 09-12-2025].

[49] ETSI. Intelligent transport systems (its); facilities layer function; interference management zone message (imzm); release 2. `https://www.etsi.org/deliver/etsi_ts/103700_103799/103724/02.01.01_60/ts_103724v020101p.pdf`. [Accessed 09-12-2025].

[50] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; decen-

tralized environmental notification service; release 2. https://www.etsi.org/deliver/etsi_ts/103800_103899/103831/02.01.01_60/ts_103831v020101p.pdf. [Accessed 09-12-2025].

[51] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; facilities layer protocols and communication requirements for infrastructure services. https://www.etsi.org/deliver/etsi_ts/103300_103399/103301/01.03.01_60/ts_103301v010301p.pdf. [Accessed 09-12-2025].

[52] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/01.03.01_30/en_30263702v010301v.pdf. [Accessed 09-12-2025].

[53] ETSI. Intelligent transport systems (its); vehicular communications; basic set of applications; release 2. https://www.etsi.org/deliver/etsi_tr/102600_102699/102638/02.01.01_60/tr_102638v020101p.pdf. [Accessed 09-12-2025].

[54] ETSI. Vehicular communications; basic set of applications; cooperative awareness service; release 2. https://www.etsi.org/deliver/etsi_ts/103900_103999/103900/02.01.01_60/ts_103900v020101p.pdf. [Accessed 09-12-2025].

[55] ETSI. Etsi ts 103 324 - collective perception service (cps). https://www.etsi.org/deliver/etsi_ts/103300_103399/103324/02.01.01_60/ts_103324v020101p.pdf, 2023. [Accessed 13-08-2025].

[56] CCAM EU. Ai4ccam project. https://www.ai4ccam.eu/. [Accessed 09-12-2025].

[57] CCAM EU. Fame project - ccam knowledge base. https://www.connectedautomateddriving.eu/projects/. [Accessed 09-12-2025].

[58] CCAM EU. Sunrise project. https://ccam-sunrise-project.eu/. [Accessed 09-12-2025].

[59] CCAM EU. Synergies project. https://synergies-ccam.eu/. [Accessed 09-12-2025].

[60] Andreas Festag. Cooperative intelligent transport systems standards in europe. *IEEE communications magazine*, 52(12):166–172, 2014.

[61] Matt Franchi. Webots. hpc: A parallel robotics simulation pipeline for autonomous vehicles on high performance computing. *arXiv preprint arXiv:2108.00485*, 2021.

[62] Alexander Frötscher, Bernhard Monschiebl, Anastasios Drosou, Erol Gelenbe, Martin J Reed, and Mays Al-Naday. Improve cybersecurity of c-its road side infrastructure installations: the seriot-secure and safe iot approach. In *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pages 1–5. IEEE, 2019.

[63] Gazzetta Ufficiale gazzettaufficiale.it. Decreto smart road. https://www.gazzettaufficiale.it/eli/id/2018/04/18/18A02619/SG. [Accessed 01-07-2025].

[64] Christian Geller, Benedikt Haas, Amarin Kloeker, Jona Hermens, Bastian Lampe, Till Beemelmanns, and Lutz Eckstein. Carlos: An open, modular, and scalable simulation framework for the development and testing of software for c-its. In *2024*

IEEE Intelligent Vehicles Symposium (IV), pages 3100–3106. IEEE, 2024.

[65] ITS Georgia. History of intelligent transportation systems. `https://www.itsga.org/wp-content/uploads/2016/08/ITS-JPO-History-of-ITS.pdf`, 2019. [Accessed 12-08-2025].

[66] Thomas Gillespie. *Fundamentals of vehicle dynamics*. SAE international, 2021.

[67] Ellen F Grumert, Andreas Tapani, and Xiaoliang Ma. Characteristics of variable speed limit systems. *European transport research review*, 10:1–12, 2018.

[68] Hendrik-Jorn Günther, Raphael Riebl, Lars Wolf, and Christian Facchi. Collective perception and decentralized congestion control in vehicular ad-hoc networks. In *2016 IEEE Vehicular Networking Conference (VNC)*, pages 1–8. IEEE, 2016.

[69] Rodrigo Gutiérrez-Moreno, Rafael Barea, Elena López-Guillén, Javier Araluce, and Luis M Bergasa. Reinforcement learning-based autonomous driving at intersections in carla simulator. *Sensors*, 22(21):8373, 2022.

[70] Farah Haidar, Arnaud Kaiser, Brigitte Lonc, and Pascal Urien. C-its pki protocol: Performance evaluation in a real environment. In *2019 15th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, pages 52–55. IEEE, 2019.

[71] Mohammad Jan Haidari and Zeki Yetgin. Veins based studies for vehicular ad hoc networks. In *2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*, pages 1–7. IEEE, 2019.

[72] Badis Hammi, Yacine Mohamed Idir, Sherali Zeadally, Rida Khatoun, and Jamel Nebhen. Is it really easy to detect sybil attacks in c-its environments: a position paper. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):18273–18287, 2022.

[73] Badis Hammi, Jean-Philippe Monteuuis, and Jonathan Petit. Pkis in c-its: Security functions, architectures and projects: A survey. *Vehicular Communications*, 38:100531, 2022.

[74] Markus Hofbauer, Christopher B Kuhn, Goran Petrovic, and Eckehard Steinbach. Telecarla: An open source extension of the carla simulator for teleoperated driving research using off-the-shelf components. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 335–340. IEEE, 2020.

[75] M. Hossain and M. Abdel-Aty. A comprehensive review of connected vehicle technology in smart signal and speed control systems. *Sensors*, 22(1):1, 2022.

[76] Dongyao Jia, Jie Sun, Anshuman Sharma, Zuduo Zheng, and Bingyi Liu. Integrated simulation platform for conventional, connected and automated driving: A design from cyber–physical systems perspective. *Transportation Research Part C: Emerging Technologies*, 124:102984, 2021.

[77] Mohannad Jooriah, Daryna Datsenko, João Almeida, Ana Sousa, João Silva, and Joaquim Ferreira. A co-simulation platform for v2x-based cooperative driving automation systems. In *2024 IEEE Vehicular Networking Conference (VNC)*, pages 227–230. IEEE, 2024.

[78] A. Kalašová, K. Čulík, P. Fiľo, and L. Šarkan. Comparative analysis of traffic monitoring technologies in urban environments. *Future Transportation*, 7(2):59, 2025.

[79] F. Kamran and S. Hassan. An adaptive traffic signal control in a connected vehicle environment: A systematic review. *Journal of Advanced Transportation*, 2017:1–20, 2017.

[80] Alžbeta Kanáliková, Mária Franeková, and Emília Bubeníková. Trends in the area of security within c2c communications. *Annals of the Faculty of Engineering Hunedoara*, 17(1):181–188, 2019.

[81] Junhee Kang, Sehyun Tak, and Sungjin Park. Analyzing the impact of c-its services on driving behavior: A case study of the daejeon–sejong c-its pilot project in south korea. *Sustainability*, 15(16):12655, 2023.

[82] Yue Kang, Hang Yin, and Christian Berger. Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments. *IEEE Transactions on Intelligent Vehicles*, 4(2):171–185, 2019.

[83] Tamás Márton Kazár1a, Zsombor Pethő, Gábor Vida, and Árpád Török. Simulation of road traffic accidents related to adas systems in prescan. *The First Conference on ZalaZONE Related R&I Activities of Budapest University of Technology and Economics 2022*, 2022.

[84] Sakib Mahmud Khan, Kakan C Dey, and Mashrur Chowdhury. Real-time traffic state estimation with connected vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 18(7):1687–1699, 2017.

[85] Matthijs Klomp, Mats Jonasson, Leo Laine, Leon Henderson, Enrico Regolin, and Stefan Schumi. Trends in vehicle motion control for automated driving on public roads. *Vehicle System Dynamics*, 57(7):1028–1061, 2019.

[86] Djibrilla Amadou Kountché, Jean-Marie Bonnin, and Houda Labiod. The problem of privacy in cooperative intelligent transportation systems (c-its). In *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 482–486. IEEE, 2017.

[87] Gergely Attila Kovács and László Bokor. Integrating artery and simu5g: A mobile edge computing use case for collective perception-based v2x safety applications. In *2022 45th international conference on telecommunications and signal processing (TSP)*, pages 360–366. IEEE, 2022.

[88] Radosław Kucharski and Andrzej Drabicki. Assessment and comparison of macroscopic traffic flow models for motorways. *Applied Sciences*, 11(21):9914, 2021.

[89] L. Li, E. I. Vlahogianni, and Y. Wang. A review of connected vehicle-based adaptive signal control: Data, models, and control strategies. *Digital Transportation and Safety*, 1(1):1–20, 2022.

[90] Xun Li and Yinhai Wang. A review of hybrid physics-based machine learning approaches for traffic state estimation. *ITSL Intelligent Transportation Systems*, 2023.

[91] Xiruo Liu, Lily Yang, Ignacio Alvarez, Kathiravetpillai Sivanesan, Arvind Merwaday, Fabian Oboril, Cornelius Buerkle, Manoj Sastry, and Leonardo Gomes Baltar. Misov: Misbehavior detection for collective perception services in vehicular communications. In *2021 IEEE intelligent vehicles symposium (IV)*, pages 369–376. IEEE, 2021.

[92] Ying Liu, Hongwei Xue, Weichao Zhuang, Fa'an Wang, Liwei Xu, and Guodong Yin. Ct2-mds: Cooperative trust-aware tolerant misbehaviour detection system for connected

and automated vehicles. *IET Intelligent Transport Systems*, 16(2):218–231, 2022.

[93] Zongwei Liu, Wang Zhang, and Fuquan Zhao. Impact, challenges and prospect of software-defined vehicles. *Automotive Innovation*, 5(2):180–194, 2022.

[94] Nikita Lyamin, Denis Kleyko, Quentin Delooz, and Alexey Vinel. Real-time jamming dos detection in safety-critical v2v c-its using data mining. *IEEE Communications Letters*, 23(3):442–445, 2019.

[95] Nikita Lyamin, Alexey Vinel, Dieter Smely, and Boris Bellalta. Etsi dcc: Decentralized congestion control in c-its. *IEEE Communications Magazine*, 56(12):112–118, 2018.

[96] Sassi Maaloul, Hasnaâ Aniss, Mohamed Kassab, and Marion Berbineau. Classification of c-its services in vehicular environments. *IEEE Access*, 9:117868–117879, 2021.

[97] Mehdi Maleki, Mateen Malik, Peter Folkesson, Behrooz Sangchoolie, and Johan Karlsson. Modeling and evaluating the effects of jamming attacks on connected automated road vehicles. In *2022 IEEE 27th Pacific Rim International Symposium on Dependable Computing (PRDC)*, pages 12–23, 2022.

[98] Johann Marquez-Barja, Bart Lannoo, Dries Naudts, Bart Braem, Carlos Donato, Vasilis Maglogiannis, Siegfried Mercelis, Rafael Berkvens, Peter Hellinckx, Maarten Weyn, et al. Smart highway: Its-g5 and c2vx based testbed for vehicular communications in real environments enhanced by edge/cloud technologies. In *EuCNC2019, the European Conference on Networks and Communications*. IEEE, 2019.

[99] Mathworks. Vehicle body 3dof. `https://it.mathworks.com/help/vdynblks/ref/vehiclebody3dof.html`, 2019. [Online; Accessed 27 Sep 2025].

[100] Vicente Milanés and Steven E Shladover. Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data. *Transportation Research Part C: Emerging Technologies*, 48:285–300, 2014.

[101] Mobility and transport. Cooperative, connected and automated mobility (CCAM). `https://transport.ec.europa.eu/transport-themes/smart-mobility/cooperative-connected-and-automated-mobility-ccam_en`. [Accessed 19-08-2025].

[102] Jean Tshibangu Muabila, Sebti Mouelhi, Patrick Leserf, and Amar Ramdan-Cherif. Striving for urban traffic optimization: Eclipse mosaic-backed predictive iov applications. In *International Conference on Computer and Communication Engineering*, pages 197–208. Springer, 2024.

[103] Emmanuel Mumba, Abubakar Sulaiman Gezawa, and Chibiao Liu. Enhanced autonomous driving within webots simulation for student experiments. *Intelligent Service Robotics*, pages 1–21, 2025.

[104] Gordon Frank Newell. A simplified car-following theory: a lower order model. *Transportation Research Part B: Methodological*, 36(3):195–205, 2002.

[105] DR Niranjan, BC VinayKarthik, et al. Deep learning based object detection model for autonomous driving research using carla simulator. In *2021 2nd international conference on smart electronics and communication (ICOSEC)*, pages 1251–1258. IEEE, 2021.

[106] ns 3. ns-3 is a discrete-event network simulator for internet systems. https://www.nsnam.org/. [Accessed 09-12-2025].

[107] João Oliveira, Emanuel Vieira, João Almeida, Joaquim Ferreira, and Paulo C Bartolomeu. A maneuver coordination analysis using artery v2x simulation framework. *Electronics*, 13(23):4813, 2024.

[108] Omnet++. Omnet++. https://omnetpp.org/. [Accessed 09-12-2025].

[109] Fang Pan, Xuan Di, Zuduo Zheng, and Henry X. Liu. A fundamental diagram based hybrid framework for traffic flow estimation and prediction by combining a markovian model with deep learning. *Transportation Research Part C: Emerging Technologies*, 161:104565, 2024.

[110] Stefanos Pasios and Nikos Nikolaidis. Carla2real: A tool for reducing the sim2real appearance gap in carla simulator. *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[111] Michał Pietruch, Andrzej Młyniec, and Andrzej Wetula. An overview and review of testing methods for the verification and validation of adas, active safety systems, and autonomous driving. *Mining–Informatics, Automation and Electrical Engineering*, 58(1):19–27, 2020.

[112] Postman. Postman. https://www.postman.com/. [Accessed 09-12-2025].

[113] Meenakshi Prabhakar, Valenteena Paulraj, Joshuva Arockia Dhanraj, Seenu Nagarajan, Dhusyant Arumukam Karthi Kannappan, and Adithyaa Hariharan. Design and simulation of an automated guided vehicle through webots for isolated covid-19 patients in hospitals. In *2020 IEEE 4th Conference*

on Information & Communication Technology (CICT), pages 1–5. IEEE, 2020.

[114] Robert Protzmann, Karl Schrab, Moritz Schweppenhäuser, and Ilja Radusch. Implementation of a perception module for smart mobility applications in eclipse mosaic. In *SUMO Conference Proceedings*, volume 3, pages 199–214, 2022.

[115] H. Rakha, K. Ahn, J. Du, and M. Farag. Quantifying the impact of cellular vehicle-to-everything (c-v2x). Technical report, Virginia Tech Transportation Institute, 2023.

[116] Felix Rampf, Georgios Grigoropoulos, Patrick Malcolm, Andreas Keler, and Klaus Bogenberger. Modelling autonomous vehicle interactions with bicycles in traffic simulation. *Frontiers in Future Transportation*, 3:894148, 2023.

[117] Marco Rapelli, Francesco Raviglione, and Claudio Casetti. Oscar: An etsi-compliant c-its stack for field-testing with embedded hardware devices. In *2024 22nd Mediterranean Communication and Computer Networking Conference (MedComNet)*, pages 1–4. IEEE, 2024.

[118] Francesco Raviglione, CM Risma Carletti, Marco Malinverno, Claudio Casetti, and Carla-Fabiana Chiasserini. ms-van3t: An integrated multi-stack framework for virtual validation of v2x communication and services. *Computer Communications*, 217:70–86, 2024.

[119] Raphael Riebl, Hendrik-Jörn Günther, Christian Facchi, and Lars Wolf. Artery: Extending veins for vanet applications. In *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 450–456. IEEE, 2015.

[120] Giedre Sabaliauskaite, Jin Cui, Lin Shen Liew, and Fengjun Zhou. Integrated safety and cybersecurity risk analysis of cooperative intelligent transport systems. In *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*, pages 723–728. IEEE, 2018.

[121] Saif Sabeeh and Krzysztof Wesołowski. Congestion control in autonomous resource selection of cellular-v2x. *IEEE Access*, 11:7450–7460, 2023.

[122] Karl Schrab, Maximilian Neubauer, Robert Protzmann, Ilja Radusch, Stamatis Manganiaris, Panagiotis Lytrivis, and Angelos J Amditis. Modeling an its management solution for mixed highway traffic with eclipse mosaic. *IEEE Transactions on Intelligent Transportation Systems*, 24(6):6575–6585, 2022.

[123] Karl Schrab, Moritz Schweppenhäuser, Robert Protzmann, Kay Massow, and Ilja Radusch. Leveraging eclipse mosaic for modeling and analyzing ride-hailing services. *arXiv preprint arXiv:2404.07547*, 2024.

[124] Karl Schrab, Moritz Schweppenhäuser, Robert Protzmann, Kay Massow, and Ilja Radusch. Modeling and analyzing ride-hailing services with eclipse mosaic. *Advances in Transdisciplinary Engineering*, 27:349, 2024.

[125] Moritz Schweppenhäuser, Robert Hilbrich, and Michael Behrisch. Leveraging eclipse sumo and eclipse mosaic: Unleashing the power of digital twins for efficient urban mobility management in berlin. *EclipseCon 2023*, 2023.

[126] Hichem Sedjelmaci, Makhlouf Hadji, and Nirwan Ansari. Cyber security game for intelligent transportation systems. *Ieee Network*, 33(4):216–222, 2019.

[127] Fetulhak Abdurahman Shewajo, Abdelwahab Boualouache, Sidi Mohammed Senouci, Ines El-Korbi, Bouziane Brik, and Kinde Anlay Fante. Integrating blockchain technology with pki for secure and interoperable communication in 5g and beyond vehicular networks. In *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*, pages 998–1001. IEEE, 2024.

[128] Ivo Silva, Hélder Silva, Fabricio Botelho, and Cristiano Pendão. Realistic 3d simulators for automotive: A review of main applications and features. *Sensors (Basel, Switzerland)*, 24(18):5880, 2024.

[129] Pranav Kumar Singh, Shivam Gupta, Ritveeka Vashistha, Sunit Kumar Nandi, and Sukumar Nandi. Machine learning based approach to detect position falsification attack in vanets. In *International Conference on Security & Privacy*, pages 166–178. Springer, 2019.

[130] Christoph Sommer, David Eckhoff, Alexander Brummer, Dominik S Buse, Florian Hagenauer, Stefan Joerer, and Michele Segata. Veins: The open source vehicular network simulation framework. In *Recent advances in network simulation: the OMNeT++ environment and its ecosystem*, pages 215–252. Springer, 2019.

[131] Santhosh Kumar Sripathi Venkata Naga, Rajkumar Yesuraj, Selvi Munuswamy, and Kannan Arputharaj. A comprehensive survey on certificate-less authentication schemes for vehicular ad hoc networks in intelligent transportation systems. *Sensors*, 23(5):2682, 2023.

[132] Vitaly G Stepanyants and Aleksandr Y Romanov. A survey of integrated simulation environments for connected automated vehicles: Requirements, tools, and architecture. *IEEE Intelligent Transportation Systems Magazine*, 16(2):6–22, 2023.

[133] Stevan Stević, Momčilo Krunić, Marko Dragojević, and Nives Kaprocki. Development and validation of adas perception application in ros environment integrated with carla simulator. In *2019 27th Telecommunications Forum (TELFOR)*, pages 1–4. IEEE, 2019.

[134] Andrea Stocco, Brian Pulfer, and Paolo Tonella. Mind the gap! a study on the transferability of virtual versus physical-world testing of autonomous driving systems. *IEEE Transactions on Software Engineering*, 49(4):1928–1940, 2022.

[135] Tram Thi Minh Tran, Callum Parker, and Martin Tomitsch. A review of virtual reality studies on autonomous vehicle–pedestrian interaction. *IEEE Transactions on Human-Machine Systems*, 51(6):641–652, 2021.

[136] Martin Treiber and Arne Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 227:228, 2013.

[137] European Union. Eu road safety: Towards "vision zero". `https://cinea.ec.europa.eu/publications/digital-publications/eu-road-safety-towards-vision-zero_en`. [Accessed 10-09-2025].

[138] European Union. Regulation - 2019/2144 - EN - EUR-Lex — eur-lex.europa.eu. `https://eur-lex.europa.eu/eli/reg/2019/2144/oj/eng`, 2019. [Accessed 11-08-2025].

[139] Rens Wouter Van Der Heijden, Stefan Dietzel, Tim Leinmüller, and Frank Kargl. Survey on misbehavior detection in cooperative intelligent transportation systems. *IEEE Communications Surveys & Tutorials*, 21(1):779–811, 2018.

[140] Vector. Vn4610 - 802.11p/can (fd)/gnss interface. `https://www.vector.com/us/en/products/products-a-z/hardware/network-interfaces/vn4610/`. [Accessed 09-12-2025].

[141] Christian Vitale, Nikos Piperigkos, Christos Laoudias, Georgios Ellinas, Jordi Casademont, Josep Escrig, Andreas Kloukiniotis, Aris S Lalos, Konstantinos Moustakas, Rodrigo Diaz Rodriguez, et al. Caramel: results on a secure architecture for connected and autonomous vehicles detecting gps spoofing attacks. *EURASIP Journal on Wireless Communications and Networking*, 2021(1):115, 2021.

[142] Tamás Wágner, Tamás Ormándi, Tamás Tettamanti, and István Varga. Spat/map v2x communication between traffic light and vehicles and a realization with digital twin. *Computers and Electrical Engineering*, 106:108560, 2023.

[143] Chia-Sui Wang, Ding-Yu Liu, and Kuei-Shu Hsu. Simulation and application of cooperative driving sense systems using prescan software. *Microsystem Technologies*, 27(4):1201–1210, 2021.

[144] Jian Wang, Yameng Shao, Yuming Ge, and Rundong Yu. A survey of vehicle to everything (V2X) testing. *Sensors*, 19(2):334, 2019.

[145] Jingwen Wang, Ivan Topilin, Anastasia Feofilova, Mengru Shao, and Yadong Wang. Cooperative intelligent transport systems: The impact of c-v2x communication technologies on

road safety and traffic efficiency. *Sensors (Basel, Switzerland)*, 25(7):2132, 2025.

[146] Y. Wang, S. Li, K. Wang, Y. Liu, and K. Li. Traffic state estimation with stochastic three-detector modeling considering heteroscedasticity. *Journal of Transportation Engineering, Part A: Systems*, 151, 2025.

[147] Cohda Wireless. Cohda mk5 obu. `https://cohdawireless.com/solutions/hardware/mk5-obu/`. [Accessed 09-12-2025].

[148] Tong Wu, Xuefei Ning, Wenshuo Li, Ranran Huang, Huazhong Yang, and Yu Wang. Physical adversarial attack on vehicle detector in the carla simulator. *arXiv preprint arXiv:2007.16118*, 2020.

[149] Lan Yang, RunMin Wang, XiangMo Zhao, ZhiGang Xu, and YiPeng Yang. Cavtest: A closed connected and automated vehicles test field of chang'an university in china. *SAE International Journal of Connected and Automated Vehicles*, 4(12-04-04-0032):423–435, 2021.

[150] Shi Ye, Tiantian Chen, Oscar Oviedo-Trespalacios, Yasir Ali, Taeho Oh, and Inhi Kim. How do the drivers react to different c-v2x-based communication conditions in dilemma zones? a driving simulator study. *Accident Analysis & Prevention*, 221:108208, 2025.

[151] Takahito Yoshizawa, Dave Singelée, Jan Tobias Muehlberg, Stephane Delbruel, Amir Taherkordi, Danny Hughes, and Bart Preneel. A survey of security and privacy issues in v2x communication systems. *ACM Computing Surveys*, 55(9):1–36, 2023.

[152] Jiawei Zhang, Cheng Chang, Zimin He, Wenqin Zhong, Danya Yao, Shen Li, and Li Li. Cavsim: A microscopic traffic sim-

ulator for evaluation of connected and automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 24(9):10038–10054, 2023.

[153] Xinhai Zhang, Jianbo Tao, Kaige Tan, Martin Törngren, José Manuel Gaspar Sánchez, Muhammad Rusyadi Ramli, Xin Tao, Magnus Gyllenhammar, Franz Wotawa, Naveen Mohan, et al. Finding critical scenarios for automated driving systems: A systematic literature review. *arXiv preprint arXiv:2110.08664*, 2021.

[154] L. Zhao and B. Yu. Traffic state estimation using data-driven models: A review. *Journal of Big Data Analytics in Transportation*, 5:1–20, 2023.

[155] Hong Zhong, Fan Yang, Lu Wei, Jing Zhang, Chengjie Gu, and Jie Cui. Dataset for evaluation of ddos attacks detection in vehicular ad-hoc networks. In *International Conference on Wireless Algorithms, Systems, and Applications*, pages 249–260. Springer, 2022.