Simulation Modelling Practice and Theory 34 (2013) 99–125

Contents lists available at SciVerse ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

iTETRIS: A modular simulation platform for the large scale evaluation of cooperative ITS applications

Michele Rondinone[a,*], Julen Maneros[b], Daniel Krajzewicz[c], Ramon Bauza[a], Pasquale Cataldi[d], Fatma Hrizi[d], Javier Gozalvez[a,*], Vineet Kumar[e], Matthias Röckl[f], Lan Lin[e], Oscar Lazaro[g], Jérémie Leguay[h], Jérôme Härri[d], Sendoa Vaz[b], Yoann Lopez[h], Miguel Sepulcre[a], Michelle Wetterwald[d], Robbin Blokpoel[i], Fabio Cartolano[j]

[a] Uwicore Laboratory, University Miguel Hernández of Elche, Avenida de la Universidad s/n, 03202 Elche, Alicante, Spain
[b] CBT Comunicación & Multimedia, Carretera Asua 6, 48930 Getxo, Spain
[c] Institute of Transportation Systems, German Aerospace Center (DLR), Rutherfordstr. 2, 12489 Berlin, Germany
[d] Mobile Communication Department, Institut Eurecom, 2229 Route des Crêtes, 06560 Sophia-Antipolis, France
[e] Hitachi Europe SAS, 1503 Route des Dolines, 06560 Valbonne, Sophia-Antipolis, France
[f] Institute of Communications and Navigation, German Aerospace Center (DLR), Münchner Str. 20, 82234 Weßling, Germany
[g] Innovalia Association, Calle Rodriguez Arias, 48008 Bilbao, Spain
[h] Thales Communications, 160 Boulevard de Valmy, 92706 Colombes, France
[i] Peek Traffic, Basicweg 16, 3821 BR Amersfoort, Netherlands
[j] Comune di Bologna, Piazza Maggiore 6, 40124 Bologna, Italy

ARTICLE INFO

Article history:
Received 6 May 2012
Received in revised form 18 December 2012
Accepted 21 January 2013
Available online 15 March 2013

Keywords:
Cooperative ITS systems
Simulation platform
Software architecture modularity
Large scale evaluations

ABSTRACT

Cooperative ITS systems are expected to improve road traffic safety and efficiency, and provide infotainment services on the move, through the dynamic exchange of messages between vehicles, and between vehicles and infrastructure nodes. The complexity of cooperative ITS systems and the interrelation between its components requires their extensive testing before deployment. The lack of simulation platforms capable to test, with high modelling accuracy, cooperative ITS systems and applications in large scale scenarios triggered the implementation of the EU-funded iTETRIS simulation platform. iTETRIS is a unique open source simulation platform characterized by a modular architecture that allows integrating two widely adopted traffic and wireless simulators, while supporting the implementation of cooperative ITS applications in a language-agnostic fashion. This paper presents in detail the iTETRIS simulation platform, and describes its architecture, standard compliant implementation, operation and new functionalities. Finally, the paper demonstrates iTETRIS large scale cooperative ITS evaluation capabilities through the implementation and evaluation of cooperative traffic congestion detection and bus lane management applications. The detailed description and implemented examples provide valuable information on how to use and exploit iTETRIS simulation potential.

1. Introduction

Road mobility significantly contributes to the society's economy and welfare. However, the increase in number of vehicles and the higher mobility frequency are creating new problems and challenges that need to be addressed to ensure a safe, sus-

* Corresponding authors. Tel.: +34 96 665 8955; fax: +34 96 665 8903.
E-mail addresses: mrondinone@umh.es (M. Rondinone), j.gozalvez@umh.es (J. Gozalvez).

1569-190X/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.simpat.2013.01.007

tainable and efficient road mobility. To address these challenges, numerous Intelligent Transportation Systems (ITS)[1] solutions have been developed and deployed over the past years (e.g. video cameras, road sensors, etc.). However, novel active safety and dynamic traffic management technologies are needed. In this context, a major technological breakthrough will be cooperative ITS systems. Cooperative ITS systems will provide new opportunities for more intelligent Smart Mobility services through the ubiquitous and continuous exchange of information between vehicles (Vehicle-to-Vehicle, V2V), and between vehicles and road infrastructure nodes (Vehicle-to-Infrastructure, V2I). The exchange of information such as the vehicles' position and speed will allow detecting potential road dangers and traffic congestions with sufficient time to react and avoid them. In addition, cooperative ITS communications will offer new means for the provision of Future Internet services on road transport. Cooperative ITS systems are being fostered by international standardization efforts at the 5.8–5.9 GHz band: ETSI Technical Committee on ITS (ITS G5 [1]), IEEE 802.11p and WAVE/1609 standards, and ISO TC204 WG16 (also referred to as CALM). Despite the importance of 5.8–5.9 GHz standards, heterogeneous communications and networking solutions exploiting the capabilities of cellular, WiMAX and DVB technologies will also have a major impact on the provision of cooperative ITS services. Authorities, automotive and telecommunications sectors have also recognized internationally the industrial strategic importance of cooperative ITS systems giving rise to the establishment of initiatives such as the iMobility Forum, IntelliDrive or the past Car-2-Car Communications Consortium (C2C-CC).

The complex, large scale and highly dynamic nature of cooperative ITS systems introduces significant challenges before efficient and reliable solutions and applications can be deployed. The existence of adequate experimentation facilities is therefore needed, and will provide significant opportunities for a more effective and efficient design of cooperative ITS systems. Field Operational Tests (FOTs) have recently demonstrated the effectiveness and potential of cooperative ITS systems under small scale scenarios in localized areas and with a reduced number of vehicles. However, the possibility to test large scale traffic experimentation scenarios (e.g. at city level and during extended time periods) is crucial for the evaluation of cooperative traffic management solutions and other info-mobility and infotainment services. In this context, simulation tools represent the most adequate and viable alternative. Simulating cooperative ITS systems requires the capability to model vehicular mobility and wireless communications, in addition to implement and execute novel cooperative ITS applications. There are currently various platforms for modelling vehicular traffic and wireless communications, but the study of cooperative ITS systems requires modelling the interaction between them. In this context, this paper presents iTETRIS, a unique standard compliant and open source simulation platform developed under the European FP7 Program (iTETRIS: an Integrated Wireless and Traffic Platform for Real-Time Road Traffic Management Solutions, http://ict-itetris.eu/) for investigating cooperative ITS systems and services. iTETRIS integrates and extends SUMO and ns-3, two widely referenced open source platforms for vehicular mobility and wireless communications simulations, and allows the implementation of cooperative ITS applications in various programming languages. Its open source nature and modular architecture facilitate the future expansion of the platform. The platform is also capable to simulate large scale scenarios, which represents a very appealing feature for the investigation of cooperative ITS systems. To demonstrate iTETRIS' capabilities, the paper will also present two cooperative ITS traffic management applications developed during the project, and implemented and tested using the iTETRIS platform. The analyzed applications will demonstrate the potential of cooperative ITS systems, and the value of the iTETRIS platform not only for the research community, but also for road operators and public authorities that need to efficiently design, plan, assess and optimize cooperative ITS applications before their deployment. Interested readers can download the iTETRIS' source code by accessing the iTETRIS' community webpage: http://www.ict-itetris.eu/10-10-10-community/.

The paper is organized as follows. Section 2 reports the state of the art on cooperative ITS simulation. Section 3 briefly describes ns-3 and SUMO. Section 4 outlines the ETSI Architecture for ITS Communications used as reference for the iTETRIS implementation. Section 5 presents the overall iTETRIS modular architecture and simulation approach, whereas Sections 6 and 7 describe the extensions realized over ns-3 and SUMO to meet the iTETRIS requirements. Section 8 specifies the functionalities of the iTETRIS' central control entity: the iCS (iTETRIS Control System). Section 9 highlights iTETRIS' advances in the simulation of cooperative ITS systems. Finally, Section 10 demonstrates the capability of iTETRIS to implement and evaluate cooperative ITS traffic management applications, and Section 11 draws the main conclusions derived from this work.

2. State of the art

An adequate modelling of cooperative ITS systems requires the interaction between vehicular mobility and wireless communications simulation processes. This is the case because a vehicle might change its route as a result of the reception of a wireless message, and such reception highly depends on the positions of the vehicle sender and receiver. Several studies have developed integrated simulation platforms with varying degrees of integration and modelling accuracy. For example, MoVES [2] presents a framework for parallel and distributed simulations based on a modular and layered modelling of both vehicular and wireless scenarios integrated with mobile applications. AutoMesh [3] includes a set of modules representing driving behavior and radio propagation, and connects them in a control loop able to reproduce their mutual influence. By using three-dimensional maps and digital elevation models, it is able to reproduce realistic radio propagation effects in urban areas. The VANET simulator [4] models the transmission and reception of messages and the vehicles' GPS position updates as a series of discrete events. These events are tied by mutual relationships ensuring the generation of new events upon their continuous execution. The GrooveSim tool [5] is a mobility and communications simulator tailored to assess the performance of geo-

[1] A list of the most significant abbreviations used in the paper can be found in Appendix A.

graphic routing in vehicular networks. GrooveSim includes various modular mobility, trip, communications, and traffic density models, and presents interesting features like visual playback of driving logs. In addition, it supports "hybrid" simulations integrating vehicular communications systems deployed in the real world and simulated vehicles. MoVES, AutoMesh, VANET and GrooveSim developed ad hoc implementations of wireless and/or mobility models. However, such implementations might need more validations before their trustfulness can be acknowledged. In fact, as works such as [6,7] claim, they lack extensive use and testing compared to realistic models generated and validated in dedicated open source projects and exploiting the contribution of thousands of users. In this context, the importance of realistically emulating vehicular mobility is discussed in [8], where the authors demonstrate that using inaccurate mobility models results in overestimating the performance of wireless communications protocols. Similar conclusions were found in [9] where the importance of properly modelling the radio channel effects is analyzed. Trying to improve realism and modelling accuracy to enable a correct study of cooperative ITS systems, some works propose to embed vehicular mobility models into validated network simulators. For example, [10] embeds into SWANS the Street Random Waypoint (STRAW) tool, which is able to parse real street map data and model complex intersection management using traffic lights and traffic signs. SWANS is a scalable and efficient network simulator built on top of the JiST (Java in Simulaton Time) platform (http://jist.ece.cornell.edu/). Similarly to [10,11] presents a collection of SWANS modules, called ASH (Application-aware SWANS with Highway mobility), to model customizable highway topologies, car following and lane changing models, as well as inter-vehicle Geocast data dissemination protocols allowing simulations of cooperative ITS applications. Using a similar approach, the network simulator NCTUns incorporates from its version 5.0 the support for road network construction and microscopic vehicle mobility models in a tight coupling with its wireless simulation [7]. Also, [12] extends the open source ns-3 network simulator (http://www.nsnam.org/) with a set of classes to realistically emulate the behavior of vehicles over highway scenarios including lane changing and car following models. The resulting ns-3 objects are fully configurable and give users the possibility to implement event handlers to generate network messages, or alter vehicles' mobility upon wireless receptions or vehicles' position updates. A fast feedback loop between wireless communications and vehicular mobility models allows these schemes to mutually interact in real-time. However, embedded integrated solutions require users and developers to have a complete knowledge of the network simulator platforms, and can result in certain difficulties to evolve their code or replace certain modules. A different integration approach is the direct coupling of separated traffic and wireless simulation platforms. This approach was adopted in [13] using CORSIM (http://mctrans.ce.ufl.edu/featured/tsis/version5/corsim.htm) and QualNet (http://www.scalable-networks.com/products/qualnet/), and in [14] using VISSIM (http://www.vissim.de/) and ns-2 (http://www.isi.edu/nsnam/ns/). ns-2 is the precursor open source tool to ns-3, and thanks to its acceptance in the research community it includes a large number of modelling components. However, QualNet, CORSIM and VISSIM are commercial platforms that, although ensuring higher modelling accuracy, provide less freedom or even compromise the possibility to integrate new cooperative ITS features over the solutions presented in [13,14]. In this context, TraNS (http://lca.epfl.ch/projects/trans) was the first attempt combining two independent open source traffic and wireless simulators, namely SUMO (http://sumo.sf.net/) and ns-2 [15]. More recently, the fully open source approach of TraNS was also adopted by the Veins Simulator [16] that combines the OMNET++ network simulator (http://www.omnetpp.org/) with SUMO. To allow the interaction, communication modules are implemented over both the interconnected platforms, and a manager entity is created in OMNET++ to send commands to SUMO (e.g. to impose a given driving behavior to a vehicle after a wireless message reception) and receive updates from it (e.g. new vehicles' positions) at regular time steps. The Online Vehicular Network Integrated Simulation (OVNIS) platform [17] is the most recently issued platform among those reported in this review. It presents the combination of SUMO and ns-3, and the inclusion of an ns-3 module implementing user-defined cooperative ITS applications. In the OVNIS platform, ns-3 is extended to be a "traffic aware network manager" not only able to simulate wireless transmission between vehicles according to the positions retrieved by SUMO, but also control the whole simulation process during its execution as well as the relative interactions between the connected blocks. TraNS, Veins and OVNIS present very interesting approaches but share a nonnegligible limitation: by assigning the simulation control to a manager entity that is included in one of the coupled simulators, the capability for evolution of the resulting platform cannot be totally ensured. In fact, if the simulator implementing the management and controlling tasks gets obsolete, its replacement in favor of a newer simulator would be challenging. In addition, all the tools require cooperative ITS application developers to integrate their applications into the existing simulation platforms, which requires becoming familiar with them. This increases the time for development, and therefore reduces the usability of the platform for cooperative ITS application developers and testers. In this context, iTETRIS moves a step beyond the current state of the art through an interaction between open source reference simulators that is independent from their respective implementations. As a result, it offers better evolution perspectives, and facilitates the potential substitution of one of its interconnected simulation platforms. In addition, it allows for a language-agnostic implementation and testing of cooperative ITS applications through the inclusion of a novel central and interfacing entity. Finally, a key feature of iTETRIS with respect to other cooperative ITS simulation platforms is that it is standard compliant with the ETSI Technical Committee on ITS Communications architecture [18].

3. iTETRIS' reference simulators

iTETRIS was created with the purpose to build on existing open source simulators, and avoid their tight integration. Such integration could hinder the evolution and use of iTETRIS taking into account the required multidisciplinary approach to design and deploy future cooperative ITS systems. As a result, commercial wireless and traffic simulation platforms were discarded, and an in-depth analysis of the available open source platforms was then realized.

3.1. Wireless simulation

To simulate wireless communications, the iTETRIS consortium analyzed the ns-2, ns-3 and OMNeT++ wireless simulators. The three platforms include a high number of communication libraries, and are widely used by the wireless communications and networking communities. Key factors for selecting the iTETRIS wireless platform were the available communication modules, the platform's stability and the community actively developing new functionalities, and what proved to be the most demanding requirement, the platform's scalability under large scale simulation scenarios. A lower physical layer modelling accuracy provides OMNET++ more scalability compared to ns-2 and ns-3. However, when the modelling of the physical layer in OMNeT++ is improved, its computation time and required memory resources considerably increase [19]. The capability of ns-2 and ns-3 to support large scale simulations was then compared in [20]. The conducted study showed that ns-2 had strong RAM memory requirements, and was not able to handle simulations with over 8000 nodes. On the other hand, ns-3 was capable to simulate scenarios with up to 20,000 nodes at the cost of long simulation times. However, the authors identified mechanisms to significantly reduce the simulation time by optimizing and simplifying the channel and interference modelling without compromising the accuracy of the simulation results. In this context, iTETRIS finally adopted ns-3 as its wireless reference simulator. ns-3 is a discrete event-driven communications network simulator that provides its code under the GNU General Public License (GPL). Compared to its predecessor ns-2, ns-3 presents a more modular architecture, as well as multi-channel and multi-technology support. In addition, ns-3 has been fully developed in C++ making use of programming solutions such as smart pointers, templates and object factories that highly ease the creation of new modules. Network entities are represented in ns-3 as *Nodes*. Nodes can include multiple applications, communication protocols and technologies. In fact, in ns-3 objects (or modules) are incrementally aggregated to a node. In this context, ns-3 *Channels* are classes defined to represent the effects generated by the (wired or wireless) communications between nodes. Channels are linked to classes called *NetDevices* modelling the Physical and Data Link layers of specific communication technologies installed on a node. The Network and Transport layers are modelled in ns-3 through the implementation of the IP communication stack. Moreover, several classes implementing applications such as *PacketSink*, *Ping*, or *UDPClient/Server* are available. Specific *Helper* classes are provided by ns-3 to create nodes, include over the nodes the required communication modules, interlink nodes through communication channels, and assign the addresses related to the installed NetDevices at MAC (Medium Access Control) and Network level. Helpers are also used to easily create simulation scenarios in the form of C++ programs to be compiled (instead of TCL scripts as in the case of ns-2).

3.2. Traffic simulation

In terms of open source traffic simulators, the iTETRIS consortium analyzed the VanetMobiSim (http://vanet.eurecom.fr/) and SUMO platforms. VanetMobiSim was developed as a tool to retrieve vehicular mobility patterns (featuring microscopic mobility modelling and real world road topologies) to study cooperative vehicular communications at the expense of its traffic modelling accuracy. On the other hand, SUMO was developed as a tool for the evaluation of pure traffic engineering solutions at large scale, and therefore offers a more complete and accurate solution than VanetMobiSim. SUMO was then selected as iTETRIS' reference vehicular mobility simulator. SUMO is a microscopic, space-continuous and time-discrete simulator. Like ns-3, its code is publicly available under the GNU GPL license. By only using standard C++ libraries, it provides high portability to different Windows and Linux platforms. SUMO is also highly interoperable with external supporting applications thanks to the adoption of XML data in many of its components. SUMO simulations are purely *microscopic* in the sense that each vehicle is modelled explicitly and individually, with a dedicated characterization in terms of mobility dynamics and route through the road network. SUMO supports the definition of different *Vehicle Types* characterized by distinct values of maximum speed, acceleration and length (among others). Interaction between vehicles is modelled based on the Krauß car-following model [21]. The road network is represented as single roads (edges), lanes and road intersections. The latter can be modelled to represent different transit rules (e.g. traffic lights or direction-based priority). The ability to manage road networks with more than 10,000 edges with relatively fast execution times makes SUMO suitable for large scale simulations [22]. Besides the traffic simulator itself, SUMO includes a number of additional tools and applications. For example, SUMO includes a graphical user interface to visualize road traffic mobility. Other tools allow generating synthetic road networks and traffic flows, or importing real road network representations from different sources and convert them into formats reusable in SUMO. In addition, SUMO can enrich the road network characterization by adding additional infrastructure information such as bus stops or inductive loop sensors. An important feature of SUMO is that it allows external applications to connect to the simulator through the use of an API called TraCI (Traffic Control Interface) that relies on a socket connection [23]. TraCI allows external applications to retrieve or modify values characterizing SUMO objects (e.g. vehicles' speed). SUMO outputs can be collected in variable time intervals as aggregated measurements per-vehicle, lane or road. All outputs are written into XML-type files, and can also be retrieved on-line through the TraCI interface.

4. ETSI ITS communications architecture

To increase its impact and maximize its usability, iTETRIS was designed to be aligned and compliant with international standards. In particular, iTETRIS has been designed to be compliant with ETSI's architecture for Intelligent Transport Systems Communications (ITSC) [18]. The ITSC defines four main communication sub-systems for the execution of cooperative ITS applications. A *Personal ITS sub-system* is a handheld communications device (e.g. a smartphone). A *Central ITS sub-system* is a Traffic Management Center (TMC) responsible for the centralized control of the road traffic. In order to disseminate road traffic information to vehicles or collect floating car data, a *Central ITS sub-system* can be connected to an 802.11p or ITS G5-based *Roadside ITS sub-system* (also referred to as Roadside Unit or RSU) or other infrastructure nodes (e.g. cellular, WiMAX or DVB base stations). Finally, a *Vehicle ITS sub-system* is a connected vehicle capable to communicate with other vehicles and the infrastructure nodes, and execute cooperative ITS applications.

The ITSC communication sub-systems implement the architecture illustrated in Fig. 1. Cooperative ITS applications can be supported by various radio access technologies. The IEEE 802.11p standard was specifically developed to enable reliable communications between vehicles and with RSUs, and has been adapted at European level through the ETSI ITS G5 standard [1]. Although 802.11p or ITS G5 are expected to be the dominant communications technology for cooperative ITS communications and applications, other technologies could also support certain cooperative ITS applications. As a result, the *Access Technologies* layer includes a variety of communications technologies enabling short range, broadcast and cellular-type communications. The *Transport & Network* layer includes two different protocol stacks. The GeoNetworking or Car-to-Car (C2C) Stack implements specific addressing schemes, georouting and transport protocols based on ITS G5. The IP Stack contains pre-existing TCP/UDP transport and IP networking protocols. The *Facilities* layer includes a set of common functionalities and data structures to support cooperative ITS applications and communications. They can be classified into *Application Support*, *Information Support* and *Communication Support* facilities. An important Facility is the Local Dynamic Map (LDM), which stores and manages the dynamic information characterizing the local neighborhood of a vehicle or RSU. This information can be collected through received cooperative messages or on-board sensors. The information stored in the LDM can be used by the cooperative ITS *Applications* that exploit the underlying functionalities to provide road safety, traffic management and infotainment services. The vertical *Management* layer is responsible for monitoring and management functionalities. For example, it coordinates the cross-layer exchange of information, and determines the optimal mapping of cooperative ITS applications onto the available set of radio access technologies, transport and network protocols. Finally, the *Security* layer provides security services for the complete communications stack in order to prevent external attacks, guaranteeing the user's privacy, and ensuring secure and trustworthy exchange of information.

5. iTETRIS architecture

5.1. Platform architecture

iTETRIS supports the simulation of cooperative ITS applications running on TMCs, individual connected vehicles or RSUs. iTETRIS does not model the communications backbone linking the TMC to communications infrastructure units. As a result,

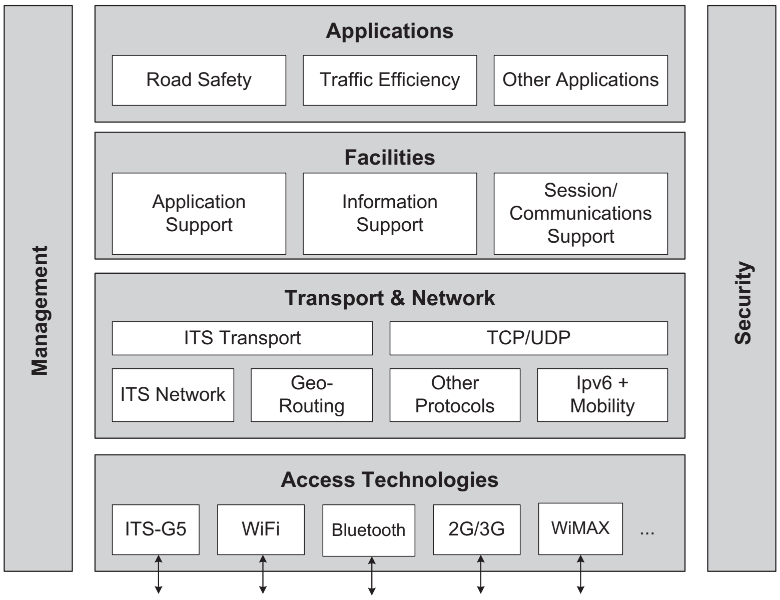

Fig. 1. ETSI ITS Architecture for Intelligent Transport Systems Communications (ITSC).

the iTETRIS representation of a TMC is only concerned with the execution of cooperative ITS applications, and does not require its modelling either in SUMO or ns-3. On the other hand, vehicles and RSUs wirelessly exchange information with other nodes, and therefore need to be represented in ns-3. Vehicles are also represented in SUMO to simulate their mobility. To handle the representation of a node over different platforms, iTETRIS implements a new central block referred to as *iCS* (*iTETRIS Control System*). The iCS handles SUMO and ns-3 interaction, in addition to preparing, triggering, coordinating and controlling the execution of iTETRIS simulations. The resulting iTETRIS architecture is represented in Fig. 2 that also maps the real world aspects to be modelled and simulated over the distinct blocks of the platform. Cooperative ITS applications running on vehicles, RSUs or TMCs are implemented in external blocks referred to as *iTETRIS Applications* (*iAPPs* in Fig. 2). SUMO models and simulates transportation aspects like vehicles' mobility at microscopic level, road intersections transit policies, pollutant and noise emissions produced by the vehicles, as well as their fuel consumption. It also supports detailed representations of real world large scale traffic scenarios in terms of road networks, traffic demands and traffic lights management. ns-3 supports accurate and realistic modelling and simulation of wireless transmissions for cooperative ITS systems in heterogeneous communications scenarios. As discussed in the next sections, ns-3 includes suitable models to emulate the radio propagation effects and to reproduce functionalities and protocols for every layer of the communications protocol stack. An important feature is the fact that iTETRIS is aligned with the ETSI ITSC architecture (white blocks in Fig. 2). As a result, ns-3 implements models for all the communications-related ITSC layers; an exception is made for the security layer that is not included in the initial iTETRIS release, but could be easily integrated. The iCS provides some supporting functionalities for the cooperative ITS applications implemented on the iAPP. Consequently, the implementation of the ITSC Facilities has been split between ns-3 and iCS. In particular, the Facilities more closely related to cooperative ITS applications (and thereby requiring a higher interaction with the iAPP) are implemented on the iCS (*iCS Facilities* in Fig. 2), while those needed to support communication sessions have been implemented in ns-3 (*ns-3 Facilities* in Fig. 2). This implementation approach results in that the iCS and ns-3 do not have to call from an external block the Facilities needed for their internal operations, which significantly reduces the exchange of messages between ns-3 and the iCS, the required computing resources, and the simulation time.

For each tested cooperative ITS application, only one implementation instance (iAPP) has to be present in iTETRIS. This instance is shared by all the nodes running the application. The modular architecture depicted in Fig. 2 permits that cooperative ITS applications can be defined and implemented in a language-agnostic fashion, while ns-3 and SUMO can be separately and independently extended with new features. All modules interact through the iCS central unit through a set of implemented open interfaces linking the iCS with the other iTETRIS building blocks. This approach increases modularity, and allows easily updating or replacing any of the iTETRIS modules without interfering with the others (in case of replacement, only the open interfaces will need to be re-programmed). iTETRIS interfaces its blocks simply through IP-based

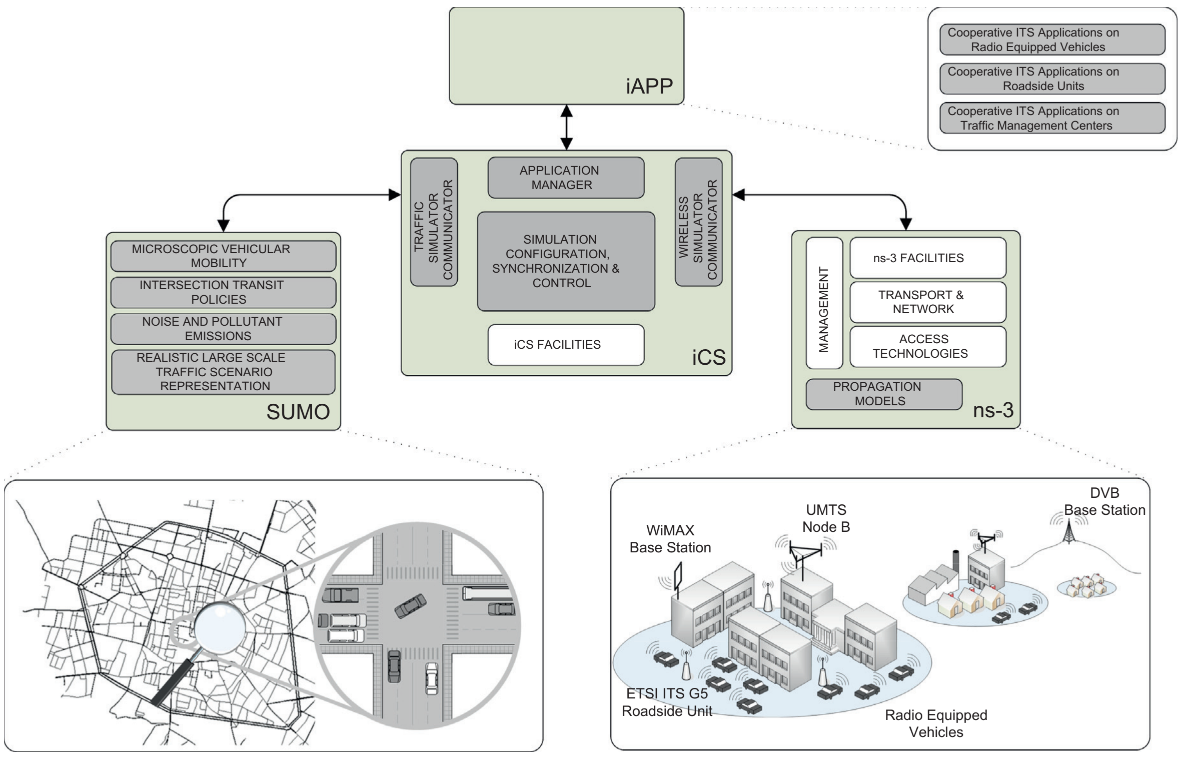
iAPP
Cooperative ITS Applications on Radio Equipped Vehicles
Cooperative ITS Applications on Roadside Units
Cooperative ITS Applications on Traffic Management Centers
APPLICATION MANAGER
TRAFFIC SIMULATOR COMMUNICATOR
SIMULATION CONFIGURATION, SYNCHRONIZATION & CONTROL
WIRELESS SIMULATOR COMMUNICATOR
iCS FACILITIES
iCS
MICROSCOPIC VEHICULAR MOBILITY
INTERSECTION TRANSIT POLICIES
NOISE AND POLLUTANT EMISSIONS
REALISTIC LARGE SCALE TRAFFIC SCENARIO REPRESENTATION
SUMO
MANAGEMENT
ns-3 FACILITIES
TRANSPORT & NETWORK
ACCESS TECHNOLOGIES
PROPAGATION MODELS
ns-3
WiMAX Base Station
UMTS Node B
DVB Base Station
ETSI ITS G5 Roadside Unit
Radio Equipped Vehicles

Fig. 2. iTETRIS architecture.

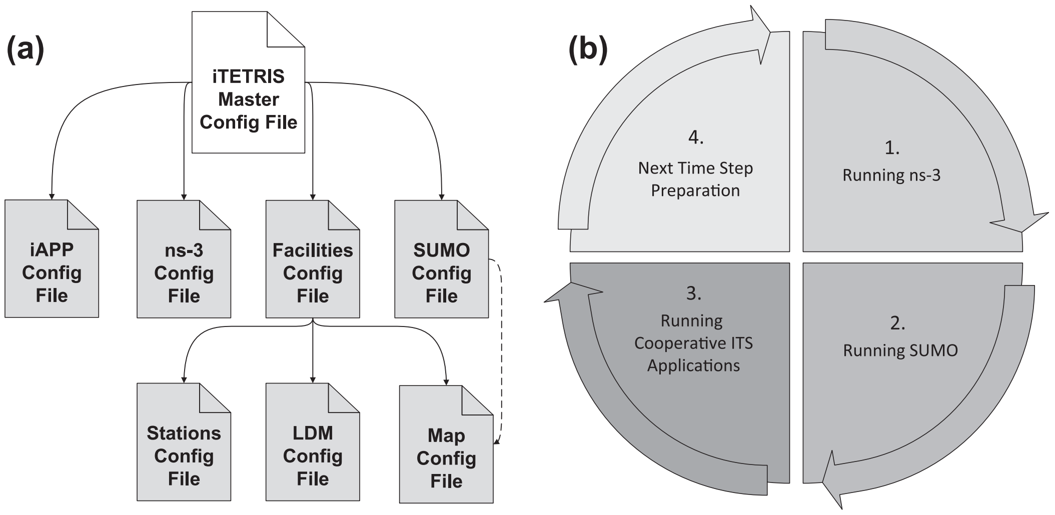

Fig. 3. iTETRIS' configuration files hierarchy (a) and run-time loop iteration (b).

sockets. In this context, the iCS can communicate with the other blocks by just specifying on which IP address and port the SUMO, ns-3 and iAPP blocks have to listen to. A "client/server" association is adopted with the iCS always acting as controller. For this purpose, the iCS implements three internal components that handle the communications with the rest of the iTETRIS blocks: the *Traffic Simulator Communicator*, the *Wireless Simulator Communicator*, and the *Application Manager* (Fig. 2). Through dedicated client entities implemented in these components, the iCS triggers actions to be executed in the other blocks, and actively requests the resulting outcomes. On the other hand, server entities implemented in the other iTETRIS blocks reactively accomplish the tasks requested by the iCS. Considering that the iCS is basically implemented as a controller, the adopted client/server approach allows focusing on the design of a highly efficient protocol for the interaction between the iCS and the other blocks. In fact, this is done independently from any overlying interfacing architecture, and therefore prevents to obey to architectural specifications, which might result in complex implementations.

5.2. iTETRIS simulation process

The execution of iTETRIS simulations is controlled by the iCS. First, the iCS sets up the simulation environment by initialising the various iTETRIS configurable objects. A hierarchical XML configuration file structure is adopted (Fig. 3a) to improve the readability of the simulation configuration, and facilitate the customization of the various iTETRIS blocks. In this context, the master configuration file defines general parameters such as the duration (in seconds) of the wanted simulated time period, the penetration rate of vehicles equipped with each simulated radio access technologies, and the sockets' IP addresses and port numbers needed to communicate the iCS with ns-3, SUMO and the iAPP. The master configuration file also includes the path to the files used to configure ns3, SUMO and the iAPP. When iTETRIS is started, SUMO and ns-3 are launched by the iCS with their executables registered in the system in separate threads so that, from that point on, they can receive commands. At the same time, the iCS allocates dedicated execution threads for each of the simulated cooperative ITS applications (iAPPs), and reads the configuration file needed to create the iCS Facilities. SUMO and ns-3 configuration files are then read to prepare the traffic and wireless environments (e.g. road map, vehicular traffic flows, type and communications parameters of the simulated wireless technologies, etc.). Interested readers can find a complete description of how to write iTETRIS configuration files, as well as build and simulate cooperative ITS applications over the iTETRIS platform on the "iTETRIS Building, Installation and Configuration Guidelines" available at the iTETRIS community webpage: http://www.ict-itetris.eu/10-10-10-community/.

iTETRIS simulations consist of subsequent iterations of a loop (Fig. 3b) in which ns-3, SUMO and the iAPPs are sequentially triggered by the iCS to execute their tasks. The simulated time period specified in the master configuration file is divided into simulation time steps of one second.[2] For each simulation time step, the different iTETRIS blocks simulate all the application, traffic or wireless communications events scheduled for the corresponding time step. The entry point in the run-time loop is the simulation of the transmission of wireless messages in ns-3. The application's payload for these messages is created and stored in the iCS. When the iCS schedules message transmissions in ns-3, a reference to these payloads is passed to ns-3. Once ns-3 has simulated all the events scheduled for the current time step, the iCS retrieves the simulation results as lists of successful wireless transmissions (i.e. messages that have been correctly received by their recipient nodes). The iCS can

[2] iTETRIS has been initially developed to evaluate cooperative ITS traffic management applications under large scale scenarios. A one second-simulation time step is sufficient for a comprehensive and accurate evaluation of traffic management applications. However, the iCS can be modified to accommodate for smaller simulation time steps if required to evaluate other types of cooperative ITS applications.

then match the received messages with the previously stored payloads, and update specific communications-related iCS Facilities (e.g. LDM database) which can then be accessed and used by the applications implemented in the iAPPs. In the following stage of the run-time loop, SUMO is triggered to simulate all the traffic mobility events corresponding to the established simulation time step. As required by the iCS' Traffic Simulator Communicator, SUMO provides as outcome the updated position and speed of active vehicles, along with the position and speed of vehicles entering the simulated scenario in the current time step. For these new vehicles, the iCS establishes whether they are equipped with a given communications technology following the penetration rates identified in the master configuration file. If a vehicle is equipped with a communications technology, it can then run cooperative ITS applications, and a structure for this vehicle is created at the iCS in order to link its SUMO and ns-3 representations. Upon retrieving the SUMO simulation outcomes, the iCS updates the mobility-related iCS Facilities to store the information related to active or new vehicles that could be used by the applications implemented in the iAPPs. The iCS passes then the simulation token to the iAPP block. iAPP is asked by the iCS' Application Manager to ''subscribe'' to the SUMO and ns-3 simulation results that are needed for the execution of the application implemented in it. Based on these subscriptions, the iCS forwards the needed information to the iAPP. After executing the applications implemented in the iAPPs during the corresponding simulation time step, the iCS retrieves the iAPPs' results that in turn may generate new actions to be executed over SUMO or ns-3 (e.g. the transmission of a new wireless message or the traffic rerouting of a vehicle). The last stage of the run-time loop is devoted to prepare the execution of the next time step in ns-3. In particular, the iCS' Wireless Simulator Communicator schedules the transmission of new messages, commands ns-3 to update the position of nodes based on SUMO's outcomes, and instructs ns-3 to create the new connected vehicles that have just entered the simulation scenario. Then, the iCS updates the time step counter, and checks whether its value is equal to the previously configured simulated time period's duration. If this is the case, the simulation is ended; otherwise a new iteration loop is performed. To end a simulation, the iCS cleans up the objects in the memory, closes logging files (if they were defined), shuts down the connection with ns-3, SUMO and the iAPPs, and eliminates the threads in which they were executed.

6. iTETRIS ns-3 implementation

The transmission of wireless messages between iTETRIS nodes has been implemented in ns-3 following ETSI's ITSC architecture. Fig. 4 illustrates the implemented ns-3 modules, and the interaction between ns-3 and the iCS using the iTETRIS iNCI interface.

6.1. iNCI interface

Differently from SUMO, the default version of ns-3 does not provide interfaces for the interaction with external modules. Since such interaction is necessary between ns-3 and the iCS, the *iTETRIS Network simulator Control Interface* (*iNCI*) has been created. The bidirectional exchange of information between ns-3 and the iCS is carried out by means of primitives between an iCS client (*iNCI Client*) implemented in the Wireless Simulator Communicator and an ns-3 server entity (*iNCI Server*). Client and Server communicate using IP sockets and implement buffers to optimize the communications between ns-3 and the iCS. The iNCI Server consists of two main entities: the *Node Manager* and the *Packet Manager.*

6.1.1. iNCI Node Manager

The Node Manager implements specific primitives that allow the iCS to dynamically create new ns-3 nodes and update their position and speed at each simulation time step. As defined in Section 3.1., the modularity of ns-3 offers the possibility to incrementally aggregate modules to a node. Following this approach, iTETRIS defines *Communications Modules* as sets of ns-3 classes modelling components of the various ITSC layers, and that can be separately installed on a node.[3] To install specific communications modules, iTETRIS defines dedicated *Communications Module Installers*. When a Node Manager's primitive to create new ns-3 nodes is called, these installers aggregate communications modules on the nodes according to the instructions defined by the user in the ns-3 configuration file. Currently, it is possible to create three types of ns-3 nodes with iTETRIS: vehicles, *Communications Infrastructure Units* (*CIUs*) and *Middleware* (*MW*) nodes. Vehicles can be equipped with more than one radio access technology, and can transmit using either the C2C or IP stacks. As a result, various communication module installers are required to create a vehicle in ns-3 and configure its communications capabilities. In addition, installers are also needed to integrate into vehicles the ns-3 Facilities and the Management layer functionalities. CIUs refer to ITS G5 RSUs and other communications infrastructure nodes such as UMTS, WiMAX or DVB base stations. RSUs only communicate using ITS G5 and the C2C stack.[4] In this context, creating a RSU in ns-3 only requires installers for ITS G5, the C2C stack and ns-3 Facilities. On the contrary, UMTS, WiMAX or DVB base stations can only communicate over the IP stack. As a result, creating these nodes in ns-3 only requires installers for the corresponding radio interface, IP stack, and necessary ns-3 Facilities. The MW node is an entity defined by iTETRIS to assist the TMC in the centralized selection of the most appropriate CIU to disseminate traffic information over a geographical target area. MW nodes are virtually tied to both the TMC and the controlled CIUs through a

[3] For example, the ITS G5 communications module includes all the ns-3 classes used by the simulator to model the operation of the ITS G5 access technology. Similarly, communications modules for the components of the ITSC Transport & Network layer (namely C2C and IP stacks) are defined, and so on.

[4] Although ITS G5 RSUs could also communicate using the IP stack, the current iTETRIS platform only considers the C2C stack in a RSU. The extension to enable IP communications over an ITS G5 RSU could be easily done.

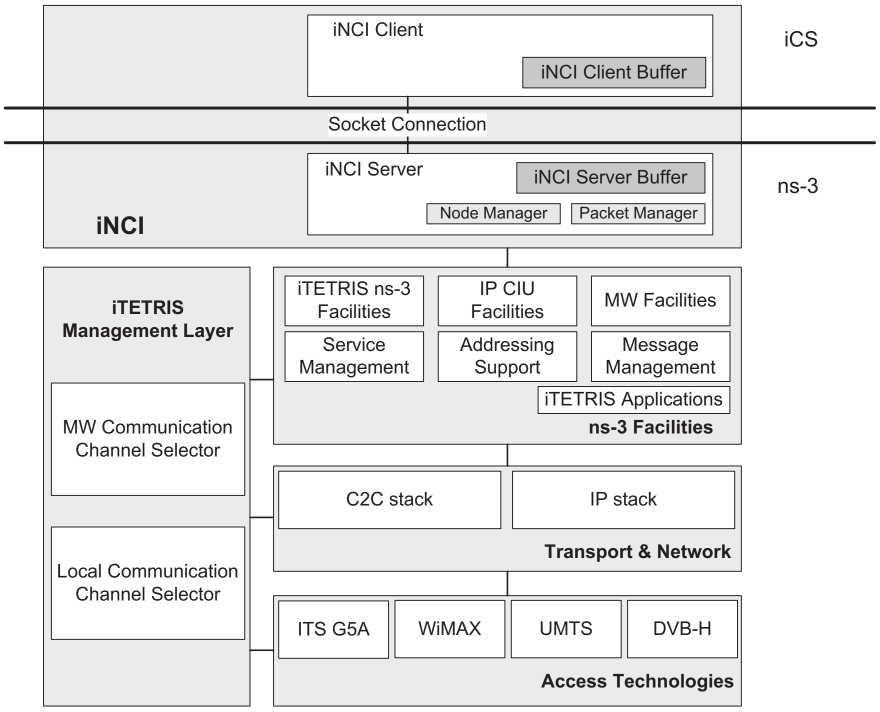

Fig. 4. iTETRIS ns-3 implementation and iNCI interface.

backbone network. Since iTETRIS focuses on wireless communications between vehicles, and between vehicles and infrastructure nodes, backbone networks are not modelled, and MW nodes only require the installer for the needed ns-3 Facilities and Management layers in order to be able to select the most appropriate CIU to run a cooperative ITS application.

6.1.2. *iNCI Packet Manager*

The Packet Manager offers primitives for the iCS to activate and deactivate in ns-3 the simulation of message transmissions, and to retrieve information about correctly received messages. Some of these primitives are listed in Table 1; the primitives are labelled following the type of transmission used to perform a given cooperative service. For example, Cooperative Awareness Message (CAM) transmissions are used by vehicles and RSUs to periodically broadcast information about their status (e.g. speed and position) to neighboring nodes, performing in this way the Cooperative Awareness Basic Service defined in the ETSI standard [24]. Similarly, geobroadcast transmissions can be used to perform the Decentralized Environmental Notification Basic Service through which vehicles and RSUs disseminate Decentralized Environmental Notification Messages (DENMs) in their neighborhood [24]. As illustrated in Table 1, the Packet Manager uses each primitive to communicate with specific nodes that have to activate or deactivate the transmission of service messages in ns-3. For example, ACTIVATE_CAM_TXON and DEACTIVATE_CAM_TXON are used to start and stop the periodic transmission of CAM messages.[5] Each of the primitives in Table 1 is defined through a different set of parameters, whose meaning is described in Table 2.

The value of the primitives' parameters can be directly defined by the active cooperative ITS applications implemented in the iAPPs. The iCS introduces these values into the corresponding primitives when the execution of a service message transmission in ns-3 is triggered. When a primitive is called, the Packet Manager identifies the ns-3 node that will transmit the corresponding messages, and calls the necessary functions at ns-3 Facilities level to generate the messages to be transmitted. After generating and simulating the transmission of these messages, ns-3 indicates to the iCS which nodes correctly received them. For this purpose, the iCS calls the Packet Manager's primitive CMD_GET_RECEIVED_PACKETS whose implementation works with classes called *InciPacketLists* that are attached to every ns-3 node and are used to keep

[5] This example shows how iTETRIS has optimized the interaction between the iCS and ns-3 in order to reduce the number of messages to be exchanged between the two blocks. In particular, iCS only needs to trigger ns-3 twice to handle the transmission of periodic messages (e.g. CAMs), once to start the transmission and another time to stop it. This significantly reduces the exchange of messages between the iCS and ns-3 compared to the case in which iCS had to trigger ns-3 to transmit every single CAM message.

Table 1
iNCI packet manager primitives.

Primitive	Description	Parameters
ACTIVATE_CAM_TXON	Requests ns-3 to activate the transmission of CAM messages on a given node (Vehicle or RSU)	NodeId, PayloadLength, TransmissionFrequency
DEACTIVATE_CAM_TXON	Requests ns-3 to deactivate the transmission of CAM messages on a given node (Vehicle or RSU)	NodeId
ACTIVATE_DENM_TXON	Requests ns-3 to activate the transmission of DENM messages on a given node (Vehicle or RSU) and on a geographic destination area	NodeId, PayloadLength, Destination, TransmissionFrequency, MsgRegenerationTime
DEACTIVATE_DENM_TXON	Requests ns-3 to deactivate the transmission of DENM messages on a given node (Vehicle or RSU)	NodeId
ACTIVATE_TOPO_TXON	Requests ns-3 to activate the topobroadcast transmission of a message on a given node (Vehicle or RSU)	NodeId, ServiceId, TransmissionFrequency, Destination, PayloadLength, MsgRegenerationTime, MsgLifetime, NumHops
ACTIVATE_GEO_BROAD_TXON	Requests ns-3 to activate the geobroadcast transmission of a message on a given node (Vehicle or RSU)	NodeId, ServiceId, TransmissionFrequency, Destination, PayloadLength, MsgRegenerationTime, MsgLifetime
ACTIVATE_ID_BASED_TXON	Requests ns-3 to activate (on a vehicle or RSU) the transmission of a message based on the ID of the destination. This primitive can be used to activate either unicast or broadcast transmissions (in the latter case, the destination ID is a broadcast constant). If the sender is a vehicle and the destination is the TMC, the transmission can be executed over one of the different radio access technologies and communications stack the vehicle is equipped with according to a given communication profile suggested by the iAPP	NodeId, ServiceId, Comm Profile, ListOfTechnologies TransmissionFrequency, PayloadLength, Destination, MsgRegenerationTime, MsgLifetime
ACTIVATE_IPCIU_TXON	Requests ns-3 to activate on a CIU the transmission of a message based on the ID of the destination. This primitive can be used to activate IP unicast, broadcast or multicast transmissions (in case of broadcast or multicast transmissions, the destination ID is a broadcast/multicast constant)	NodeId, ServiceId, TransmissionFrequency, PayloadLength, Destination, MsgRegenerationTime
ACTIVATE_MW_TXON	Requests ns-3 to activate the transmission of a notification message to a geographical area using the MW node. The selection of the most appropriate CIU to transmit the message will be requested by the MW node to the Management Layer according to a given communication profile suggested by the iAPP	NodeId, ServiceId, Comm Profle, ListOfTechnologies, TransmissionFrequency, PayloadLength, Destination, MsgRegenerationTime, MsgLifetime
GET_RECEIVED_PACKETS	Requests ns-3 to return the messages received by a given node	NodeId

Table 2
Parameters of the iNCI packet manager primitives.

Parameter	Description
ServiceID	Identifier of the cooperative ITS service
NodeId	Identifier of the node over which the cooperative ITS service has to be activated or deactivated
PayloadLength	Size of the service message to be activated
TransmissionFrequency	Transmission frequency of the service message to be activated
MsgRegenerationTime	Time period over which the service message to be activated has to be periodically retransmitted
MsgLifetime	Time during which the service message to be activated have to be considered valid by recipient nodes
Destination	Destination of the service message to be activated. It can be a geographical circular destination area where a message has to be disseminated (ACTIVATE_GEO_BROAD_TXON, ACTIVATE_MW_TXON), or the identifier of a specific destination node (ACTIVATE_ID_BASED_TXON, ACTIVATE_IPCIU_TXON)
NumHops	Maximum number of hops the service message is allowed to be forwarded
ListOfTechnologies	Set of suitable radio access technologies (ITS G5, UMTS, etc.) that can be used to execute a cooperative ITS service
CommProfile	Communication profile suggested by the iAPP for the cooperative ITS service

trace of all the received *InciPackets*. An InciPacket is a structure that includes the information needed by the iCS to match ns-3 representations of transmitted messages with its internal iCS message representations. In particular, each InciPacket includes the following information: (1) the identifier of the message's sender, (2) the associated service identifier, (3) the simulation time step at which the message is transmitted, and (4) the sequence number of the message in a simulation time step.[6] Upon request from the iCS, the Packet Manager retrieves from a given node all the InciPackets that are stored in its InciPacketList.

[6] This information is needed by the iCS to distinguish among various messages associated to the same service and transmitted by the same sender in the same simulation time step.

6.2. ns-3 Facilities

On each ns-3 node, its Facilities generate the messages to be transmitted, and implement the functions required to forward these messages to the lower layers of the communications protocol stack. The ns-3 Facilities are also responsible to forward received simulated messages to the iNCI Packet Manager, so that the iCS can retrieve the required information about these messages as previously explained. To generate and receive service messages at Facilities level, iTETRIS defines the ns-3 *iTETRISApplications*, which are conceptually similar to other ns-3 applications (e.g. Ping or UDPClient/Server) used to generate communications traffic. The iTETRISApplication is a parent class for a number of specific applications such as those handling CAM and DENM services or those supporting IP-based services. Another interesting example of iTETRISApplication is the *DTNiTETRISApplication,* which controls the transmission of messages following the guidelines established in the Delay-tolerant Networking (DTN) architecture (http://tools.ietf.org/html/draft-irtf-dtnrg-arch-08).[7] When a node is created in ns-3, specific iTETRISApplications supporting distinct services are installed on it according to the instructions specified in the ns-3 configuration file (vehicles and RSUs have the CAM and DENM services installed by default). These services are locally stored on every node in a *ServiceList*, and are associated to a string ServiceId. Using these identifiers, the iCS can call and execute, when required, the correct iTETRISApplication to start the transmission of specific service messages.

Besides the iTETRISApplication classes, the implementation of the ns-3 Facilities layer on a given ns-3 node requires distinct additional classes depending on the type of node. In particular, vehicles and RSUs use the *iTETRISns3Facilities* class, the rest of CIUs (UMTS, WiMAX and DVB base stations) the *IPCIUFacilities* class, and MW nodes the *MWFacilities* class (Fig. 4). When the iNCI Packet Manager wants to trigger the execution of one of the iTETRISApplications installed on a node, it calls specific functions provided by the above mentioned ns-3 Facilities classes. For example, the iTETRISns3Facilities class includes functions to activate or deactivate the periodic transmission of CAM messages, or to activate transmissions over the C2C or IP stacks. By calling these functions, many of the parameters indicated in the Packet Manager primitives (Table 2) are reused. If the ListOfTechnologies parameter defines more than one candidate radio access technology, then one of them has to be selected to execute the service following the specification of a given CommProfile suggested by the iAPP. In the current iTETRIS release, this situation can only occur in the iTETRISns3Facilities class when a vehicle equipped with multiple radio access technologies wants communicate with the TMC (in response to the primitive ACTIVATE_ID_BASED_TXON defined in Table 1), or in the MWFacilities class when the TMC wants to disseminate a traffic message over a target area and can use different types of communications infrastructure nodes to do so (ACTIVATE_MW_TXON primitive in Table 1). In both cases, the selection will be performed by calling functions of the iTETRIS ns-3 Management layer that are described in the next section.

Once the adequate radio access technology has been selected, iTETRISns3Facilities, IPCIUFacilities and MWFacilities use additional ns-3 Facilities classes to generate the service messages. In particular, the *AddressingSupport*, *ServiceManagement* and *MessageManagement* classes are defined. Fig. 5 illustrates some of the functions provided by these classes and how they are sequentially called when a service message is activated at Facilities level. As it can be seen, once all the communications parameters have been set on the iTETRISApplication identified by ServiceID, the MessageManagement class can finally activate its transmissions. From this point on, the called iTETRISApplication starts to autonomously generate sporadic or periodic service messages as requested by the simulated cooperative ITS application. At this stage, the active iTETRISApplication generates all the information needed for the InciPackets, and associates it to each of the transmitted messages. When a message is received at a node's ns-3 Facilities layer, the active iTETRISApplication extracts from it the InciPacket information, and forwards it to the InciPacketList by the means of callbacks. Using this information, the iCS can track the reception of messages previously scheduled for transmission in ns-3.

6.3. Management layer

The Management layer is a vertical cross-layer component of an ITS station that can coordinate the operation of the Access Technologies, Transport & Network, Facilities and Application layers. The iTETRIS ns-3 Management layer focuses on communications management issues, and is in charge of selecting the most suitable combination of radio access technology and communication protocols (in the following referred to *Communication Channel* or *Communication Profile* as defined by ETSI in [24]) to operate each cooperative ITS application. This selection should be taken dynamically based on the application requirements and the current status of the available radio access technologies. In iTETRIS, the selection can be local or global. A local communication channel selection is adopted by vehicles equipped with several radio access technologies. On the other hand, a global communication channel selection is necessary for the TMC to select the most suitable CIU to disseminate traffic messages over a target area. For this purpose, iTETRIS uses the previously defined MW node. The current iTETRIS release partially simplifies the communication channel selection process to facilitate the execution of large scale simulations. However, since iTETRIS is open source, interested readers could implement more advanced selection algorithms. In the cur-

[7] DTN protocols are used to cope with the absence of radio connectivity in environments with sparse presence of nodes. These protocols allow routing and forwarding messages after being locally stored and carried on a given node. In the iTETRIS DTN extension, the DTN stack is designed as a message overlay in which a common message header format and common mechanisms for handling the DTN message forwarding are defined. These mechanisms are implemented in the DTNiTETRISApplication to control the transmission of messages at each hop using information from lower layers (e.g. the network neighborhood), and allow replication-based dissemination.

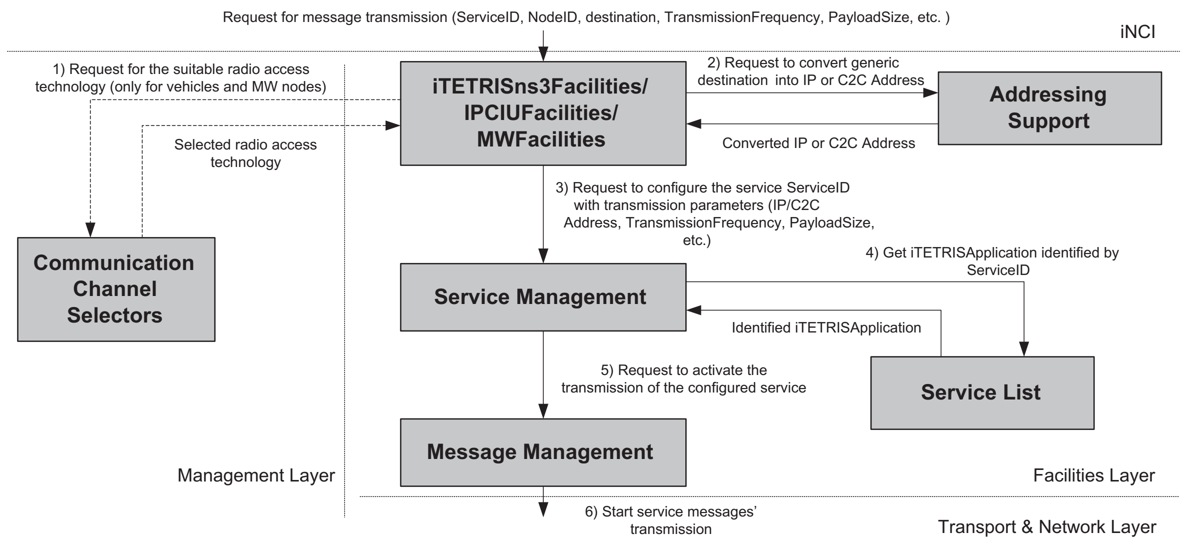
Request for message transmission (ServiceID, NodeID, destination, TransmissionFrequency, PayloadSize, etc.)
iNCI
1) Request for the suitable radio access technology (only for vehicles and MW nodes)
iTETRISns3Facilities/ IPCIUFacilities/ MWFacilities
2) Request to convert generic destination into IP or C2C Address
Addressing Support
Converted IP or C2C Address
Selected radio access technology
3) Request to configure the service ServiceID with transmission parameters (IP/C2C Address, TransmissionFrequency, PayloadSize, etc.)
Communication Channel Selectors
4) Get iTETRISApplication identified by ServiceID
Service Management
Identified iTETRISApplication
5) Request to activate the transmission of the configured service
Service List
Message Management
Management Layer
Facilities Layer
6) Start service messages' transmission
Transport & Network Layer

Fig. 5. iTETRIS ns-3 Facilities architecture.

rent iTETRIS release, the iTETRISns3Facilities class calls the *LocalCommChSelector* class implemented in the Management layer to realize the local selection on vehicles (Fig. 5). To do so, the list of suitable radio access technologies (ListOfTechnologies) and the suggested Communication Profile (CommProfile) are passed to the LocalCommChSelector. Based on the communication profile, the message to be transmitted is mapped into a set of *Generic Profiles* that represent different requirements of cooperative ITS applications. Each of these profiles is then associated to one or more suitable radio access technologies according to the rules defined in the communication channel selector. Once the list of suitable radio access technologies has been established for the identified profile, the selector compares it with the ListOfTechnologies parameter received from the Facilities layer. It then checks whether one of the suitable radio access technologies is present on the vehicle and can actually be used by it. To do so, the *VehicleStaMgnt* class, available at every vehicle, is used. This class returns the RSUs and CIUs that currently offer radio coverage to the vehicle. The first radio access technology that is found suitable based on the information retrieved from VehicleStaMgnt is selected to transmit the message. A similar approach is used for the global communication channel selection on MW nodes. In this case, the MWFacilities class calls the MWCommChSelector class (Fig. 5) to which the CommProfile and ListOfTechnologies parameters are passed in addition to the geographical coordinates of the target area where the TMC wants to disseminate a given message (Destination). Using this information, together with the knowledge of the position of the deployed CIUs and the number of vehicles they serve (this information can be retrieved from specific *CIUMngt* classes), the selector can decide which is the most suitable CIU to disseminate a given message. In the current iTETRIS release, the first access technology in the ListOfTechnologies that offers coverage to a satisfactory number of vehicles is selected to transmit the message.

6.4. Transport and network layer

iTETRIS enriches the official ns-3 release by implementing a C2C stack based on ETSI ITSC's GeoNetworking stack. The ITSC's GeoNetworking stack offers the transport and networking capabilities needed in vehicular environments using ETSI's ITS G5 radio access technology. In this context, *Geonetworking* or *Georouting* refers to a communications paradigm in which the messages are forwarded through relaying nodes that are selected based on their geographical position. Four basic georouting transmission modes (*Geounicast, Geobroadcast, Topobroadcast* and *Geoanycast*) are implemented in iTETRIS following the specifications of the ETSI standard [25].

iTETRIS C2C stack (Fig. 6) is similar in structure and operation to the existing ns-3 IPv4 and IPv6 stacks, except for the use of specific *C2CAddresses* (based on the nodes' ID and geographic position) and the existence of *C2CInterfaces* to ITSC lower layers. *C2CSockets* implement an asynchronous socket API system enabling the iTETRISApplications to bind and listen to specific port numbers for the transmission and reception of messages. The *C2CTransportProtocol* class' implementation follows the lightness requirements imposed by vehicular environments to transport layer functionalities. *C2CL3Protocol* is responsible for the routing of transmitted and received messages according to the established georouting transmission mode. As a result, it exploits the functionalities offered by the *GeoRoutingProtocol* class, which defines the core algorithms of the above mentioned basic georouting transmission modes. In addition, the *Beaconing Protocol* and the *Location Table*, two important supporting functionalities for georouting transmissions [25], are implemented. The Beaconing Protocol is used by all vehicles and RSUs to periodically broadcast beacon messages to notify their position and speed to the neighboring nodes. The Location Table is a dynamic database where vehicles and RSUs store and update the position and speed information received from neighboring nodes.

When a message to be transmitted is generated by an iTETRISApplication on a given node, it is sent to the C2CTransportProtocol through the C2CSocket. As shown in Fig. 6, the C2CTransportProtocol requests the GeoRoutingProtocol to select the next forwarder of the message. GeoRoutingProtocol identifies the next forwarder by analyzing the C2C address of the message's final destination, which varies according to the requested georouting transmission mode. The C2C address specifies a circular destination area for the geobroadcast and geoanycast transmission modes, a number of hops for the topobroadcast transmission mode, and a geounicast address (represented as a combination of the ID and geographical position of the destination node) for the geounicast transmission mode. While identifying the next forwarder, the GeoRoutingProtocol also extends the transmitted message by adding a transmission mode-specific network header [25]. Once the next forwarder has been identified, the extended message is passed down to the C2CL3Protocol, which in turn delivers the message to the node's lower layers over a C2CInterface connecting the C2C stack to the attached Access Technologies. When the C2CL3Protocol receives a message from the lower layers, it calls GeoRoutingProtocol. GeoRoutingProtocol analyzes the information included in the message's network header, and decides whether to locally deliver the message to the upper layers (if the receiving node is the message's destination node), forward it (if the receiving node is not the message's destination node), or discard it in case of errors. Although iTETRIS only implements the four basic georouting transmission modes as defined in [25], future deployments could require the coexistence and cooperation of several georouting protocols. To account for this possibility, iTETRIS' C2C stack proposes an implementation supporting a list of routing protocols over the C2CListRouting class. Each of these routing protocols can be installed on ns-3 nodes, and be assigned a given priority. When needed during a simulation, these routing protocols are sequentially called to identify the next forwarder of a message; the final routing decision is taken according to the protocols' priority. In iTETRIS, GeoRoutingProtocol is installed by default with the lowest priority on vehicles and RSUs. By modifying the ns-3 configuration file, advanced georouting protocols with higher priorities could be added and used.

6.5. Access technologies

As shown in Fig. 4, iTETRIS has implemented in ns-3 four different radio access technologies to offer a heterogeneous communications and networking environment to support cooperative ITS applications. In particular, iTETRIS implements in ns-3 the vehicular ETSI ITS G5A standard [1], the cellular UMTS technology [26], the WiMAX (or IEEE 802.16 [27]) technology, and the DVB-H broadcasting system [28]. Each of these technologies is implemented using independent ns-3 NetDevices. To achieve a satisfactory tradeoff between the computational cost of conducting large scale simulations and the modelling accuracy, different modelling decisions were taken. iTETRIS has accurately modelled the radio propagation effects using validated propagation models. Moreover, in iTETRIS the effect of the probabilistic nature of radio propagation is modelled by implementing the Packet Error Rate (PER) performance as a function of the experienced Signal to Noise and Interference Ratio (SINR). It is important to note that iTETRIS is not a simulator designed to investigate and optimize the performance of UMTS, WiMAX or DVB-H, but rather a simulator designed to exploit their transmission capabilities to efficiently support the deployment of cooperative ITS applications. As a result, most of the network management and transmission control functionalities that are part of the UMTS, WiMAX and DVB-H standards have been simplified without negatively

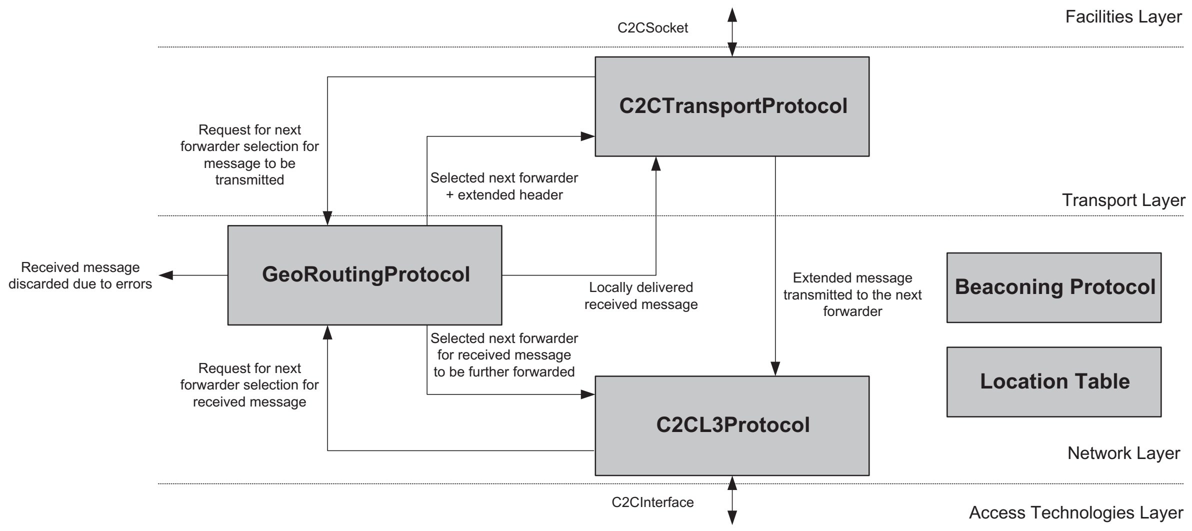
C2CSocket
Facilities Layer
C2CTransportProtocol
Request for next forwarder selection for message to be transmitted
Selected next forwarder + extended header
Transport Layer
Received message discarded due to errors
GeoRoutingProtocol
Locally delivered received message
Extended message transmitted to the next forwarder
Beaconing Protocol
Location Table
Selected next forwarder for received message to be further forwarded
Request for next forwarder selection for received message
C2CL3Protocol
Network Layer
C2CInterface
Access Technologies Layer

Fig. 6. iTETRIS C2C stack architecture.

influencing the simulation results or iTETRIS's validity to investigate cooperative ITS communications and applications in large scale scenarios. It is worth stressing that none of the iTETRIS implementations of the described radio access technologies corresponds to pre-existing modules in the form they are distributed in ns-3 or ns-2 releases. Even if in some cases the iTETRIS radio access technologies have been built on top of existing ns-3 or ns-2 implementations, they have always required modifications and extensions. In some cases they were needed to adapt to the new ETSI ITSC standards (e.g. ETSI ITS G5 built on the top of the ns-3 WiFi models), while in other cases certain wireless control and management functionalities unnecessary for the evaluation of cooperative ITS applications were simplified (e.g. in the case of UMTS and WiMAX).

6.5.1. ETSI ITS G5A

The ETSI ITS G5 standard is an evolution of the IEEE 802.11a standard including communication functions required to operate in rapidly varying vehicular environments and exchange messages with short connection establishment delays. As a result, ITS G5 stations do not need to perform procedures such as scanning, association and authentication before establishing communications with other nodes. ITS G5 distinguishes different operation modes for various types of cooperative ITS applications and frequency bands. However, iTETRIS only implements the ITS G5A mode which will be used for road safety and traffic management applications. The ITS 5GA operation mode uses a channel bandwidth of 10 MHz instead of the 20 MHz employed by the IEEE 802.11a standard. This is done to combat the inter-symbol interference caused by multipath propagation effects. In fact, the time guard interval at 10 MHz bandwidth is longer, and is therefore more capable to overcome the worst case delay spread resulting from multipath propagation in vehicular environments. The use of a 10 MHz channel bandwidth results in that all 802.11a OFDM timing parameters are doubled, and consequently all the data rates are halved, leading to transmission rates ranging from 3 to 27 Mbps. The 30 MHz bandwidth (5875–5905 MHz) assigned to the ITS G5A mode is divided into 3 sub-channels of 10 MHz, one devoted to operate as control channel (G5CC) and the other two as service channels (G5SCs). Vehicles and RSUs broadcast their presence over the G5CC using CAMs and beacon messages, while they announce available services that can be exploited on G5SC1 or G5SC2.

iTETRIS realized a new implementation of the ITS G5A radio access technology by modifying and extending the existing ns-3 WiFi models with a set of new functions and modules. The resulting *ITS G5A Communications Module* depicted in Fig. 7 includes all the needed communications functions required to operate in vehicular environments following the specifications of the standard [1]. The capability stated by [1] to simultaneously receive over the control channel and one of the service channels is realized by installing two ITS G5A NetDevices on vehicles and RSUs: one NetDevices always operates on the control channel, while the other one switches among the two service channels (G5SC1 and G5SC2). The ITS G5A NetDevices required changes in the existent ns-3 WiFi PHY and MAC layers implementation in order to allow the 10 MHz channel bandwidth. To direct messages coming from upper layers to the corresponding ITS G5A channel, a *NetDevice Router* has been implemented. The default ns-3 WiFi module was unable to perform channel switching. This capability has been then introduced in iTETRIS through the implementation of an *ITS G5A Switching Manager*. The switching manager allows the cancellation, suspension and resumption of pending message transmissions that were blocked before switching channels. The iTETRIS ITS G5A implementation also provides the possibility for the upper or management layers to control the transmission parameters on a per-message basis. This is done by defining a set of ns-3 packet tags which can specify the channel, transmission power and data rate parameters to be used for transmitting each message.

6.5.2. WiMAX

The ns-3 WiMAX module of iTETRIS is built from an existing ns-3 implementation developed by INRIA [29]. Unicast and broadcast data transmission functionalities have been accurately modelled in iTETRIS. On the other hand, the implementation of network management functionalities has been simplified. This has been done since these functionalities are more related to the actual management of WiMAX networks than the use of WiMAX transmission capabilities to assist in the future deployment of cooperative ITS applications. Their detailed implementation is therefore not required for iTETRIS purposes. In addition, such a detailed implementation and modelling can significantly increase the simulation time, which would have limited iTETRIS capability to conduct large scale investigations. In this context, channel scanning, synchronization, and the process that WiMAX mobile stations periodically perform to get connected to a fixed WiMAX base station are emulated in iTETRIS by consulting an *Infrastructure Location Map* containing the position of fixed stations. A similar approach has been followed for the dynamic adaptation of the mobile stations' modulation, coding and transmission power. For this purpose, an

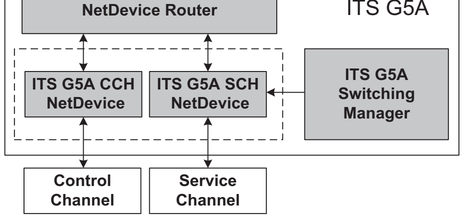

Fig. 7. Implementation of iTETRIS's ITS G5A communications module.

Adaptive Modulation & Coding (AMC) Map has been defined to assist in the decision process. The dynamic parameters are selected based on the distance between mobile stations and the fixed stations to which they are attached, and on the current radio channel conditions. Finally, iTETRIS also includes *Command Managers* (attached to each WiMAX node) to simulate the transfer of control information between mobile and base stations for management and maintenance of the WiMAX network. As a result, the registration and connection establishment processes are performed through the exchange of primitives, and there is no need to simulate the transmission of any control message.

6.5.3. UMTS

The implementation of UMTS in iTETRIS has followed a similar process to that adopted for WiMAX, i.e. to start from an existing UMTS implementation, accurately implement those aspects that would influence the outcome of iTETRIS investigations, and simplify those that are more related to the management of UMTS networks and not needed for iTETRIS purposes. iTETRIS UMTS implementation is built from the UMTS module developed at the University of Strathclyde, and originally implemented in ns-2 (http://nsnam.isi.edu/nsnam/index.php/Contributed_Code). As in the original implementation, two NetDevices are implemented, one for the UE (User Equipment, which in TETRIS corresponds to an UMTS-equipped vehicle) and another one for UMTS base stations (Node B). To reduce the complexity of the original communications architecture implementation, the iTETRIS UMTS Node B NetDevice acquires the necessary intelligence and functionalities of the RNC (Radio Network Controller). Moreover, iTETRIS introduces the *UMTS Manager* to model control procedures such as handovers or the setup of new connections. This entity is in charge of processing the demands related to the control level, and maintains a list of pointers to all the Nodes B deployed in the simulated scenario. When the UMTS Manager receives a petition to perform any control function (e.g. a setup request to a certain Node B), it notifies the petition to the RRC (Radio Resource Control) layer of the corresponding Node B and forwards the response to the originator UE node.

6.5.4. DVB-H

iTETRIS implements a new simplified DVB-H module that is not based on any existing ns-3 or ns-2 implementation. The implementation differentiates between two types of nodes. The *DVB-H Base Station* is in charge of the management and delivery of data services. The *DVB-H User Equipment* represents the consumer of DVB-H data services. The implemented DVB-H NetDevices include three different levels, each of them modelling different functionalities as defined by the DVB-H standard. The upper layer is referred to as the *DVB-H Manager*, and is mainly in charge of operations related to the creation and management of services, the management of the network resources, and the establishment and maintenance of connections between base and mobile stations. The *Link Layer* is devoted to the creation and processing of MPE-FEC (Multi-Protocol Encapsulation – Forward Error Correction) sections and PSI/SI (Program-Specific Information, and Service Information) tables, and the operation of the time slicing functionality. Finally, the *Physical Layer* handles the transmission and reception of MPEG-2 Transport Streams, and the creation and processing of Orthogonal Frequency Division Multiplex (OFDM) blocks, in addition to the estimation of the Packet Error Rate experienced during DVB-H transmissions.

6.6. Propagation models

Different studies have demonstrated that radio propagation significantly influences the operation and performance of cooperative vehicular communications [9]. As a result, iTETRIS implements in ns-3 radio propagation models that aim to realistically reproduce radio propagation effects. After a careful review of the state of the art, iTETRIS has selected from the available models those that more accurately take into account the characteristics of urban and highway scenarios for V2V and V2I communications (Table 3). The implemented models carefully reproduce the path-loss, shadowing and multipath fading effects. The urban models distinguish between LOS (Line-of-Sight) and NLOS (Non-Line-of-Sight) propagation conditions since they significantly impact the received signal strength.

As an example, urban iTETRIS channel models are implemented in dedicated ns-3 classes defining functions to compute the path-loss, shadowing and multipath fading under both LOS and NLOS conditions. For this purpose, iTETRIS implements a basic *Visibility Model* that detects the visibility conditions between communicating nodes based on their relative position with respect to the buildings present in the simulated road network scenario. At the beginning of an iTETRIS simulation, the visibility model reads a file with the coordinates of the walls forming the perimeter of the buildings present in the scenario. Each wall is represented as a segment. The model considers that there are NLOS conditions between two nodes if the segment connecting them intersects any of the walls present in the simulated scenario.

7. iTETRIS SUMO implementation

Readers interested in a detailed presentation of SUMO can refer to [22]. This section focuses on the description of the new functionalities added to SUMO based on the iTETRIS requirements.

Table 3
Propagation models adopted in iTETRIS.

Scenario	Urban	Highway
V2V	WINNER B1 – Urban microcell [31]	Cheng and Stancil [30]
V2I	WINNER B1 – Urban microcell [31]	WINNER D1 – Rural [31]

7.1. TraCI interface

As previously explained, TraCI allows external applications to retrieve values from SUMO (*Value Retrieval Commands*), or reactively impose actions to simulate in SUMO's vehicular environment (*State Change Commands*). Compared to the first version presented in [23], iTETRIS includes a reworked and extended version of TraCI. The first change concerns the naming system for the TraCI internal representation of simulated nodes. In iTETRIS, TraCI adopts the SUMO-native naming system based on string IDs instead of the old system based on integers. TraCI was first implemented in the context of TraNS to interconnect ns-2 and SUMO. As a result, the first version of TraCI used integers since integers were used to represent nodes in ns-2. In this context, the new TraCI solution based in string IDs avoids expensive mapping procedures aimed at translating the simulated objects' representations. The new TraCI interface also includes a generic implementation of the access functions used to retrieve data from SUMO or to provide instructions to it. This has been done by reusing much of the original interface, but defining a system based on commands and variable identifiers that SUMO can easily interpret. The existing *Vehicle Value Retrieval* and *Vehicle State Change* commands used to retrieve values from simulated vehicles or to impose changes over them have also been extended in iTETRIS by adding new capabilities. In particular, it is now possible to retrieve information about fuel consumption, noise and pollutant emissions. In addition, iTETRIS has modified the support to dynamically change (e.g. upon reception of V2V or V2I messages) pre-established vehicles' routes during the simulation.[8] iTETRIS's TraCI implementation has also been extended through the integration of new access functions. In particular, the *Simulation Control Access Functions* allow tracking changes in the state of SUMO objects. As a result, it is now possible to, for example, retrieve the number and list of new vehicles entering the simulation scenario in a particular time step, as well as the number and list of vehicles that leave the scenario or reach their destination. TraCI also includes new access functions that allow retrieving and controlling data regarding the road network (e.g. roads' edges, junctions, etc.) and other simulated objects such as inductive loops and traffic lights. iTETRIS' iCS uses these access functions to collect from SUMO data requested by the applications implemented at the iAPPs, or to control certain objects. For example, the new access functions allow retrieving information from the traffic lights, such as the current state (signal phases), the active program, or the time to the next switch. Similarly, the new access functions provide iTETRIS with the capability to control specific traffic lights by replacing their active program or directly changing their signal phase.

In iTETRIS, and similarly to the iNCI, TraCI is also divided into a client implemented in the iCS' Traffic Simulator Communicator, and a server integrated in SUMO and communicating through IP sockets. In particular, the mechanism through which the iCS TraCI client connects to SUMO is based on the concept of *Subscriptions*. A TraCI subscription identifies the data to be accessed, the SUMO object that holds this data, and the simulation time steps when the data has to be retrieved. Subscriptions are initiated through dedicated TraCI *Subscription Commands* similar in format to the above mentioned *Value Retrieval Commands*. After simulating the actions scheduled for the current time step, SUMO provides the iCS with all the subscribed data. In particular, the list of vehicles that entered and left the SUMO scenario in the current time step is first provided. Subscriptions to retrieve SUMO variables for the new entering vehicles are performed at this stage; these variables are required to update their representation within the iCS in the next time steps. On the other hand, the internal iCS representation of all vehicles leaving the SUMO scenario is removed. The TraCI Client receives then the subscribed parameters of vehicles already present in the SUMO scenario, and updates their internal iCS representation. Further interactions with SUMO can take place, for example for triggering actions to be simulated in SUMO through the execution of TraCI *State Change Commands*.

7.2. Importing traffic scenarios

iTETRIS is able to simulate the operation of cooperative ITS systems over large scale scenarios consisting of road networks and vehicular traffic demands that constitute the inputs to the traffic simulator SUMO. In this context, SUMO improved its tools to translate road networks and traffic demands representations derived from the real world and initially described in other formats (e.g. VISSIM, VISUM and OpenStreetMap, among others) into SUMO-compatible representations. Some of the SUMO-compatible road networks representations resulting from the application of these tools are depicted in Fig. 8. The figure shows different areas of the Italian city of Bologna that were selected in the iTETRIS project to test the effectiveness of cooperative ITS traffic management applications in large scale scenarios. The selected areas include interurban (Fig. 8a), city center (Fig. 8b), and urban neighborhood (Fig. 8c) scenarios. SUMO uses different traffic scenario importers depending on the

[8] In this context, new algorithms have been developed to compute alternative vehicles' routes that are no longer based solely on achieving the minimum travel time, but that also take into account the minimum environmental impact in large scale scenarios.

source format. VISSIM scenarios are able to describe road networks with a high degree of resolution: they include information such as the number of lanes, and even account for minor streets. Traffic lights, including their positions and signal plans are also defined. The given passenger vehicles' demands are described in terms of numbers of vehicles entering the scenario from certain roads located at the network's border. When passing specific routing decision points, vehicles are assigned new routes according to a given distribution. This method reproduces the turn percentages at intersections measured in the real world (other traffic demand descriptions used by VISSIM can be similarly imported by SUMO). In addition to the passenger vehicles, the public bus transport in terms of bus stops, bus routes and schedules is also described. Although SUMO's VISSIM road network importer is capable to import VISSIM road networks representations in a trustful manner, the complexity of VISSIM representations generally requires a manual validation of the resulting networks. On the other hand, SUMO's VISSIM traffic demands converter very effectively translates VISSIM representations into SUMO-compatible routes. Along with these tools, SUMO also includes an additional utility that permits extracting and importing into SUMO representations the position of bus stops and bus routes from VISSIM files.

Importing VISUM scenarios requires using different SUMO converters. Differently from VISSIM, VISUM is a macroscopic traffic assignment tool. Traffic assignment tools are used to compute which routes would be chosen by a given vehicles' demand within a given network, and thereby permit retrieving traffic loads and travel times of different streets on a macroscopic scale. As a result, VISUM representations do not necessarily include information about the number of lanes per street and make use of Connectors to model macroscopic traffic flows; VISUM connectors result in additional roads connecting distinct districts defined in VISUM files, and produce altered intersections compared to the real world. To convert such VISUM network representations additional network descriptors obtained from a Geographic Information System (GIS) database might be necessary. On the top of these SUMO-compatible converted road networks, the traffic demands stored in VISUM descriptions can then be mapped.

7.3. Models for pollutant emissions, noise emissions, and fuel consumption

A major SUMO extension within iTETRIS is the implementation of modules to compute the pollutant (e.g. CO, CO_2, HC, PM, NO_x) and noise emissions produced by the simulated vehicles as well as their fuel consumption. These new modules allow evaluating the environmental benefits of cooperative ITS systems. To model the pollutant emissions and fuel consumption, iTETRIS used the HBEFA (Handbook of Emission Factors, http://www.hbefa.net/) database which includes emission information for a large variety of vehicles. HBEFA provides lookup tables classifying the emission values as a function of the type of emission, road, or vehicle category, among others. To apply the HBEFA information in the new SUMO modules used to compute emissions and fuel consumption, a functional description able to fit the HBEFA lookup tables has been derived. Fitting functions for fuel consumption and pollutant emissions have been obtained individually. Instead of embedding all the 550 resulting functions, a clustering was performed first, joining functions for vehicle classes with similar emission values. This was done to ease the definition of a scenario's vehicle population. Finally, the implementation of these new modules for pollutant emission and fuel consumption has required the inclusion of the emission class information in the SUMO Vehicle Types. Computing the emissions or fuel consumption for a specific vehicle is done through methods that obtain the vehicle's emission class, its speed and acceleration [32]. If the acceleration is not positive, each of these methods returns a zero value. Otherwise, the methods retrieve the appropriate HBEFA fitting functions and realize the required computation. To compute

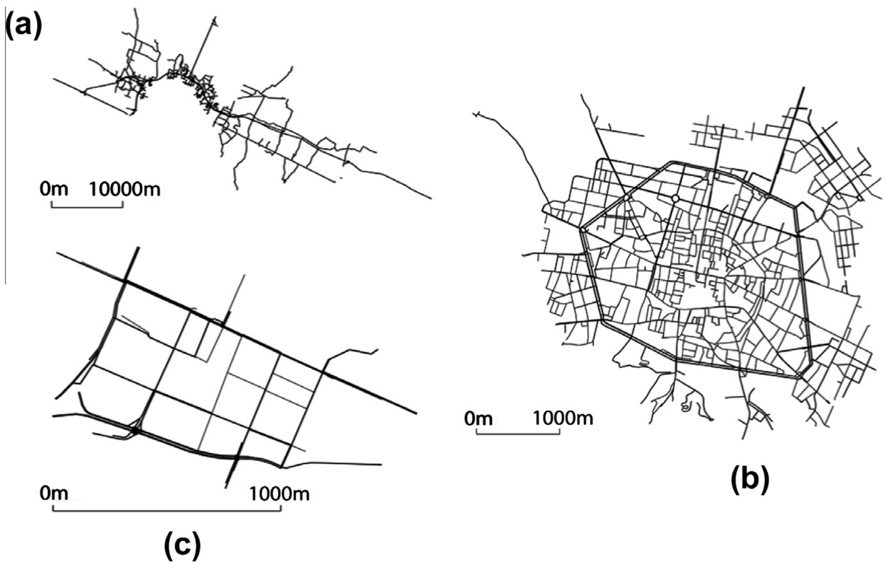

Fig. 8. Bologna traffic scenarios imported to iTETRIS.

vehicular noise emissions, iTETRIS has adopted the Harmonoise model [33] that describes different type of noises. In particular, the model includes the *rolling* and *tractive* noises produced by a vehicle, and *acoustically* sums them up to model the total noise. The Harmonoise model has been embedded into SUMO without major changes, apart from excluding the noise's propagation model.

8. iCS iTETRIS implementation

As previously explained, iTETRIS defines and implements a new central coordinating block referred to as iCS. The iCS implements the functionalities required to administrate the nodes' representations over the integrated wireless and traffic simulators. In addition, it defines the interfacing system with the other iTETRIS blocks and governs the execution of the simulation process as explained in Section 5.2. This section presents a more detailed description of the iCS implementation.

8.1. iCS architecture and functionalities

The iCS is structured into four main components. The *Traffic Simulator Communicator* is the component handling the communications with SUMO, and constitutes the client part of the TraCI interface. The *Wireless Simulator Communicator* manages the interaction with ns-3 through the client part of the iNCI. The *Applications Manager* is the component that administrates the communications with the iAPPs. Finally, the *iCS Facilities* includes the ITSC Facilities that directly support the execution of cooperative ITS applications and that for this reason are implemented in the iCS. The Traffic and Wireless Simulator Communicators were explained when the TraCI and iNCI interfaces were described in Section 7.1 and Section 6.1, respectively. As a result, the following sections describe the remaining iCS components.

8.1.1. iCS facilities

The iCS implements the ITSC Facilities that are more related and used by the iAPPs. Fig. 9 shows the structure of the iCS Facilities, where it is important to highlight the presence of the *Facilities Manager* that acts as an interface to access and modify the data contained in the different iCS Facilities. As illustrated in Fig. 9, the iCS Facilities block is made up of *Storage* and *Action Entities*. The main function of storage entities is to store data related to the traffic and wireless simulators. For example, the *Map Database* includes the static information used to describe the simulated road network topology. This database is obtained at the beginning of an iTETRIS simulation by parsing the map used in the traffic simulator as indicated in the corresponding configuration file (Fig. 3a). Locating a representation of the map also at the iCS Facilities improves the data management between iTETRIS components. In fact, the adopted solution prevents that several queries are performed to the traffic simulator every time certain road map data is needed, which would significantly slow down the simulation execution time. In addition, it allows enriching the *Map Database* by adding additional information (e.g. points of interest) to the streets, junctions or traffic lights. The *Station Database* contains information related to all the vehicles and fixed ITS stations (RSUs and other CIUs) simulated within an iTETRIS scenario. For each ITS station, the database contains static (e.g. the station type and available radio access technologies) and dynamic parameters (e.g. the position of nodes) that can be updated during the simulation. The *Messages Database* manages the application payloads of the simulated wireless messages. The payload can be built out of information retrieved from the Station Database, the Map Database and, if necessary, the cooperative ITS applications implemented in the iAPPs. To reduce the computational load, the created payloads are not sent to ns-3, but are rather stored in the *iTETRIS Facilities Payload Table* (*iFPT*) located at the Messages Database. A reference to these payloads, including their size, is passed to the iCS, which then requests ns-3 to transmit these messages. The iCS will later be informed of which nodes correctly received the transmitted messages, and will check their references to retrieve the messages' payload from the iFPT. The iCS will then create another table (*iTETRIS Facilities Message Table - iFMT*) in the Messages Database to associate a message's payload to the list of nodes that received it. This process allows a very scalable iCS implementation of the ITSC Local Dynamic Map (LDM) Facility functions, since only one table is required for all the simulated nodes (instead of one per each node).

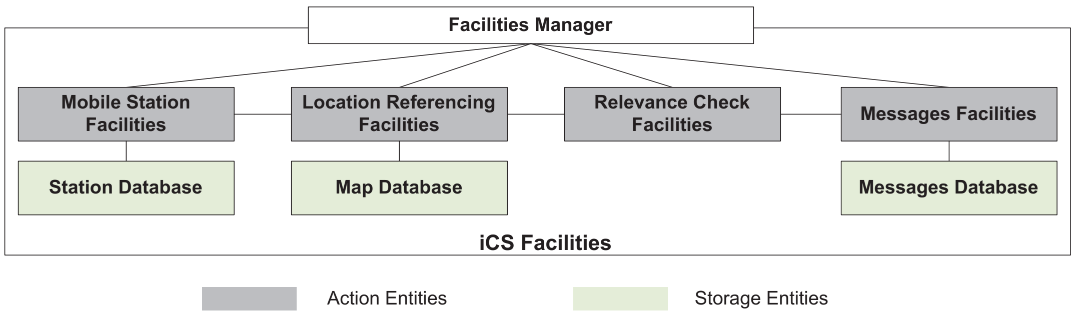

Fig. 9. Structure of the iCS facilities.

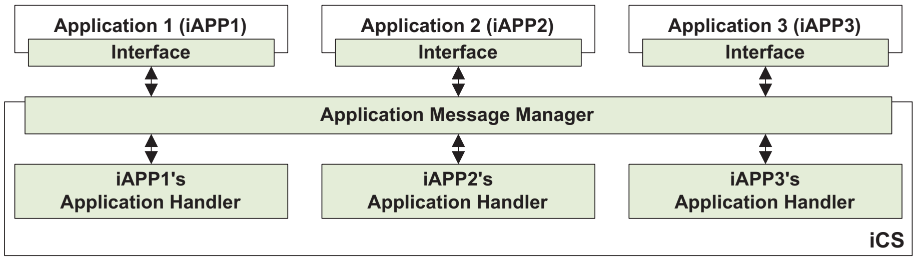
Application 1 (iAPP1)
Interface
Application 2 (iAPP2)
Interface
Application 3 (iAPP3)
Interface
Application Message Manager
iAPP1's Application Handler
iAPP2's Application Handler
iAPP3's Application Handler
iCS

Fig. 10. Structure of the applications manager.

The storage entities are accessed and controlled by the action entities. The *Mobile Station Facilities* retrieves information about the simulated vehicles (position, direction, speed, acceleration and active radio access technologies), and accordingly updates the Station Database. The *Location Referencing Facilities* provides geographic references (e.g. the road lane, street or intersection) for nodes' positions within the simulated road topology represented in the Map Database. The *Relevance Check Facilities* implements the rules to decide whether a message received by a given node is relevant for the active iAPPs. These rules have to be specified by the iAPPs during the configuration phase, and may require some location referencing and information about the receiving nodes. As a result, the Relevance Check Facilities is connected to the Location Referencing Facilities and the Mobile Station Facilities. The Messages Facilities creates the applications' payloads in the iFPT and stores the received messages in the iFMT. It is connected to the Relevance Check Facilities to identify and manage the received messages that are relevant to the iAPPs, and is also in charge of eliminating old and irrelevant messages from the iFMT and iFPT.

8.1.2. iCS applications manager

The Applications Manager provides the iAPPs with an interfacing support to interact with the iCS. It also provides the functionalities to determine the simulation timing and the data needed for their execution. To this aim, the Applications Manager relies on the *Application Handlers* and the *Application Message Manager* (Fig. 10). The Application Handlers define the time steps at which the cooperative ITS applications implemented on the iAPPs shall be executed, and the data that has to be retrieved from the iCS Facilities for their execution. This information is defined based on the iAPP's configuration files. To reduce the consumption of memory and computational resources, only one Application Handler is created for each cooperative ITS application implemented on the iAPPs. This approach also allows the iCS to distinguish between the many instances of the same application simultaneously running on different nodes. Thanks to the use of Application Handlers, the iCS just acts as an information supplier to the cooperative ITS applications implemented on the iAPPs, and thereby does not need to be redefined for every new application tested under iTETRIS. The Application Handlers are connected to the Application Message Manager, which acts as the interface between the iCS and the iAPPs. The Application Message Manager makes use of IP sockets to transfer data to/from the iAPPs by means of a predefined set of open APIs. The IP sockets allow decoupling the iCS and iAPPs from a content and a programming language point of view, which in turn avoids having to rewrite existing applications code or enforce an iTETRIS user to develop the applications' source code in a specific language. In this context, cooperative ITS applications (iAPPs) need to include the interfaces to communicate with the iCS (Fig. 10). To facilitate the use of iTETRIS, the current platform already provides open interfaces in Java, C++, and Python.

8.2. Interaction between applications and the iCS

The interaction between the iAPPs and the iCS is based on a mechanism relying on *Subscriptions* and *Result Containers*. The iAPPs contain application algorithms that use data retrieved from the iCS, and return results that may either be reused by other iAPPS or trigger actions in the traffic or wireless simulators (e.g. rerouting a vehicle after receiving a traffic jam notification). The iCS defines a *Subscription-based* mechanism to retrieve the information needed by the iAPPs. Using the subscriptions, the iAPPs can access data required by the active applications; for example, a public subscription is available in iTETRIS to retrieve the payload contained in received CAM messages. Similarly, subscriptions can be used to impose actions from the iAPPs on the traffic or wireless simulations; for example, an application implemented in an iAPP may impose the transmission of CAMs to vehicles located on a given area. The Result Containers have been implemented for the local storage of results from iAPPs that could be reused by the iCS or other iAPPs.[9]

[9] The current iTETRIS release includes a set of predefined subscriptions and result containers that cover the most common needs for simulating cooperative ITS applications (e.g. retrieving vehicle positions, enable CAM transmissions, etc.). However, application developers can easily define and integrate new types of subscriptions and result containers into iTETRIS.

The iAPPs *subscribe* to the iCS in order to periodically, or on demand, retrieve the necessary data for their internal operation, and to trigger actions over the traffic and wireless simulations. Since a large part of the information passed to/from the iCS is common to various iAPPs, a set of generic and modular *push* and *pull* subscriptions have been developed in iTETRIS[10]:

- *Subs-Get-Facilities-Info* is a pull subscription through which the iAPPs retrieve information the iCS holds in its Facilities. In particular, it is possible to retrieve information about the static road network (e.g. edgeID, junctionID stored in the Map Database), dynamic vehicle information (e.g. position, speed, acceleration stored in the Station Database), or information contained in the received CAM messages (CAM payloads stored in the iFMT database).
- *Subs-App-Cmd-Traff-Sim/Subs-App-Result-Traff-Sim* are respectively push and pull subscriptions to retrieve data from SUMO (through the iCS) or to impose SUMO actions. Typical pull data includes dynamic travel information (e.g. vehicles' travel time or route), whereas a common push data can be assigning a new route to a vehicle.
- *Subs-App-Msg-Send/Subs-App-Msg-Receive Subs-App-Msg-Send* is a push subscription that iAPPs can use to impose, using the iNCI interface, the wireless transmission of a message on an ns-3 node. The subscription currently allows iAPPs to define a specific transmission mode (geobroadcast, geounicast, unicast, etc.) as well as the transmission parameters listed in Table 2 (ServiceID, CommProfile, ListofTechnologies, PayloadLength, MsgLifeTime). *Subs-App-Msg-Receive* is a pull subscription set by the subscriber iAPP to be notified of correct ns-3 message receptions. To filter received messages, the subscription specifies the targeted message using the same format as used by the *Subs-App-Msg-Send* subscription.
- *Subs-X-App-Data* is a pull subscription used by an iAPP to retrieve particular results of other iAPPs. For example, a dynamic vehicle rerouting application may be triggered by a traffic monitoring application that detects traffic congestion in a particular area of the road network.

The algorithms implemented on the iAPPs may create results to be stored at the iCS for their further use. The iCS defines and implements application Result Containers in order to avoid over-specifying either the content or the format of such storage entities in one of its databases. From a modelling point of view, a Result Container is designed as a storage object for data processing and sharing with different nodes or iAPPs. Given the type of Result Container that a given iAPP is related with, the iCS becomes aware of the procedures to undertake whenever the iAPP generates a result value. The exact result yet remains transparent to the iCS. In this context, a user that may want to implement and test a new application over iTETRIS, will also have to implement the necessary Result Containers. The iAPP's configuration file shall specify the Result Container that will be used for collecting its results.

The iCS uses the Application Handlers to manage the interactions with the iAPPs and control the execution of the associated application algorithms. At every simulation time step, the iCS iterates through the set of nodes currently participating in the simulation to retrieve their associated Application Handlers. When an Application Handler is given the control of the communications process with its iAPP, it performs the following tasks through the Application Messenger:

1. Ask its iAPP whether it requires new subscriptions.
2. Ask its iAPP whether an existing subscription has to be dropped starting from a particular simulation time; if it is the case, the iAPP will stop receiving the subscribed data or triggering the subscribed actions over the traffic and wireless simulators.
3. Transfer the subscribed information from the iCS to the iAPP; the Application Handler checks all the existing subscriptions of the node it refers to, and according to the nature of the subscription (traffic or communications subscription), sends the appropriate data to the iAPP.
4. Inform the iAPP that it can execute its algorithms, and ask whether it is going to provide the iCS with a result that may imply triggering new actions over the traffic and wireless simulators.

9. iTETRIS beyond the state of the art

This section outlines how iTETRIS makes important advances in the field of cooperative ITS simulation by comparing its technical features and characteristics with existing cooperative ITS simulation platforms. To the authors' knowledge, none of the existing cooperative ITS simulation platforms jointly provide iTETRIS' capability to:

- Allow language-agnostic implementation and simulation of cooperative ITS applications thanks to interfaces that abstract application developers from the intrinsic technological aspects of both traffic and wireless simulators.
- Extend its open source wireless and traffic simulators independently from the internal implementation of the rest of the iTETRIS blocks. This capability is enabled by the central iTETRIS Control System (iCS) and its open interfaces.
- Provide implementation modules that are fully compliant with the ETSI standards for Intelligent Transport Systems Communications (ITSC). Consequently, iTETRIS allows testing and optimizing novel cooperative ITS applications using standard compliant systems prior to a prototype implementation and field tests.

[10] Interested readers can find examples of the encoding format used to implement the subscriptions by consulting the iAPP implementations provided along with the iTETRIS source code available on the iTETRIS community webpage: http://www.ict-itetris.eu/10-10-10-community/.

Table 4
Comparison of iTETRIS features with existing cooperative ITS simulation platforms.

Simulation Platform	Communications modelling accuracy	Mobility modelling accuracy	Modularity in comms, mobility, and application blocks	Open source availability	Capability for evolution	Support for external applications coded in different languages	Cooperative ITS communications protocols standard compliance	Support for large scale simulations
MOVES [2]	Medium	Medium	Low	Not proved	Low	No	No	Yes
AutoMesh [3]	High	Medium	Medium	Not proved	Low	Not proved	No	Yes
VANET [4]	Medium	Medium	Low	Not proved	Low	Not proved	No	Yes
GrooveSim [5]	Medium	Medium	Low	Yes	Low	No	IEEE 802.11p/1609	Yes
SWANS Estensions [10,11]	High	Medium	Low	Yes	Low	No	No	Not proved
ns-3 Extensions [12]	Very high	Medium	Low	Yes	Low	No	No	Not proved
NCTUns [7]	Very high	Medium	Low	Yes	Low	Yes	IEEE 802.11p/1609	Yes
CORSIM/ Qualnet [13]	High	High	Medium	No	Low	No	No	Not proved
VISSIM/ns-2 [14]	Very high	High	Medium	No	Low	No	No	Yes
TraNS [15]	Very high	High	Medium	Yes	Low	No	No	Not proved
Veins [16]	Very high	High	Medium	Yes	Low	No	IEEE 802.11p/1609	Yes
OVNIS [17]	Very high	High	Medium	Yes	Low	No	No	Yes
iTETRIS	Very high	High	High	Yes	High	Yes	ETSI ITSC	Yes

- Support realistic large scale simulations while accurately modelling standard-compliant cooperative ITS systems. It is worth noting that iTETRIS allows modifying the modelling accuracy based on the study objectives and constraints. This is particularly relevant in the case of modelling the wireless physical layer since it has a significant impact on the simulation time.

As demonstrated in the previous sections, iTETRIS provides several innovative modelling implementation choices and solutions concerning both wireless communications and transportation aspects. Strategic tradeoffs between modelling accuracy and simulation computational efficiency have been realized in the communications modelling. On the one hand, the already acknowledged ns-3 modelling accuracy is improved by implementing all the functionalities of the ETSI ITSC architecture at each level of the communications protocol stack, and by using validated radio propagation models tailored for each simulation scenario. On the other hand, computational efficiency is obtained by smartly simplifying most of the UMTS, WiMAX and DVB-H standards' system functionalities that are not critical in the evaluation of cooperative ITS applications at large scale. For what concerns the simulation of transportation aspects, iTETRIS improves SUMO with modelling advances in computing the pollutant and noise emissions produced by simulated vehicles, as well as their fuel consumption. Moreover, SUMO was extended with appropriate solutions for the realistic assessment of cooperative ITS applications at large scale. In this context, specific methodologies were improved to import real world road networks and traffic demands representations from other formats. Finally, new methods were implemented to retrieve values from SUMO simulated vehicles (e.g. fuel consumption, noise and pollutant emissions) or to impose changes over them (e.g. speed or route changes).

Table 4 compares iTETRIS features with those of the existing cooperative ITS simulation platforms described in Section 2. The table compares important aspects such as the modelling accuracy, the simulator's modularity and capability to be extended, support for external applications, implementation of standard-compliant cooperative ITS communications protocols and technologies, and capability to simulate large scale scenarios critical to adequately evaluate the impact and effectiveness of cooperative ITS traffic management solutions. Although all the other existing cooperative ITS simulation platforms provide high modularity in protocol implementations, Table 4 confirms that iTETRIS exhibits a higher modularity in the way its communications, mobility and application blocks are separated and can be recombined. This results in a high capability for evolution, as it allows for an independent development, or even replacement, of its interconnected simulators. Moreover, its open iCS' interfacing approach permits the language agnostic-implementation of cooperative ITS applications that can interact efficiently with the ns-3 and SUMO simulators through the iCS. In addition, iTETRIS is one of the few platforms that allow for large scale evaluations of cooperative ITS applications over a complete set of cooperative ITS standard-compliant communications protocol implementations. To the authors' knowledge, iTETRIS is currently the only platform implementing the ETSI ITSC architecture.

10. Evaluation of cooperative ITS applications using iTETRIS

Several cooperative ITS applications were implemented and evaluated during the iTETRIS project to demonstrate the benefits that cooperative ITS systems can provide to improve road traffic management. To illustrate the capabilities and potential of the iTETRIS platform, this section presents two of these applications.

10.1. Cooperative traffic congestion detection

CoTEC (COoperative Traffic congestion detECtion) [34] is a novel technique that exploits ITS G5A vehicle to vehicle communications to detect road traffic congestions without any fixed infrastructure sensors. The technique uses the periodic CAM messages received from vehicles in its local neighborhood to estimate the local traffic density. The local traffic density is provided as input, together with the vehicle's speed, to a fuzzy logic process (CoTEC fuzzy rules) that continuously estimates the local traffic congestion level; the congestion level is defined as a continuous output value in the range [0–1], with 0 representing free flow conditions and 1 severe congestion. To this aim, CoTEC uses SkyComp's congestion classification system, which relates traffic density, speed and congestion levels [35]. If a congestion level exceeding a predefined threshold C_{th} is locally detected, CoTEC activates a cooperative process. The process shares and correlates estimations made by different vehicles in order to accurately characterize the overall traffic congestion over a particular road segment. For this purpose, vehicles located close to the front end of the traffic jam are responsible for the periodic generation of Cooperative Traffic Estimation Messages (CTEMs). A CTEM, initially containing the local congestion estimation level, is multi-hop forwarded towards the rear end of the jam. Every vehicle receiving a CTEM retransmits it only if it also detects a traffic congestion (i.e. its traffic density estimation $>C_{th}$). Before retransmitting the CTEM, each vehicle updates the information included in the CTEM message using its own estimate in order to generate a cooperative correlation and achieve a coherent and reliable detection. The updating process is based on the collection of congestion estimates following grouped frequency distributions. The range of congestion levels to be monitored $[C_{th} - 1]$ is divided into a number of equal congestion intervals. The CTEM payload includes as many data fields as congestion intervals are defined. Every time a vehicle forwards a CTEM, it increases the frequency of the interval in which its congestion estimate lies by the number of neighboring vehicles it detects through the reception of their CAM messages. This approach increases the statistical accuracy since it takes advantage of the fact that estimations made by vehicles geographically close to each other are relatively similar. When the CTEM finally reaches the rear end of the traffic jam, the median statistic of the traffic congestion estimates is computed based on the frequency intervals. Its value is considered to represent the overall traffic congestion estimate. Finally, it is important to emphasize that by multi-hop forwarding the CTEMs, the vehicles situated at the rear end of a traffic jam can obtain a global and complete vision of the road's congestion level. In fact, an important novelty of CoTEC is that it is not only capable to estimate the congestion level, but also the length of the traffic jam.

CoTEC has been implemented in C++ as an external iTETRIS application block (iAPP), and uses the iCS subscriptions system previously described. In particular, the CoTEC application running on vehicles uses the *Subs-Get-Facilities-Info* subscription to obtain from the iCS Facilities information about its position, speed and list of CAM messages received by neighboring vehicles. CoTEC's Application Handler is in charge of providing this information to the application at each simulation time step. At each time step, the CoTEC application first estimates the traffic density in a vehicle's local neighborhood, and then decides whether generating a CTEM message based on the detection of traffic congestion. If this is the case, the application uses *Subs-App-Msg-Send* subscriptions to request the iCS to activate the simulation of a CTEM transmission. The iCS does so by calling an iNCI primitive that will initiate the simulation of the CTEM transmission in ns-3. When such simulation is over, the iCS uses another iNCI primitive to retrieve the list of vehicles that received the CTEM. This list is used to update the iCS storage structures, which are then accessed by CoTEC's *Application Handler* to supply CoTEC's application with the list of received messages. To receive this list, CoTEC's application had to previously subscribe to it using a *Subs-App-Msg-Receive* subscription. CoTEC updates then the content of the CTEM following the mechanism previously described, and decides whether the CTEM has to be forwarded or not. If CoTEC detects that the receiving vehicle is located outside the traffic jam, it estimates the road's traffic congestion level using the median statistic previously explained. CoTEC's performance has been evaluated in iTETRIS using a real world highway scenario covering an area of 50 km by 20 km around the Italian city of Bologna (Fig. 8a). The performance evaluation has been conducted under three different traffic densities before a traffic jam takes place: Scenario A (5 veh/km/lane), Scenario B (10 veh/km/lane) and Scenario C (15 veh/km/lane). Each simulation corresponds to more than 2 h of real traffic flow. In total, up to 4500 vehicles are present in the simulated scenarios, all equipped with ETSI ITS G5A radio interfaces to transmit CAMs (every 0.5 s) and CTEMs using a transmission power of 10 dBm. Traffic congestions are artificially generated in SUMO by gradually reducing the maximum speed limit from 130 km/h to 10 km/h, which can increase the traffic density to 70 veh/km/lane. Several simulation runs have been conducted for each scenario to guarantee the statistical validity of the obtained results. CoTEC performance is compared against that obtained by an infrastructure-based monitoring solution using inductive loops. The inductive loops system locally calculates vehicular densities and average speeds, and uses them as input parameters for the same fuzzy logic process as used by CoTEC (this is done for a fair comparison). The comparison has been conducted considering different separation distances between inductive loops (from 100 m to 1000 m).

Fig. 11 depicts the congestion detection probability (i.e. the probability of successfully detecting a congestion event); *ILx* refers to inductive loops placed every *x* meters. The obtained results show that CoTEC and inductive loops are able to detect traffic congestion with a probability higher than 80% under Scenarios A and B, characterized by low and medium traffic densities in the absence of traffic congestion. However, under more dense traffic conditions (Scenario C), the performance obtained with inductive loops strongly deteriorates when the distance between loops increases. This is not the case with CoTEC that is still capable to guarantee a high congestion detection probability. Fig. 11 also shows that the probability of not detecting a congestion event (false negative detection) is notably low for CoTEC (less than 10% in all simulated scenarios). On the contrary, using Inductive Loops can increase the number of false negative detections, especially in case of dense traffic conditions (in Scenario C, false negatives are detected with a probability of almost 45% for IL1000). The results indicate that whereas CoTEC is always able to reliably detect congestion, the technique based on inductive loops can overlook congestion under certain traffic conditions. The number of false positive detections (situations in which congestion is estimated while the traffic was actually experiencing free-flow) was observed to be negligible (less than 1%) in the analyzed simulation scenarios for both compared techniques. This is due to different factors related to the functioning of the used detection mechanisms. First, two different traffic metrics (speed and density) are taken into account to estimate congestion. This guarantees

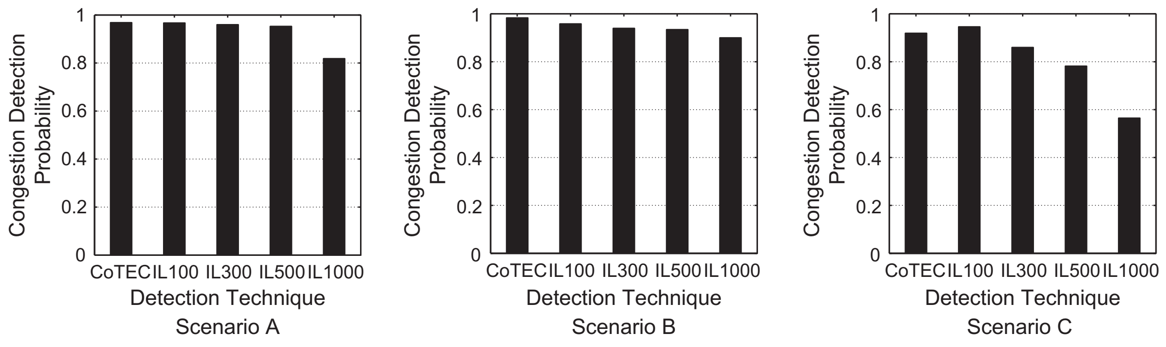

Fig. 11. Traffic congestion detection probability.

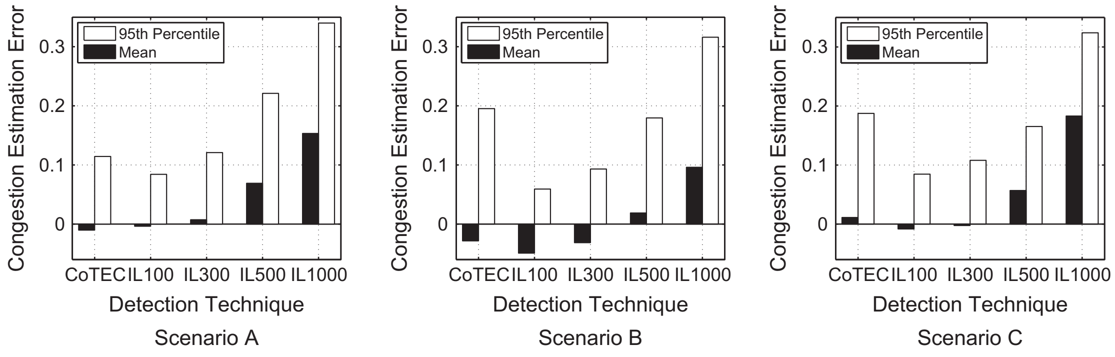

Fig. 12. Traffic congestion estimation error.

a more reliable detection in comparison with other techniques that only consider one metric (e.g. speed). In addition, the compared techniques employ a moving average window to smooth out the speed and traffic density estimates prior to the calculation of the congestion level. This prevents that short-lived variations in the traffic flow can be interpreted as traffic congestion. Additionally, CoTEC implements a cooperative process to correlatively determine whether there is a traffic jam or not. As a result, a congestion event is only detected when multiple vehicles detect it.

The techniques' accuracy in detecting the congestion level has been estimated by comparing the congestion estimates achieved with CoTEC and inductive loops to that obtained by an idealistic centralized monitoring solution that would have full access to all the traffic information for the congested road segment (this solution directly retrieves speeds and traffic densities from the SUMO traces). The obtained congestion estimation error (mean and 95th percentile) is shown in Fig. 12; a negative sign of the estimation error indicates that the congestion level is underestimated. The obtained results demonstrate that CoTEC achieves a low mean congestion estimation error close to that obtained with an expensive infrastructure-based monitoring solution that would require installing inductive loops every 100 m on the highway. CoTEC's performance is slightly degraded when analyzing the 95th percentile, especially under Scenarios B and C, although it still achieves an accuracy level greater than that obtained with inductive loops placed every 1000 m. The obtained results demonstrate that CoTEC can successfully and accurately detect and characterize congestion conditions without requiring the deployment of infrastructure sensors. In this context, iTETRIS was able to demonstrate at large scale that cooperative vehicular communications can represent an efficient and cost-attractive solution for road authorities to detect traffic jams and characterize their entity.

10.2. Cooperative bus lane management

Another cooperative ITS application implemented in the iTETRIS project is cooperative bus lane management [36]. The application automatically detects conditions of high traffic density, and tries improving the traffic flow by temporary allowing private vehicles to use the bus lanes. To do so, RSUs deployed along the road network continuously monitor the vehicles' speed through their CAM messages. The RSUs compute then the average speed over time periods of 5 min. If the estimated average speed is below a certain threshold representing normal driving conditions, the RSUs will inform ITS G5A vehicles using a broadcast message that they can use the bus lanes. The vehicles use the bus lanes if they reduce their travel time, which is estimated using their navigation system.

The implementation of the bus lane management application is left out of the paper since it follows similar principles as those described for the cooperative traffic congestion detection application. The bus lane management application was evaluated in a 2 km by 2 km urban scenario around a football stadium in the Italian city of Bologna (Fig. 8c). Nine RSUs were deployed at the intersections that had a major influence on the traffic and operation of bus lanes. The simulations were conducted using real traffic demands and vehicle routes measured during a time period of one hour just before the beginning of a football match. The total number of simulated vehicles is larger than 4000 for all the simulation runs. The percentage of vehicles equipped with ITS G5A (penetration rates) was varied from 10% to 100%. ITS G5A vehicles transmit CAM messages with a 1 Hz frequency. To evaluate the benefits of the cooperative bus lane management application, the average vehicles' travel time was compared when the application was active and when it was not (Fig. 13). The depicted results differentiate between distinct vehicle classes: buses, vehicles equipped with ITS G5A (equipped), vehicles not equipped with ITS G5A (not equipped), and vehicles equipped with ITS G5A that decide to use the bus lane as a result of a broadcast notification (rerouted); as previously mentioned, ITS G5A vehicles will only use the bus lane if they estimate it will improve their travel time. Buses are distinguished in Fig. 13 in order to analyze whether public transport is negatively influenced by the deployment of cooperative bus lane management applications.

The results illustrated in Fig. 13 show that the travel time can be reduced for almost all vehicle classes with ITS G5A penetration rates below 60%. However, the travel time increases with higher penetration rates. This is due to the fact that if too

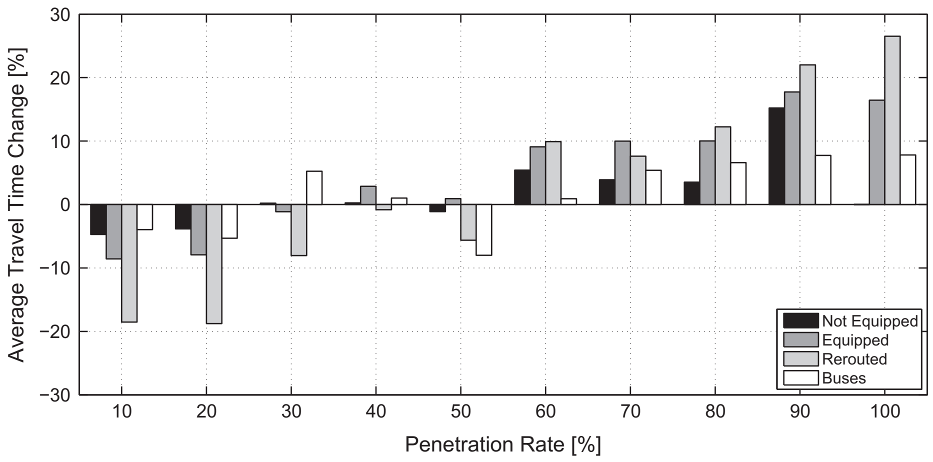

Fig. 13. Average travel time difference per vehicle class and ITS G5A penetration rate.

many vehicles reroute to use the bus lanes, the probability for vehicles to be queued behind a bus augments. The simulated urban Bologna scenario contains many narrow streets where vehicles cannot overtake buses. As a result, increasing the probability for vehicles to be queued behind a bus increases the travel time. The obtained results provide an interesting insight into the potential of cooperative bus lane management applications, but also show that to obtain the maximum benefit more advanced solutions than the one here analyzed need to be designed. In particular, the negative effects measured under high ITS G5A penetration rates are due to the fact that rerouted vehicles took the decision to modify their route without considering the impact on the use of bus lanes that other rerouted vehicles will create. Considering such impact in the travel time estimation before taking a rerouting decision has been shown here to be of critical importance; the optimization of the cooperative bus lane management is out of the scope of this paper. In this context, it is important to emphasize that the conducted iTETRIS cooperative ITS applications study has highlighted both the benefits and challenges that cooperative ITS technologies might create in the future.

10.3. iTETRIS simulation performance

Analysis conducted within the iTETRIS project showed that simulating wireless communications (in particular, simulating the wireless physical layer with significant modelling accuracy) had a higher impact on the simulation time than the exchange of messages among iTETRIS blocks (to date, the platform has been used executing all these blocks on the same server). Similar observations were reported in [20] that explain how the detailed modelling of the lower layers of the wireless communications stack can considerably increase the computation time and required memory resources. When simulating large scale standard compliant cooperative ITS deployments, the simulation time increases as a result of the exchange of standard periodic broadcast CAM messages among hundreds of vehicles. However, the modelling accuracy can be modified based on the study objectives. For example, while a study of cooperative safety applications would probably require a 10 Hz CAM transmission frequency, such frequency could be reduced to just 1 Hz in the case of traffic management applications without having a negative impact on the outcomes of the study. To quantify the iTETRIS simulation execution times, simulation runs of the use cases described in Section 10.1 were analyzed on a 2.93 GHz Quad-Core Xeon HP Proliant DL 580 G5 server with 8 GB RAM. The resulting execution times refer to 2 h of real time in the traffic scenario B, with over 800 vehicles present on average at each moment and all periodically broadcasting CAM messages with a frequency of 2 Hz. The simulations were conducted with a time step of one second, which is the value supported by the current iTETRIS release, but can be modified by changing part of the iCS internal logic. With these settings, the maximum required simulation execution time was 311 min and the maximum RAM consumption was 1.2 GB, which are acceptable values for cooperative large scale scenarios.

11. Conclusions

Cooperative ITS systems will play a fundamental role towards achieving a safer, greener and more efficient road traffic mobility. The complexity and holistic nature of cooperative ITS technologies require an extensive testing and evaluation of their operation and performance before considering their on-field deployment. In this context, this paper has presented the iTETRIS simulation platform, an efficient and modular open source simulation platform developed to study and optimize cooperative ITS applications in large scale scenarios. The platform is characterized by a modular architecture that integrates two open source traffic and wireless simulators through a central coordinating entity, and facilitates the interfacing with

Table A1
List of abbreviations.

C2C	Car-to-Car	ITS	Intelligent Transportation Systems
CALM	Communications Access for Land Mobiles	ITS G5	European profile standard for the PHY and MAC layer of ITS operating in the 5 GHz frequency band
CAM	Cooperative Awareness Message	ITSC	Architecture for Intelligent Transport Systems Communications
CIU	Communications Infrastructure Unit	LDM	Local Dynamic Map
CTEM	Cooperative Traffic Estimation Message	LOS	Line-of-Sight
CoTEC	COoperative Traffic congestion detECtion	MW	Middleware Node
DENM	Decentralized Environmental Notification Message	NLOS	Non-Line-of-Sight
ETSI	European Telecommunications Standards Institute	PER	Packet Error Rate
FOTs	Field Operational Tests	RSU	Roadside Unit
GIS	Geographic Information System	SINR	Signal to Noise and Interference Rate
HEBFA	Handbook of Emission Factors	TMC	Traffic Management Center
iAPP	iTETRIS implementation of a cooperative ITS APPlication	TraCI	Traffic Control Interface
iCS	iTETRIS Control System	V2I	Vehicle-to-Intersection
iFMT	iTETRIS Facilities Message Table	V2V	Vehicle-to-Vehicle
iFPT	iTETRIS Facilities Payload Table	WAVE	Wireless Access in Vehicular Environments
iNCI	iTETRIS Network Simulator control Interface		

external modules where cooperative ITS applications can be implemented in different programming languages. The extensions made to the ns-3 and SUMO platforms enable the study of fully ETSI ITS standard compliant cooperative ITS systems. The paper includes a detailed description of the platform, and shows how cooperative ITS applications can be integrated in iTETRIS. In this context, the paper also includes examples of cooperative ITS applications implemented and tested in iTETRIS, highlighting the capability of iTETRIS to conduct unprecedented large scale cooperative ITS investigations with high modelling accuracy.

Acknowledgments

This work has been partly funded by the European Commission through the FP7 ICT Project iTETRIS: An Integrated Wireless and Traffic Platform for Real-Time Road Traffic Management Solutions (Project Number: FP7 224644). The authors wish to acknowledge the Commission for the support.

Appendix A

See Table A1.

References

[1] ETSI TC ITS, Intelligent Transport Systems (ITS); European Profile Standard on the Physical and Medium Access Layer of 5 GHz ITS, Standard ETSI ES 202 663 v1.1.0, 2010.

[2] L. Bononi, M. Di Felice, G. D'Angelo, M. Bracuto, L. Donatiello, MoVES: a framework for parallel and distributed simulation of wireless vehicular ad hoc networks, Elsevier Computer Networks 52 (1) (2008) 155–179.

[3] R. Vuyyuru, K. Oguchi, Vehicle-to-vehicle ad hoc communication protocol evaluation using realistic simulation framework, in: Proceedings of the 4th IEEE/ IFIP Wireless on demand Networks and Services (WONS '07), 2007, pp. 100–106.

[4] C. Gorgorin, V. Gradinescu, R. Diaconescu, V. Cristea, L. Iftode, An integrated vehicular and network simulator for vehicular ad-hoc networks, in: Proceedings of the European Simulation and Modelling Conference (ESM), 2006, pp. 1–8.

[5] R. Mangharam, D. Weller, R. Rajkumar, P. Mudalige, F. Bai, GrooveNet: a hybrid simulator for vehicle-to-vehicle networks, in: Proceedings of the Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services, 2006, pp. 1–8.

[6] J. Härri, M. Fiore, F. Filali, C. Bonnet, Vehicular mobility simulation with VanetMobiSim, SAGE SIMULATION 87 (4) (2011) 275–300.

[7] S.Y. Wang, C.L. Chou, NCTUns tool for wireless vehicular communication network researches, Elsevier Simulation Modelling Practice and Theory 17 (7) (2009) 1211–1226.

[8] M. Fiore, J. Haerri, The networking shape of vehicular mobility, in: Proceedings of the Ninth ACM Symposium on Mobile Ad-hoc Networking Computing (MobiHoc), 2008, pp. 261–272.

[9] J. Gozalvez, M. Sepulcre, R. Bauza, Impact of the radio channel modelling on the performance of VANET communication protocols, Springer Telecommunication Systems 50 (3) (2012) 149–167.

[10] D. Choffnes, F. Bustamante, An integrated mobility and traffic model for vehicular wireless networks, in: Proceedings of the second ACM Workshop on Vehicular Ad Hoc Networks (VANET), 2005, pp. 69–78.

[11] K. Ibrahim, M.C. Weigle, ASH: Application-aware SWANS with highway mobility, in: Proceedings of the IEEE INFOCOM Workshops, 2008, pp. 1–6.

[12] H. Arbabi, M. Weigle, Highway Mobility and Vehicular ad-hoc Networks in ns-3, in: Proceedings of the 2010 Winter Simulation Conference, 2010, pp. 2991–3003.

[13] H. Wu, J. Lee, M. Hunter, R.M. Fujimoto, R.L. Guensler, J. Ko, Efficiency of simulated vehicle-to-vehicle message propagation on Atlanta's I-75 corridor, Transportation Research Record: Journal of the Transportation Research Board (2005) 82–89.

[14] Multiple Simulator Interlinking Environment (MSIE) for C2CC in VANETs. <http://www.cn.uni-duesseldorf.de/projects/MSIE>.

[15] M. Piorkowski, M. Raya, A. Lezama Lugo, P. Papadimitratos, M. Grossglauser, J.P. Hubaux, TraNS: realistic joint traffic and network simulator for VANETs, ACM SIGMOBILE Mobile Computing and Communications Review 12 (1) (2008) 31–33.
[16] C. Sommer, R. German, F. Dressler, Bidirectionally coupled network and road traffic simulation for improved IVC analysis, IEEE Transactions on Mobile Computing 10 (1) (2011) 3–15.
[17] Y. Pigné, G. Danoy, P. Bouvry, A platform for realistic online vehicular network management, in: Proceedings of IEEE GLOBECOM Workshops, 2010, pp. 595–599.
[18] ETSI TC ITS, Intelligent Transport Systems (ITS); Communications Architecture, Standard ETSI EN 320 665, v 1.1.1, 2010.
[19] E. Egea-Lopez, J. Vales-Alonso, A. Martinez-Sala, P. Pavon-Mario, J. Garcia-Haro, Simulation scalability issues in wireless sensor networks, IEEE Communications Magazine 44 (7) (2006) 64–73.
[20] J. Gozalvez et al., iTETRIS: the framework for large scale research on the impact of cooperative wireless vehicular communications systems in traffic efficiency, in: Proceedings of the ICT-MobileSummit 2009, 2009, pp. 1–10.
[21] S. Krauß, Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics, PhD Thesis, Mathematical Institute of the University of Cologne, 1998.
[22] M. Behrisch, L. Bieker, J. Erdmann, D. Krajzewicz, SUMO – simulation of urban MObility: an overview, in: Proceedings of the Third International Conference on Advances in System Simulation (SIMUL 2011), 2011, pp. 63–68.
[23] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrueck, S. Fischer, J.P. Hubaux, TraCI: An Interface for Coupling Road Traffic and Network Simulators, in: Proceedings of the 11th Communications and Networking Simulation Symposium (CNS), 2008, pp. 155–163.
[24] ETSI TC ITS, Intelligent Transport Systems (ITS); Users & Applications requirements; Facilities layer structure, functional requirements and specifications, Draft ETSI TS 102 894 v1.0.2, 2012.
[25] ETSI TC ITS, Intelligent Transport Systems (ITS); Communications; Architecture; Vehicular Communications, Part 4: Geographical Addressing and Forwarding for Point-to-Point and Point-to-Multipoint Communications; Sub-part 1: Media-Independent Functionality, Draft ETSI EN 302 636–4-1 v0.0.2, 2012.
[26] B.H. Walke, P. Seidenberg, M.P. Althoff, UMTS: The Fundamentals, Wiley, 2003.
[27] IEEE Standards Association, Local and Metropolitan Networks - Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems, Amendment 2: Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands and Corrigendum 1, Standard IEEE 802.16e-2006, 2006.
[28] ETSI, Digital Video Broadcasting (DVB); Transmission System for Handheld Terminals (DVB-H), European Standard EN 302 304, 2004.
[29] J. Farooq and T. Turletti, An IEEE 802.16 WiMAX Module for the NS-3 Simulator, in: Proceedings of the 2nd International Conference on Simulation Tools and Techniques (SIMULTOOL 2009), 2009, pp. 1–11.
[30] L. Cheng, B.E. Henty, D.D. Stancil, Fan Bai, P. Mudalige, Mobile vehicle-to-vehicle narrow-band channel measurement and characterization of the 5.9 GHz dedicated short range communication (DSRC) frequency band, IEEE Journal on Selected Areas in Communications, 25 (8) (2007) 1501–1516.
[31] WINNER consortium, D1.1.2, WINNER II channel models, WINNER European Research project Public Deliverable, 2007.
[32] W. Schnabel, D. Lohse, Grundlagen der Straßenverkehrstechnik und der Verkehrsplanung, Verlag für Bauwesen, Berlin, 1997. pp. 557–577.
[33] R. Nota, R. Barelds, D. Van Maercke, Harmonoise WP 3 – Engineering Method for Road Traffic and Railway Noise after Validation and Fine-Tuning, Harmonoise Technical, Report HAR32TR-040922-DGMR20, 2005.
[34] R. Bauza, J. Gozalvez, Traffic congestion detection in large-scale scenarios using vehicle-to-vehicle communications, Elsevier Journal of Network and Computer Applications, Special Issue on Vehicular Communications and Applications. Available from: <http://dx.doi.org/10.1016/j.jnca.2012.02.007>.
[35] Skycomp, Major Highway Performance Ratings and Bottleneck Inventory – State of Maryland – Spring 2008, 2009.
[36] L. Bieker, D. Krajzewicz, Evaluation of opening bus lanes for private traffic triggered via V2X communication, in: Proceedings of the IEEE Forum on 1101 Integrated and Sustainable Transportation System (FISTS), 2011, pp. 48–53.