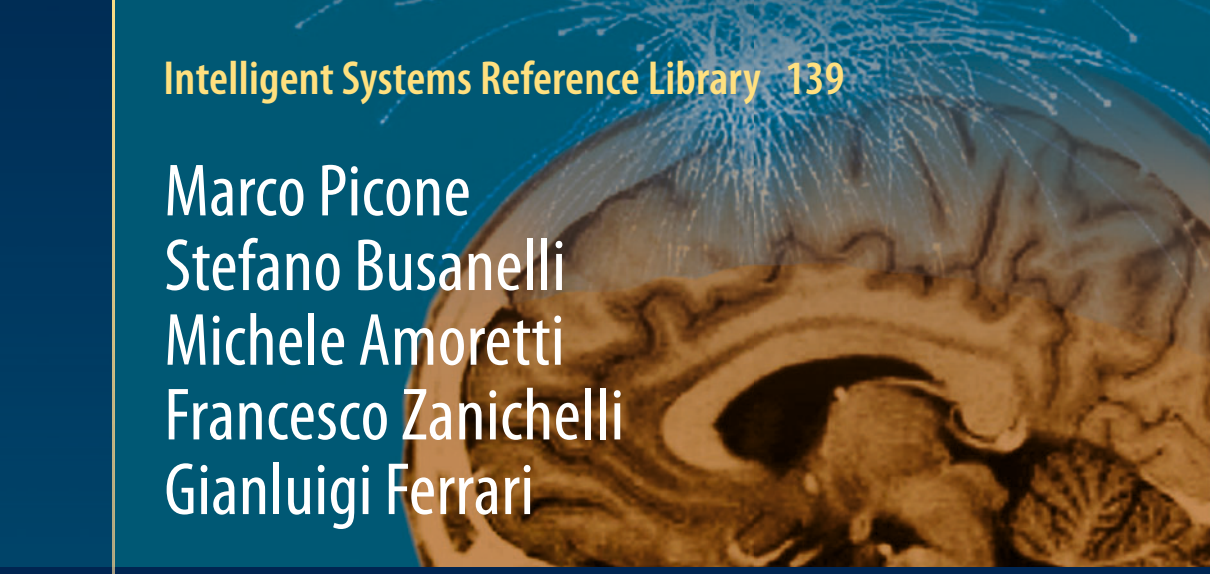
Intelligent Systems Reference Library 139



Marco Picone
Stefano Busanelli
Michele Amoretti
Francesco Zanichelli
Gianluigi Ferrari

Advanced Technologies for Intelligent Transportation Systems

Springer

Intelligent Systems Reference Library

Volume 139

About this Series

The aim of this series is to publish a Reference Library, including novel advances and developments in all aspects of Intelligent Systems in an easily accessible and well structured form. The series includes reference works, handbooks, compendia, textbooks, well-structured monographs, dictionaries, and encyclopedias. It contains well integrated knowledge and current information in the field of Intelligent Systems. The series covers the theory, applications, and design methods of Intelligent Systems. Virtually all disciplines such as engineering, computer science, avionics, business, e-commerce, environment, healthcare, physics and life science are included.

More information about this series at http://www.springer.com/series/8578

Marco Picone · Stefano Busanelli
Michele Amoretti · Francesco Zanichelli
Gianluigi Ferrari

Advanced Technologies for Intelligent Transportation Systems

Springer

Marco Picone
Michele Amoretti
Francesco Zanichelli
Gianluigi Ferrari
Dipartimento di Ingegneria
dell'Informazione
Università degli studi di Parma
Parma
Italy

Stefano Busanelli
Università degli studi di Parma
Parma
Italy

ISSN 1868-4394
ISBN 978-3-319-10667-0
DOI 10.1007/978-3-319-10668-7

ISSN 1868-4408 (electronic)
ISBN 978-3-319-10668-7 (eBook)

Library of Congress Control Number: 2014947668

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

We cannot teach people anything; we can only help them discover it within themselves.

Galileo Galilei

The beginning of all science is wondering why things are the way they are.

Aristotele

To my parents and all the people that supported me during these years

Marco Picone

To all who have loved me, to all whom I have loved

Stefano Busanelli

To my parents and Annalisa, for their endless support

Michele Amoretti

To Niccolò and Rita, my only wealth

Francesco Zanichelli

To Sofia, Viola, and Anna, the north stars of my sky

Gianluigi Ferrari

Foreword

Intelligent Transportation Systems (ITS) have lately earned center stage as some of the most relevant technologies to academia, industries, and the society in general. Their strong suit is *transversality*, as the development of ITS straddles several different research fields (e.g., from communication to logistics), and has a dramatic impact on many real-world aspects.

Applications supported by ITS range from safety to entertainment, and all of them will lead to tangible improvements of our daily life. Among such applications, those that aim at reducing energy consumption and carbon footprint deserve special mention, as they make ITS one of the green technologies that will define an eco-friendly society.

It is therefore evident that introductory and teaching material on ITS, as well as in-depth studies in this field, are precious resources for students, engineers, researchers, and anyone who would like to enhance her knowledge on the world we live in.

This book, *Advanced Technologies for Intelligent Transportation Systems*, is the work of one of the most renowned groups of experts in the field. It first presents the fundamental aspects of ITS in a tutorial manner, then it moves toward more advanced topics. It successfully covers a vast range of technical aspects such as communication, networking, security, applications. Additionally, the book provides an insightful overview of the major analytical and experimental methodologies for the study of ITS, which is one of its most unique merits.

This book thus represents the definitive guide to ITS: an excellent reference for students as well as for researchers working in the field. I am certain that all readers will enjoy it.

Turin, May 2014 Carla Fabiana Chiasserini

Preface

This books is by no means a treatise on all aspects of Intelligent Transportation Systems (ITSs). Rather, it attempts to present a unified perspective on ITS, encompassing a few advanced technologies which we came in touch with during part of our research activity in the last years. In particular, one of the peculiarities of this book is the presentation of possible solutions at various communication layers, encompassing both computer science-oriented (high layers) and telecommunication-oriented (low layers) perspectives. Along the way, we describe, in a coherent fashion, a number of interwoven innovative technologies. The approach is thus inherently cross-layer, in the sense that we cover different wireless communication protocols, but we also take into account application-level services. The intended audience is academic and industrial professionals, with good technical skills in information and communication technologies. To ease reading, we have limited as much as possible the mathematical details, which are mostly reported in the appendices of the book.

The contents of the book flow from a preliminary regulatory overview to more technical issues. The synopsis can be summarized as follows. The *first chapter* presents ITS principles and a brief standardization history, comparing European and US visions. Emerging worldwide ITS architectures are also illustrated, together with the most relevant envisioned ITS applications. The *second chapter* goes more deeply into the analysis of the communication paradigms and technologies that enable ITSs. Key challenges in vehicular networks are discussed, taking into account Vehicle-to-X (V2X) communications. A survey of the literature on centralized client/server and decentralized Peer-to-Peer (P2P) vehicular networks is proposed. This chapter terminates with the presentation of the most important enabling communication technologies for future ITSs, namely: cellular networks, WiFi, IEEE 802.11p, WAVE and ETSI ITS. The *third chapter* is fully devoted to wireless communications for Vehicular Ad hoc NETworks (VANETs). We first investigate probabilistic broadcast protocols with silencing, a recursive analytical performance evaluation framework and simulations. Then, we analyze the performance of VANETs as distributed wireless sensor networks. The *fourth chapter* presents X-NETAD, a hierarchical architecture for "cross-network" ITS

communications. Experimental results are illustrated and discussed. The *fifth chapter* focuses on application-level distributed algorithms for ITS. In particular, the Distributed Geographic Table (DGT) P2P overlay scheme is presented, and its performance is evaluated, relying on both analytical and simulation results. The DGT for Vehicular Networks (D4V) architecture, supporting a number of ITS applications, is finally presented.

We remark that the specific protocols and architectures considered in this book are "representative," as opposed to "optimal." In other words, we set to write this book mainly to provide the reader with our (limited) view on the subject. Our hope is that this book will be interpreted as a starting point and a useful comparative reference. Some of the tools used in the book (for example, the simulator DEUS) are open-source and available to the interested reader.

It is our pleasure to thank all the collaborators and students who were with us during the years of research which have led to this book, collaborating with our two groups at the Department of Information Engineering of the University of Parma: the Wireless Ad hoc and Sensor Networks (WASN) Lab and the Distributed Sytems Group (DSG). We cannot thank them one by one, but their contributions were instrumental to get here. Finally, we express our sincere gratitude to Springer for giving us the opportunity to complete this project. In particular, we are indebted to Dr. Cristoph Bauman, who believed in this project from the very beginning, and to Mrs. Janet Sterritt-Brunner, our production project coordinator, who was very kind and (above all) very patient.

Parma, June 2014

Marco Picone
Stefano Busanelli
Michele Amoretti
Francesco Zanichelli
Gianluigi Ferrari

Contents

Acronyms

ABC	Always Best Connected
AC	Access Class
ACK	Acknowledgment
AIFS	Arbitration Inter-Frame Space
AP	Access Point
AS	Application Server
ASV	Advanced Safety Vehicle
BA	Basic Access
BC	Backoff Counter
BPAB	Binary Partition Assisted Protocol
BPSK	Binary Phase Shift Keying
BS	Base Station
BSS	Basic Service Set
BTP	Basic Transport Protocol
C2C-CC	Car 2 Car Communications Consortium
CALM	Communications Access for Land Mobiles
CAM	Cooperative Awareness Message
CCK	Complementary Code Keying
CCP	Cluster Confirmation Packet
CDF	Cumulative Distribution Function
CHE-IF	Cluster-Head Election IF
CIP	Cluster Initialization Packet
C-ITS	Cooperative ITS
COTS	Commercial Off-The-Shelf
CP	Coverage Percentage
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
D2D	Device-to-Device
D4V	DGT for VSN
DCF	Distributed Coordination Function
DENM	Decentralized Environmental Notification Message

DFE	Distance From Event
DGT	Distributed Geographic Table
DHT	Distributed Hash Table
DIFS	Distributed InterFrame Space
DK	DGT Kernel
DMH	DGT Message Handler/Dispacher
DSRC	Dedicated Short-Range Communications
DSSS	Direct Sequence Spread Spectrum
EC	European Commission
ECDA	Enhanced Distributed Channel Access
ECU	Engine Control Unit
EG	Event Generator
EIFS	Extended IFS
EMDV	Emergency Message for Vehicular Environments
ERP	Extended Rate PHY
ETSI	European Telecommunications Standards Institute
EU	European Union
FCC	Federal Communications Commission
FCS	Frame Control Sequence
FHSS	Frequency Hopping Spread Spectrum
FHWA	Federal Highway Administration
FNTP	Fast Networking and Transport Layer protocol
FSAP	Fast Service Advertisement Protocol
FTM	Fluid Traffic Model
GB	GeoBucket
GDP	Gross Domestic Product
GM	Geo-Bucket Manager
GP	Global Position
GPA	Global Positioning System
HCCA	HCF Controlled Channel Access
HCF	Hybrid Coordination Function
HR/DSSS	High Rate Direct Sequence Spread Spectrum
I2V	Infrastructure-to-Vehicle
IBSS	Independent BSS
ICT	Information Communication Technologies
IDM-LC	Intelligent Driver Motion with Lane Changes
IF	Irresponsible Forwarding
ITS	Intelligent Transportation System
ITSC	ITS Communication
IVC	Inter-Vehicular Communication
IVCES	In-Vehicle Communications and Entertainment System
LM	Location Manager
LocHNESs	Localizing and Handling Network Event System
LTE-A	LTE Advanced
M2M	Machine-to-Machine

MAC	Medium Access Control
MAF	Mass Air Flow
MANET	Mobile Ad-hoc NETworks
MBMS	Multimedia Broadcast Multicast Service
MCDS	Minimum Connected Dominant Set
MIMO	Multiple-Input Multiple-Output
ML	Map Loader
MM	Mobility Model
NHTSA	National Highway Traffic Safety Administration
NPE	Node Position Error
OBU	On-Board Unit
OFDM	Orthogonal Frequency Division Multiplexing
OIS	On-board Infotainment System
P2P	Peer-to-Peer
PAF	Probability Assignment Function
PATH	Partners for Advanced Transportation TecHnology
PCF	Point Coordination Function
PDF	Probability Density Function
PER	Packet Error Rate
PMF	Probability Mass Function
PP	Probe Packet
PU	Primary User
QoS	Quality of Service
RB	Resource Block
RE	REachability
RITA	Research and Innovative Technology Administration
RSU	Road Side Unit
RTD	Round-Trip Delay
RTS/CTS	Ready-To-Send/Clear-To-Send
RTSM	Real-Time Simulation Monitoring
SB	Smart Broadcast
SDO	Standards Development Organizations
SIFS	Short Inter Frame Space
SL	Sip2Peer Layer
SM	Subscription Manager
SNR	Signal to Noise Ratio
SS	Switch Station
SSWE	Switch Station Web Editor
TD	Transmission Domain
TE	Transmission Efficiency
TSF	Timing Synchronization Function
TTL	Time To Live
TXOP	Transmission Opportunity
UDP	User Data Protocol
UI	User Interface

UMB	Urban Multihop Broadcast
USA	United States of America
USDOT	United States Department of Transportation
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-X
VSN	Vehicular Sensor Network
WAVE	Wireless Access in Vehicular Environments
WLAN	Wireless Local Area Network
WPA	WiFi Protected Access
WSA	WAVE Service Advertisement
WSMP	WAVE Short Message Protocol
X-NETAD	Cross-Network Effective Traffic Alert Dissemination

Chapter 1
Introduction

1.1 Principles and Challenges

Intelligent Transportation Systems (*ITSs*) promise to hugely improve safety, efficiency and sustainability of our transportation system, by means of a massive adoption of Information Communication Technologies (ICTs) [1–3]. Not surprisingly, in last decades ITSs have attracted the worldwide interest of researchers, automotive companies, and governments. In order to create an economically sustainable ITS ecosystem, a large number of projects have been conducted by institutions from all around the world [4]. For instance, the Advanced Safety Vehicle (ASV) program in Japan [5], the IntelliDrive project in the United States [6], and, in Europe, the numerous projects coordinated by the Car 2 Car Communications Consortium (C2C-CC) [7], strongly supported by the European Commission [8] and by the European Telecommunications Standards Institute (ETSI) [9].

In the marketplace, ITSs boast a long series of success histories, carried out by either car manufacturers (with active safety systems), toll road infrastructures operators (with Electronic Tolling Systems), insurance companies (with black boxes), Internet companies (with traffic information systems). However, current ITS hardware, software, and communication technologies are closed, i.e., unable to share data and cooperate together. In other words, current ITS applications are implemented as "silos", thus yielding to equipment duplication and no data sharing. Such a fragmented approach is typical of the first development phase of new technologies, where innovation is driven by pioneers. Figure 1.1 illustrates some significant ITS applications, implemented according to the stand-alone or not-cooperative approach.

Next years' biggest challenge will be to achieve a *Cooperative ITS* (*C-ITS*) ecosystem, where secured data are shared across several ITS applications developed by independent actors, leveraging on a solid basis of international standards, as represented in Fig. 1.2. Such a C-ITS ecosystem would facilitate actions and decisions that improve transportation safety, sustainability, efficiency and comfort beyond that achievable by stand-alone ITS systems.

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7_1

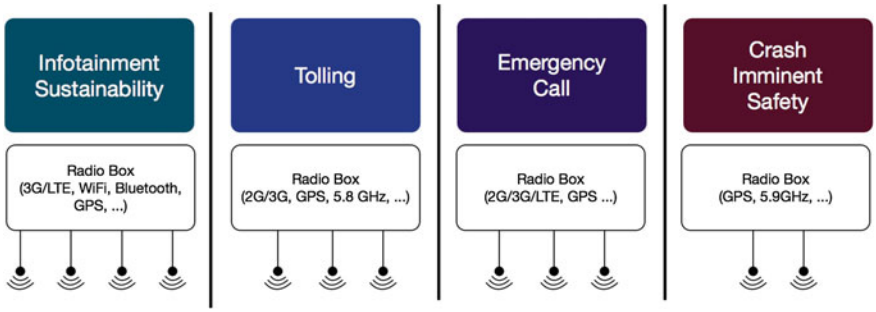


Fig. 1.1 Stand alone ITS

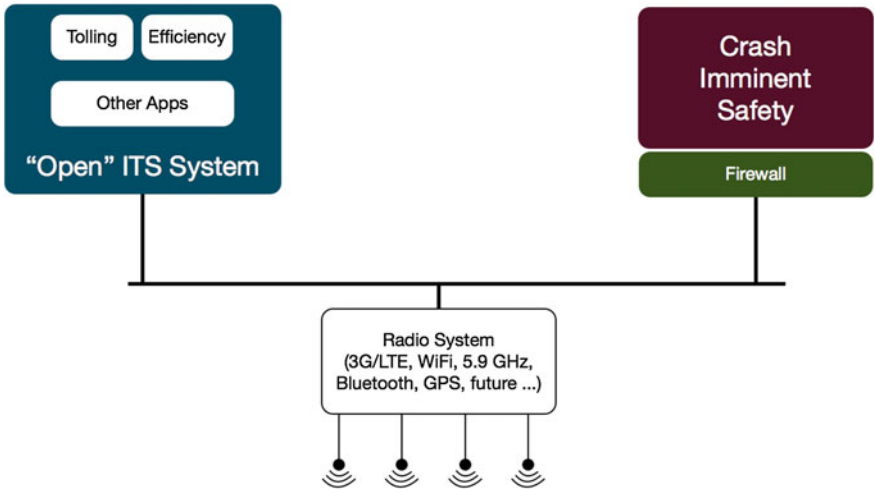


Fig. 1.2 Cooperative ITS

Future integrated ITSs will exploit vehicles provided with sensorial, cognitive, decision and communication functionalities. Therefore, such *smart* vehicles will be able to perceive the surrounding environment, collecting both public-interest information (e.g., air pollution measurements) or obtaining data for the autonomous real-time management of the vehicle itself [10–13]. Current vehicles are equipped with a large number of sensors—over 100, but it depends on the maker and model. Examples are oxygen sensor, crankshaft and camshaft position sensors, mass air flow (MAF) sensor, coolant temperature sensor, etc. All these sensors send data to the *Engine Control Units* (*ECUs*), which are embedded systems that control one or more of the electrical systems/subsystems of the vehicle. In a recent interview, Bill Ford, executive chairman of Ford Motor Company, observed that as computer involvement

becomes more active, cars will drive themselves in platoons—groups of vehicles linked on the highway for efficiency, eliminating traffic accidents at intersections.[1]

1.2 Standardization History and Open Issues

The realization of ITSs involves an exceptionally high number of stakeholders, including public administrations, transportation authorities and companies coming from a variety of industrial sectors (vehicles manufacturers, OEM and tier-1 producers, telecommunications companies, consumer electronic, service providers). From both technological and industrial perspectives, ITSs are one of the hardest challenge faced by ICT community. The presence of worldwide harmonized standards is a key requirement to drive the success of ITSs and exploit all their potentiality.

Since the end of '90s, industrial stakeholders as well as European Union (EU), United States of America (USA), Asia governments have invested a huge amount of economical resources in the standardization process of ITSs, involving numerous Standards Development Organizations (SDOs)—IEEE, ISO, European Committee for Standardization (CEN), CENLEC, ETSI, IETF, Society of Automotive Engineers (SAE), FCC, and others.

1.2.1 Worldwide Standardization Process

ISO/TC 204 is the ISO workgroup responsible for the overall system aspects and infrastructure aspects of ITSs, as well as the coordination of the overall ISO Work Program in this field, including the schedule for standards development, taking into account the work of existing international standardization bodies. Standardization efforts in ISO TC 204 produced the Communications Access for Land Mobiles (CALM) concept—defined in standard ISO 21217:2010 and subsequent ISO 21217:2013 [14] by the Work Group 16 (WG16)—and ETSI TC ITS, based on the recent European ITS Communication Architecture. CALM and ETSI TC ITS have paved the way towards C-ITS.

The CALM System Architecture, illustrated in Fig. 1.3, is a layered solution, enabling continuous Vehicle-to-Vehicle (V2V), Vehicle-to-Infrastructure (V2I) and even Infrastructure-to-Infrastructure 2I communications. It is based on multi-channel terminals capable of connecting to a wide range of potential carriers, and on IPv6, as an unification layer of underlying technologies. Moreover, it is able to select the optimal wireless telecommunications media that are available in any particular location, and to switch to a different media when necessary. Validated by the European project called CVIS, CALM has been conceived to be able to exploit a large number of different media: Cellular Networks (2G, 3G), Infrared (highly directive

[1] http://www.wired.co.uk/news/archive/2012-02/28/bill-ford-mwc.

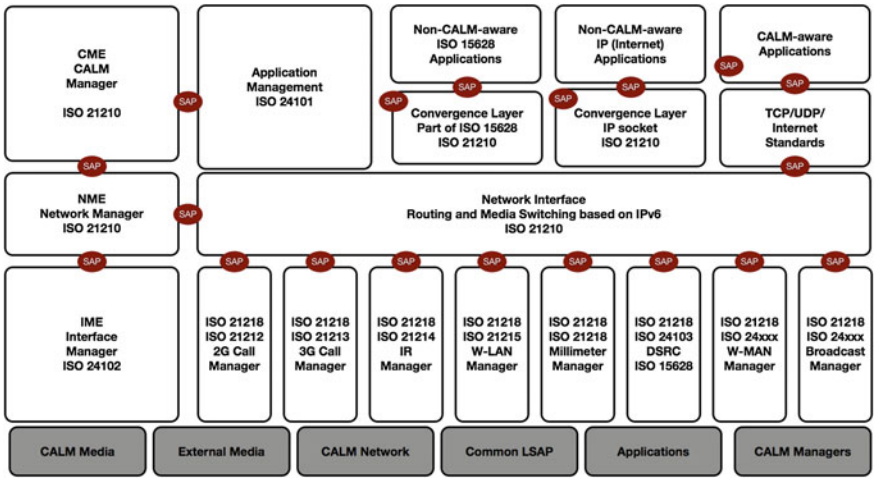


Fig. 1.3 Communications access for land mobiles (CALM) architecture

beams), Microwave CALM M5 ISO 21215—i.e., IEEE 802.11 a/b/g (WiFi) and IEEE 802.11p (mobile WiFi), providing roughly 6 Mbps up to 300 m radius—, Millimeters waves (CALM MM), and Microwaves CEN DSRC.

1.2.2 European Vision

Historically, Europe has been a worldwide leader in ITS development, principally for being the homeland of many important vehicle manufacturers. In last decades, several initiatives have been promoted either by public governments, industrial entities and standardization bodies.

1.2.2.1 Government Actions

Since long time, the European Union (EU) is quantitatively aware of the negative effects of traffic congestion, in terms of Gross Domestic Product (GDP) loss, energy efficiency reduction and CO_2 emission boost. ITSs are considered very important instruments to cope with these issues. The interest on ITSs has been materialized in a series of important political acts, with huge economical resources granted to international research projects.

One of the first European political act supporting ITSs was the foundation of the *Ertico-ITS* Europe organization in 1991, an initiative of leading members of the European Commission (EC), Transport Ministries and European companies. The goal of Ertico-ITS Europe is to bring intelligence into mobility, working together

in public/private partnerships towards zero accidents, zero delays, reduced impact on the environment, fully informed people, with affordable and seamless services, respected privacy and ensured security [15].

On June 2005, the EC started the *European Information Society 2010* (*i2010*) initiative, a comprehensive strategy for modernizing and deploying all EU policy instruments to encourage the development of the digital economy. i2010 consists of three pillars: a Single European Information Space, Innovation and Investment, and an Inclusive European Information Society. The *Intelligent Car Initiative* is one of the three Flagship initiatives proposed within the third pillar, with the objective to raise the visibility of the vital contribution of ICT to the quality of life. The Intelligent Car Initiative on smart, safe and clean transport, focuses on road vehicles and addresses safety and environmental challenges caused by increased road use.

Then, in 2008 the EC took a major step towards the deployment and use of ITS in road transport, by adopting a dedicated Action Plan [16], whose goal was to create the momentum necessary to speed up market penetration of rather mature ITS applications and services in Europe. As a direct consequence of the Action Plan, in July 2010, the European Parliament and the European Council adopted the Directive 2010/40/EU [17], establishing a legal framework for the deployment of ITSs in the field of road transport and for interfaces with other modes of transport. The Directive 2010/40/EU is an important instrument for the coordinated implementation of ITS in Europe, aiming to establish interoperable and seamless ITS services while leaving Member States the freedom to decide which systems to invest in. According to the requirements of the Directive 2010/40/EU, the EC has to define and adopt, by 2017, specifications for compatibility, interoperability and continuity of ITS solutions across the EU. The main priorities are traffic and travel information, the eCall emergency system and intelligent truck parking.

Figure 1.4 summarizes the sequence of ITS programmes funded by the European Commission (EC) in the last two decades. The first ITS project was the famous *PROgraMme for a European Traffic of Highest Efficiency and Unprecedented Safety* (*PROMETHEUS*) [18], started in 1987, that was the largest R&D project ever in the field of driverless cars. Then, there was a series of projects focused on all aspects of ITS. The first program with interventions in the field of vehicular communication has been the Fifth Framework Programme (FP5), in the 2000–2003 period, where projects like FleetNet and CarTalk2000 have been carried out. In subsequent FPs (FP6, FP7), many other projects related to car communications have been founded, such as Safespot, NoW, Coopers, GeoNet and CVIS.

1.2.2.2 Industrial Cooperation

In the ITS domain, European industries and companies have carried out a profitable cooperation since a long time. The most brilliant example, in the field of cooperative vehicular communications, is represented by the *CAR 2 CAR Communication Consortium* (*C2C-CC*),[2] a nonprofit, industry-driven organization initiated by European

[2] http://www.car-to-car.org.

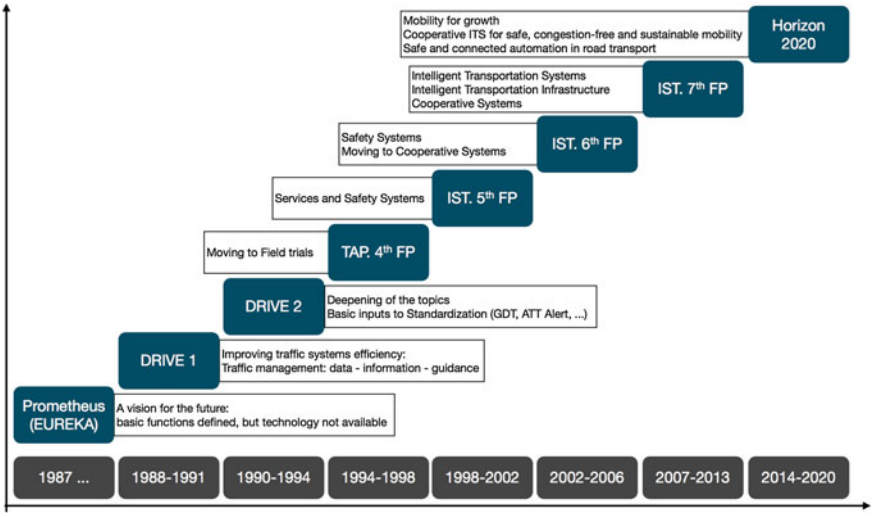


Fig. 1.4 European union roadmap on ITSs

vehicle manufacturers and supported by equipment suppliers and research organizations. C2C-CC, directly or through its partners, is usually involved in R&D projects funded by the EC, and actively cooperate with standardization bodies. The mission and the objectives of the CAR 2 CAR Communication Consortium are:

- to create and establish an open European (possibly worldwide) industry standard for CAR 2 CAR Communication Systems;
- to guarantee inter-vehicle operability;
- to enable the development of active safety applications by specifying, prototyping and demonstrating the CAR 2 CAR system;
- to promote the allocation of a royalty free European-wide exclusive frequency band for CAR 2 CAR applications;
- to push the harmonization of CAR 2 CAR Communication standards worldwide;
- to develop deployment strategies and business models to speed-up the market penetration.

Recently, C2C-CC has constituted another voluntary cooperation platform of leading European ITS stakeholders, denoted as *The Amsterdam Group*.[3] Other important members of C2C-CC are: the European professional association of operators of toll road infrastructures (ASECAP), the European organization for national roads administration (CEDR), the network of European cities and regions working together to develop innovative technologies and policies for local transport (POLIS). All the members of Amsterdam Group have signed a Memorandum of Understanding (MoU), which creates a framework for the harmonized implementation and deployment of cooperative ITS in Europe by 2015.

[3] https://amsterdamgroup.mett.nl.

1.2.3 American Vision

1.2.3.1 Government Actions

ITS development in the USA has followed an evolutive process which in in most aspects overlaps with the European and Japanese ones. In the same period of the launch of the first European ITS projects (namely Drive and Prometheus), and the second wave of Japan projects (i.e., RACS), a number of initiatives started all around the USA, under the initiative of private companies, universities, State Governments and the Federal Government. Just to name a few, we recall Smart Corridor, Pathfinder, GuideStar, and the Partners for Advanced Transportation TecHnology (PATH) projects [19]. However, at least in first years, in the USA the amount of public funds dedicated to ITS projects was significantly smaller than in Japan and Europe [19].

The principal organ of the Federal Government involved in ITS development has always been the United States Department of Transportation (USDOT), which has ofter operated through one of their agencies, such as Federal Highway Administration (FHWA), National Highway Traffic Safety Administration (NHTSA), and the Research and Innovative Technology Administration (RITA), which was created in 2005 with the mission to advance transportation science, technology, and analysis, and to improve the coordination of transportation research within the Department and throughout the transportation community. Furthermore, under the USDOT initiative, in 1991 the *Intelligent Transportation Society of America* (*ITS America*),[4] the largest organization dedicated to advance the research, development and deployment of ITSs, was founded. ITS America has played a major role in laying the groundwork at the Federal Communications Commission (FCC) and other agencies, for the use of the electromagnetic spectrum and other telecommunication infrastructures as the foundation for almost all ITSs. Thanks to such an effort, in 1999, a 75 MHz spectrum in the 5.9 GHz bandwidth was set aside by the FCC. The idea was to allocate a spectrum that could be used for the development of Vehicle-to-X (V2X) communications, without fearing potential signal interference from non-automotive users.

1.2.3.2 The ITS Strategic Research Plan

Since 2009, the strategic directions of USDOT's ITS are established in a 5-years long program, denoted as ITS Strategic Research Plan. At the time of writing this book, the ITS Strategic Research Plan 2010–2014 is being completed, while the subsequent 2015–2019 program is in course of definition. The vision of the ITS Program for 2010–2014 is to provide USA with a national, multimodal transportation system, which delivers connectivity among vehicles of all types, the infrastructure, and portable devices, thus realizing an integrated connected vehicle environment [20]. The expected outcomes of such a research include the determination of the potential

[4] http://www.itsa.org/.

benefits of connected vehicle technologies and evaluation of driver acceptance of vehicle-based safety systems, as well as the identification of research gaps and the ways to address them. Other outcomes include factual evidence needed to support a 2013 NHTSA agency decision on the deployment of these technologies for light vehicles. The ITS strategic research plan involves several ITS research areas, including connected vehicles, mobility, environment, road weather management, integrated corridor management, ITS asset viewer and multimodal transportation systems. The Connected Vehicle project is a pillar of current ITS plan and will be discussed in next section.

1.2.3.3 Connected Vehicle Program

The Connected Vehicle program is a large set of research activities related to vehicles equipped with communications and processing power, able to communicate with each other and with the surrounding infrastructure. V2V connections allow for crash prevention, while V2I connections enable safety, mobility, and environmental benefits. Moreover, connections among vehicles, infrastructure, and wireless devices to provide continuous real-time connectivity to all system users.

The Connected Vehicle project supports both non-DSRC and DSRC technologies, but for all security-related applications it does strongly rely on DSRC, because of its high availability and very low latency characteristics. For this reason, USDOT has participated in the development of all DSRC-related standards that are critical to the connected transportation environment, including IEEE 802.11p (amendment to IEEE 802.11) [21], the vehicle-centric IEEE 1609 series (known as IEEE 1609.x) [22] and the SAE J2735 DSRC message set standard [23].

Connected vehicle safety applications are designed to increase situational awareness and reduce or eliminate crashes, by means of V2V and V2I data communications. Connected vehicle mobility applications provide a data-rich travel environment. Such communications should support driver advisories, driver warnings, and vehicle and/or infrastructure controls, by capturing real-time data from automobiles, trucks, and buses, and within the transportation infrastructure. Data are transmitted wirelessly and are used by transportation managers in a wide range of dynamic, multi-modal applications, to manage the transportation system for optimum performance. As part of this, connected vehicle environmental applications both generate and capture environmentally relevant real-time transportation data, and use such data to support and facilitate green transportation choices, thus reducing the environmental impact of each trip. In August 2012, the USDOT started the Connected Vehicle Safety Pilot project, a 1-year length trial involving over 2,800 vehicles, in Ann Arbor (Michigan, USA). The goal of the trial is to assess the capacity of vehicular communication technology to improve safety. In detail, the pilot is not only testing the technical reliability of Dedicated Short-Range Communications (DSRC) devices in real-world conditions, but also how drivers adapt to the technology, and how they respond to in-vehicle warnings. The trial was initially supposed to last one year, but it has been extended by another 6 months. At the end, National Highway Traffic

Safety Administration (NTHSA) will use the results from the Safety Pilot to decide whether to advance the technology through regulatory proposals, additional research, or a combination of both. The cost of the trial is 25 million dollars, 80 % funded by the USDOT.

1.3 ITS Architecture

1.3.1 A Global Standardization Effort

As discussed in previous sections, in last decades governments and private companies have been involved in global ITS standardization and harmonization efforts, coordinated by a large number of SDO. Thanks to this long cooperation history, nowadays there is a general consensus about the ITS architecture and related communication protocols, but we are still far from having market-ready implementations.

In order to enable effective collaboration, by establishing a common vocabulary, experts from a number of SDOs developed the concept of *ITS station* (*ITS-S*), which is described in standard ISO 21217 (CALM). At the highest level of abstraction, an ITS-S is a set of functionalities in a bounded, secured, managed domain, which provides communication services to resident applications (ITS-S applications). From an architectural perspective, an ITS-S is a set of functionalities in an Open Systems Interconnection (OSI)-like layered model (from ISO/IEC 7498-1). Example functionalities are those to securely manage applications and communication resources. The ITS-S concept and its architecture have been adopted by CEN TC278, by ETSI TC ITS and by ISO TC204, and is discussed in next section. A different approach is the WAVE one, which is discussed in a dedicated section.

1.3.2 ISO/ETSI ITS Station Architecture

Starting points for the definition of a common ITS Communication (ITSC) architecture are the ETSI EN 302 665 standard [24], and the Communications Access for Land Mobiles (CALM) [25] family of standards—in particular, ISO 21217:2013 and its predecessor ISO 21217:2010 [14].

The ITSC architecture is designed around the concept of ITS station (ITS-S), a modular computing unit provided by communication capabilities, which can be installed virtually anywhere. As shown in Fig. 1.5, the ETSI EN 302 665 standard defines four main ITS-S types:

- Vehicle ITS Stations: embedded or after-market devices in road-enabled vehicles (cars, trucks, bus, motorcycles), both in motion or parked;
- Roadside ITS Stations: installed at the roadside, at road gateways, on traffic lights;

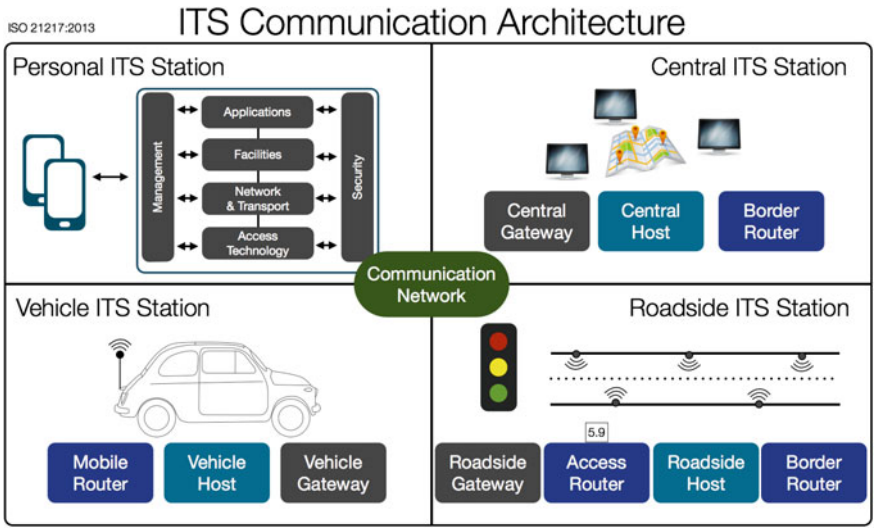


Fig. 1.5 Main ITS components

- Central ITS Remote Stations: installed in back offices, it is a key component of the centralized ITS management/service systems, needed to coordinate the whole system and to collect, store, and process information;
- Personal ITS Stations: handheld or nomadic devices such as smartphones and tablets.

With the exception of the Personal ITS-S, which it is composed by a single ITS-S host entity, all stations are based on a number of independent entities, which can be classified as hosts, gateways, routers and border routers. Hosts have both applicative and communication functionalities, while other entities have only specific communication functionalities, namely, protocol translation for the gateways, and routing for the routers. For example, in Fig. 1.6 it is shown the architecture of a vehicle ITS-S composed by a gateway, a host and a router. The gateway translates messages interchanged with the proprietary internal network of the vehicle, while the router is charged of routing packets through a series of heterogeneous networks.

Figure 1.7 shows the layered architecture of the ITS-S Host, which is the most significant entity. The ITS-S Host is constituted by four horizontal logical layers and two vertical layers. Starting from the bottom, one first encounters the access technologies layer, which groups together the corresponding physical and link layers of the ISO/OSI stack. Networking and Transport can be straightforwardly mapped with the homonym layers of the ISO/OSI stack. The transport layer includes TCP, UDP and dedicated ITS transport protocols, such as the ETSI Basic Transport Protocol (BTP) [26]. The networking layer includes a large variety of protocols, such as GeoNetworking [27], IPv6 networking with mobility support, developed at IETF and ISO specified in [28], IPv6 over GeoNetworking as specified in [29], CALM

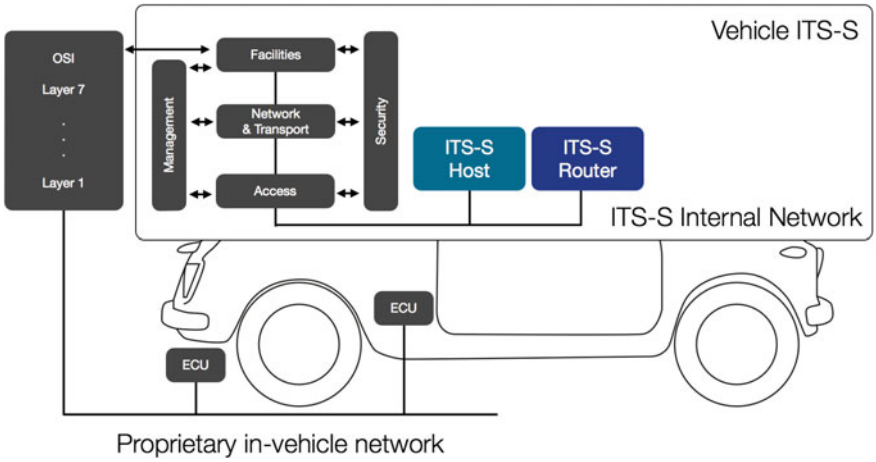


Fig. 1.6 Architecture of a vehicle ITS station

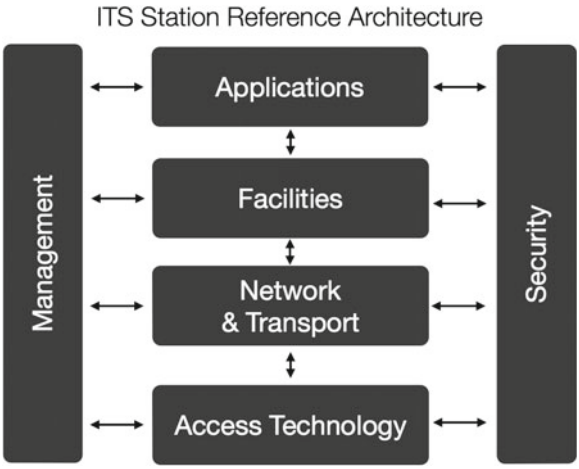


Fig. 1.7 Architecture of an ITS host

FAST protocol, as specified in [30], also known as Fast Networking and Transport Layer protocol (FNTP),[5] Fast Service Advertisement Protocol (FSAP), which follows closely the functionality of IEEE for WAVE Service Advertisement (WSA).

The ITSC Facilities layer contains functionalities from the OSI application layer, the OSI presentation layer (e.g., ASN.1 encoding, decoding and encryption) and the OSI session layer (e.g., inter-host communication), with amendments dedicated to ITSC. Within ITSC Facilities also lie some functionalities not directly related to

[5] The IEEE WSMP protocol is closely related, and there are serious attempts to harmonize FNTP and WSMP.

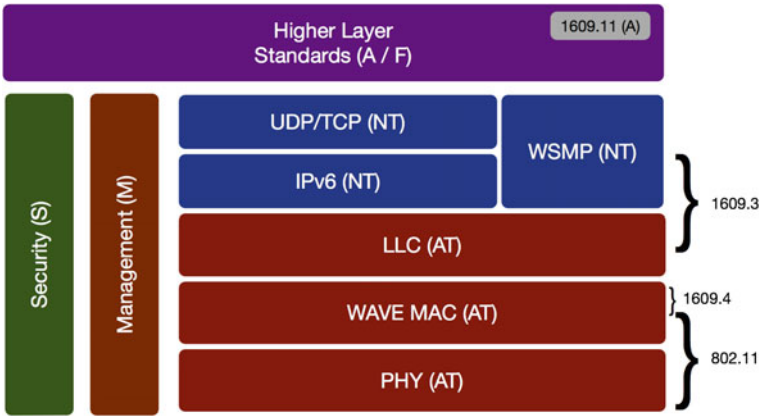


Fig. 1.8 Architecture of a WAVE ITS-S

communications, such as the Human Machine Interface functionality. The Applications layer is composed by all ITS-S applications, built on top of the previous three layers. Finally, the architecture explicitly includes management and security stacks, which are interrelated with all the previous.

1.3.3 WAVE Station Architecture

While the IEEE WAVE reference architecture standard P1609.0 has yet to be published, Fig. 1.8 has been adopted by the IEEE 1609 WG and contains a subset of the functionalities shown in the CEN/ETSI/ISO ITS station architecture presented in Sect. 1.3.2. Note that the current IEEE WAVE reference architecture does not explicitly include Facilities or Applications layer functions. The colors indicate the correspondences with the ITS station architecture from CEN/ETSI/ISO in a relaxed way. The CEN/ETSI/ISO approach is intended to support (but does not require) multiple network stacks from the outset, while IEEE work is focused on a 5.9 GHz radio interface.

IEEE 1609 is defined by four sub-protocols, with different functions and a different grade of maturity. More specifically, IEEE 1609.1 is a sufficiently mature standard (it dates 2006) and it basically defines a resource manager [31] acting as an "outsourcing" manager. In other words, it allows to physically separate the applications from the physical radio interfaces, either Road Side Units (RSUs) or On-Board Units (OBUs). For example, an application can run on an external device, such as a smartphone or a Global Positioning System (GPS) [32] navigator, without adding computational load and complexity to the OBU. This should allow to reduce the cost and increase the reliability of the OBU and RSUs.

The IEEE 1609.2 standard [33] defines security services for the WAVE networking stack and for applications that are intended to run over the stack, such as authentication of STAs and encryption of messages. IEEE 1609.2 provides mechanisms to authenticate WAVE management messages, to authenticate messages that do not require anonymity, and to encrypt messages to a known recipient.

The IEEE 1609.3 defines the networking services for IVCs, but its specifications are still in draft form [34]. The WAVE networking services can be divided into two sets: (i) data-plane services, with data-bearing functions; (ii) management-plane services, charged of the system configuration and maintenance. WAVE supports both two network-layer protocols: (i) the traditional IPv6 routing protocol [35], together with the transport protocols associated with it; (ii) the new WAVE Short Message Protocol (WSMP), expressly designated for accommodate high-priority, time-sensitive communications [36].

The IEEE 1609.4 specification, that is still a draft, defines the organization of multiple channels operations [37], and therefore it has a strong relation to the EDCA mechanism, better described in Sect. 2.5.3. IEEE 1609.4 envisions the presence of a single Control CHannel (CCH), reserved for system control and safety messages, and up to six Service CHannels (SCHs) used to exchange non-safety data packets (e.g., IP traffic) and WAVE-mode Short Messages (WSM). According to the multi-channel operation, all vehicular devices have to monitor the CCH during common time intervals (the CCH intervals), and to (optionally) switch to one SCH during the SCH intervals. The described operation allows the safety warning messages to be transmitted on CCH using the WSM protocol, while non-safety data applications, either running over IP or WSM packets, use the SCHs.

1.4 ITS Applications

As vehicles become integrated in an ITS, their "horizon of awareness" drastically increases and an entirely new ecosystem of applications can be created, and even pre-existent applications can greatly enhance their efficiency [38]. New applications, especially which in the domain of transportation safety and efficiency, are the main drivers for the development of new systems. These applications shall cope with new challenges created by high vehicle speeds and highly dynamic operating environments, and shall guarantee high packet delivery rates and low packet latency.

As represented in Fig. 1.9, ITS applications can be classified in three categories acting in three primary directions [39]: transportation safety, transportation efficiency, and user services delivered to the vehicles, typically in the field of connectivity and convenience. Due to their nature, safety applications require to be executed in dedicated reliable hardware, while the remaining applications can be delivered through consumer electronic devices such as smartphones, or in-vehicle embedded devices. Obviously, a better integration with the vehicle can provide

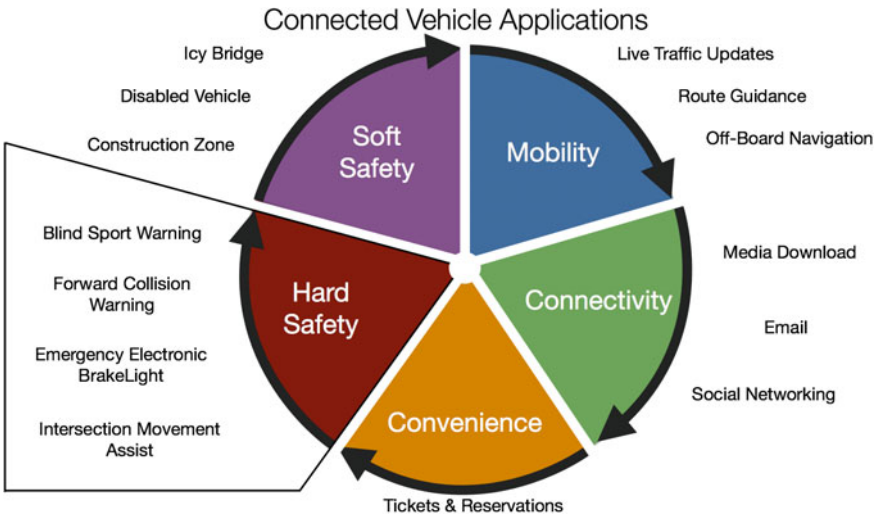


Fig. 1.9 Applications classification

additional advantages. The vehicles display and sound system can offer a user interface designed to minimize driver distraction.

Hard safety applications are targeted to avoiding imminent crashes and minimizing the damage when these crashes become unavoidable. These applications impose the most stringent requirements on the communication system. The communication latency has to be minimized in order to offer the driver sufficient time to take action and the communication system must provide high levels of reliability such as high message reception probabilities. There is also a subset of safety-applications, with less time-critical requirements, that can be denoted as soft-safety applications. These applications increase driver safety but do not require immediate driver reaction, because the hazards are not imminent. Examples include warning the driver of weather, road, traffic, icy roads, construction zones, reduced visibility, pot holes, and traffic jams. Typical actions in response to soft safety application alerts would be to proceed with caution or take alternate routes to avoid the dangerous conditions ahead.

Transportation efficiency applications focus on improving traffic flow. Examples include navigation, road guidance, traffic information services, traffic assistance, and traffic coordination. The last family of general purposes applications focus on making driving more enjoyable and providing greater convenience. Examples include point-of-interest notification, email, social networking, media download, and applications update. All these applications can tolerate long delays but may occasionally demand high data throughput.

1.4.1 Traffic Information Services

Traffic information Services combine historical, real-time, and predictive information, to enable optimized (re-)routing and reliable Estimated Time of Arrival computing. Major navigation brands, such as Nokia (with HERE Drive) and Google (with Google Maps), leverage probe data from their customer base, to offer free traffic updates. Smaller vendors, like TomTom, shifted to lifetime traffic offers, bundled with the navigation device.

TIS rely on Embedded On-board Infotainment Systems (OISs), which combine entertainment, multi-media and driver information functions in one module; on unintegrated information systems, such as smartphones and tablets, which may extend the functionalities of OISs, or operate independently of them; on general-purpose communication infrastructures, such as the cellular network; on dedicated communication infrastructures, such as Road Side Units (RSUs) based on IEEE 802.11p; and on remote services (e.g., cloud-based) which provide/collect and process information collected by RSUs and/or vehicles.

OISs, also known as In-Vehicle Communications and Entertainment System (IVCES), combine entertainment, multi-media and driver information functions in one module. They offer AM/FM or satellite radio, DC/DVD player for music and video, navigation system, data and multi media ports (USB, Bluetooth, line in, line out, video in) as well as general and vehicle status information. Recent OISs are also networked, e.g., by means of 802.11n, an amendment which improves upon the previous 802.11 standards by adding multiple-input multiple-output antennas (MIMO), operating at a maximum net data rate from 54 to 600 Mbit/s [40].

An OIS is an embedded hardware module, powered by dedicated embedded operating systems and middleware, able to provide passengers with several services, including audio/video entertainment, navigation assistance, telephony, car setup and diagnostic, driver information, Internet connectivity, and smartphone integration. To achieve these results, an OIS must interact with the diagnostic and multimedia buses of the vehicle (CAN BUS, FlexRay, MOST), to offer a multimodal friendly HMI (including touch-screens, steering wheel buttons, vocal controls). On the basis of the desired level of functionality, an OIS could also be equipped with a certain number of network interfaces (Bluetooth, WiFi, 3G/4G, USB), auxiliary inputs and positioning systems (i.e., GPS).

From a historical perspective, the need for entertaining car passengers was born with cars themselves. The first car radio, developed by Motorola, appeared on the market during the 1930s [41].[6] However, during the whole 20th century, OISs have been devices able to offer a limited set of functionalities—mainly audio entertainment, diagnostics and navigation services (the latter, only in last decades)—without interoperability and connectivity capabilities. Since the appearance of the Bluetooth technology,[7] in 2000, OISs became more and more influenced by mobile phone

[6] http://www.motorola.com/us/consumers/about-motorola-us/About_Motorola-History-Timeline/About_Motorola-History-Timeline.html.

[7] http://www.bluetooth.com/Pages/History-of-Bluetooth.aspx.

technologies. Later, the Bluetooth technology triggered the development of a new generation of OISs able to offer integrated phone services, interoperating with mobile phones. The smartphone revolution started with the launch of the iPhone, in 2007, and forced a further change of paradigm for OISs manufacturers. Smart devices, such as smartphones and tablets, have quickly achieved a pivotal role in vehicle infotainment, thanks to their flat Internet connectivity, their application stores with thousands of apps, and their vertically integrated cloud services.

According to many external observers, the producers of OISs cannot compete with mobile phone companies from a technologically perspective, and soon or later smart devices will become the core of car infotainment, leaving a mere auxiliary role to OISs. However, car manufacturers have not yet accepted this idea, and are figuring out a business model able to guarantee all the advantages of the smart devices ecosystem, without loosing the control of the chain value. For this reason, it is possible to find in the market many different examples of integration between OISs and smart devices. Typically, the integration goal is to leverage on OISs' HMI capabilities to exploit smart devices' resources, including basic phone functionalities (phone call, contact list, SMS), navigation assistance (i.e., Google send-to-car), multimedia resources, Internet connectivity offered by smart devices or by OISs (typically through WiFi access points), total integration—a smart device and an OIS device operate as an unique platform. The latter approach is followed, for example, by MirrorLink [42], which offers seamless connectivity between a smartphone and the OIS itself, allowing to gain access to phone applications through car controls.

The contamination between mobile phones and vehicles clearly emerges by observing the software conception of modern OISs, which can be classified according to the following categories:

- monolithic software that can be expanded only by replacing the whole firmware (Fiat Blue&ME [43]);
- software expandable through apps realized by the car manufacturer itself (i.e., Mercedes-Benz Apps Store [44]);
- software expandable through apps realized by independent developers, by using the official SDK (Ford AppLink [45], BMW ConnectedDrive [46], Renault R-Link [47]);
- full mirroring with the smart device: in this scenario, the applications of the OIS are the applications (at least a subset) of the smart device itself (i.e., MirrorLink approach).

1.5 Chapter Outlines

The remainder of this book has the following organization. Chapter 2 presents the state of the art in ITS-enabling communication technologies, network topologies, as well as centralized and decentralized approaches. Chapter 3 illustrates novel wireless communication strategies for VANETs. Chapter 4 describes a hierarchical

architecture for cross layer ITS communications. Chapter 5 focuses on the application layer, describing a structured overlay network called DGT, and a DGT-based architecture enabling ITS services—in particular, TIS services. Appendix A illustrates DEUS, the simulation tool we used to evaluate the algorithms illustrated in Chap. 5. Appendix B provides an overview of the mathematical methods we adopted throughout the book. Appendix C illustrates in detail the group key distribution protocol used in Chap. 4.

References

1. Bishop, R.: A survey of intelligent vehicle applications worldwide. In: Proceedings of of IEEE Symposium on Intelligent Vehicles (IV 2000), pp. 25–30. Dearborn, MI, USA (2000)
2. Figueiredo, L., Jesus, I., Machado, J., Ferreira, J., Martins de Carvalho, J.: Towards the development of intelligent transportation systems. In: Proceedings of IEEE Conference on Intelligent Transportation Systems, pp. 1206–1211. Oakland, CA, USA (2001)
3. Qu, F., Wang, F.Y., Yang, L.: Intelligent transportation spaces: vehicles, traffic, communications, and beyond. IEEE Commun. Mag. **48**(11), 136–142 (2010)
4. Hartenstein, H., Laberteaux, K.: VANET vehicular applications and inter-networking technologies. Wiley Press, New Jersey (2010)
5. Ministry of Land, Infrastructure, Transport and Tourism, Japan, A.: (2010). Website: http://www.mlit.go.jp
6. Administration, F.H.: IntelliDrive: safer, smarter greener. Public Roads **74**(1), 18–22 (2010)
7. Car-to-Car Communication Consortium: Website: http://www.car-2-car.org/
8. Communication from the Commission to the European Parliament the Council, t.E.E., Committee, S., the Committee of the Regions: Towards Europe-wide safer, cleaner and efficient mobility: The first intelligent car report (2007)
9. European Telecommunications Standards Institute: Website: http://www.etsi.org/
10. Willke, T., Tientrakool, P., Maxemchuk, N.: A survey of inter-vehicle communication protocols and their applications. Commun. Surv. Tutorials IEEE **11**(2), 3–20 (2009)
11. Lee, U., Magistretti, E., Gerla, M., Bellavista, P., Corradi, A.: Dissemination and harvesting of urban data using vehicular sensing platforms. IEEE Trans. Veh. Technol. **58**(2), 882–901 (2009)
12. Kato, S., Tsugawa, S., Tokuda, K., Matsui, T., Fujii, H.: Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications. IEEE Trans. Intell. Transp. Syst. **3**(3), 155–161 (2002)
13. Broggi, A.: VisLab's vehicles just reached Siberia: driverless! Now Kazakhstan and then China until Shanghai [ITS events]. Intell. Transp. Syst. Mag. IEEE **2**(3), 43–44 (2010)
14. International Organization for Standardization: ISO 21217:2013 intelligent transport systems—communications access for land mobiles (CALM)—architecture (2013)
15. Europe, E.I.: Ertico vision. http://www.ertico.com/about-ertico-weare/
16. Commission, E.: Action plan and legal framework for the deployment of intelligent transports systems (its) in Europe (2008). http://ec.europa.eu/transport/themes/its/road/action_plan/index_en.htm
17. Parliament, E.: Directive 2010/40/EU of the European parliament and of the council on the framework for the deployment of intelligent transport systems in the field of road transport and for interfaces with other modes of transport (2010). http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32010L0040:EN:NOT
18. Eureka: Programme for a european traffic system with highest efficiency and unprecedented safety (prometheus). http://www.eurekanetwork.org/project/-/id/45
19. Jurgen, R.K.: Smart cars and highways go global. Spectr. IEEE **28**(5), 26–36 (1991)

20. Research, R., Administration, I.T.: ITS strategic research plan 2010–2014 (2010–2014). http://www.its.dot.gov/strategic_plan2010_2014/
21. IEEE: IEEE standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 6: wireless access in vehicular environments. IEEE Std 802.11p-2010 (amendment to IEEE Std 802.11-2007) pp. 1–51 (2010)
22. Insitute of Electrical and Electronics Engineers: IEEE 1609–2006. IEEE trial-use standard for wireless access in vehicular environments (WAVE) (2006)
23. International, S.: SAE J2735—dedicated short range communications (DSRC) message set dictionary (2009). http://www.sae.org/standardsdev/dsrc/
24. Intelligent Transport Systems (ITS): Communications architecture. ETSI ES 202 665 V1.1.1, pp. 1–44 (2010)
25. Communications Access for Land Mobiles (CALM), I.: Website: http://www.calm.hu/
26. Intelligent Transport Systems (ITS): Vehicular communications; geonetworking; part 5: Transport protocols; sub-part 1: Basic transport protocol. ETSI TS 102 636-5-1, pp. 1–45 (2011)
27. Intelligent Transport Systems (ITS): Vehicular communications; geonetworking. ETSI TS 102 636 V1 (all parts), pp. 1–44 (2011)
28. International Organization for Standardization: ISO/IEC 21210:2012 Intelligent transport systems—communications access for land mobiles (CALM)—ipv6 networking (2012)
29. Intelligent Transport Systems (ITS): Vehicular communications; geonetworking; part 6: Internet integration; subpart 1: Transmission of IPv6 packets. ETSI TS 102 636-6-1, pp. 1–45 (2011)
30. International Organization for Standardization: ISO/IEC 29281:2013 intelligent transport systems—communications access for land mobiles (CALM)—non-ip networking (2013)
31. Insitute of Electrical and Electronics Engineers: IEEE 1609.1-2006. IEEE trial-use standard for wireless access in vehicular environments (WAVE)—resource manager (2006)
32. Kaplan, E., Hegarty, C.: Understanding GPS: Principles and applications. Artech House Publishers, Boston (2006)
33. Insitute of Electrical and Electronics Engineers: IEEE 1609.2-2006. IEEE trial-use standard for wireless access in vehicular environments (WAVE)—security services for applications and management messages (2006)
34. Insitute of Electrical and Electronics Engineers: IEEE 1609.3-2010. IEEE draft standard for wireless access in vehicular environments (WAVE)—networking services (2010)
35. RFC 2460 Internet Protocol: Version 6 (IPv6) specification. The internet engineering task force (IETF)
36. Uzcategui, R., Acosta-Marum, G.: WAVE: a tutorial. IEEE Commun. Mag. **47**(5), 126–133 (2009)
37. Insitute of Electrical and Electronics Engineers: IEEE 1609.4-2010. IEEE draft standard for wireless access in vehicular environments (WAVE)—multi-channel operation (2010)
38. Papadimitratos, P., La Fortelle, A., Evenssen, K., Brignolo, R., Cosenza, S.: Vehicular communication systems: enabling technologies, applications, and future outlook on intelligent transportation. Commun. Mag. IEEE **47**(11), 84–95 (2009)
39. Zhang, T., Delgrossi, L.: Vehicle safety communications: Protocols, security, and privacy. Wiley Press, New Jersey (2012)
40. IEEE Standard for Information technology-Part 11: Wireless LAN medium access control (MAC) and physical layer (PHY) specifications amendment 5: Enhancementss for higher throughput. IEEE Std 802.11n-2009, pp. 1–211 (2005)
41. Salcedo, I., Rodriguez, F.J.: Calling all cars: Motorola's early police radios. In: 2010 2nd IEEE Region 8 Conference on the History of Telecommunications Conference (HISTELCON), pp. 1–4. IEEE (2010)
42. Consortium, C.C.: Mirrorlink. http://www.mirrorlink.com/
43. Fiat: Fiat blue&me. http://www.blueandme.net
44. Mercedes-Benz: Mercedes-benz apps. http://apps.mercedes-benz.com/

45. Ford: Ford applink. https://developer.ford.com/
46. BMW: Bmw connecteddrive. http://www.bmw.com/com/en/insights/technology/connected-drive/2013/index.html
47. Renault: Renault r-link. https://ie.rlinkstore.com

Chapter 2
Communication Paradigms and Literature Analysis

2.1 Vehicular Networks

2.1.1 Terminology and Definition

According to the ISO architecture presented in Sect. 1.3, within a ITSC system there are 4 type of entities: Onboard ITS subsystem of a vehicle, which is usually denoted as On Board Unit (OBU), Roadside ITS subsystem, typically denoted as RoadSide Unit (RSU), central ITS subsystem, and Personal ITS subsystems (e.g., smart devices). In this book, a network that includes all these entities is referred as a vehicular network. In this type of network, several communication technologies and communication protocols can be used at the same time, thus realizing a truly heterogeneous network. Since an endpoint of the communication is always given by a vehicle (OBU), while the other endpoint varies depending on the application, communications in a vehicular network are typically denoted with the acronym V2X (Vehicle-to-Device), where the "X" denotes any kind of entity discussed above.

When a vehicular network is exclusively formed by OBUs and RSUs, it is customary referred as vehicular ad-hoc network (VANET). A VANET is less heterogeneous than a generic vehicular network, and typically based on a single dedicated communication technology (DSRC-like) and a limited set of communication protocols. When both endpoints of a communication are OBUs, the communication is referred as Vehicle-to-Vehicle (V2V), while when the first endpoint is an OBU and the second endpoint is an RSU or a centralized server, the communication is denoted as Vehicle-to-Infrastructure (V2I) or Infrastructure-to-Vehicle (I2V), depending on the direction of the information flow.

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7_2

2.1.2 Key Challenges in Vehicular Networks

A node member of a vehicular network cannot be considered as a mere communication device (OBU), but as a complex entity jointly constituted by the OBU itself and the vehicle where the OBU is installed on. This entity is characterized by a series of unique features and compelling challenges that should be accounted by any communication protocol designed for inter-vehicular communications.

2.1.2.1 Advantages

A vehicle is equipped with computing and sensing capabilities (e.g., GPS transceivers, accelerometers, cameras) able to provide information about the dynamic state of the vehicle and the surrounding environment [1], [2], [3], [4]. This actively collected information can be integrated with a considerable amount of side-information derived by the constrained mobility of the vehicles, that are forced to move along defined roadways, or by the information pro-actively provided by the driver itself (i.e., path-planning). A vehicle also provides a virtually unlimited amount of stored energy, which is available for sensing, communication and computation tasks, while other types of mobile wireless networks (i.e., wireless sensor networks, cellular networks) are usually energy-constrained. Another peculiarity of vehicular networks is that information is typically exchanged between entities without any user interaction, like in Machine-to-Machine (M2M) networks, thus allowing to predict communication patterns [5]. These combined factors imply that a node of a vehicular network can achieve a higher awareness level than in traditional wireless networks, which can be exploited by communication protocols to improve several functionalities, such as topology management, congestion control [6], handover management [7] and performance optimization [8].

2.1.2.2 Limitations

The main drawbacks of vehicular networks are related to the high vehicle mobility, which requires a huge effort for maintain up to date network topology maps. The problem is complicated by the high variability of the environment in which vehicular networks operate. A lot of different configurations are possible, ranging from highways where relative inter-vehicle speeds of up to 300 km/h may occur, and the spatial density is typically low, to city roads where relative inter-vehicle speeds can be in the order of few tens of km/h and spatial density can be high, especially during rush hour. On the one hand, when the vehicular spatial density is low, networks are often partitioned in sets of disconnected clusters; on the other hand, when the density is high, the wireless channel can be congested and affected by interference.

The variability of the environment is not only important from a topological point of view, but also from the perspective of the physical wireless channel. Vehicles

can run in a desert countryside, inside a tunnel, or in a urban canyon studded with skyscrapers. Furthermore, because of their metallic nature, surrounding vehicles have a huge impact on the number of multi-path reflections experienced by the receiver. Also the antenna radiation pattern is highly influenced by its placement with respect to the vehicle (i.e., inside, on the roof). Moreover, high relative speeds produce Doppler shifts, and other vehicles induce shadowing phenomena [9].

Finally, the third category of issues that affect vehicular networks is more abstract and directly influenced by the nature of the vehicular market and applications requirements. The market has a global scale or, at minimum, a country-wide scale, and therefore also a vehicular network shall assume a dimension comparable with that of a country-wide road infrastructure. However, for most applications, interesting information is local, and can be retrieved by interacting with neighbor vehicles, or with active points of interest and service providers, positioned in the nearby. In other words, even if vehicular network can have a huge scale in terms of geographical extension and number of nodes, they provide (mostly) localized services. This bipolar nature is manifested in the network architecture that it is often a combination of centralized and decentralized network topologies.

Common communication CPEs have a limited lifetime, which rarely overcomes the threshold of 10 years, indeed it is typically much shorter. Conversely, the lifetime of a vehicle often overcomes the threshold of 10 years and can also be significantly longer. This lifetime mismatch shall be accounted when designing a vehicular communication technology, and an OBU shall be reliable and easily upgradable in order to not excessively impact on the vehicle maintenance cost. Furthermore, a longer lifetime implies a slow vehicles substitution rate and a slow adoption rate of off-the-rack vehicular communication technology. This is big issue since most ITS applications and services are effective only if the number of equipped vehicles in a given spatial region (i.e, the spatial density) is sufficiently high, or, in other words, greater than a certain threshold. This is a typical example of a *chicken-and-egg* problem, which poses serious issues on the appealing of communication-based ITSs system [10]. To mitigate such a problem, it is necessary to increase the adoption rate of ITS equipment, a result that can be obtained by pursuing three main strategies illustrated below.

- Enforcing proper rules and policies for the adoption of ITS equipment. For example, the European Commission has mandated the adoption of the e-Call platform in all new vehicles, starting from 2015.
- The development of suitable after-market devices, installable on old vehicles, to speed up the transition.
- Reusing as much as possible some widely available technologies, such as cellular networks or standard WiFi, which can be used to provide services to vehicle without dedicated equipment.

2.1.3 Network Topology

With the expression *network topology* it is possible to refer both to the **physical** network topology, given by the nodes and by the physical wired or wireless links between them, or to the **logical** network topology constituted by a network of virtual nodes and virtual links set over the real links.

2.1.3.1 Physical Network Topology

The physical topology of a mobile wireless network is determined by the nodes' positions and by the directionality and transmission range of the adopted communication technologies. With the exception of a partial control that can be achieved by employing power control and/or smart antenna techniques, once chosen the communication technology the designer cannot modify the network topology, as it cannot control or predict the nodes' movements.

Vehicular networks are mobile wireless networks where it possible to have a better topology control, for two reasons: as seen in Sect. 2.1.2, the vehicles' movements are constrained by the road infrastructure and at some extent predictable; moreover, vehicles are often equipped with heterogeneous communication technologies, thus leading to an additional degree of topology freedom. On the basis on their impact on network topologies, today's vehicular communication technology can be classified in three main groups, illustrated in the following.

- Unidirectional broadcast communication technologies, such as FM-based digital radio broadcast, where communications can only happens between a centralized service center through a network of radio station. In this case the network has always a star topology.
- Technologies with a limited transmission range (less than 1km) supporting direct communications between vehicles and broadcast transmissions. Such a category includes DSRC-like technologies like IEEE 802.11p, CALM M5, ETSI G5 and traditional WiFi standards (IEEE 802.11a/b/g/n), in either ad-hoc or direct mode. In this case, the physical topology is strictly correlated to the vehicles mobility and to the communication range. According to many studies, cars naturally tend to form isolated clusters of vehicles [11]. The cluster size, the cluster lifetime and the number of clusters depends on the environment. Within a cluster, vehicles assume a highly-dynamic mesh topology, with a highly-variable degree of connectivity, ranging from line topology to fully-connected topology. The presence of RSUs do not alter significantly the network topology, however they can help in connecting isolated vehicle clusters.
- Technologies with a wide area coverage, not supporting direct communications and broadcast transmissions. WiMAX [12], [13] and all current cellular technologies, starting from GSM to LTE [14], [15], fall in this category. In this case, the physical topology is quite simpler since from a link-layer perspective all communications are directed to the base station, and therefore the network topology is constituted by

a series of star-topology networks. Obviously, a node out of coverage is intrinsically isolated.

Clearly, when vehicles are equipped with both technology types, the two categories of topology merge together, leading to hybrid topologies.

We observe that the introduction of the upcoming LTE Advanced (LTE-A) [16], [17] standard will drastically change the panorama and will form a category alone. In fact, LTE-A promises to bring some direct communication and broadcast transmission capacities, thus achieving the properties of both second and third categories.

2.1.3.2 Logical Network Topology

The concept of logical network topology can be ambiguous, as—depending on the context—in some cases it consists of overlays created at low layers of the ISO/OSI stack (MAC and routing), while in other cases it is more convenient to define logical topologies at higher layers, such as the application layer. When a vehicular networks collapses in a pure VANET constituted only by OBUs and RSUs, application and routing levels often coincide. In these cases, it is possible to analyze them together. Conversely, in more complex vehicular networks that encompass heterogeneous technologies and include remote nodes or mobile terminals, it is difficult to keep a coherent topological vision at low layers, and it is more convenient to focus on the application layer. Greater details on topological overlay at application layer will be provide in Sects. 2.3 and 2.4, which are devoted to VANETs.

It is a costly operations maintain a logical topology view on highly mobile networks such as as vehicular networks. On the one hand, protocols that proactively build a topology map require a constant heartbeat traffic, which is expensive in terms of resource usage and can lead to congestion in highly dense networks. On the other hand, protocols that actively (on demand) construct a topology map have to face many obstacles. A non exhaustive list of logical topology that characterize VANETs is reported below.

- No topology knowledge—In most dissemination applications based on local broadcast, there are no significant advantages on actively build up the network topology. Therefore in this case the sender simply has no knowledge of the network topology.
- Source-originated tree-based topology—In this case each node of the network build its own topology tree starting from itself (the root). Such a tree can be constructed only when necessary, in order to reduce the overhead costs due to maintenance. In most cases the tree-based topology collapse in a star topology where the nodes only know 1-hop distant nodes. This partial knowledge of the network can be exploited in unicast or broadcast multi-hop dissemination protocols.
- Cluster-based topology—In this case, the logical topology keeps the same structure of the underlining physical topology. Most cluster protocols requires to individuate some supernodes, such as the clusterhead and some kind of gateways with special function, as which of routing the packets towards other clusters in the nearby.

In this family of protocols, within the cluster there is a shared knowledge of the network,
- Mesh topology. In this case, all network nodes share a common vision of the network. Also in this case, the topology is temporary and can be proactively constructed by means of a multi-hop routing protocol.

2.2 Vehicle-to-X Communications

2.2.1 Key Features of a V2X Communication Protocol

As a result of the numerous issues described in Sect. 2.1, it is difficult to design an unique V2X communication protocol able to cope with the extreme complexity of a vehicular network, in terms of mobility, environment dynamism, technology heterogeneity, and to satisfy the often contradictory requirements of vehicular applications. For this reason, V2X communications are based on a set of V2X communication protocols, each of them able to deal with specific types of scenarios and applications.

A communication protocol could be analyzed in two different manners, by observing the service it provides or by considering its inner characteristics (e.g., its communication mechanisms). The service type provided by a V2X protocol is fully specified by the following parameters [18].

- Target applications.
- The nature of communicated data (size, real-time requirements, bulk data).
- The direction of the data flow (unidirectional or bidirectional).
- The quality of service that can be fully specified by three aspects: latency, end-to-end throughput and packet success ratio.
- The circumstances under which the communication is initiated (the trigger event).
- The type and number of involved endpoints (OBU, RSU, centralized server of mobile terminals).

The communication mechanism is characterized by the following factors.

- The traffic pattern—it is considered as unicast when the communication involves only two endpoints, while on the contrary a broadcast traffic pattern involves a single traffic source, and the destination nodes are all the network nodes. A geocast traffic pattern can be considered as a localized broadcast protocol, where the destination nodes are not all the network nodes, but only the nodes positioned in a precise and constrained geographical area.
- The network model, determined by the number of communication hops (measured at the link layer). In a single-hop protocol, the communication involves only two nodes. In a multi-hop protocol the communication involves a larger number of nodes, that can act either as source, destination or relay. A multi-hop protocol allows to extend the dimension of the network and to solve coverage problems, but

it complicates network operation and reduce the performance in terms of latency and throughput in comparison to a single-hop solution.

- Transaction type and transaction frequency.
  - Protocols based on frequent periodic messages sent multiple times per second, such as the Cooperative Awareness Messages (CAM) used by the CAM Basic Service [19]. CAMs are periodically broadcasted by the facility layer at a given frequency satisfying both road safety application requirements and transport and network layer requirements (*network heartbeat*). The CAM frequency may be determined by the communication management entity, taking into account the supported road safety application's operating requirements, transport layer requirements and the current channel load.
  - Event-triggered messages suddenly sent upon a certain event happens, with an unpredictable frequency, but usually in the order of once a second or less. A significant example is represented by the Decentralized Environmental Notification Messages (DENMs) used by the DENM Basic Service [20].
- Transaction size.
  - Small: constituted by single message.
  - Medium: constituted by multiple messages—but the transaction can be still completed in a time smaller than the links lifetime.
  - Large: the transaction cannot be completed during a link lifetime, but it shall involve several links or technologies.
- Session type.
  - Individual messages with a loose session concept (as in broadcast protocols).
  - Unicast local session, generally with OBUs or RSUs.
  - Unicast session with a remote endpoint, which can be a server or a mobile terminal. The remote session can be maintained across several V2I communication sessions, or relying on wide area networks technologies.
- Protocol type.
  - IP based protocol. Often V2X communication protocols natively rely on IPv6 routing protocol to have greater efficiency, larger addressing space and better mobility support.
  - Protocol based on dedicated custom messages, such as the WAVE Short Message Protocol (WSMP).

2.2.2 Vehicle-to-X Communication Paradigms

By combining the inner properties of a V2X communication protocol and the characteristics of the provided services, it is possible to define four main communication paradigms: V2V Local Broadcast, V2V Multi-Hop Dissemination, I2V

Local Broadcast, V2I Bidirectional Communications [21]. Conversely, Beaconing, Geobroadcast, Unicast routing, Advanced information dissemination and Information aggregation, are media independent communication paradigms, in theory, but in practice most of them are effective only with dedicated communication technologies (i.e., IEEE 802.11p or similar) [18].

Beaconing is a special case of vehicle-originated broadcast, used to continuously informing neighboring cars of each other's current position, heading, and speed. Communication is unidirectional and single-hop, hence only neighbors in the communication range of the sender are able to receive its messages. Information exchange through beaconing has a local value and serves as the foundation for cooperative applications aimed at collision avoidance. Beaconing is effective only if performed with dedicated DSRC technologies, with native broadcast support. The aforementioned CAM Basic Service is a potential consumer of a beaconing communication service.

Geobroadcast is another case of unidirectional vehicle-originated broadcast used to inform cars positioned in a certain spatial region in the same area of the sender. Geobroadcast is typically based on V2V multi-hop dissemination mechanisms, and messages from one vehicle are relayed by other vehicles to reach destinations that are outside the source's communication range. When the number of hops is low, this paradigm can be used to support hard safety applications such as emergency vehicle approach, slow vehicle, emergency electronic brake lights and forward collision warnings. These represent the most time-critical and safety-critical category of connected vehicle applications, since they are used to warn drivers of imminent-crash hazards. Geobroadcast dissemination could also be used for soft safety purposes such as the distribution of hazardous road and traffic information. Geobroadcast messages are typically sent upon a certain external event, and they often requires very low latency, especially when used for hard-safety purposes. The aforementioned DENM Basic Service is a potential consumer of a Geobroadcast communication service.

With *I2V Local Broadcast*, vehicles receive local broadcasts from the roadside infrastructure, in support of safety, mobility or sustainability applications. Infrastructure-originated broadcasts are used to disseminate data that are relevant to all vehicles in the vicinity of a specific road infrastructure location, where an RSU is installed. Therefore, it can be implemented through DSRC transceivers deployed along the roadside. However, it can also be implemented using cellular, satellite, or digital radio broadcast services, to reach all the vehicles in a large geographical region. These broadcasts use individual, single-hop I2V messages, involving small transactions, with frequent transmission. The quality of service requirements depend on the target application (traffic controller signal phase and timing information, dangerous road condition information).

Unicast routing has the purpose to transmit data through the network to a specific destination, which can be positioned in the nearby of the sender or remotely. Every endpoint type (OBU, RSU, central server, mobile terminal) can assume either the role of sender or destination. The communication may consist of only a single hop, or route messages over multiple hops toward the destination. These messages are

generally unicast local sessions with low time criticality, low transaction frequency and small transactions, without any refined control or management mechanism.

Advanced information dissemination or multi-RSU session is required to transfer large quantities of data or to execute transactions that take considerable time, which cannot be accommodated within a single encounter between a moving vehicle and one roadside ITS station, but must be maintained across several V2I/I2V unicast communication sessions with a remote server. These transactions are single-hop, with low time-criticality and low frequency. It is necessary to connect with a service provider across a network, but the logical communication session needs to persist across multiple paths, signal between the OBU and a series of RSUs offering access to the backhaul. The persistence may be provided at the application level, the transport layer or the network layer. Example applications would be downloads of large infotainment/media or map update files, some concierge services or continuous web surfing.

Finally, *V2I Bidirectional Communications* are required by many mobility applications, such as navigation, Internet access for browsing or email, electronic transactions for purchasing goods or services, and media download, but also to exchange messages between vehicles, through infrastructure applications servers. Bidirectional communications mode can be supported by dedicated DSRC technologies, but more often with wide area networks, such as cellular networks or WiMAX.

2.3 Centralized Client/Server Technologies

Traditionally proposed architectures are based on a centralized approach, where one or more central servers have the responsibility to manage all position updates and queries from involved users—related, for example, to a specific point of interest, to neighborhood discovery or to path planning. In order to manage a huge number of active users at the same time, with a high quality of service (QoS), usually those solutions require relevant computational power on the server side, and are provided by big companies such as Google and TomTom.

Google Latitude has been introduced by Google in 2009 [22]. It is a location-aware Web/mobile application which allows a mobile phone user to share his/her current location with a group of people, i.e., both real and social friends (Fig. 2.1). By means of his/her Google Account, the location corresponding to the user's cell phone is mapped on Google Maps. The user can control the accuracy and details of what other users can see. An exact localization can be allowed, or it can be limited to identify only the city. Also, a location can be manually entered. For privacy reasons, the localization feature can be turned off. Recently, Google added the possibility to share the user's daily check-ins with Latitude friends, thus merging the functionalities of another application, namely Google Places.

Another example of centralized system is TomTom's HD Traffic, a real-time traffic service which tries to give accurate and up-to-date traffic information. HD Traffic is part of TomTom's LIVE Services, which deliver information to drivers, helping them to save time, money and fuel. In order to be able to provide such an accurate

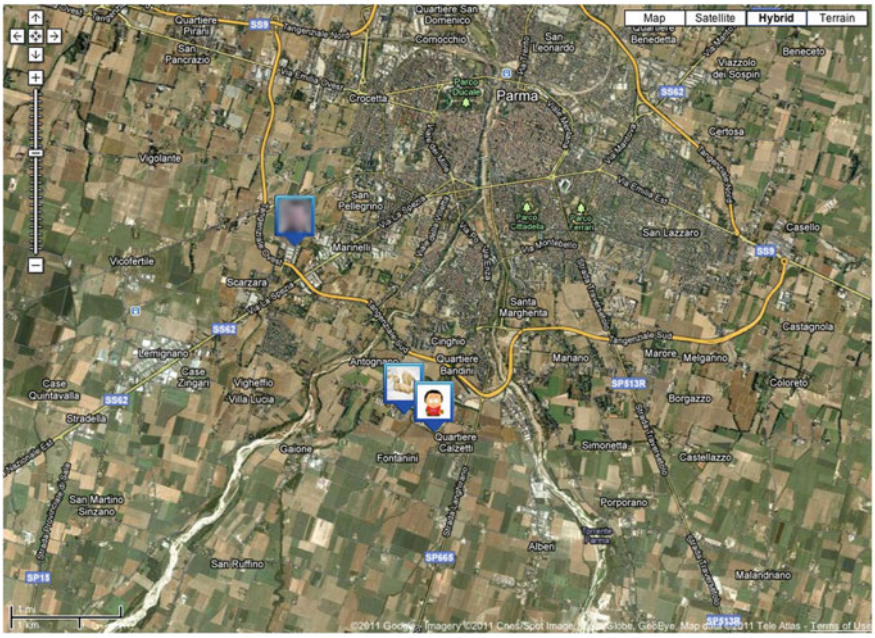


Fig. 2.1 Google Latitude application (Web version) showing friend locations on a map view

real-time information on all major and secondary roads, TomTom's patented HD Traffic technology uses, above all, traffic data generated by the movement patterns of mobile phones inside vehicles, which are collected anonymously from mobile carrier networks. These patterns are then combined with anonymous data from TomTom devices, as well as other traditional sources of traffic information, to provide one of the most advanced traffic information services. Traffic information is relayed in real-time and securely to TomTom devices, thanks to Vodafone Italy's patented Machine to Machine (M2M) solutions [23], and includes a SIM card with a GPRS connection, which is embedded into the navigation device. Processed information and evaluation results, such as traffic status, accident and road monitoring, are then available on TomTom devices or through a Web interface (Fig.2.2).

Among academic research projects, MIT's CarTel [24] combines mobile computing and sensing, wireless networking, and data-intensive algorithms running on servers in the cloud to address these challenges. CarTel is a distributed, mobile sensing and computing system using phones and custom-built on-board telematics devices—one may think of it as a "vehicular cyber-physical system". A CarTel node is a mobile embedded computer coupled to a set of sensors. Each node in the system gathers and locally processes sensor readings, before delivering them to a central portal, where data are stored in a database for further analysis and visualization. CarTel provides a simple query-oriented programming interface, handles large amounts of heterogeneous data from sensors, and copes with intermittent and

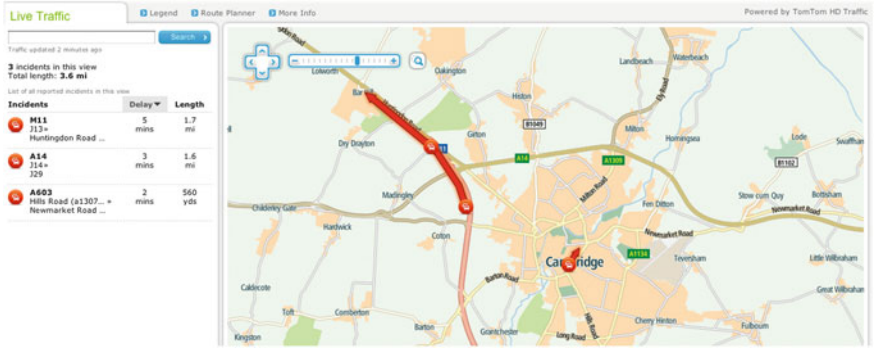


Fig. 2.2 TomTom's HD Traffic Web Interface

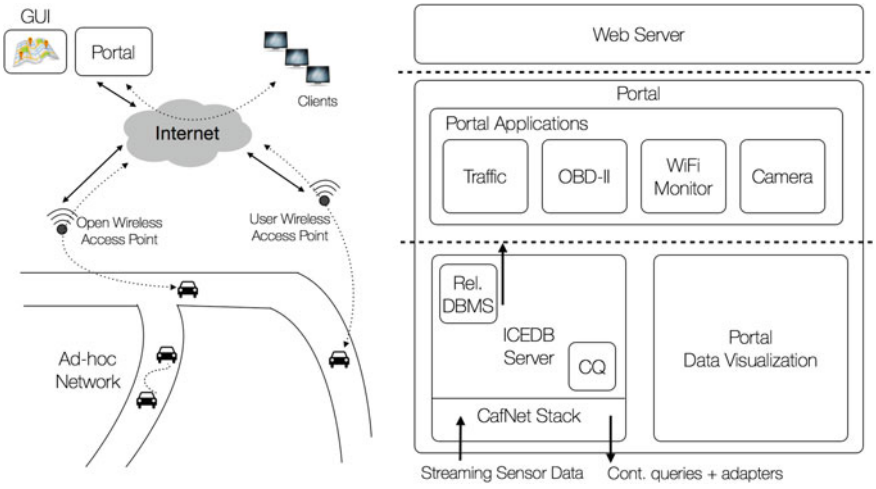


Fig. 2.3 MIT's CarTel architecture

variable network connectivity. CarTel nodes primarily rely on opportunistic wireless (e.g., Wi-Fi, Bluetooth) connectivity to the Internet, or to "data mules", such as other CarTel nodes, mobile phone flash memories, or USB keys, to communicate with CarTel applications running on a Web portal. A delay-tolerant continuous query processor, called ICEDB, allows to specify how mobile nodes should summarize, filter, and dynamically prioritize data. Figure 2.3 illustrates the system architecture, with the different components of the platform. Cars collect data as they move, and log them to their local ICEDB databases. As connectivity becomes available, data on cars is delivered to the Web portal, where users can browse and query it by means of the visual interface.

These examples of centralized solutions are only a small subset of the huge amount of existing solutions. They give an idea about problems associated to location based

Fig. 2.4 Screenshot of Waze's Live map

solutions, the amount and heterogeneity of information that could be potentially involved, and the number of simultaneously active users that participate in such a system.

Particularly innovative is Waze [25], a *social driving* mobile application. By connecting drivers to one another, Waze helps people create local driving communities that work together to improve the quality of everyone's daily driving. That might mean helping them avoid the frustration of sitting in traffic, cluing them into a police trap, or shaving five minutes off of their daily travel by showing them new routes they never even knew about (Fig. 2.4). In October 2012, the Waze community consisted of 28 million drivers. New features are periodically added, according to user preferences. For example, the features released in Waze 3.5 (Driving to a restaurant with friends, Picking up your spouse, Let the kids know you'll be home for dinner) have been introduced taking into account an Omnibus survey of 1,000 American drivers, commissioned by Waze, which found that over 50 % of drivers regularly pick up friends and family.

2.4 Decentralized and Peer-to-Peer Systems

Due to the progressive improvement of Internet connections, and in particular of mobile devices capabilities, during the last decades the research community focused on decentralized architectures, and in particular on peer-to-peer (P2P) overlay networks, which are characterized by highly distributed algorithms for sharing data and resources like computing cycles, storage, and bandwidth. Target applications were (and still are) file sharing, social networking, live and on demand streaming, as well as decentralized geolocation services. Every time a peer wants to know a specific information, e.g., which peers are located in a certain area, it does send a number of lookup queries to a subset of the known peers. Such queries are routed within the overlay network, towards those nodes which may store the desired information.

Distributed localization is a clear example of *geocollaboration service*, usually implemented by recursively dividing the 2D space into smaller areas, in order to assign each peer the responsibility of a space region. Instead of employing a number of centralized servers (either dedicated or selected among participating nodes) to carry the load for the entire network, the peers share the load of indexing and searching data that refers to its area. The idea of hierarchical partitioning comes from the indexing of data structures for multidimensional data sets, such as the R-tree [26], which is widely used in centralized databases. R-trees are tree data structures used for spatial access methods, i.e., for indexing multi-dimensional information such as geographical coordinates, rectangles or polygons. Such an approach has been used in both research and real-world applications, for example to store spatial objects such as restaurant locations, or the polygons which typical maps are made of—streets, buildings, outlines of lakes, coastlines, etc. Moreover, it allows to quickly answer queries such as "*find all museums within 2 km of my current location*", "*retrieve all road segments within 2 km of my location*", or "*find the nearest gas station*". The key idea of the data structure is to group nearby objects and represent them with their minimum bounding rectangle in the next higher level of the tree. Since all objects lie within this bounding rectangle, a query that does not intersect the bounding rectangle cannot intersect any of the contained objects. At the leaf level, each rectangle describes a single object, while at higher levels it describes the aggregation of an increasing number of objects. This can also be seen

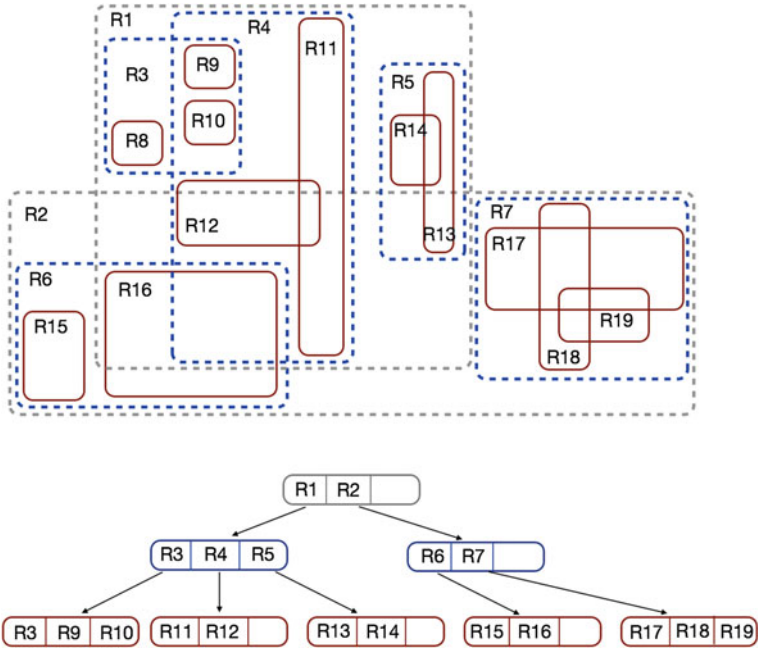


Fig. 2.5 Simple example of an R-Tree with 2D rectangles

as an increasingly coarse approximation of the data set. Scheme in Fig. 2.5 shows a R-Tree examples in the 2D.

Examples of general-purpose hierarchical peer-to-peer schemes supporting geo-collaboration services are HZSearch [27], DPTree [28], DiST [29]. These and other research works propose strategies for supporting complex queries over multi-dimensional data, such as "select five available buildings closest to the airport" [30], [31], [32].

According to the use of wireless technologies and system designs, Tsao et al. categorized decentralized traffic information systems into four different architectures: single-tier VANET, single-tier P2P over VANET, single-tier infrastructure-based P2P, and two-tier VANET/P2P [33]. Such categories are illustrated in Fig. 2.6.

In single-tier VANETs, vehicles communicate with each other through Inter-Vehicular Communication (IVC), and periodically broadcast their current speeds and positions to neighboring vehicles. A part of the traffic information a vehicle receives may also be propagated to its neighbors through broadcast messages. Based on the received messages, a vehicle can generate traffic reports.

In a single-tier P2P over VANET, vehicles form an application-layer P2P overlay network on top of the VANET. The P2P overlay can be either unstructured or structured. Vehicles share their resources (i.e., traffic information) and retrieve resources from others through the P2P overlay. The application-layer P2P overlay communication relies on the routing protocol of the underlying VANET. The key difference between the P2P over VANET architecture and the previous architecture is the traffic information lookup. In the previous approach, a vehicle floods a query message to all neighboring vehicles within the IVC range. In this architecture, a vehicle explicitly

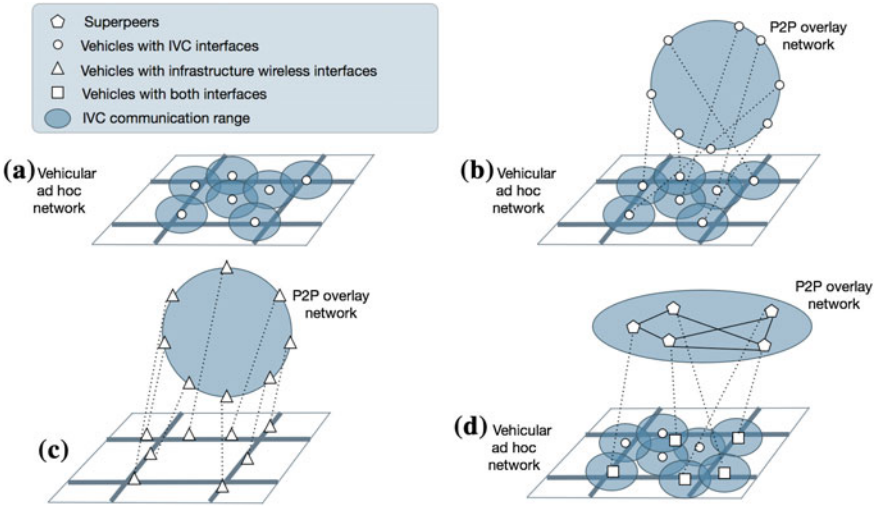


Fig. 2.6 Decentralized traffic information systems: **a** single-tier VANET; **b** single-tier P2P over VANET; **c** single-tier infrastructure-based P2P; **d** two-tier VANET/P2P

forwards the query by exploiting the application-layer P2P lookup mechanism and the VANET routing.

Another single-tier architecture involves forming a P2P overlay through an infrastructure network. Vehicles are required to have a broadband wireless interface to access the infrastructure network. Vehicles communicate with each other through infrastructure communication instead of ad-hoc communication. Also in this case, the P2P overlay could be unstructured or structured.

A two-tier VANET/P2P architecture exploits both VANET and P2P technology. In the low tier, vehicles form a VANET via inter-vehicle communication to exchange traffic information. Some vehicles in are selected to form a high-tier P2P overlay through a broadband wireless infrastructure. These vehicles are called superpeers and serve as a bridge between the high-tier and low-tier networks to handle message exchanges and lookups. According to Tsao et al., the two-tier architecture achieves much higher lookup success rates than VANET-based systems and outperforms single-tier infrastructure-based P2P systems in terms of success rate, latency, and maintenance cost [33]. Nevertheless, we claim that in many cases the single-tier infrastructure-based P2P architecture as several advantages. In the following of this section we discuss some noticeable example.

The specific problem of geographic localization is addressed by Globase.KOM (Geographical LOcation BAsed SEarch) [34], which adopts an enhanced tree-based P2P overlay.The main focus is to enable search over all peers in a defined geographical area, which can be either circular or rectangular. A peer is enabled to search for a node with a particular location, or for the geographically closer peers. Together with the information about its geographical location, a peer can publish any other data describing the service it offers (e.g., a video stream from a webcam), the object it represents (e.g., restaurant, police station, sightseeing, gasoline station), or some additional information (e.g., menu, prices, opening hours). For example, users can find the closest gasoline station or can find all restaurants in some area and see their menu or video streams from webcams.

The Globase.KOM scheme is based on supernodes, i.e., powerful nodes, with best network connectivity, which tend to stay online for a long time. Supernodes are responsible for indexing all nodes/services in one clearly defined geographical area. Other nodes in the network simply offer and consume services, with no additional responsibilities. The idea is that the world projection is divided into disjoint, non-overlapping zones. Each zone is assigned to a supernode (located inside the zone itself), which has to collect, store and maintain the overlay/underlay contact addresses of all nodes in that zone (Fig. 2.7).

Active peers in Globase.KOM perform three main location-related operations, i.e., area search, address lookup, and discovery of the geographically closest nodes. Area search is performed using the SEARCH message, which includes a description of the geographical area (center and radius), plus metadata describing the targeted service/object. When a superpeer receives a SEARCH query from one of its peers, it calculates the searched ellipse onto the map projection. Next, it checks if that ellipse intersects the zone it is responsible for. Figure 2.8 illustrates an example of area search. A peer in the zone of superpeer B sends a SEARCH message containing a

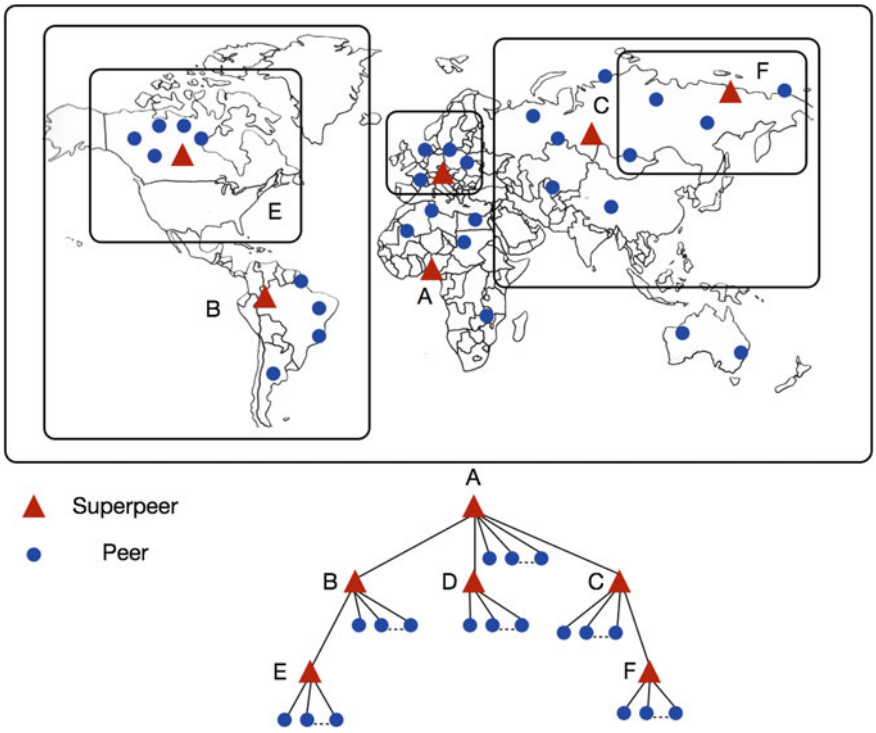


Fig. 2.7 Globase.KOM: division of world projection in multiple non-overlapping zones

description of the marked zone. As the zone does not intersect the zone for which superpeer B is responsible, the SEARCH message is forwarded to the superpeer A. In the end, superpeers A, C, and D reply with the list of the matching results.

Every superpeer maintains the contact addresses of the peers inside its zone (but not those of the peers which are into inner zones), of the superpeers responsible for inner zones (called child nodes), of its parent in the tree, of the root superpeer, and of interconnected superpeers. Each normal peer maintains the contact addresses of the parent superpeer, of the root superpeer, as well as an interconnection list and a cached list of already contacted peers. Peers/Superpeers are identified by their unique PeerID (Fig. 2.9), which contains the GPS coordinate of the node, if it is a supernode, the zone it is responsible for, and a random part, in order to support the existence of more than one peer at the same location.

The address lookup operation is used to determine the IP address and port of a peer, given its geographical location. Each superpeer knows the IDs/locations of all nodes it is responsible for. Therefore, a lookup operation basically means routing the LOOKUP message to the superpeer responsible for the peer with the given location.

When a peer wants to find the closest peer, it first calculates the closest border of the zone it belongs to. This is possible by using the ID of the parent superpeer,

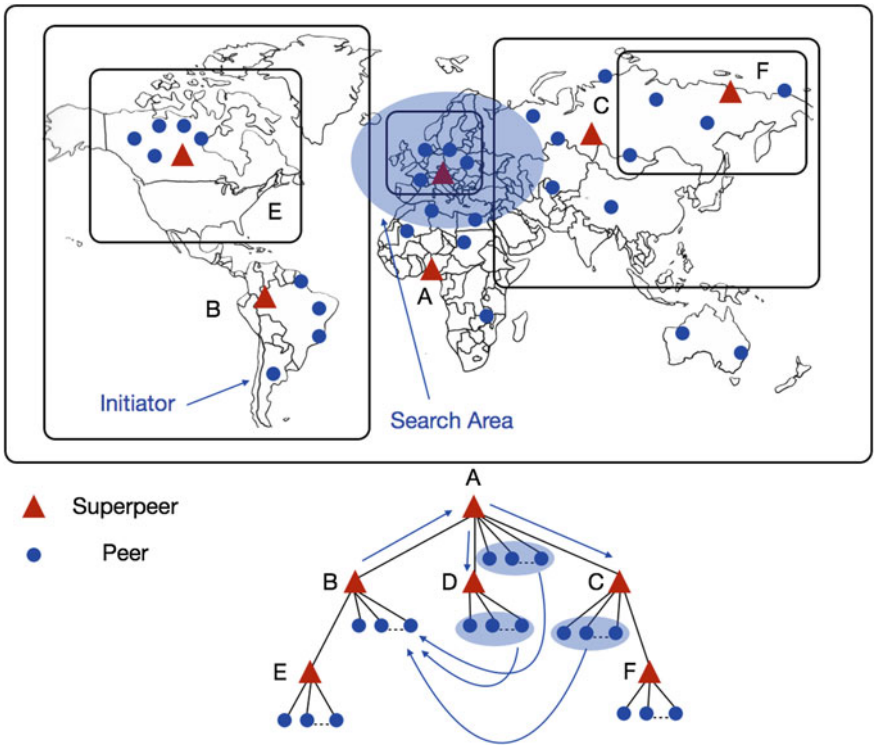
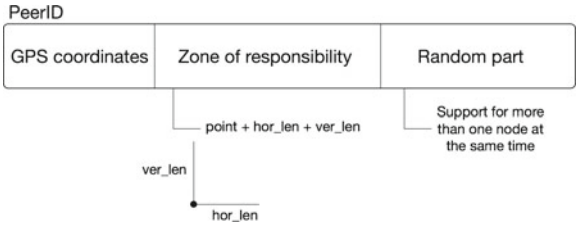


Fig. 2.8 Globase.KOM: example of Area Search procedure

Fig. 2.9 Globase.KOM: structure of PeerID



which contains a vector representation of the zone. Then, the peer sends a FIND CLOSEST message to its parent superpeer, containing the calculated distance to the closest border of the zone.

Another architecture, called GeoP2P [35], still performs a hierarchical partitioning of the 2D geographic space, but adopts a fully decentralized peer-to-peer overlay scheme, with overlay maintenance and query routing performed without super or special peers. The system consists of large number of peers, distributed across a 2-dimensional space with rectangular boundary. Each peer is placed in the 2D space, and is responsible for providing information which is relevant to its location. A peer

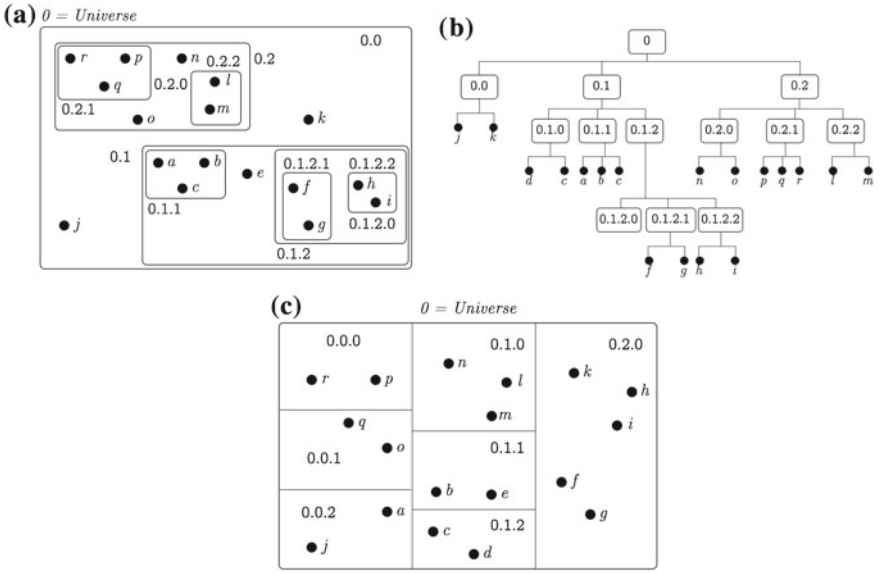


Fig. 2.10 GeoP2P: **a** Zoning by clustering; **b** Zoning Hierachy; **c** Zoning by splitting

can be associated to a single sensor, such as a surveillance camera, or to a database which contains information about the local environment, such are a hotel, or a gas station. Each piece of data stored in the overlay is updated independently. Additionally, any peer can be interested in any region in the space and send a query. The purpose of the overlay network is to route queries to relevant peers.

Also in GeoP2P, the universe is hierarchically divided into zones. At the top level of the hierarchy, the zone representing the universe is divided into a number of sub-zones, each one being further divided into sub-sub-zones at the next level of the hierarchy, and so on. Thus the zones can be conceptually organized into a tree, where the root of the tree represents the universe and each tree-node represents a zone. Figure 2.10a illustrates an example division of the universe, and the corresponding tree representation is shown in Fig. 2.10b.

Each peer maintains a routing table which lists all the other peers it knows. To resolve a query about any region in the universe, a peer tries to find a peer that belongs to the leaf zones intersecting the query region. To do that, each peer needs to have some structured knowledge to cover the globe, such that for any zone, it either knows all the peers belonging to that zone, or at least knows some peers which know more about that zone. Any query can thus be either resolved or forwarded to a peer that has better knowledge of the queried region.

The routing table is organized in d rows, one for each level of hierarchy, from 1 to d. Each row maintains information regarding $k - 1$ sibling zones of that level, plus some information for the self-zone. For each sibling zone, the table maintains the network address of one (or more) contact peer, associated with the rectangular

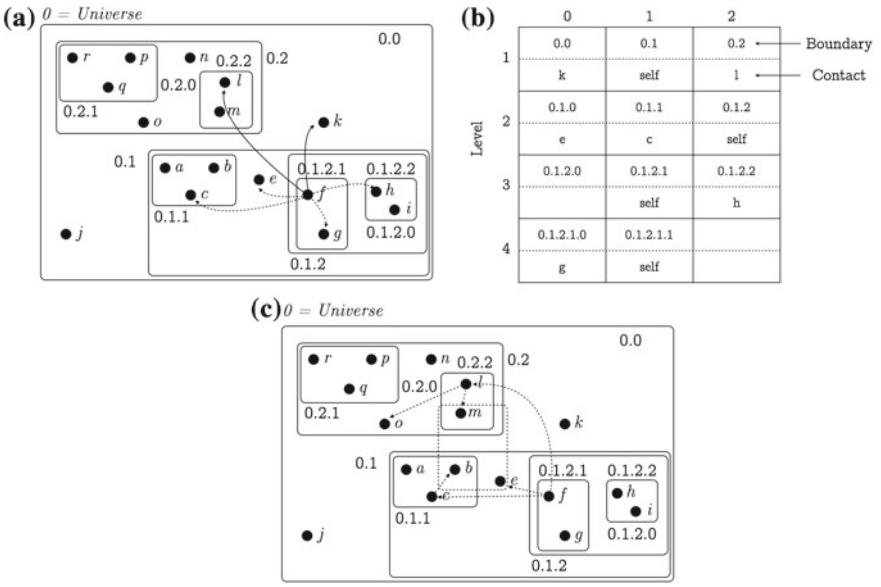


Fig. 2.11 GeoP2P: **a** overlay neighbors of peer f; **b** routing table of peer f; **c** routing path of a range query

boundary (coordinates of bottom-left and top-right corner) of the zone. Siblings can be organized into columns, based on the segment of the zone identifier which corresponds to the branching at that level. For the self-zone, only the zone boundary must to be maintained, and it can be stored in the corresponding column based the identifier of the self-zone. Level d stores the information regarding the leaf zone, and there the siblings are individual peers instead of zones. Thus, in this case the coordinates of the peers are stored, instead of the rectangular boundaries. Figure 2.11b shows an example routing table of a peer. The corresponding overlay neighborhood is illustrated in Fig. 2.11a.

The main drawback of the hierarchical approach is that peers representing higher level regions may become bottlenecks for query routing, and possible points of failure for the whole system. Moreover, none of the state-of-art solutions has been demonstrated to work in presence of mobile peers.

Rybicki et al., with Peers on Wheels [36], PeerTIS [37], and GraphTIS [38] proposed P2P architectures where participating cars are peers organized in a Distributed Hash Table (DHT), to receive and distribute useful information to improve the vehicle travel time using dynamic route guidance. In PeerTIS, roads are divided into segments, each with a unique ID that is used as key in the DHT. The main idea is that each node is responsible for a certain part of the ID space and, consequently, for a certain number of road segments. GraphTIS was designed to overcome the main issue of PeerTIS, i.e., that the geographical coordinates of road segments are not uniformly distributed over the key space (it was observed that, in a realistic simulated

scenario, almost 40 % of all peers do not maintain any data at all). in GraphTIS, graph partitioning algorithms are employed to dissect the graph into a number of (almost) equally sized disjoint sub-graphs. The aim is still to minimize the effort for looking up a group of road segments. Whenever a query crosses a boundary between two peers, it must be transmitted over the network, causing network traffic and increasing the query latency. Thus, the partitions should be formed in such a way that typical queries touch as few partitions (and hence as few peers) as possible, i.e., the partitions should be chosen in a way that minimizes the number of cut graph edges. It is known that optimal graph partitioning is NP-hard. In GraphTIS, a combination of graph growing and the Kernighan-Lin algorithm [39] are used. The hierarchical partitioning and labeling is constant for a given map, so it is pre-calculated once, before the system starts running, and stored along with the street map data. At runtime each peer maintains all (key, value) mappings for the sub-graph represented by one node in the partitioning tree.

2.5 Enabling Technologies

As we have seen in Chap. 1, the approach followed by international standardization bodies, notably ETSI and ISO, consists in adopting an open communication architecture embracing a mix of protocols and technologies, either dedicated or general purpose. In Table 2.1, we have summarized the main characteristics of principal communication technologies that can be currently used in ITSs. We have considered the dedicated IEEE 802.11p standard, a mix of general purpose existing technologies including IEEE 802.11, and two generations of cellular networks, namely UMTS and LTE.

In such a table, IEEE 802.11p is the unique dedicated technology and therefore it natively offers most of the features required by ITSs applications, including support for all broadcast and V2X communication capabilities, including multihop. It is also boasts a small latency and it allows to perform communications without a preemptive setup of the network. This is a key feature for hard-safety applications that require a low delay. As today, the low market penetration is the most critical issue of IEEE 802.11p.

WiFi (with this term we refer to classical IEEE 802.11a/b/g standards) shares some characteristics with IEEE 802.11p, including native support for broadcast and V2X communication capabilities, even if at less reliable degree than IEEE 802.11p. On the other hand, WiFi suffers of the limited data range and of the limited mobility support, due to the high network setup time. However, WiFi can exhibit a very high market that can be exploited for both in-vehicles and V2X communications.

UMTS and LTE exhibit characteristics that are almost orthogonal with respect to WiFi and IEEE 802.11p. In fact, both UMTS and LTE exhibit ubiquitous coverage, high mobility support and V2I capabilities. However, they lack support for direct V2V communications, and they support local broadcast only through dedicated protocol specifications, notably Multimedia Broadcast Multicast Service (MBMS) for UMTS

and evolved Multimedia Broadcast Multicast Service (eMBMS) for LTE [40], [41]. However, the deployment of MBMS and eMBMS is totally dependent on the mobile operators' willingness. Currently, the diffusion of eMBMS is pretty scarce, and, at the best of our knowledge, the are no implementation of MBMS. In terms of performance, there is a clear gap between UMTS and LTE, in terms of data rate, end-to-end delay and setup type. While LTE copes with the exigences of a large number of ITSs applications, including safety dedicated tasks, UMTS can be employed only for a limited class of services.

2.5.1 Cellular Networks

The cellular networks deployed today are based on a heterogeneous mix of different generations of communication technologies, starting from 2G, and including 3G, 3.5G and LTE. This heterogeneity and the frequent generation upgrades has historically discouraged the use of cellular networks in ITSs applications, in particular in USA, where car owners and state DOTs feared to constantly upgrade equipment in vehicles and intersections, respectively.[1]

There is a pervasive diffusion of compatible devices (smartphones, tablets, OISs) and the existing network infrastructure guarantees a worldwide coverage and it is constantly upgraded by operators. This means that there is no need for additional

Table 2.1 Performance indicators of main ITSs communication technologies

Feature	IEEE 802.11p	WiFi	UMTS	LTE
Type	Dedicated	General purpose	General purpose	General purpose
Market Penetration	Low	High	High	Potentially High
Bit Rate	3–27 Mbit/s	6–54 Mbit/s	2 Mbit/s	up to 300 Mbit/s
End-to-end delay	10 ms	10 ms	50–100 ms	10 ms
Setup time	0	a few seconds	100 ms up to seconds	50-100ms
Maximum Range	1 km	0.1 km	10 km	30 km
Coverage	Intermittent	Intermittent	Ubiquitous	Ubiquitous
Mobility support	Medium	Low	High	Very High
V2V Local Broadcast	Yes	Yes	Through server	Through server
V2V Multihop	Yes	Yes	Through server	Through server
V2I Bidirectional	Yes	Yes	Yes	Yes
I2V Local Broadcast	Yes	Yes	Partially (MBMS)	Partially (eMBMS)

[1] http://www.itsa.org/industryforums/connectedvehicle.

dedicated infrastructures. Moreover, current generations of cellular technology (3G, 3.5G and LTE) offers low latency and high throughput, and may operate with vehicle speed up to 300Km/h. For these reasons, cellular technologies are included in both ETSI and ISO architectures, and are already used in vertically integrated ITS applications (i.e, insurance company black box), or to provide TIS-related services [42].

However, current cellular technologies are affected by a series of technical limitations that make them unsuitable to fit the requirements of safety-related applications, which play a pivotal role in ITS. The main limitations, that emerge clearly from Table 2.1, are related to the lack of support for broadcast communications and V2V communications and for the significant latency, which is too high for most hard-safety ITS applications (e.g., collision avoidance). The upcoming LTE-Advanced (LTE-A) generation promises to be a game changer. LTE-A is committed to provide technologies for high data rates and system capacity. Further, LTE-A was defined to support new components for LTE, to meet new communication demands coming from the ITS community and from Machine-to-Machine (M2M). On the one hand, LTE-A will provide more advanced local broadcast eMBMS mechanisms, thus bringing benefits to ITS communications. On the other hand, LTE-A will be provide Device-to-Device (D2D) communication capabilities to be exploited in V2V services. In D2D mode, terminals may communicate directly with their neighbors, thus bypassing the data plane of cellular base stations (denoted as eNodeB in LTE), while maintaining their leadership role in the control plane [43].

2.5.2 WiFi and WiFi Direct

The IEEE 802.11 standard specifies physical and MAC layers to set up wireless local area networks (WLANs) [44]. Although its first release was published in the far 1997, it is not yet an obsolete technology. In order to accommodate the introduction of new functions, the standard has been continuously modified, with specific amendments processes. An up-to-date version of the standard, aggregating several of these amendments, was released in the 2007 and denoted as IEEE 802.11-2007 [45]. In particular, the *a*, *b*, and *g* amendments were introduced, respectively as the chapter 17, 18, and 19, while the amendment *e*, introducing the support for Quality of Service (QoS) at MAC level, has been merged with chapter 9 of the standard. Two important amendments, IEEE 802.11p [46] and IEEE 802.11n [47], have been recently included in the standard, and the IEEE 802.11ac amendment will be standardized soon.

The basic building block of an IEEE 802.11 network is the Basic Service Set (BSS), a group of STAtions (STAs) that may communicate with each other. IEEE 802.11 offers different opportunities to build a BSS. For instance, nodes can form an Independent BSS (IBSS) with no central coordination authority, or, as in environments with infrastructure (i.e., AP) be part of an infrastructure BSS, with an individual identification number. In both cases, the channel access mechanism is based on a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

strategy. When STAs operate in an IBSS, the network operates in ad-hoc mode, which is the only form that allows to perform V2V communications. However, the ad-hoc mode is affected by two main limitations, one concerning synchronization issues, and the other affecting communication security.

Synchronization issues are related to the intrinsic characteristics of the CSMA/CA mechanism. In fact, in order to correctly transmit and receive data, all the STAs within the same IBSS have to be locked to a common clock [45]. In order to reach this goal, the standard defines a suitable mechanism, known as Timing Synchronization Function (TSF) [44]. In the case of infrastructure-based networks, all the STAs are synchronized with the AP's clock. On the other hand, in ad-hoc networks it is necessary to employ a suitable distributed algorithm to synchronize the STAs. In all the cases, synchronization information is obtained through the transmission of a special frame, denoted as beacon. It can be observed that in the case of ad-hoc networks the synchronization issue limits the maximum number of nodes that can be belong to the same IBSS [48]. Some issues are related to the nature of the Wi-Fi Protected Access (WPA e WPA2) mechanisms that provide the security layer of the IEEE 802.11 protocol [49]. In fact, if applied in ad-hoc networks, WPA introduces significant scalability problems, as it requires to exchange a number of keys directly proportional to the number of STAs.

In order to overcome these limitations, the WiFi Alliance has recently realized a new protocol, denoted as WiFi Direct [50], with the goal to overcome the numerous limits of the ad-hoc mode. However, the WiFi Direct mode has currently a low market share, since it is available in a small number of top level devices. Therefore, it is not possible to realize an application with a large diffusion based on WiFi Direct.

WiFi Direct devices build topologies based on groups, following the classic concept of BSS. Each group has a leader which can be considered an access point to the network. One of the main peculiarity of WiFi Direct with respect to the classic IEEE 802.11 standard is the fact that the access point of the network can be dynamically chosen. The leader of the group is dynamically designed and not fixed, unlike infrastructure-based networks. For this reason, all WiFi Direct devices should be able to become group leaders, supporting the functionality of device discovery and mechanisms for the coexistence with other types of IEEE 802.11 networks. In fact, WiFi Direct has been conceived to be used simultaneously to a infrastructure-based network, thus yielding to a higher level of scalability than pure ad-hoc networks. However, WiFi Direct is strongly limited by the fact that it cannot be directly used to perform multihop communications. To overcome such a limit, it is possible to use the advanced functionalities of WiFi Direct, as the support for multiple groups and to transverse communications. Unfortunately, most of these features are not supported by the devices available on the market.

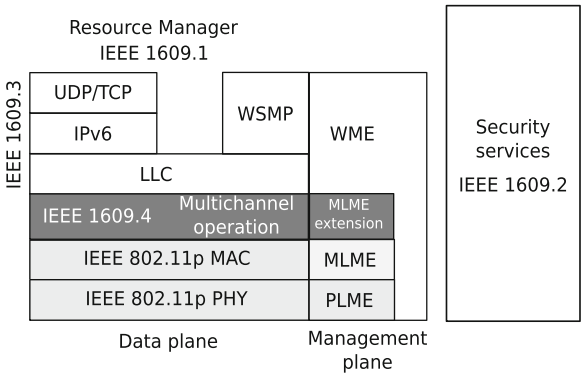


Fig. 2.12 Protocol stack jointly defined by the IEEE 1609 and the IEEE 802.11p standards

2.5.3 IEEE 802.11p and WAVE

The IEEE has realized a totally new protocol stack, commonly denoted as WAVE, which copes with the IVC requirements: highly dynamic and mobile environment, message transmission in an ad-hoc manner, low latency, and operation in a reserved multi-channel frequency range. As shown in Fig. 3.4, WAVE maintains the traditional ISO/OSI protocol stack [51] and is composed by a collection of standards, namely, IEEE 802.11p the IEEE 1609 standards family [52]. The latter—which has been already discussed in Sect. 1.3.3—defines higher layer services, such as system architecture, security, resource management and communication model [53], while IEEE 802.11p is focused on physical and MAC layers.

In November 2004, the IEEE 802.11p Task Group was formed to develop an amendment to the IEEE 802.11 standard, to add *wireless access in vehicular environments (WAVE)*, by defining enhancements to IEEE 802.11 including data exchange between high-speed vehicles and between vehicles and roadside infrastructure, in the licensed ITS band of 5.9 GHz (5.85–5.925 GHz). IEEE 802.11p was considered for vehicle-based communication networks—in particular, for toll collection applications, vehicle safety services, and commerce transactions via cars. The ultimate vision was a nationwide network enabling communications between vehicles and roadside access points or other vehicles. The last approved amendment of IEEE 802.11p was incorporated in IEEE 802.11 standard [45], published in 2012 (Fig. 2.12)

The IEEE 802.11p standard differs form the existing IEEE 802.11a standard in three main aspects [54]: (i) the definition of BSS; (ii) some details of the physical layer; (iii) the MAC layer. In the WAVE mode, data packets transmission is only allowed to occur within an IBSS, which is established in a fully ad-hoc manner, without any need for active scanning, association or authentication procedures. A node that initiates an IBSS is called provider, while a node that joins an IBSS is called user. To establish an IBSS, the provider has to periodically broadcast on CCH an IBSS announcement message, which includes the WAVE Service Advertisement (WSA).

The latter message contains all the information identifying WAVE applications and associated network parameters, necessary to join an IBSS (e.g., the ID, the SCH index, and timing information). A node should monitor all WSAs on CCH to learn about the existence and the operational parameters of the available IBSSs.

The IEEE 802.11p physical layer is an amended version of the IEEE 802.11a specifications, thus it is based on OFDM modulation. It mandates the use of 10 MHz channels, which offer a greater resistance with respect to the channel delay spread, thanks to their double guard time (1.6 μs). The 10 MHz frequency leads to halved data rates, and the maximum sustainable data rate becomes 27 Mbit/s. We remark that, differently from the IEEE 802.11a, whose use was forbidden for several years in Europe, the use of IEEE 802.11p is already allowed, regulated by the ETSI European Standard 202 663 [55].

The IEEE 802.11p MAC layer has the same Enhanced Distributed Channel Access (EDCA) core mechanism of introduced in the IEEE 802.11e amendment [56]. EDCA maintains the distributed approach of the CSMA/CA protocol as in legacy DCF, but introduces four Access Categories (ACs), each one defining a priority level for channel access and having a corresponding transmission queue at the MAC layer. Each AC in the queue behaves like a virtual STA, and it follows its own DCF algorithm, independently contending with the others to obtain the channel access. Each i-th AC has a set of distinct channel access parameters, including Arbitration Inter-Frame Space (AIFS) duration and contention window size ($CW_{min}[i]$ and $CW_{max}[i]$). The AIFS has the same meaning of DIFS parameter in the DCF algorithm but the different duration, and it is defined as:

$$T_{\text{AIFS}}[i] = T_{\text{SIFS}} + \text{AIFSN[i]} \cdot T_{\text{SLOT}},$$

where AIFSN[i] is an adimensional parameter different for every AC. Clearly, when AIFSN[i] $= 2$, $T_{\text{AIFS}}[i]$ becomes identical to T_{DIFS}. The amendment [56] has also introduced the possibility of sending a train of consecutive frame by the concept of Transmission Opportunity (TXOP), but this feature is not exploited by the IEEE 802.11p amendment.

In the IEEE 802.11p standard the access mechanism is properly modified to work in the multi-channel WAVE environment, by implementing two separate EDCA functions, one for CCH and one for SCH, which handle different sets of queues for packets destined to be transmitted on different channel intervals, as shown by Fig. 2.13. Table 2.2 summarizes the most interesting parameters of the physical and MAC layers of IEEE 802.11, with the exception of the EDCA parameters, which are listed in Table 2.3. From Table 2.3 we observe that IEEE 802.11p uses the same CW_{min} and CW_{max} values of the original IEEE 802.11e specification [56], but slightly modified AIFSN values. While in standard WLAN the AC_VI and AC_VO correspond, respectively, to Video and Voice, in the case of IEEE 802.11p, AC_VI and AC_VO must be interpreted as ACs reserved for prioritized messages (e.g., critical safety warnings).

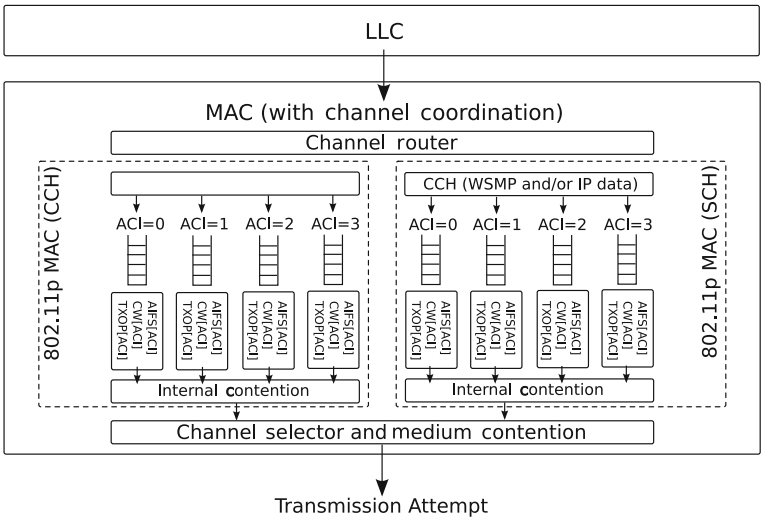


Fig. 2.13 Enhanced Distributed Channel Access mechanism defined by the IEEE 802.11p standard

Table 2.2 Main parameters of the IEEE 802.11p standard

Parameter	IEEE 802.11p
Carrier Frequency (GHz)	5.9
Bandwidth (MHz)	10
OFDM Guard Time (μs)	1.6
CW_{min}	See Table 2.3
CW_{max}	1023
T_{SLOT} (μs)	13
T_{SIFS} (μs)	32
Data rates (Mbit/s)	3, 4.5, 6, 9, 12, 18, 24, 27

Table 2.3 EDCA parameters of the IEEE 802.11p standard

AC	CW_{min}	CW_{max}	AIFSN
AC_BK	15	1,023	9
AC_BE	15	1,023	6
AC_VI	7	15	3
AC_VO	3	7	2

2.5.4 ETSI ITS Protocol Stack

The main candidate protocol stack for ITS applications designed by ETSI is illustrated in Fig. 2.14. The physical and MAC layers have been standardized in 2009 by ETSI in the ITS-G5 protocol [55], which is largely based on IEEE 802.11p. The design goals and principles of ITS are the following:

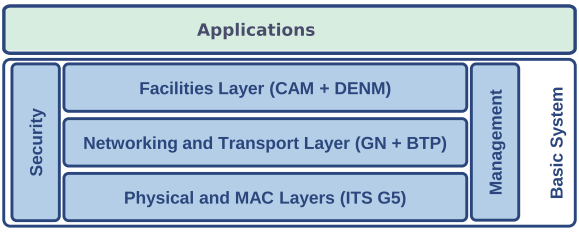


Fig. 2.14 ETSI protocol stack

- quick media access (low latency broadcast/unicast communication);
- ad-hoc communication (no infrastructure requirements);
- allocated spectrum for ITS (communication reliability);
- 200-800m communication range;

The ITS-G5 protocol supports the Basic Transport Protocol (BTP) [57] and the GeoNetworking protocol for V2X communication [58], based on results from project GeoNet. GeoNetworking, in turn, uses both the BTP protocol and a own location-based addressing for all communications, including single-hop communication between ITSs. Finally, within the facilities, lay the two types of safety messages standardized by ETSI, referred as Cooperative Awareness messages (CAMs) [19] and Decentralized Environmental Notification Messages (DENMs) [20]. CAMs are heartbeat periodic messages, delivered to vehicles laying in the awareness range of the sender. DENMs are event-triggered messages delivered to vehicles laying in the relevant geographical area of the triggering event. Such a region of interest can span several hundred meters.

References

1. Willke, T., Tientrakool, P., Maxemchuk, N.: A survey of inter-vehicle communication protocols and their applications. IEEE Commun. Surv. Tutor. **11**(2), 3–20 (2009)
2. Lee, U., Magistretti, E., Gerla, M., Bellavista, P., Corradi, A.: Dissemination and harvesting of urban data using vehicular sensing platforms. IEEE Trans. Veh. Technol. **58**(2), 882–901 (2009)
3. Kato, S., Tsugawa, S., Tokuda, K., Matsui, T., Fujii, H.: Vehicle control algorithms for cooperative driving with automated vehicles and intervehicle communications. IEEE Trans. Intell. Transp. Syst. **3**(3), 155–161 (2002)
4. Broggi, A.: VisLab's vehicles just reached siberia: driverless! now Kazakhstan and then China until Shanghai [ITS Events]. IEEE Intell. Transp. Syst. Mag. **2**(3), 43–44 (2010)
5. Machine to machine communications (M2M);use cases of automotive applications in M2M capable networks. ETSI TS 102 898 pp. 1–46 (2013)
6. Sepulcre, M., Mittag, J., Santi, P., Hartenstein, H., Gozalvez, J.: Congestion and awareness control in cooperative vehicular systems. Proc. IEEE **99**(7), 1260–1279 (2011)
7. Zhu, K., Niyato, D., Wang, P., Hossain, E., Kim, D.: Mobility and handoff management in vehicular networks: a survey. Wirel. Commun. Mob. Comput. **11**(4), 459–476 (2011)

8. Naumov, V., Gross, T.R.: Connectivity-aware routing (CAR) in vehicular ad-hoc networks. In: Proceedings of IEEE Conference on Computer Communication (INFOCOM), pp. 1919–1927. IEEE (2007)
9. Mecklenbrauker, C.F., Molisch, A.F., Karedal, J., Tufvesson, F., Paier, A., Bernado, L., Zemen, T., Klemp, O., Czink, N.: Vehicular channel characterization and its implications for wireless system design and performance. Proc. IEEE **99**(7), 1189–1212 (2011)
10. Vandenberghe, W., Moerman, I., Demeester, P.: On the feasibility of utilizing smartphones for vehicular ad hoc networking. In: ITS Telecommunications (ITST), 2011 11th International Conference on, pp. 246–251. IEEE (2011)
11. Fiore, M., Härri, J.: The networking shape of vehicular mobility. In: Proceedings of ACM International Symposium on Mobile ad hoc networking and computing (MobiHoc), pp. 261–272. ACM, New York, NY, USA (2008)
12. Andrews, J.G., Ghosh, A., Muhamed, R.: Fundamentals of WiMAX: understanding broadband wireless networking. Pearson Education (2007)
13. Insitute of Electrical and Electronics Engineers: IEEE Std 802.16TM-2009. Part 16: Air Interface for Broadband Wireless Access Systems (2009)
14. Sesia, S., Toufik, I., Baker, M.: LTE: the UMTS long term evolution. Wiley Online Library (2009)
15. Dahlman, E., Parkvall, S., Skold, J., Beming, P.: 3G evolution: HSPA and LTE for mobile broadband. Elsevier, Amsterdam (2010)
16. Dahlman, E., Parkvall, S., Skold, J.: 4G: LTE/LTE-Advanced for Mobile Broadband: LTE/LTE-Advanced for Mobile Broadband. Academic Press, Waltham (2011)
17. Bleicher, A.: 4g gets real. IEEE Spectr **51**(1), 38–62 (2014)
18. Schoch, E., Kargl, F., Weber, M., Leinmuller, T.: Communication patterns in VANETs. IEEE Commun. Mag. **46**(11), 119–125 (2008)
19. Intelligent Transport Systems (ITS), vehicular communications, basic set of applications, part 2: Specification of cooperative awareness basic service. ETSI TS 102 637–2, pp. 1–18 (2011)
20. Intelligent Transport Systems (ITS), vehicular communications, basic set of applications, part 3: Specifications of decentralized environmental notification basic service. ETSI TS 102 637–3, pp. 1–46 (2010)
21. Zhang, T., Delgrossi, L.: Vehicle Safety Communications: Protocols, Security, and Privacy. Wiley Press, Hoboken (2012)
22. Inc., G.: Google latitude. https://latitude.google.com/latitude/b/0
23. Vodafone: Vodafone machine to machine. http://m2m.vodafone.com/home/
24. Thiagarajan, A., Ravindranath, L.S., LaCurts, K., Toledo, S., Eriksson, J., Madden, S., Balakrishnan, H.: Vtrack: Accurate, energy-aware traffic delay estimation using mobile phones. In: 7th ACM Conference on Embedded Networked Sensor Systems (SenSys). Berkeley, CA (2009)
25. Mobile, W.: Waze. http://www.waze.com
26. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: 14th ACM SIGMOD, vol. 14, pp. 47–57. ACM (1984)
27. Tran, D.A., Nguyen, T.: Hierarchical multidimensional search in peer-to-peer networks. Comput. Commun. **31**(2), 346–357 (2008)
28. Li, M., Lee, W., Sivasubramaniam, A.: Dptree: A balanced tree based indexing framework for peer-to-peer systems. In: 14th IEEE ICNP (2006)
29. Namand, B., Sussman, A.: Dist: fully decentralized indexing for querying distributed multidimensional datasets. In: 20th IPDPS (2006)
30. Tanin, E., Harwood, A., Samet, H.: Using a distributed quadtree index in peer-to-peer networks. VLDB J. **16**(2), 165–178 (2007)
31. Liu, B., Lee, W.C., Lee, D.L.: Supporting complex multi-dimensional queries in p2p systems. In: 25th IEEE International Conference on Distributed Computing Systems (ICDCS '05) (2005)
32. Harwood, A., Tanin, E.: Hashing spatial content over peer-to-peer networks. In: Australasian Telecommunications, Networks and Applications Conference (ATNAC) (2003)

33. Tsao, S.L., Cheng, C.M.: Design and evaluation of a two-tier peer-to-peer traffic information system. IEEE Commun. Mag. **49**(5), 165–172 (2011)
34. Kovacevic, A., Liebau, N., Steinmetz, R.: Globase.kom - a p2p overlay for fully retrievable location-based search. In: 7h IEEE International Conference on Peer-to-Peer Computing (2007)
35. Asaduzzaman, S., von Bochmann, G.: Geop2p: An adaptive peer-to-peer overlay for efficient search and update of spatial information. CoRR abs/0903.3759 (2009)
36. Rybicki, J., Scheuermann, B., Kiess, W., Lochert, C., Fallahi, P., Mauve, M.: Challenge: peers on wheels—a road to new traffic information systems. In: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (2007)
37. Rybicki, J., Scheuermann, B., Koegel, M., Mauve, M.: Peertis: a peer-to-peer traffic information system. In: Proceedings of the sixth ACM International Workshop on VehiculAr Inter-NETworking (2009)
38. Rybicki, J., Scheuermann, B., Mauve, M.: Peer-to-peer data structures for cooperative traffic information systems. Pervasive Mob. Comput. **8**(2), 194–209 (2012)
39. Kernighan, B., Lin, S.: An efficient heuristic procedure for partitioning graphs. Bell Syst. Tech. J. **49**(1), 291–308 (1970)
40. Intelligent transport systems (ITS); framework for public mobile networks in cooperative ITS (C-ITS). Technical report (2012)
41. Valerio, D., Ricciato, F., Belanovic, P., Zemen, T.: Umts on the road: broadcasting intelligent road safety information via mbms. In: Vehicular Technology Conference, 2008. VTC Spring 2008. IEEE, pp. 3026–3030. IEEE (2008)
42. Araniti, G., Campolo, C., Condoluci, M., Iera, A., Molinaro, A.: LTE for vehicular networking: a survey. IEEE Commun. Mag. **51**(5), 148–157 (2013)
43. Doppler, K., Rinne, M., Wijting, C., Ribeiro, C., Hugl, K.: Device-to-device communication as an underlay to lte-advanced networks. IEEE. Commun. Mag. **47**(12), 42–49 (2009)
44. Gast, M.: 802.11 Wireless Networks: The Definitive Guide, Second edn. O'Reilly Media, Sebastopol (2005)
45. Insitute of Electrical and Electronics Engineers: IEEE Std 802.11TM-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications (2012)
46. IEEE: IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007), pp. 1–51 (2010)
47. IEEE Standard for Information technology-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 5: Enhancementss for Higher Throughput. IEEE Std 802.11n-2009, pp. 1–211 (2005)
48. Zhou, D., Huang, L., Lai, T.H.: On the scalability of IEEE 802.11 ad-hoc-mode timing synchronization function. Wirel. Netw. **14**, 479–499 (2008)
49. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6:Medium Access Control (MAC) Security Enhancements. IEEE Std 802.11i-2004, pp. 1–190 (2004)
50. Wi-Fi Alliance: Wi-Fi CERTIFIED Wi-Fi Direct™ (2010). Disponibile: http://www.wi-fi.org/register.php?file=wp/Wi-Fi/Direct/20101025/Industry.pdf
51. Zimmermann, H.: OSI reference model-The ISO model of architecture for open systems interconnection. IEEE Trans. Commun. **28**(4), 425–432 (2002)
52. Uzcategui, R., Acosta-Marum, G.: WAVE: a tutorial. IEEE Commun. Mag. **47**(5), 126–133 (2009)
53. Insitute of Electrical and Electronics Engineers: IEEE 1609–2006. IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE) (2006)
54. Jiang, D., Delgrossi, L.: IEEE 802.11 p: Towards an international standard for wireless access in vehicular environments. In: Vehicular Technology Conference, pp. 2036–2040. Marina Bay, Singapore (2008)

55. European profile standard for the physical and medium access control layer of Intelligent Transport Systems operating in the 5 ghz frequency band. ETSI ES 202 663 V1.1.0, pp. 1–27 (2010)
56. IEEE Standard for Information technology-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements. IEEE Std 802.11e-2005, pp. 1–211 (2005)
57. Intelligent Transport Systems (ITS); vehicular communications; geonetworking; part 5: Transport protocols; sub-part 1: Basic transport protocol. ETSI TS 102 636-5-1, pp. 1–45 (2011)
58. Intelligent Transport Systems (ITS); vehicular communications; geonetworking. ETSI TS 102 636 V1 (all parts), pp. 1–44 (2011)

Chapter 3
Wireless Communications for Vehicular Ad-Hoc Networks

3.1 Information Dissemination in Loosely-Coupled VANETs

The goal of this book is to explore the possibility to enable ITSs services by exclusively leveraging on existing technologies such as WiFi networks and cellular networks. In this chapter, we evaluate the feasibility of using WiFi networks to disseminate or collect information on a small geographical area with a pure V2V approach. As it does not make sense to build a novel WiFi infrastructure for vehicular communications, we will investigate the possibility to realize V2V-based services by leveraging on commodity hardware and software, which can be currently found on smartphones, tablets and OISs. The main advantage of this approach is the dramatic economic cost reduction, while the main drawback is the heritage of all the characteristics of the WiFi technology, without any chance of adapting it for the vehicular communications scenario. In fact, relying on already deployed devices prevents from performing any technical optimization in some crucial areas such as low layer protocol stacks (physical and MAC layer) and antenna design.

As a consequence, our scenario is affected by a series of issues. First of all, the transmission range of a device maxes out at 100 m but it is often far smaller. The physical layer of WiFi technology has not been conceived for communication in the harsh conditions that characterizes a vehicular wireless channel, which reduces the reliability of the communications. Furthermore, the infrastructure mode of the WiFi protocol has a high setup time and does not scale very well, making difficult to adapt it to highly dynamic vehicular network topologies. For these scenarios, the WiFi ad-hoc mode can be a better choice, at least in terms of scalability.

Because of these important issues, the IEEE 802.11 technology cannot be used to build and maintain stable VANETs. Instead, it can only used to build loosely-coupled VANETs, whose members have weak bonds and the concept itself of network topology is feeble. In such VANETs, it is infeasible to use unicast V2V protocols to disseminate information, as the network topology is not sufficiently stable. Therefore, in loosely-coupled VANETs, information can be disseminated exclusively by means of broadcast or geocast protocols, either single- or multi-hop. Single-hop broadcast

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7_3

protocols have an intrinsic higher transmission efficiency, but they have a limited transmission range and they can therefore send information only in a small region centered in the source. Conversely, multi-hop broadcast have lower transmission efficiency due to the half-duplex nature of the employed radios, but allow to reach vehicles that are out of the transmission range of the original source. Furthermore, with multi-hop broadcast protocols, it is possible to provide some geocast functionalities also in loosely-coupled networks.

In this chapter, we focus on both information dissemination applications and data collection applications. In particular, we present a novel theoretical framework for the analytical performance evaluation of a family of multihop broadcast dissemination protocols that can be really installed on already deployed WiFi-ready devices, and therefore do not require to modify the lower layers of the protocol stack. Such a low complexity theoretical framework is useful to characterize the main performance metrics of a family of probabilistic multihop broadcast protocols with applications to VANET scenarios. First, we show that the average positions of a given number of points of a PPP falling in a segment with finite length are equally spaced. Then, assuming a *silencing* mechanism at each hop, we derive a recursive (hop-wise) theoretical performance evaluation framework which exploits the assumption of fixed and equally spaced vehicles positions in each retransmission hop. In particular, such a performance analysis is likely to be representative of the average (with respect to the nodes' spatial distribution) performance of the broadcast protocols at hand, as it is confirmed by simulations carried out with the Network Simulator 2 (ns-2) [1]. Moreover, the proposed analytical model applies also to other vehicle spatial distributions, provided that the average inter-vehicle distance is fixed. The impact of node mobility will also be evaluated. Although we consider two novel illustrative broadcast protocols, we underline that our approach is general.

As aforementioned, we also analyze a data collection scenario, where the VANET acts as a vehicular sensor network. The presented approach consists in the creation, during a downlink phase, of a clustered VANET topology during fast broadcast data dissemination, from the Access Point (AP), through a novel clustering protocol, denoted as Cluster-Head Election IF (CHE-IF). This clustered topology is then exploited, during an uplink phase, to collect information from the vehicles and perform distributed detection. Our results highlight the existing trade-off between decision delay and energy efficiency. Unlike classical sensor networks for distributed detection, the proposed vehicular distributed detection schemes exploit the natural vehicle clustering and have to cope with their "ephemeral" nature. More precisely, vehicle mobility has a direct impact on the maximum amount of data which can be collected, thus leading to the concept of decentralized detection on the move. In particular, we analyze the performance of vehicular decentralized detection schemes, based on the observation, by all vehicles of a VANET, of a spatially constant phenomenon of interest.

The chapter is structured as follows. In Sect. 3.2, multihop broadcast protocols for linear networks are introduced. Section 3.3 is devoted to the derivation of the average distribution of a given number of points of a PPP in a segment with finite length. In Sect. 3.4, a succinct overview of the IEEE 802.11b standard is provided. In

Sect. 3.5, the family of probabilistic broadcast protocol with silencing is accurately described. In Sect. 3.6, the analytical framework for performance evaluation of the probabilistic broadcast protocols of interest is presented. In Sect. 3.7, after the validation of the analytical framework by means of numerical simulation, the performance of the novel probabilistic broadcast protocols is investigated and compared with that of other (known) protocols. Finally, in Sect. 3.8 we analyze the data collection scenario, where the VANET acts as a vehicular sensor network.

3.2 Multihop Broadcast Protocols

Reducing the number of redundant packets, while still ensuring good coverage and low latency, is one of the main objectives in multi-hop broadcasting. In fact, a too large number of transmissions acts unavoidably leads to unsustainable levels of latency, retransmissions, and collisions: the overall phenomenon is typically referred to as broadcast storm problem [2] and it mainly affects dense networks. The problem of minimizing the number of transmissions has been deeply investigated by the Mobile Ad-hoc NETworks (MANETs) research community: the theoretically optimal solution consists in designating, as relays, the nodes belonging to the Minimum Connected Dominant Set (MCDS) of the network [3]. The nodes within the MCDS have the following properties: (i) they form a connected graph; (ii) every other node of the network is one-hop connected with a node in the MCDS; (iii) the MCDS has the lowest cardinality over all the possible collections of nodes that satisfy the previous two requirements.

The general goal of a multihop broadcast protocol is to attain the widest network coverage in the shortest possible time. This can be obtained by pursuing three intermediate goals: (i) minimizing the number of communication hops; (ii) minimizing the number of effective retransmissions in every hop; (iii) minimizing the latency associated with a single hop. The number of transmission hops can be minimized by designating, as relays, the nodes forming the MCDS. However, the number of retransmissions and the latency are directly affected by the protocol characteristics, and there is no general rule for minimizing them.

Following the "idealized" MCDS-based design approach, a plethora of multihop broadcast protocols have been recently proposed in the VANET literature. Some of them, such as the Emergency Message for Vehicular environments (EMDV) protocol [4], achieve remarkable performance by exploiting partial or complete knowledge of the network topology [5]. However, since collecting this type of information may be very expensive in terms of overhead, other techniques (requiring a reduced information exchange) have been proposed. An efficient IEEE 802.11-based protocol, denoted as Urban Multihop Broadcast (UMB), was proposed by Korkmaz et al. [6, 7]. UMB suppresses the broadcast redundancy by means of a black-burst contention approach [8], followed by a Ready-To-Send/Clear-To-Send (RTS/CTS)-like mechanism. According to this protocol, a node can broadcast a packet only after having secured channel control. A different approach is adopted by another IEEE

802.11-based protocol, denoted as Smart Broadcast (SB) [9]. Similarly to UMB, SB partitions the transmission range of the source, associating non-overlapping contention windows to different regions. The Binary Partition Assisted Protocol (BPAB) [10] uses concepts from both UMB and SB, thus presenting similar performance, with an improvement, with respect to the SB protocol, in VANETs with low vehicle spatial density and irregular topologies. Finally, a different approach is considered when analyzing the class of *probabilistic* broadcast protocols, designed around the idea that each node forwards a received packet according to a characteristic Probability Assignment Function (PAF), computed by each node in a distributed manner [11, 12]. An entire class of probabilistic broadcast protocols is proposed and analyzed by Wisitpongphan et al. [13].

3.2.1 Reference Scenario

Figure 3.1 shows the linear network topology of reference for a generic multihop broadcast protocol: a static one-dimensional wireless network with a source and N (receiving) nodes. The assumption of static nodes is not restricting. In fact, from the perspective of a single transmitted packet, because of the very short transmission time (with typical IEEE 802.11 transmission rates), the network appears as static [14]. At the same time, a one-dimensional network is suitable for analyzing highway-like VANETs, where the width of the road (lying in the interval [10−40] m) is significantly smaller than the transmission range of an IEEE 802.11 network interface. These motivations are supported by simulation results, illustrated in Sect. 3.7.

We consider a deterministic free-space propagation model (i.e., without fading) and a fixed transmit power: therefore, each vehicle has a fixed transmission range, denoted as z (dimension: [m]). The network size (the line length) is set to L (dimension: [m]). For generality, we denote as *normalized network size* the positive real number $\ell_{\text{norm}} \triangleq L/z$. Generally, $\ell_{\text{norm}} > 1$ and this motivates the need for multihop communication protocols.

On the basis of empirical traffic data [15], the nodes' positions are generated according to a Poisson Point Process (PPP) of parameter ρ_s, where ρ_s is the vehicle

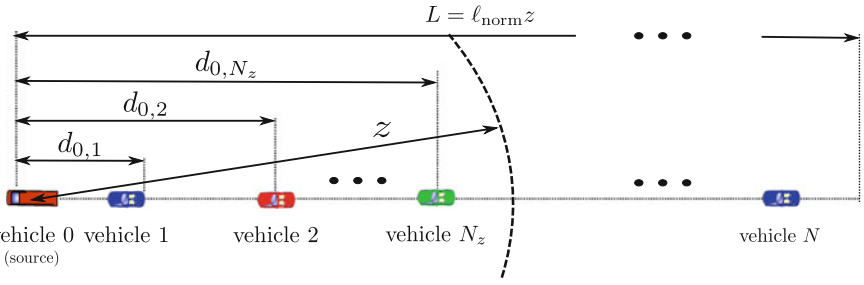


Fig. 3.1 A typical linear network topology of a VANET

(linear) spatial density (dimension: [veh/m])—the symbol "veh" it is not a realistic unit of measure, but it will be used for the sake of clarity. Consequently, N is a random variable characterized by a one-dimensional Poisson distribution with parameter $\rho_s L$. Similarly, the random variable N_z, denoting the number of nodes lying in the transmission range of the source (e.g., within the interval $(0, z)$), has a Poisson distribution with parameter $\rho_s z$. Thanks to the properties of the Poisson distribution, the inter-vehicle distance is exponentially distributed with parameter ρ_s and the (constant) average distance between two consecutive vehicles is $1/\rho_s$.

As shown in Fig. 3.1, the source node, denoted as node 0, is placed at the west end of the network, and we assume a single propagation direction (eastbound). Each of the remaining N nodes is uniquely identified by an index $i \in \{1, 2, \ldots, N\}$. The distance between the i-th and j-th nodes ($i, j \in \{1, 2, \ldots, N\}$, $i \neq j$) is denoted as $d_{i,j}$. Each vehicle can exactly estimate the value of $d_{i,j}$, thanks to the following assumptions: (i) the position of the source is a-priori known by every node; (ii) each vehicle knows its own position under the assumption of the presence (on board) of a Global Positioning System (GPS) receiver; (iii) each rebroadcaster inserts its own geographical coordinates within the packet. In the one-dimensional and with a single propagation direction scenario described in Fig. 3.1, the operational principle of a multihop broadcast protocol is quite simple. The initial transmission of a new packet from the source is denoted as the 0-th hop transmission, while the source itself identifies the so-called 0-th *Transmission Domain* (TD). After the source transmission, the packet is then received by the N_z source's neighbors, that are the potential rebroadcasters at the 1-st hop. Hence, their ensemble constitutes the 1-st TD. Each vehicle in the 1-st TD decides to forward the packet according to a PAF specified by the broadcast protocol. The use of *silencing* corresponds to the fact that the "fastest" retransmitter—among the set of those which have decided to retransmit—silences the others. Note that a collision may happen if at least two nodes of a TD retransmit simultaneously. The propagation process is therefore constituted by multiple packet retransmissions, that continue at most until the east end of the network—as will be clear in the following, with a *probabilistic* broadcasting protocol the retransmission process might terminate before reaching the end of the network.

3.2.2 Performance Metrics of Interest

In this chapter, the performance of probabilistic multihop broadcast protocols are investigated using the following average metrics: (i) the REachability (RE), (ii) the Transmission Efficiency (TE), and (iii) the end-to-end delay (D). The RE (adimensional), originally introduced by Ni et al. [2], is the fraction of nodes that receive the source packet among the set of all reachable nodes. The cardinality of the set of the reachable nodes is denoted as n_{reach}, and can be expressed as $n_{\text{reach}} = \min(N, n^*)$, where n^* is the minimum index such as the condition $d_{n^*,n^*+1} > z$ is verified. This definition is necessary since in PPP scenarios, as those considered in this Section, there can exist a pair of disconnected consecutive nodes $(n^*, n^* + 1)$. The TE (adi-

mensional) is defined as the ratio between the RE of a packet and the overall number of rebroadcast acts experienced during its transmission to the last reachable node. Finally, D (dim: [ms]) is defined as the duration of the packet trip between the source and the last reachable node. We remark that only the packets received correctly at the n_{reach}-th node of the network are considered for the evaluation of D. Therefore, this definition of D corresponds to a worst case scenario.

Owing to the symmetry of the forwarding process, the entire network can be modeled on the basis of the (local) analysis of a single TD. Therefore, in Sect. 3.3 we focus on a single TD—the reasons behind this assumption will be better clarified in Sect. 3.5.

3.3 Average Distribution of Poisson Points in a Segment with Finite Length

In one-dimensional networks, like those considered in this chapter, inter-node distance knowledge is necessary to implement the MCDS solution. For this reason, most of the proposed multihop broadcast protocols assume, at least to some extent, such a knowledge. Therefore, the first step to derive an analytical model consists in statistically characterizing the spatial distribution of the vehicles. In the literature, node positions are frequently modeled with a PPP. Despite its apparent simplicity, the derivation of an analytical performance evaluation framework based on the assumption of Poisson spatial distribution of the vehicles is not straightforward.

We now present a constructive definition of a PPP with parameter $\rho_s \in \mathbb{R}^+$, directly inspired from the one presented in Papoulis' book [16]. Given a finite interval $(-T/2, T/2) \subset \mathbb{R}$, place $n \in \mathbb{N}$ points in $(-T/2, T/2)$, under the constraint that $n/T = \rho_s$. A PPP is obtained by letting $n \to \infty$ and $T \to \infty$, under the constraint that n/T remains equal to ρ_s. A PPP has the following properties: (i) the distance between two consecutive points is a random variable with an exponential distribution with parameter ρ_s; (ii) given $z \in \mathbb{R}^+$, the number of points falling in the finite interval $\mathscr{I} \triangleq (0, z) \subset \mathbb{R}$ is a random variable with a Poisson distribution with parameter $\rho_s z$. In Fig. 3.2, an illustrative realization of a PPP with parameter ρ_s is shown. With reference to Fig. 3.2, denoting by n the number of Poisson points falling in $\mathscr{I}$ it is possible to define the n-dimensional positions vector

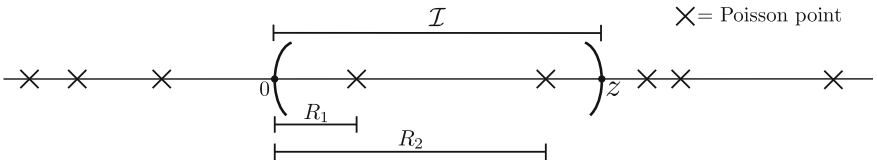


Fig. 3.2 Illustrative realization of a PPP

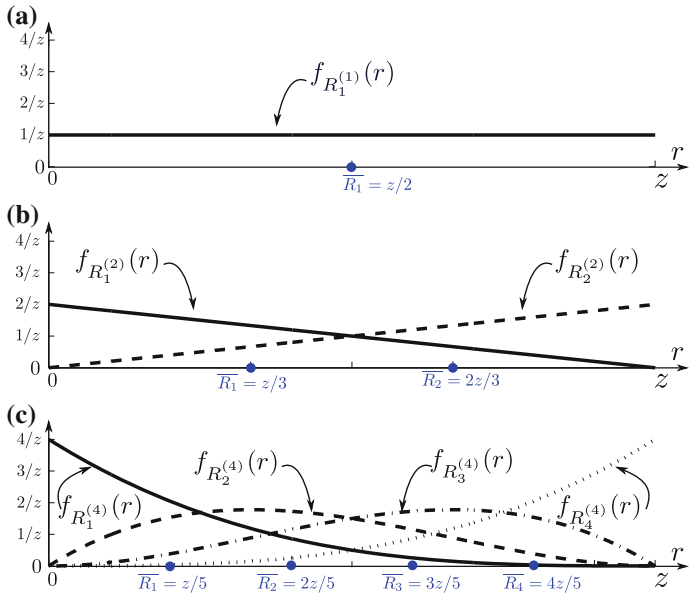


Fig. 3.3 $\{f_{R_i^{(n)}}(r)\}_{i=1}^{n}$ for different values of n: **a** $n = 1$, **b** $n = 2$, and **c** $n = 4$

$$\mathbf{R}^{(n)} = [R_1 R_2 \dots R_n] \tag{3.1}$$

where R_i ($i \in \{1, 2, \dots, n\}$) is the distance of the i-th point from the source (placed in zero)—in the illustrative case in Fig. 3.2, $n = 2$.

In Appendix B.1, it is shown that the marginal Probability Density Function (PDF) of R_j is:

$$f_{R_j}^{(n)}(r) = \begin{cases} \frac{n!}{z^n} \frac{(z-r)^{n-j}\, r^{j-1}}{(n-j)!\,(j-1)!} & r \in (0, z) \\ 0 & \text{otherwise.} \end{cases} \qquad j = 1, \dots, n \tag{3.2}$$

In Fig. 3.3, the PDFs of the positions of consecutive nodes are shown for various values of n: (a) 1, (b) 2, and (c) 4. In Appendix B.1, it is also shown that the average position of the j-th node can be expressed as follows:

$$\overline{R}_j^{(n)} = \int_0^z r \frac{n!}{z^n} \frac{(z-r)^{n-j}\, r^{j-1}}{(n-j)!\,(j-1)!}\, \mathrm{d}r_j = j \frac{z}{n+1} \qquad j = 1, \dots, n. \tag{3.3}$$

From Eq. (3.3), it clearly emerges that, for a given number of nodes falling in a finite segment $\mathscr{I}$, their average positions are equally spaced. The average node positions, for various values of the number n of nodes in $\mathscr{I}$, are also shown in Fig. 3.3.

Thanks to these results, the average performance analysis of a broadcast protocol in a network with Poisson node distribution can be carried out by simply studying a deterministic scenario, where the nodes are placed in correspondence to the average positions of the corresponding Poisson-based scenario. Moreover, this average analysis applies to other vehicle spatial distributions (e.g., taking into account the constraint on the vehicle lengths) with equally spaced average positions.

3.4 A Quick Overview of the IEEE 802.11b Standard

3.4.1 The IEEE 802.11 Standard

In this chapter, we assume that the physical and the Medium Access Control (MAC) layers of every node adhere to the IEEE 802.11b standard [17]. The IEEE 802.11 standard has been introduced in Sect. 2.5.2, hence we limit our discussion to the aspects that are of interest for this section. In particular, the PHY layer aspects are discussed in Sect. 3.4.2, while the relevant MAC layer characteristics are analyzed in Sect. 3.4.3.

3.4.2 Physical Layer

The IEEE 802.11 standard defines several PHY layers differing in terms of modulation format and carrier frequencies [17]. Because of their obsolescence we ignore some of them, namely, the legacy Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), and InfraRed (IR) modulations, respectively, defined in Chaps. 14, 15, and 16 of the standard [17]. We also ignore the IEEE 802.11n amendment [18], which defines an high rate Multiple Input Multiple Output (MIMO) modulation format, because it has not reached yet a sufficient diffusion (especially in handheld devices), and because its MIMO capabilities are not yet supported by the IEEE 802.11p amendment. We therefore focus on the remaining physical layers, introduced in the amendments *a*, *b*, and *g*.

The IEEE 802.11b amendment (now in Chap. 18 of the IEEE 802.11 standard [17]) has introduced the so-called High Rate Direct Sequence Spread Spectrum (HR/DSSS) modulation, which combines the original DSSS modulation of the legacy standard with a 8-chip Complementary Code Keying (CCK) modulation, providing a maximum data rate of 11 Mbit/s. The IEEE 802.11b standard defines 14 overlapped channels of 22 MHz width centered in the nearby of 2.4 GHz frequency. Because of overlapping, there is a strong co-channel interference, and therefore the channels cannot be used all together. Thanks to its adaptive rate selection capabilities, an IEEE 802.11b network interface can select the desired data rate in the {1, 2, 5.5, 11} Mbit/s set. Obviously, a lower data rate leads to a higher receiver sensitivity, thus allowing

to operate in harder channel conditions, with a lower Signal to Noise Ratio (SNR). As a rule of thumb, downscaling the data rate from 11Mbit/s and 1Mbit/s allows to improve the sensitivity of approximately 8 dB.

On the other hand, the IEEE 802.11a amendment (now in Chap. 17 of the IEEE 802.11 standard [17]) is based on a more robust Orthogonal Frequency Division Multiplexing (OFDM) modulation, which offers a greater maximum data rate of 54 Mbit/s. Also in this case, the radio interface can adaptively select lower data rates, scaling down up to 6 Mbit/s. Differently from IEEE 802.11b, the IEEE 802.11a works in the [5.2, 5.8] GHz frequency band. The number of channels is not fixed, as it is possible to use channels of three different size, namely, 5, 10, and 20 MHz. Each channel is separated into 52 orthogonal sub-carriers. Depending on the modulation scheme each sub-carrier encodes a specific number of bits in each symbol; for example, using the relatively simple Binary Phase Shift Keying (BPSK) modulation scheme, each sub-carrier encodes 1 bit. The signals of all sub-carriers are transformed into the time domain as symbols of fixed length. Subsequent symbols are separated by a guard interval in order to avoid interferences between distinct symbols. The duration of the guard interval is function of the channel bandwidth. In particular, with a channel size equal to, respectively, 5, 10, and 20 MHz, the corresponding guard interval length is equal to 3.2, 1.6, and 0.8 μs.

Finally, the IEEE 802.11g release (now in Chapter 19 of the IEEE 802.11 standard [17]) defines the Extended Rate PHY (ERP), a collection of different PHYs that are partially retro-compatible with the pre-existent modulation formats (especially the HR/DSSS). However, only the ERP-OFDM mode is implemented by almost all the chipsets, while the other modulation are not very spread in the market [19]. The ERP-OFDM modulation format is basically a simple transposition of the IEEE 802.11a OFDM modulation in the 2.4 GHz band, with a few minor changes to provide backwards compatibility. It supports the channel bandwidth, data rates and guard intervals of the IEEE 802.11a modulation.

3.4.3 MAC Layer

The basic building block of an IEEE 802.11 network is the Basic Service Set (BSS), a group of STAtions (STAs) that can communicate with each other. IEEE 802.11 offers different opportunities to build a BSS. For instance, nodes can form an Independent BSS (IBSS) with no central coordination authority, or, as in environments with infrastructure—i.e., Access Point (AP)—be part of an infrastructure BSS which is identified by an individual identification number.

The IEEE 802.11 standard defines three types of frames, management, control, and data, that share a set of common characteristics. In particular, all the frames include a bit field for frame control, a duration field, several addresses, the frame body and a Frame Control Sequence (FCS) for error detection. Each subtype is derived and adapted from the generic format (i.e., specific fields and data elements are added or left out). IEEE 802.11 provides several approaches for medium access control: (i)

Point Coordination Function (PCF), which is only applicable if an AP is available; (ii) Distributed Coordination Function (DCF), which can be used also in fully distributed networks; (iii) Hybrid Coordination Function (HCF), defined in the IEEE 802.11e amendment [20]. Within the HCF, there are two channel access methods, similar to those defined in the legacy 802.11 MAC: HCF Controlled Channel Access (HCCA), and EDCA—already introduced in Sect. 2.13. In both EDCA and HCCA, every packet has to be assigned to a particular Access Class (AC). In turn, every AC establishes different channel access settings, allowing to assign different priority levels to the packets [21]. Since the PCF and the HCCA mechanism are not of interest in VANETs, in the rest of the section we focus on the DCF mechanism. The DCF defines two different channel access mechanisms, both based on a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) strategy, which differ for the number of employed control packets: the Basic Access (BA) and the Ready To Send / Clear To Send Access. Due to the broadcast nature of the communications, the contention channel is managed through the BA mechanism, whose operational principles are described in the following paragraph.

3.4.3.1 Basic Access

The functioning of the Basic Access is represented in Fig. 3.4. When a node has a frame ready to be transmitted, it checks if the channel remains idle for a period of time at least longer than a Distributed InterFrame Space (DIFS): if this is the case, the node is free to immediately transmit. On the opposite, if the wireless medium is busy, the node defers its transmission until the medium remains idle for a whole DIFS without interruption. In the latter case, once the DIFS has elapsed, the node generates a random backoff period, which corresponds to an additional waiting time before transmitting (pre-backoff). The node transmits when the backoff time has elapsed. At each transmission act, the backoff time is uniformly chosen in the range $[0, cw-1]$, where cw is the current backoff window size, that is constant and equal to the minimum value defined by the standard, denoted as $CW_{\min}$, and corresponding

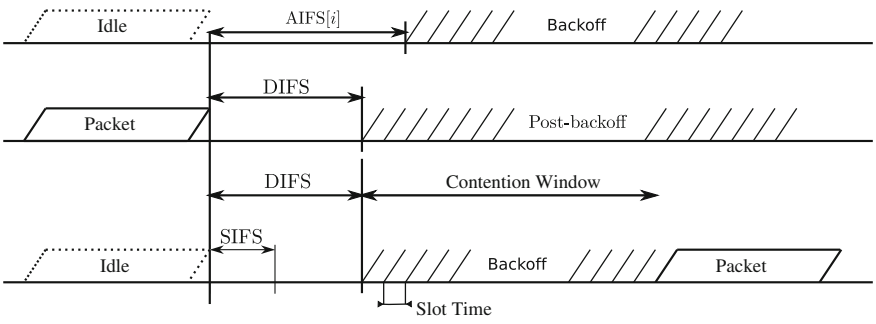


Fig. 3.4 Distributed Coordination Function mechanism defined by the IEEE 802.11 standard

to 32. The backoff period is slotted and the duration of the backoff, expressed in terms of number of backoff slots, is denoted as Backoff Counter (BC). This number is decremented as long as the medium is sensed idle, and it is frozen when a transmission is detected on the channel (this is an instance of a collision avoidance mechanism). Decrementing restarts when the medium is sensed idle again for more than a DIFS. At the end of every packet transmission, the node is forced to enter a post-backoff phase, which coincides with the subsequent pre-backoff, if the node has another packet in the transmission queue. It is important to observe that when a relay finds the channel idle, it *can* immediately transmit, but this is not mandatory. In order to reduce the number of collisions within a TD, we have interpreted the standard in a non-persistent manner, imposing that every relay enters into the pre-backoff phase, regardless of the channel status. We also remark that the extension of our approach to scenarios with IEEE 802.11p [22] communications, as envisioned in VANETs, is straightforward. Our approach (based on the IEEE 802.11b standard) is meaningful under the assumption of smartphone-based vehicular communications [23, 24].

In case of unicast transmissions, after a period of time equal to a Short InterFrame Space (SIFS), the destination STA has to send an acknowledgment (ACK) in order to confirm the (hopefully), successful reception of the packet. The SIFS is shorter than the DIFS, thus giving a higher priority to the ACK transmission. After the reception of the ACK, the sender STA is forced to begin a backoff period, denoted as *post-backoff*. Otherwise, the sender would "capture" the channel precluding access to the other STAs. So, if another frame is ready for transmission before this post backoff period ends, the STA has to execute it until the end before transmitting the frame. If the original sender does not correctly receive the ACK, and if it has not yet exceed the maximum number of retry, it can re-attempt the packet transmission from the beginning, after having doubled its CW value. We note that the CW is initialized to CW_{min}, and it cannot exceed the value CW_{max}.

In case of a broadcast transmission the use of the ACK is forbidden, determining a slightly different Basic Access behavior.[1] Without ACKs, the sender cannot known the status of the packet reception at the destination and therefore there are never retransmissions (at least at MAC layer). This leads to several consequences:

- CW is never increased and it is always equal to CW_{min};
- the transmissions are intrinsically less reliable;
- the transmission overhead is smaller.

Because of the less reliable transmissions it is necessary to adopt suitable countermeasures at the upper layers (network and transport).

We finally observe that, when the last received packet by a certain node was corrupted,[2] the DIFS period shall be replaced by a longer Extended IFS (EIFS) period. If we define the duration of a SIFS, DIFS and EIFS, as respectively, T_{SIFS}, T_{DIFS}, and T_{EIFS} (dimension: [μs]), the following relations hold:

[1] The use of the ACK is forbidden since the ACKs sent by distinct destination nodes will inevitably collide at the original sender.

[2] A received packet is considered corrupted if the corresponding FCS field is wrong.

Table 3.1 Main parameters of IEEE 802.11a and IEEE 802.11b standards

Parameter	IEEE 802.11b	IEEE 802.11a (20 MHz)
Carrier frequency (GHz)	2.4	5.8
Bandwidth (MHz)	22	20
OFDM guard time (μs)	–	0.8
CW_{min}	31	15
CW_{max}	1,023	1,023
T_{SLOT} (μs)	20	9
T_{SIFS} (μs)	10	16
Data rates (Mbit/s)	1, 2, 5.5, 11	6, 9, 12, 18, 24, 36, 48, 54

$$T_{DIFS} = T_{SIFS} + 2T_{SLOT}$$
$$T_{EIFS} = T_{SIFS} + T_{DIFS} + T_{ACK},$$

where T_{ACK} (dimension: [μs]) is the time required to transmit an ACK frame.

3.4.4 Main IEEE 802.11 Parameters

Table 3.1 summarizes the main default parameters defined for the IEEE 802.11a and IEEE 802.11b standards [17].

3.5 Probabilistic Broadcast Protocols with Silencing

3.5.1 Preliminaries Considerations

The general goal of a multihop broadcast protocol is to attain the widest network coverage in the shortest possible time. This can be obtained by pursuing three intermediate goals: (i) minimizing the number of communication hops; (ii) minimizing the number of effective retransmissions in every hop; (iii) minimizing the latency associated with a single hop. The number of transmission hops can be minimized by designating, as relays, the nodes forming the MCDS. However, the number of retransmissions and the latency are directly affected by the protocol characteristics, and there is no general rule for minimizing them—this motivates the presence, in the literature, of a large number of heuristic broadcast protocols.

A *probabilistic* broadcast protocol tries to achieve the goals outlined in the previous paragraph in a probabilistic and completely distributed manner: (i) *probabilistic*, in the sense that every intermediate node decides to retransmit a packet according

to a certain PAF, computed on a per-packet manner—even if, in general, one could introduce a per-flow PAF, in this Section we focus on single packet transmissions; (ii) *distributed*, in the sense that every node autonomously makes a retransmission decision without any coordination with its neighbors.

In "classical" probabilistic broadcast protocols (without silencing), if no suitable counter-measures are adopted, it is possible that more than one node in a TD decides to rebroadcast the packet (even without collisions). This leads to inefficiencies—besides complicating the mathematical analysis. A more efficient probabilistic broadcast protocol, regardless of the expression of the PAF, is obtained in the presence of a single retransmitting node in every TD. This can be obtained by imposing that the reception of a packet sent by a node of a TD *silences* the preceding nodes of the same TD. As a consequence, the next TD starts from the node which follows the "silencer." Note that the last TD partially overlaps with the previous one if the "silencer" is not a member of the MCDS.

In this chapter, we consider two novel probabilistic broadcast protocols with silencing, whose operations can be described as follows, with respect to the first TD.

1. The source sends a new packet (directly mapped on an IEEE 802.11 frame).
2. The nodes within a distance z from the source receive the packet and form the 1-st TD. Their number is denoted as N_z.
3. Every node in the 1-st TD probabilistically decides, according to the given PAF and taking into account its distance from the source, to retransmit (or not) the packet.
4. The potential forwarders (i.e., the nodes of the 1-st TD which have decided to retransmit) compete for channel access, by using the BA mechanism of the IEEE 802.11b standard (described in Sect. 3.4), first entering in the pre-backoff phase and, then, generating a random waiting time (denoted as BC, in Sect. 3.4). For the purpose of analytical simplicity, we assume that the BCs of the losing contenders are set to ∞.
5. The BCs are continuously decreased by all nodes, until (in the case of a successful forwarding) only one of them reaches 0, say the k-th BC. During a transmission of a node the other BCs freeze. Should there be the BCs of at least two nodes which reach simultaneously zero, both nodes would transmit and, thus, collide. We assume that the packets involved in a collision are considered undetectable and ignored by the other nodes. The corresponding k-th node retransmits the packet.
6. The remaining $N_z - 1$ nodes decode the packets, reset their timers, and discard the potentially queued packet. The nodes (spatially) preceding the k-th node will refrain from retransmitting from then on.
7. The whole process (from step 1) is restarted at the 2-nd TD, for which the k-th node acts as the source. The 2-nd TD is composed by all nodes lying in the interval $(d_{0,k}, d_{0,k} + z) \subset \mathbb{R}$, and it can also include some former nodes of the 1-st TD (those following the k-th node).

The two novel probabilistic broadcast protocols, polynomial and SIF, are described in the following two subsections.

3.5.2 Polynomial Broadcast Protocol

This protocol is characterized by a polynomial PAF, with the following form:

$$p(d, z, g) \triangleq \left(\frac{d}{z}\right)^g \tag{3.4}$$

where d denotes the distance (dimension: [m]) between the node of interest and the previous relay (or source, in the case of the first TD); z is the already introduced transmission range; $g \in \mathbb{N}$ is the polynomial order. According to the assumptions in Sect. 3.2, both z and d are known, without the need of exchanging additional messages. In fact, z can be estimated by knowing the transmit power and the channel propagation model, while d can be estimated by simply inserting the position of the source vehicle in every transmitted packet (under the assumption of having an accurate GPS receiver).

The shape of p, as a function of d, is shown in Fig. 3.5, for different values of g. It can be observed that the function p is monotonic and concave for all values of g. For high values of g, it becomes quite "selective," since it is approximately zero everywhere, but in the proximity of z. Note that the case with $g = 0$ ($p = 1$, $\forall d$) corresponds to the flooding protocol, i.e., each node retransmits. In this case, the BC value is randomly selected in $\{0, 1, \ldots, cw - 1\}$ as mandated by the IEEE 802.11 standard (Sect. 3.4).

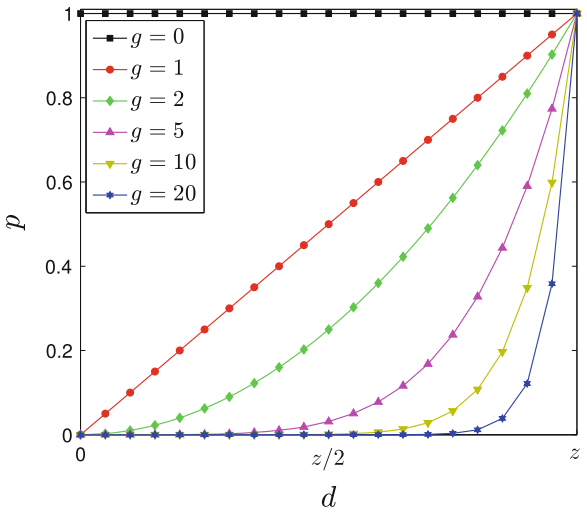


Fig. 3.5 Probability of retransmission (denoted as p) of the polynomial probabilistic protocol as a function of the distance d for several values of g

3.5.3 Silencing Irresponsible Forwarding

This broadcast protocol directly derives from the Irresponsible Forwarding (IF) protocol, originally presented in [25], with the introduction of the silencing mechanism outlined in Sect. 3.5.2. Besides this difference, IF and SIF share the same PAF, namely:

$$p(d, z, g) \triangleq \exp\left\{-\rho_s \frac{(d-z)}{c}\right\} \tag{3.5}$$

where c is an adimensional shaping coefficient and ρ_s is the vehicle spatial density. The latter can be estimated in a straightforward manner. In fact, under the assumption of knowing with a sufficient accuracy its transmission range, a node can estimate its local vehicular spatial density by simply counting the number of nodes lying within its transmission range and dividing them by the transmission range. The design of an efficient method for accurate estimation of the vehicular spatial density goes beyond the scope of this manuscript. However, intuitively it is sufficient to periodically send (and receive) Hello messages to the surrounding nodes. Alternatively, it is possible to rely on already existing beaconing mechanisms, such as the exchange of Cooperative Awareness Messages (CAMs) foreseen by the European Car-to-car consortium (broadcasted by default every 500 ms) [26].

Similarly to the PAF of the polynomial broadcast protocol, also the PAF of the SIP protocol "rewards" the farthest nodes (with respect to the transmitter). However, unlike the polynomial PAF, the SIF's PAF protocol also takes into accounts the (linear) vehicular spatial density, thus allowing to better adapt to different traffic conditions—this is the very idea of IF. The shape of p, as a function of d, is shown in Fig. 3.6, for different values of c and ρ_s. It can be observed that the SIF's PAF

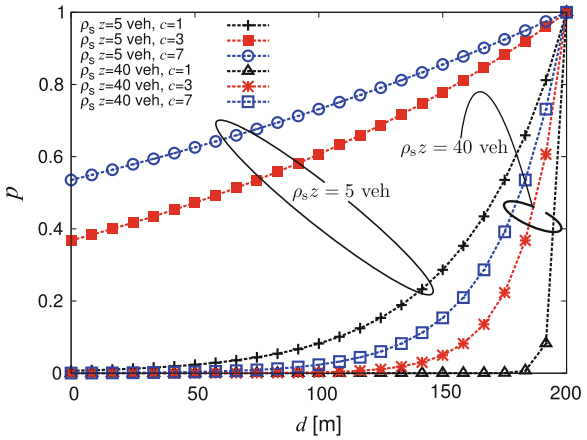


Fig. 3.6 Probability of retransmission (denoted as p) of the SIF protocol as a function of the distance d for several values of c and $\rho_s z$

is monotonically increasing and concave for all values of c. Moreover, it becomes selective far small values of c (e.g., 1), while it tends to flatten for high values of c and for low values of ρ_s. Also in this case, the BC value is randomly selected in $\{0, 1, \ldots, cw - 1\}$ as mandated by the IEEE 802.11 standard (Sect. 3.4).

3.6 A Recursive Analytical Performance Evaluation Framework

In Sect. 3.2, it has been stated that, since all TDs are statistically identical, the global behavior of the network can be modeled by analyzing a single TD. By exploiting the properties of probabilistic broadcast protocols with silencing (described in Sect. 3.5), the following assumptions hold: (i) the inter-node distance is characterized by a (memoryless) exponential distribution, so that the topology of every TD is (statistically) identical; (ii) the PAF only depends on the distance and is, therefore, memoryless; (iii) the IEEE 802.11b contention mechanism is memoryless, in the sense that it is restarted at every retransmission. Under these assumptions, every retransmission act can be interpreted as a renewal that resets the statistics of the forwarding process. Moreover, since all TDs are statistically identical, without loss of generality we can focus on the first TD.

Therefore, a complete analytical performance evaluation framework can be derived in the following manner: (i) characterizing the first TD with *local* performance metrics (e.g., the successful transmission probability and the delay); (ii) deriving *global* performance metrics (e.g., D, RE, TE), by means of a recursive approach.

In Sect. 3.6.1, the local performance (i.e., single TD) is investigated under the assumption of a given number of equally spaced nodes, by considering, without loss of generality, the first TD. In Sect. 3.6.2, we derive the global metrics for an overall deterministic network scenario, where the nodes are equally spaced in the interval $(0, L)$. Then, in Sect. 3.6.3 the results obtained in the deterministic scenario are extended to the original PPP-based scenario.

3.6.1 Local (Single Transmission Domain) Performance Analysis with a Given Number of Nodes

Without loss of generality, we focus on the first TD, corresponding to the interval $\mathscr{I}$ introduced in Sect. 3.3. We consider a deterministic scenario with a fixed number n of nodes equally spaced in the interval $\mathscr{I} = (0, z) \subset \mathbb{R}$. Every node in a TD is identified by an index $i \in \{1, 2, \ldots, n\}$. The nodes are thus positioned as in Fig. 3.3 and the positions vector $\mathbf{R}^{(n)}$ is defined as in (3.1).

According to the operational principles of the considered protocol, after the reception of a packet in a given TD, each node decides to (or not to) retransmit according to the protocol's PAF. The nodes that lose the contention set their BCs to ∞, while

the winners set their BCs according to the policy of the specific broadcast protocol. The protocol execution could lead to three different outcomes: (i) nobody decides to retransmit; (ii) some nodes decide to retransmit, but all their transmitted packets collide; (iii) some nodes decide to retransmit, and a single node transmits successfully (when its BC because zero, no other BC is zero). It is useful to define the following events, associated to the forwarding process in a TD:

$$
\begin{aligned}
\mathscr{F}_1 &\triangleq \{\text{nobody decides to retransmit}\} \\
&= \{\mathrm{BC}_i = \infty, \forall i \in \{0, 1, \ldots, n\}\} \\
\mathscr{F}_2 &\triangleq \{\text{all the transmitted packets collide}\} \\
&= \{\forall i \in \{0, 1, \ldots, n\} : \mathrm{BC}_i < \infty,\ \exists j \in \{0, 1, \ldots, n\},\ j \neq i, \mathrm{BC}_j < \infty \\
&\quad \text{such as } \mathrm{BC}_i = \mathrm{BC}_j\} \\
\mathscr{F} &\triangleq \{\text{nobody wins the contention}\} = \mathscr{F}_1 \cup \mathscr{F}_2 \\
\mathscr{S}_i &\triangleq \{\text{the node } i \text{ successfully retransmits}\} \qquad i \in \{1, \ldots, n\} \\
&= \{\mathrm{BC}_i < \infty, \mathrm{BC}_i = \min(\{\mathrm{BC}_m\}_{m=1}^{n}) \\
&\quad \cup \{\text{if } \exists j \in \{1, \ldots, n\}, i \neq j : \mathrm{BC}_j < \mathrm{BC}_i, \text{ then } \exists m \in \{1, \ldots, n\}, \\
&\quad m \neq j, m \neq i : \mathrm{BC}_j = \mathrm{BC}_m\} \qquad i \in \{1, \ldots, n\} \\
\mathscr{S} &\triangleq \{\text{a node successfully retransmits}\} = \bigcup_{i=1}^{n} \mathscr{S}_i.
\end{aligned}
$$

The probabilities of the above defined events are the following:

$$
\begin{aligned}
p_{\mathrm{rtx}}^{(n)}(i) &\triangleq \mathrm{P}\{\mathscr{S}_i\} \qquad i = 1, 2, \ldots, n \\
p_{\mathrm{succ}}^{(n)} &\triangleq \mathrm{P}\{\mathscr{S}\} = \sum_{i=1}^{n} p_{\mathrm{rtx}}^{(n)}(i) \\
p_{\mathrm{fail}}^{(n)} &\triangleq 1 - \mathrm{P}\{\mathscr{S}\} = 1 - \sum_{i=1}^{n} p_{\mathrm{rtx}}^{(n)}(i).
\end{aligned}
$$

Let us now introduce the random variable $Y \in \{0, 1, 2, \ldots, n\}$ with the following PMF:

$$
P_Y(y) = P\{Y = y\} = \begin{cases} p_{\mathrm{fail}}^{(n)} & y = 0 \\ p_{\mathrm{rtx}}^{(n)}(y) & y \in \{1, 2, \ldots, n\}. \end{cases}
$$

Since the event $\{Y = 0\}$ identifies the failure event, the random variable Y indicates either which node has effectively retransmitted or a failure. Moreover, it can be observed that:

$$
\bigcup_{y=1}^{n} \{Y = y\} = \mathscr{F} \cup \mathscr{S}.
$$

Obviously,

$$P_Y(y|\mathscr{S}) = P_Y(Y = y|\mathscr{S}) = \begin{cases} 0 & y = 0 \\ \frac{p_{\text{rtx}}^{(n)}(x)}{\sum_{i=1}^{n} p_{\text{rtx}}^{(n)}(i)} & y \in \{1, 2, \ldots, n\}. \end{cases}$$

In other words, if there is a retransmission ($\mathscr{S}$), then $P_Y(y|\mathscr{S})$ ($y \in \{0, 1, 2, \ldots, n\}$) is the probability that the y-th node has retransmitted.

As shown in Appendix B.2, the transmission probabilities $\{p_{\text{rtx}}^{(n)}(i)\}$ can be expressed as follows:

$$p_{\text{rtx}}^{(n)}(i) = p_i \sum_{m=1}^{n} q^{(m)} p_{V_i^{(n)}}(m-1) \tag{3.6}$$

where p_i denotes the value of the PAF (3.4) for the i-th node and depends on the considered protocol; $q^{(m)}$ is the probability that the i-th node wins the contention among a set of m competing nodes (the same for a given value of n); $V_i^{(n)} \in \{0, \ldots, n-1\}$ is the following discrete random variable:

$$V_i^{(n)} \triangleq \{\text{number of nodes, among the } n \text{ nodes, competing with the } i\text{-th node}\}.$$

The derivation of $q^{(m)}$ and of the PMF of $V_i^{(n)}$ can also be found in Appendix B.2.

After deriving $p_{\text{rtx}}^{(n)}(i)$, it is possible to compute the per-hop delay, denoted as D_i, of a retransmission from the i-th node. Since the per-hop delay is meaningful only if the i-th node decides to retransmit, it is of interest to study the statistical distribution of D_i conditioned on S_i. For this reason, we introduce the random variable $D_{i|i}$, which can be defined as follows:

$$D_{i|i} \triangleq T_{\text{slot}}(DIFS + N_{i|i}^{\text{bo}}) + T^{\text{tx}} \qquad i = 1, \ldots, n$$

where T^{tx} (dimension: [s]) is the transmission time; T_{slot} (dimension: [s/slot]) is the deterministic duration of the backoff slot; $DIFS$ (dimension: [slot]) is the duration of the DIFS; and $N_{i|i}^{\text{bo}}$ (dimension: [slots]) is the number of slots spent by the i-th node during the *backoff* (conditionally on the event S_i). We assume that both the packet size, denoted as P (dimension: [bits]), and the transmission rate, denoted as R (dimension: [bits/s]), are constant, thus leading to a deterministic packet transmission time $T^{\text{tx}} = P/R$. Taking into account that $DIFS$, T_{slot}, and T^{tx} are deterministic, the average value of $D_{i|i}$ becomes:

$$\overline{D}_{i|i} = T_{\text{slot}}(DIFS + \overline{N}_{i|i}^{\text{bo}}) + T^{\text{tx}} \qquad i = 1, \ldots, n \tag{3.7}$$

where, according to the derivation in Appendix B.3,

$$\overline{N}_{i|i}^{\text{bo}} = \frac{p_i}{cw\, p_{\text{rtx}}^{(N)}(i)} \sum_{v=0}^{N-1} p_{V_i}^{(N)}(v) \sum_{k=1}^{cw-1} \left[k \sum_{j=0}^{J_{k,v}} P'_v(k,j) + T^{\text{tx}} \sum_{j=1}^{J_{k,v}} j\, P'_v(k,j) \right] \tag{3.8}$$

where $J_{k,v} \triangleq \min(k, \lfloor (v/2) \rfloor)$ denotes the maximum number of collisions that can happen in slots $0, 1, \ldots, k-1$, while the matrix $\mathbf{P}'_v = \{P_v(k,j)\}$ is defined in Appendix B.3.

Proceeding in a similar manner, it is also possible to obtain the average number of retransmissions per-hop of the node i, denoted as $\overline{N}_{\text{rtx}}^{\text{hop}}(i)$:

$$\overline{N}_{\text{rtx}}^{\text{hop}}(i) = \frac{p_i}{cw\, p_{\text{rtx}}^{(N)}(i)} \left(1 + \sum_{v=0}^{N-1} p_{V_i^{(N)}}(v) \sum_{k=1}^{cw-1} \sum_{h=2}^{v} hN_{k,v}(0,h) \sum_{j=0}^{J_{k,v}} M_{k,v}(j,h) \right) \tag{3.9}$$

where the matrices $\mathbf{M}_{k,v} = M_{k,v}(j,h)$ and $\mathbf{N}_{k,v} = N_{k,v}(j,h)$ are defined in Appendix B.3.

3.6.2 Global Performance Analysis with Fixed Number of Nodes

Once the per-TD performance has been analyzed (as described in Sect. 3.6.1), the global performance metrics introduced in Sect. 3.2.2 (namely, RE, TE, and D) can be computed by following a recursive approach, based on the inductive principle. This recursive approach is extensively described, for the evaluation of D, in Appendix B.4, but can be directly re-adapted for the evaluation of RE and TE. In the remainder of this subsection, we outline the final results, trying to provide the reader with the intuition behind them.

Recall that we consider a deterministic scenario with a fixed number N of nodes equally spaced in the interval $(0, L) \subset \mathbb{R}$, where $L = z\ell_{\text{norm}}$. For simplicity, we assume that a generic TD contains $n = N/\ell_{\text{norm}}$ nodes. This corresponds to a best-case scenario, where the farthest node of each TD is the domain forwarder (the "silencer," as denoted in Sect. 3.5).

3.6.2.1 Delay

The computation of the average D is carried out by taking into account only the packets that successfully arrive at the end of the network (i.e., at the last reachable node) and ignoring the (remaining) packets that stop earlier. On the basis of the approach described in detail in Appendix B.4, the average end-to-end delay can be

given the following recursive formulation:

$$D \triangleq \overline{D}^{(N)} = \overline{T}_{\text{src}}^{\text{tx}} + \sum_{i=1}^{n} \left(\overline{D}^{(N-i)} + \overline{D}_{i|i} \right) p_Y(i|\mathscr{S}) \tag{3.10}$$

where $\overline{D}^{(N-i)}$ is the average delay in a network with $N - i$ nodes and $\overline{T}_{\text{src}}^{\text{tx}}$ is the average transmission time of the source, which differs from those of the following nodes, since the source does not contend with any other node and its transmission is not affected by collisions. Since the average time spent in the backoff is $(cw - 1)/2$, $\overline{T}_{\text{src}}^{\text{tx}}$ can be expressed as

$$\overline{T}_{\text{src}}^{\text{tx}} \triangleq T^{\text{tx}} + T_{\text{slot}} \left(DIFS + \frac{cw - 1}{2} \right). \tag{3.11}$$

3.6.2.2 RE

The average RE can be defined as follows:

$$\text{RE} \triangleq \frac{\overline{N}_{\text{reach}}}{N} \tag{3.12}$$

where N_{reach} is a random variable denoting the number of nodes reached by a packet. As a consequence of our assumptions, N_{reach} is lower bounded by n, since the transmission from the source reaches n nodes (those of the first TD) with probability 1. The average value $\overline{N}_{\text{reach}}$ can be obtained by following the approach described in Appendix B.4, but for the replacement of $p_Y(i|\mathscr{S})$ with $p_Y(i)$ and of $\overline{D}_{i|i}$ with the number of additional nodes covered by a new transmission. For example: a transmission from the 1-st node of the first TD will reach only one additional node (namely, the $(n + 1)$-th); a transmission from the 3-rd node will reach three additional nodes (namely, the $(n + 1)$-th, $(n + 2)$-th, and $(n + 3)$-th); and so on. The reader should note that, unlike the delay, in the computation of the RE we are not conditioning on the fact of reaching the N-th node of the network, i.e., the last reachable node of the network. Therefore, also the packets which stop being retransmitted are taken into account.

After the execution of the recursive approach outlined in Appendix B.4, it is sufficient to add a constant equal to n, corresponding to the number of nodes directly reached by the source at the first hop. The final expression of $\overline{N}_{\text{reach}}$ becomes (using the notation of Appendix B.4):

$$\overline{N}_{\text{reach}} = \overline{N}_{\text{reach}}^{(N)} = n + \sum_{i=1}^{n} \left(\overline{N}_{\text{reach}}^{(N-i)} + i \right) p_Y(i)$$
$$= n + \sum_{i=1}^{n} \left(\overline{N}_{\text{reach}}^{(N-i)} + i \right) p_{\text{rtx}}^{(n)}(i) \tag{3.13}$$

where $\overline{N}_{\text{reach}}^{(N-i)}$ corresponds to the average number of nodes reached in a network with $N - i$ nodes and can be recursively computed in the same way.

3.6.2.3 TE

In order to reduce the computational burden, we adopt the following approximated formulation of TE:

$$\text{TE} \triangleq \frac{\text{RE}}{N_{\text{rtx}}} \tag{3.14}$$

where $\overline{N}_{\text{rtx}}$ denotes the average overall number of retransmissions over all hops. From a computation viewpoint $\overline{N}_{\text{rtx}}$ is approximated by $\overline{N}_{\text{rtx}}^{m^{(*)}}$, where m^* corresponds to the average number of reached nodes—it is a sort of approximated indicator of the "depth" of the propagation process. Since the RE can be interpreted as the ratio between the average number of reached nodes and the total number (N) of nodes, m^* can be approximated as follows:

$$m^* \simeq N \cdot \text{RE}.$$

At this point, $\overline{N}_{\text{rtx}}^{(m^*)}$ can be computed by applying the recursive approach presented in Appendix B.4, by replacing (i) $p_Y(i|\mathscr{S})$ with $p_Y(i)$ and (ii) $\overline{D}_{i|i}$ with the average number of transmissions per hop, denoted by $\overline{N}_{\text{rtx}}^{\text{hop}}$ and given in (3.9).

3.6.3 Generalization to a PPP-Based Scenario

According to the original PPP-based model, described in Sect. 3.2, the number of nodes within $\mathscr{I}$, denoted as N_z, has the following Poisson distribution:

$$p_{N_z}(n, \rho_s z) = \frac{e^{-\rho_s z}(\rho_s z)^n}{n!} \qquad n \in \{0, 1, 2, \dots\}.$$

However, since a real vehicle has a finite length, it is not possible to have an infinite number of vehicles within $\mathscr{I}$. Therefore, it makes sense to impose an arbitrary limit to the maximum number of nodes within $\mathscr{I}$, denoted as N_c. The new truncated

Poisson random variable, denoted as N'_z, has the following distribution:

$$p_{N'_z}(n, \rho_s z) = \frac{\frac{e^{-\rho_s z}(\rho_s z)^n}{n!}}{\sum_{i=1}^{N_c} \frac{e^{-\rho_s z}(\rho_s z)^i}{i!}} \qquad n \in \{1, 2, \ldots, N_c\}$$

where we have also removed the event $n = 0$—this would correspond to an empty TD.

In order to exploit the results of Sect. 3.6.1, the stochastic network topology of the PPP needs to be mapped into a deterministic one with equally spaced nodes. In order to do this, the interval $\mathscr{I}$ is partitioned in N^{int} sub-intervals of length z/N^{int}, where $N^{\text{int}} \in \{N_c, N_c + 1, N_c + 2, \ldots\}$ is a design parameter. The computational burden and the accuracy are directly related to the value of N^{int}. After some numerical tests, we observed that the value $N^{\text{int}} = 100$ is a good tradeoff between precision and computational time. Thus, the i-th sub-interval is:

$$\mathscr{I}_i = \left[\frac{(i-i)z}{N^{\text{int}}}, \frac{iz}{N^{\text{int}}}\right] \qquad i = 1, 2, \ldots, N^{\text{int}}.$$

Every sub-interval can contain at most one node: in general, we assume that in each sub-interval there is a "virtual" node. Consequently, it is possible to associate a transmission probability $p^{\text{eq}}_{\text{rtx}}(i)$ to the generic sub-interval $\mathscr{I}_i$, defined as $p^{\text{eq}}_{\text{rtx}}(i)$, and a corresponding per-node delay, denoted as $D(i)^{\text{eq}}$ $(i = 1, \ldots, N^{\text{int}})$.

We define as $p^{(n)}_{\text{rtx}}(j)$ the probability of retransmission of the j-th node, given that there are exactly n nodes in the interval $\mathscr{I}$. Using the total probability theorem, $p^{\text{eq}}_{\text{rtx}}(i)$ can be expressed as follows:

$$\begin{aligned} p^{\text{eq}}_{\text{rtx}}(i) &= \sum_{n=1}^{N_c} \left(p^{\text{eq}}_{\text{rtx}}(i) | N'_z = n\right) P(N'_z = n) \\ &= \sum_{n=1}^{N_c} \sum_{j=1}^{n} p^{(n)}_{\text{rtx}}(j)\ f(i, j, n)\ p_{N'_z}(n, \rho_s z) \qquad i \in \{1, \ldots, N^{\text{int}}\} \end{aligned} \tag{3.15}$$

where $f(i, j, n)$ is an indicator function defined as follows:

$$f(i, j, n) \triangleq \begin{cases} 1 & \overline{\mathrm{R}}^{(n)}_j \in \mathscr{I}_i \\ 0 & \overline{\mathrm{R}}^{(n)}_j \notin \mathscr{I}_i. \end{cases} \tag{3.16}$$

The probability $p^{\text{eq}}_{\text{rtx}}(i)$ is now a function of $p^{(n)}_{\text{rtx}}(i)$ $(n \in \{1, 2, \ldots, N_c\}, i \in \{1, 2, \ldots, n\})$, which can be computed with combinatorics, since it is associated with a deterministic scenario with n static nodes equally spaced in $[0, z]$.

At this point, by using (3.6) in Eq. (3.15), it is possible to obtain a closed-form expression for $p^{\text{eq}}_{\text{rtx}}(i)$. Leveraging on the knowledge of $p^{\text{eq}}_{\text{rtx}}(i)$, by using Eq. (3.15) into Eqs. (3.7) and (3.9), it is possible to obtain, respectively, $D(i)^{\text{eq}}$

$(i = 1, \ldots, N^{\text{int}})$ and $n_{\text{rtx}}^{\text{hop}^{\text{eq}}}$. Then, it is possible to use the framework presented in Sect. 3.6.2 to derive RE, TE, and D for a deterministic network composed by $N_{\text{c}}\ell_{\text{norm}}$ nodes, since N_{c} is the (imposed) number of nodes in the interval $\mathscr{I}$ (and, thus, in each TD).

As anticipated at the end of Sect. 3.1, we remark that the presented analytical framework can be employed to study other types of broadcast protocols, not necessarily probabilistic, by simply re-adapting the definition of $p_{\text{rtx}}^{(n)}(i)$ and $D_{i|i}$. This is the subject of our current research activities.

3.7 Performance Analysis in Realistic Scenarios

3.7.1 Polynomial Protocol

In this section, we compare the results obtained with the analytical framework presented in Sect. 3.6 with those obtained through numerical simulations carried out with the ns-2 simulator [1]. In particular, the polynomial protocol has been "inserted" on top of the IEEE 802.11b model, after fixing the bugs reported by Chen et al. [27]. We observe that, conditionally on the fact of suitably scaling the packet size and the packet generation rate, from the perspective of our framework the IEEE 802.11a/p standards will offer the same performance of the IEEE 802.11b standard. All the results presented are accurate within ± 5 % of the values shown with 95 % confidence. The relevant parameters of the simulation are listed in Table 3.2. The results are obtained for a fixed node spatial density $\rho_{\text{s}} = 0.1$ veh/m, while the possible values of the transmission range z are listed in Table 3.2. In particular, the values of z are selected so that the corresponding values of $\rho_{\text{s}} z$ are between 10 veh and 40 veh. In the numerical simulations, we do not consider any case with $\rho_{\text{s}} z < 10$ veh, since this corresponds to topologies that are disconnected with a high probability, as shown in [11].

In Fig. 3.7, (a) D, (b) RE, and (c) TE are shown as functions of $\rho_s z$, for different values of g, by taking into account both the results of the analytical framework and numerical simulations, thus allowing to assess the validity of the analytical model.

Table 3.2 Main IEEE 802.11b network simulation parameters

ρ_{s}	0.1 veh/m
z	{100, 150, 200, 300, 400} m
ℓ_{norm}	8
Packet size	1,000 bytes
Carrier freq.	2.4 GHz
Data rate	1 Mbps
CW_{MIN}	31

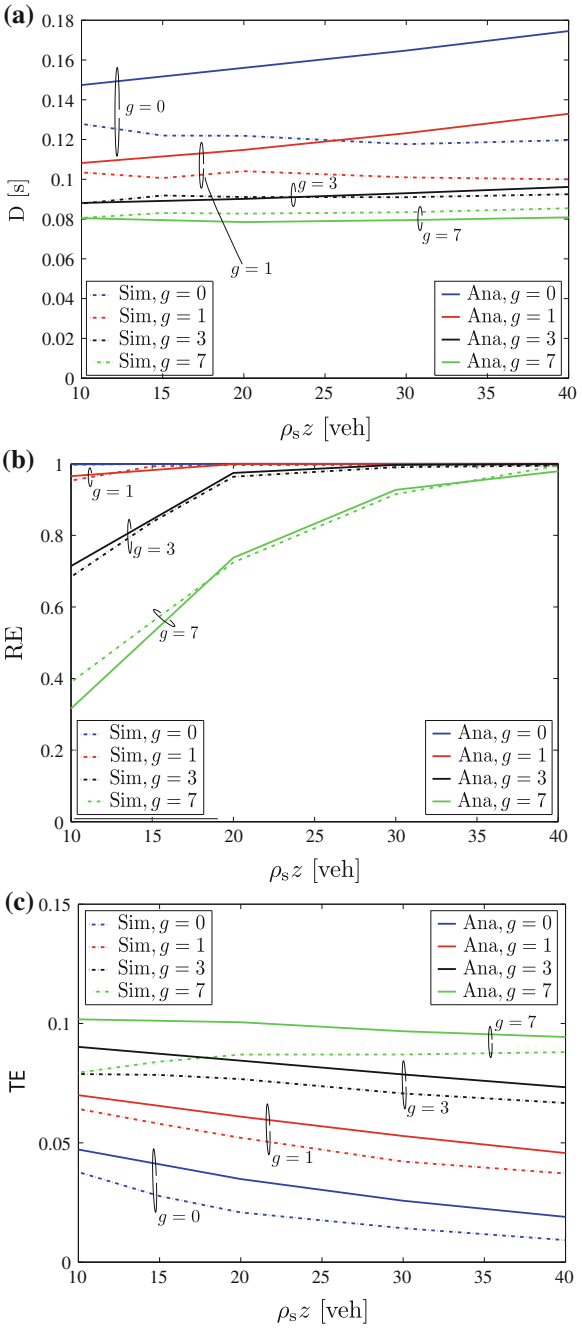


Fig. 3.7 D (**a**), RE (**b**), and TE (**c**), as a function of $\rho_s z$ obtained using the polynomial protocol and different values of g, by considering $cw = 31$, $l_{\text{norm}} = 8$, $P = 1{,}000$ bytes, and $R = 1$ Mbps. Both simulation (Sim) and analytical results (Ana) are shown

As shown by Busanelli et al. [11], using the considered values of $\rho_s z$ (between 10 and 40 veh), the network is fully connected (i.e., $n_{\text{reach}} = N$) with a high probability. From Fig. 3.7b it emerges that, in terms of RE, there is an excellent match between the results of the theoretical framework and those of the simulator. As shown by Fig. 3.7c, the agreement between analysis and simulations is still good also in terms of TE. On the other hand, the delay predicted by the analytical framework overestimates the true delay for small values of g (e.g., $g = 0$), whereas it becomes very accurate for large values of g (e.g., $g = 7$). The comparative investigation of analytical and simulation results indicates the validity of the proposed framework (especially for large values of g).

According to the results in Figs. 3.7a, c, it emerges that a higher polynomial degree leads to a better performance, regardless of the value of $\rho_s z$, in terms of both D and TE. Conversely, since the PAF is highly selective for large values of g (as shown in Fig. 3.5), this leads to poor performance in terms of RE, as shown in Fig. 3.7b. By considering small values of g (e.g., $g = 0$ corresponds to flooding), one observes the opposite phenomenon: a drastic improvement in terms of RE, at the price of a slightly higher D and a smaller TE.

In order to better understand the impact of g and $\rho_s z$ on the protocol performance: in Fig. 3.8a, D is shown, parametrized with respect to g, as a function of RE for different values of $\rho_s z$; while in Fig. 3.8b D is shown, parametrized with respect to $\rho_s z$, as a function of RE for different values of g. From the results in Fig. 3.8a, it emerges that even little variations of g lead to radically different protocol behaviors. On the contrary, $\rho_s z$ has an impact on the performance only for small values of $\rho_s z$, while for increasing values of $\rho_s z$ (e.g., larger than 20 veh) its impact vanishes.

From the results in Figs. 3.7 and 3.8, it clearly emerges that there is no optimal value of g. However, the proposed framework allows to optimize a single performance metric, after having imposed some constraints on the other metrics, on the basis of proper quality of service criteria. A possible choice consists in ignoring TE and minimizing D under the constraint of attaining a target value of RE. Since D is a decreasing function of g, it is possible to define the following quasi-optimal g^*:

$$g^*(\rho_s z) = \{\max(g) | \text{RE}(\rho_s z) > 0.95\}\,.$$

Selecting $g = g^*$ allows to achieve the minimum delay under a constraint on the RE. The obtained g^* is shown, as a function of $\rho_s z$, in Fig. 3.9a, and the following considerations can be drawn: (i) g^* is an increasing monotonic function of $\rho_s z$; (ii) with the exception of the region in proximity to $\rho_s z = 0$, where g^* tends to 0, g^* has a quasi-linear dependence with respect to $\rho_s z$. It can be shown that if $g = g^*$, RE $\simeq 1$ for each value of $\rho_s z$. Note that the selection of g^* allows to maximize RE. However, as shown in Fig. 3.9, D is always higher than 0.08 s, a delay which is instead guaranteed by the use of $g = 7$, as shown in the same figure.

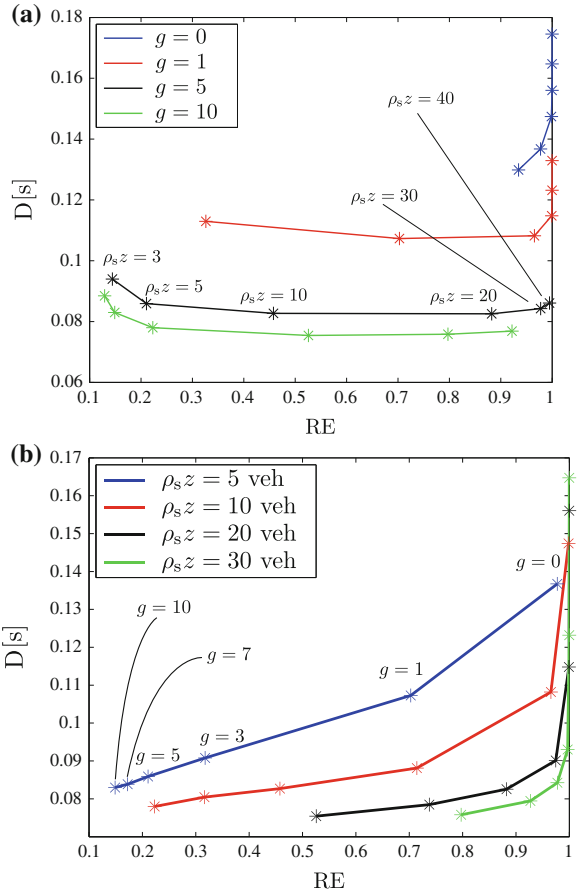


Fig. 3.8 D as a function of RE, parametrized with respect to $\rho_s z$ (for various values of g) (**a**) and g (for various values of $\rho_s z$) (**b**). The results are obtained by using the polynomial protocol and considering $cw = 31$, $l_{\text{norm}} = 8$, $P = 1{,}000$ bytes, and $R = 1$ Mbps

3.7.2 Silencing Irresponsible Forwarding

As pointed out in Sect. 3.6, the proposed framework can be applied to a large family of broadcast protocols. In this section, the framework is applied to SIF. In particular, the validity of the proposed analytical framework is clearly shown in Fig. 3.10, where (a) D, (b) RE, and (c) TE are shown, as functions of $\rho_s z$, for different values of c, by directly comparing both analytical and simulation results. As with the polynomial broadcast protocol, in this case as well there is a good agreement between the results obtained with the analytical model and the simulations. In particular, it can be observed that the accuracy of the model depends on the value of the shape parameter c (the highest average accuracy, over all metrics, is observed with $c = 7$).

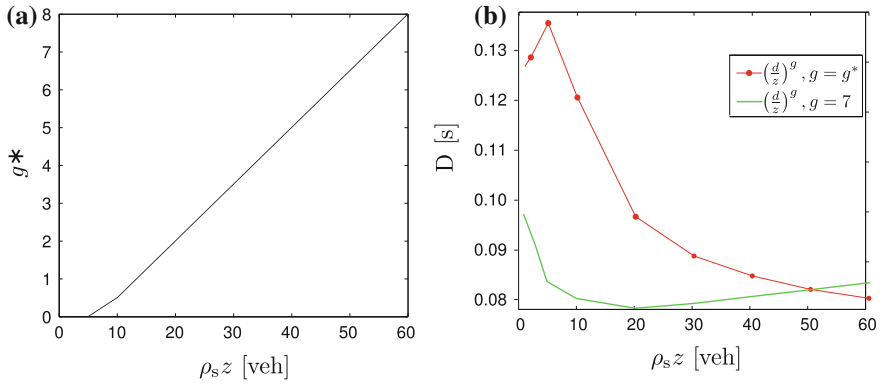


Fig. 3.9 **a** g^* and **b** D, as a function of $\rho_s z$

By comparing Figs. 3.8 and 3.10, one can observe that polynomial and SIF protocols have a different dependence on $\rho_s z$. In particular, in the case of SIF, as the product $\rho_s z$ increases RE remains roughly the same, while D decreases and TE increases. In other words, SIF performs better in dense networks. On the other hand, in the case of the polynomial protocol (Fig. 3.8), D and TE have an opposite behavior (namely, D slightly increases and TE slightly decreases for increasing values of $\rho_s z$), and RE strongly depends on $\rho_s z$, especially in sparse networks. In general, SIF outperforms the polynomial broadcast protocol.

Furthermore, from Fig. 3.10 it is clear that also for SIF there is no optimal value of the parameter c which simultaneously optimizes the performance according to all considered metrics. This fact can be better understood from Fig. 3.11, where D is shown as a function of RE, parametrized, respectively, with respect to (a) $\rho_s z$ and (b) c. In particular, from Fig. 3.11b it emerges that if one wants to guarantee a minimum value of RE (say 0.95), it is necessary to use a sufficiently high value of c. This, in turns, does not minimize D, which, as shown in Fig. 3.10a, is directly proportional to c. Moreover, the results in Fig. 3.11a strengthen the observations carried out regarding the results in Fig. 3.10. In fact, they clearly evidence two important characteristics of SIF: (i) RE is not affected by the value of $\rho_s z$, as SIF automatically adapts; (ii) counterintuitively, D is a decreasing function of $\rho_s z$ (i.e., SIF performs better in dense networks).

3.7.3 Comparison with Benchmark Protocols

As aforementioned, the theoretical framework presented in this manuscript can be used for evaluating a large number of broadcast protocols. In this subsection, it is applied to two benchmark broadcast protocols: (i) the flooding protocol (denoted with "FLOOD"), where each node forwards a received message; (ii) the optimal MCDS-

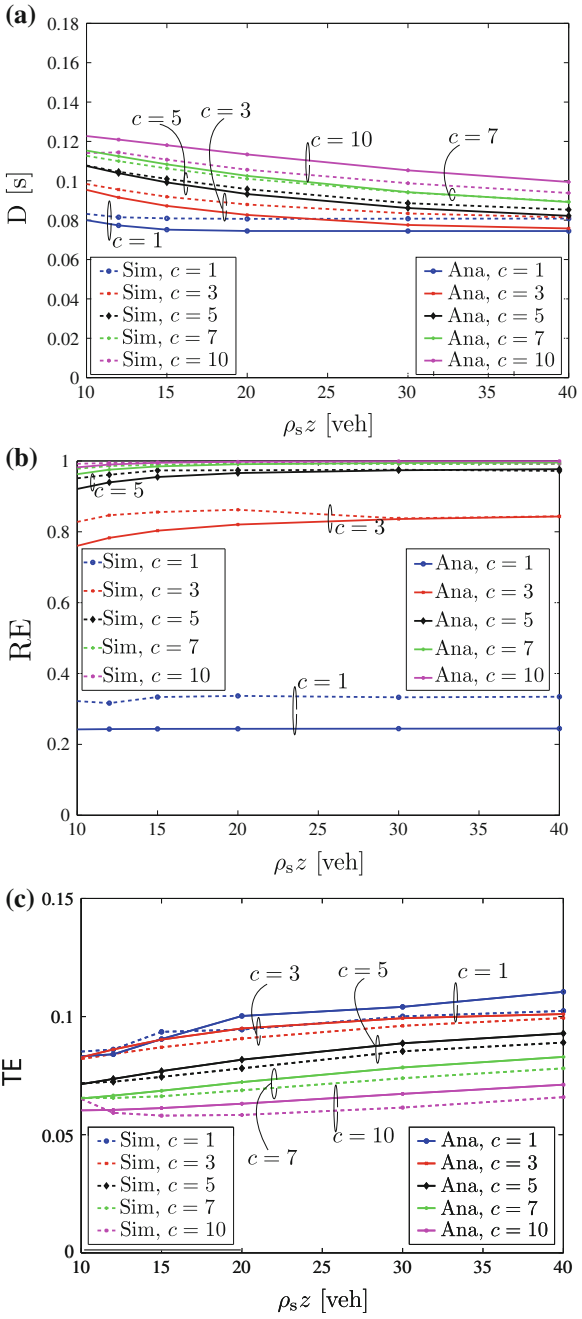


Fig. 3.10 D (**a**), RE (**b**), and TE (**c**), as a function of $\rho_s z$ obtained using the SIF protocol and different values of c, by considering $cw = 31$, $l_{\text{norm}} = 8$, $P = 1{,}000$ bytes, and $R = 1$ Mbps. Both simulation (Sim) and analytical results (Ana) are shown

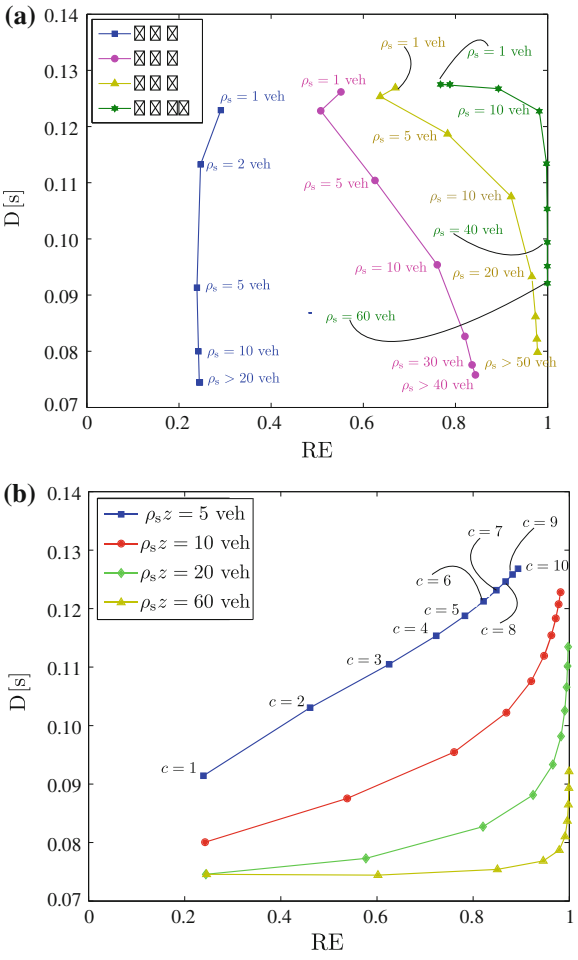


Fig. 3.11 D as a function of RE, parametrized with respect to $\rho_s z$ (for various values of c) (**a**) and c (for various values of $\rho_s z$) (**b**). The results are obtained by using the SIF protocol and considering $cw = 31$, $l_{\text{norm}} = 8$, $P = 1{,}000$ bytes, and $R = 1$ Mbps

based protocol (denoted with "MCDS"), where a hypothetical network genius selects as relays only the nodes belonging to the MCDS set (as described in Sect. 3.1). In both cases, the silencing mechanism is employed.

These benchmark protocols are compared with the SIF and polynomial protocols, considering a vehicle spatial distribution characterized by a Poisson distribution with parameter $\rho_s z = 16$ veh. In order to have a significant comparison, the optimal values of c and g ($c^* = 4.8$ and $g^* = 2.7$) are considered. These values, obtained through the analytical framework, allow to minimize D under the constraint of having a RE higher than 0.95, in a scenario with $\rho_s z = 16$ veh. The results, attained through both simulations and theoretical analysis, are shown in Fig. 3.12. From the results in Fig. 3.12, a few considerations can be drawn. First, for all considered metrics, there is a performance loss between the MCDS-based and the optimized SIF/polynomial

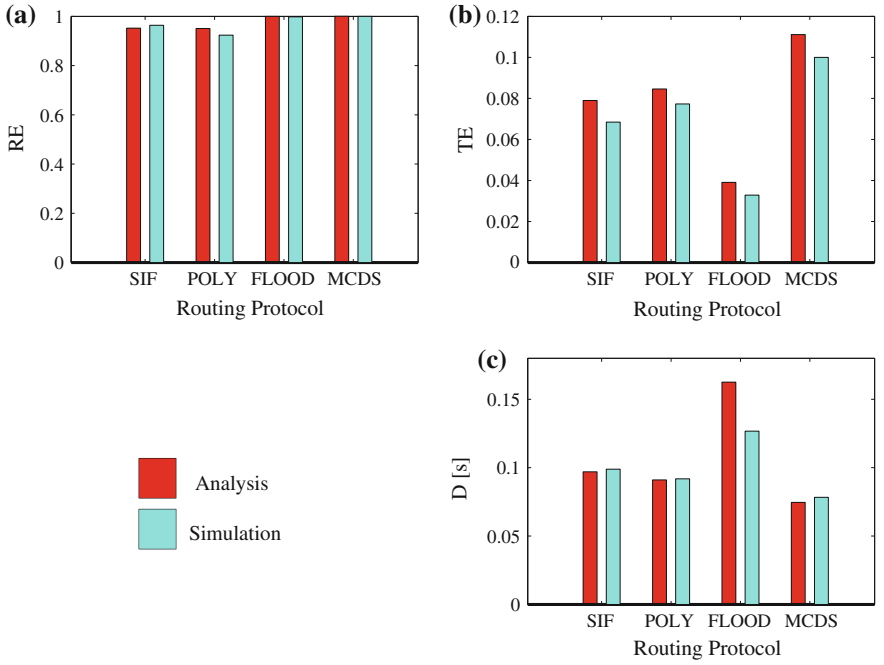


Fig. 3.12 **a** D, **b** RE, and **c** TE, obtained using the SIF, polynomial, flooding, and MCDS protocols, with $\rho_s z = 16$ veh, $c^* = 4.8$, and $g^* = 2.7$. The results are obtained through simulations by considering different topology, namely, a single-lane static network, a multi-lane static network, and a multi-lane mobile network (highway-style)

protocols. At the same time, the SIF/polynomial protocols exhibit a similar performance gain with respect to flooding (with the exception of the RE metric). It is also possible to observe that, counterintuitively, the SIF and the polynomial protocols offer a similar performance level. This result can be motivated by considering that their PAFs tend to converge to a common shape, when using, respectively, the optimal values g^* and c^* as their key parameters. Finally, an excellent match between simulation and theoretical results can be observed.

3.7.4 Highway-Style Scenarios

The goal of this subsection is to assess (a-posteriori) the validity of the assumption, made in Sect. 3.2, of considering a *uni-dimensional static* network. The validation is performed through simulations, by taking into account the protocols considered in Sect. 3.7.3 (namely, flooding, MCDS-based, SIF, and polynomial protocols). According to our assumption, we expect that the performances offered by these protocols will not be significantly affected by the network topology. To this end, we consider three

different scenarios: (i) the uni-dimensional (single-lane) static network presented in Sect. 3.2; (ii) a multi-lane static network; (iii) a multi-lane mobile network. The multi-lane static scenario is composed by $N_{\text{lane}} = 6$ adjacent lanes, each with width equal to $w_{\text{lane}} = 4$ m. Such a network is obtained by simply replicating the single-lane topology. In particular, in each lane the positions of the vehicles are generated according to a PPP of parameter $\rho_{\text{s}}/N_{\text{lane}}$. Similarly, the multi-lane mobile scenario is composed by $N_{\text{lane}} = 6$ adjacent lanes (3 per direction of movement), each with width equal to $w_{\text{lane}} = 4$ m. In this case, the vehicles are moving according to the Intelligent Driver Motion with Lane Changes (IDM-LC) mobility model [28] and, therefore, their positions do not have Poisson distribution. The mobility traces have been obtained using VanetMobiSim [29] and plugged in the ns-2 network simulator. The vehicles' speeds are independent and uniformly distributed in the interval $(20 - 40)$ m/s. Greater insights about the mobility models and the trace generation process are provided in [30]. It should be noticed that the value of the per-lane vehicular density (ρ_{s}) is time-averaged, since it is computed directly from the mobility trace and thus is time-varying. In Fig. 3.13, we show the results obtained by considering $\rho_{\text{s}} = 16$ veh and the optimal values of c and g ($c^* = 4.8$ and $g^* = 2.7$). It can be easily noticed that the performances obtained in the considered scenarios are quite similar. Hence, this proves (a-posteriori) that the assumptions made in Sect. 3.2 are

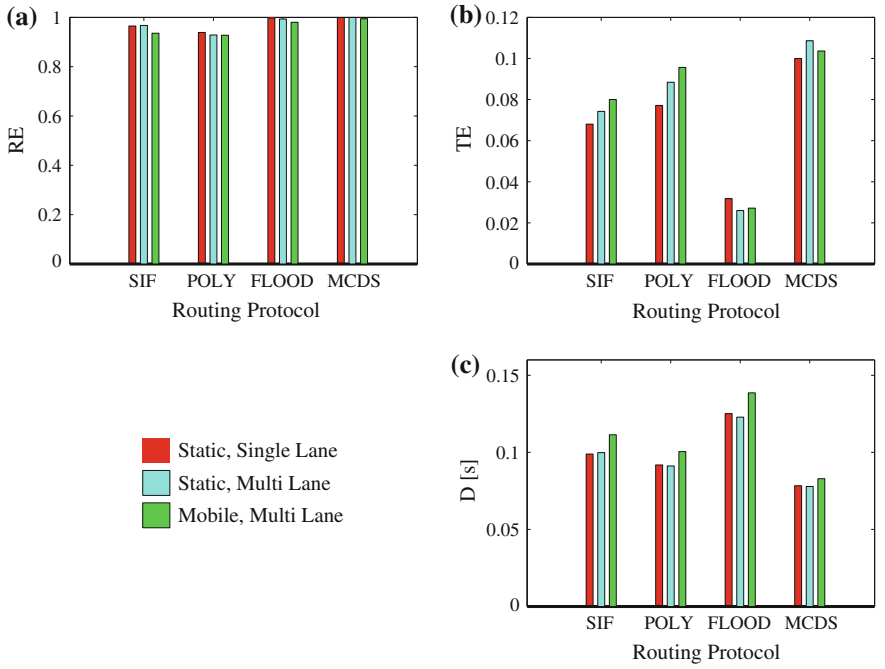


Fig. 3.13 **a** D, **b** RE, and **c** TE, obtained using the SIF, polynomial, flooding, and MCDS protocols, with $\rho_s z = 16$ veh, $c^* = 4.8$, and $g^* = 2.7$. Both simulation and analytical results are shown

substantially correct. More specifically, it can be observed that increasing the width of the network leads to very similar values of RE and D, and to slightly higher TE (this can be justified by considering that there is a higher number of nodes in the neighborhood of a vehicle). Instead, if we consider the same scenario but with mobile vehicles, one can observe that the RE becomes slightly lower, while D and TE become higher. This behavior is motivated by the tendency of mobile VANETs to form ephemeral clusters of vehicles [31], leading to a reduced RE and increased D but to a higher TE.

Finally, the limited impact of the vehicle mobility on the protocols' performance could have been expected by considering the values of the worst case transmission time (about 0.2 s) and of the the maximum allowed speed (roughly equal to 40 m/s, corresponding to 144 Km/h). In these conditions, two vehicles proceeding in opposite directions on a highway have a differential speed of 80 m/s, and this leads, in turn, to a distance variation of 16 m during a packet transmission time. A distance of 16 m (the worst-case variation) corresponds to a small fraction of the transmission range of a typical IEEE 802.11 network interface (in Fig. 3.13, we have considered $z = 160$ m).

3.8 VANETs as Distributed Wireless Sensor Networks

In this section, we consider a data collection application in a V2I scenario, where the vehicles act as a distributed wireless sensor network. In particular, we present a vehicular decentralized detection scheme, based on the observation, by all vehicles of a VANET, of a spatially constant phenomenon of interest (e.g., the average smog level or traffic situation on a given road). Our approach consists in the creation, during a downlink phase, of a clustered VANET topology during fast broadcast data dissemination, from the Access Point (AP), through a clustering protocol, denoted as Cluster-Head Election IF (CHE-IF). Such a clustered topology is then exploited, during an uplink phase, to collect information from the vehicles and perform distributed detection. Our results highlight the existing trade-off between decision delay and energy efficiency. Unlike classical sensor networks for distributed detection, the proposed vehicular distributed detection scheme exploit the natural vehicle clustering and have to cope with their "ephemeral" nature. More precisely, vehicle mobility has a direct impact on the maximum amount of data that can be collected, thus leading to the concept of decentralized detection on the move.

3.8.1 System Model

We consider a static one-dimensional wireless network with N (receiving) nodes, like the one presented in Sect. 3.2.1. The system model is the same used during the rest of the book and presented in Sect. 3.2.1. In particular, the reference scenario is represented by Fig. 3.1. All vehicles observe a spatially constant phenomenon, i.e.,

a phenomenon whose status does not change from vehicle to vehicle along the road. For example, vehicles could monitor if the average smog (or fog) level overcomes a critical threshold: the VANET would declare that it does if it happens for most of the road. The observed phenomenon can be generically defined as

$$H = \begin{cases} H_0 \text{ with probability } p_0 \\ H_1 \text{ with probability } 1 - p_0 \end{cases}$$

where $p_0 \triangleq \mathbb{P}\{H = H_0\}$, being $\mathbb{P}\{\mathscr{A}\}$ the probability that the event $\mathscr{A}$ happens. The value H_0 can be interpreted as the fact that the underlying physical phenomenon is, on average (along the road), below a given threshold, whereas the value H_1 can be interpreted as the fact that the underlying physical phenomenon is, on average (along the road), above a given threshold.

3.8.2 Clustered VANET Creation and IVCs

In this section, we derive the communication model for the vehicular distributed detection scenario. First, a downlink phase is envisioned, where the AP broadcasts a query to all vehicles in the network, in order to obtain information about the phenomenon of interest. During this phase, the CHE-IF protocol, besides guaranteeing fast information dissemination, automatically creates a clustered architecture, by opportunistically exploiting the ephemeral vehicular clusters. After a clustered network topology has been generated, during the uplink phase the decentralized detection task is performed by transferring the sensed data from the vehicles to the AP, through multi-hop communications and considering local fusion in each vehicular cluster.

3.8.2.1 Downlink

The philosophy of CIF protocol [31] is to establish a weak artificial packet flow, with the purpose to discover the presence of naturally formed clusters. Then, this information is exploited in order to optimize the forwarding procedure, increasing the reliability and the transmission efficiency, but without building up a true clustered infrastructure.

We propose a derivation of CIF, denoted as CHE-IF, that introduces some expedient mechanisms to make CIF a protocol capable to efficiently construct a stable clustered infrastructure. The new CHE-IF protocol is a totally decentralized protocol, since each node designates its own CH without pursuing a common global consensus. The purpose is to obtain an operative clustered topology in the shortest time, in order to start the data collection process as soon as possible. This behavior fits well with the intrinsic dynamic nature of a VANET, characterized by continuous topology changes that vanish the hypothetical advantages of a centralized clustering protocol. Moreover, a refinement of the cluster structure can be performed once the collection process is started, making small adjustment of the network topology.

The CHE-IF protocol is designated in order to choose a single CH among the retransmitting nodes of a transmission domain. This ideally yields to the creation of an unique set of connected CHs, able to cover the entire area of interest. After choosing the CHs, the cluster will naturally form. In fact, the nodes not designated as CHs become children of the nearest CH, leading to the formation of clusters of similar dimension.

The CHE-IF protocol defines 3 types of packets: (i) Cluster Initialization Packet (CIP); (ii) Probe Packet (PP); (iii) Cluster Confirmation Packet (CCP). The CHE-IF protocol is characterized by three phases. In the first one, through the exchange of some dedicated packets (CIPs and PPs), every node fills a temporary routing table containing the list of the potential CHs in its transmission range. During the second phase, that starts after a time $T_{\mathrm{w}}^{\mathrm{CIP}}$ (set proportionally to the length of the network), each node elects its CH based on the information contained in its routing table.[3] Due to the lack of global consensus, it is not guaranteed that node decisions match together. For instance, some nodes could designate a CH that does not believe to be a CH. For this reason, there is also a third phase, called confirmation phase, during which the AP sends a special packet (the CCP) that is retransmitted only by the CHs (with probability 1). Listening to the CCP, the network nodes can become aware of the identity of the true CHs.

While the second and the third phases are relatively simple, the first phase is more complicated, as it requires, at every hop, 4 steps that are graphically represented in Fig. 3.14 and described in the following.

The first phase consists of the transmission of the CIP by a node of the $(i-1)$-th transmission domain, which leads to the identification of the i-th transmission domain (the AP in the case of 1-st transmission domain). The CIP is sent with a transmit power $P_{\mathrm{t}}^{\mathrm{CIP}}$ and contains a unique identification (ID) and the source address of the AP.

The second step derives directly from the IF protocol and is a sort of "virtual contention." In particular, every node in the i-th transmission domain decides to become or not a potential forwarder by performing the same probabilistic election mechanism of the IF protocol. The winners of this contention will begin the third step, while the others will simply discard the packet.

The third step derives from the concept of "ephemeral cluster." Once a node wins the first virtual contention, it schedules the retransmission of a very short packet, denoted as Probe Packet (PP). A PP bears just two information items: (1) the unique identification (ID) of the CIP; (2) the distance from the node in the previous transmission domain from which it has received the packet. The PPs are intrinsically single hop, i.e., they are not forwarded. A PP is transmitted with a power defined as $P_{\mathrm{t}}^{\mathrm{PP}} = 0.25 P_{\mathrm{t}}^{\mathrm{CIP}}$, in order to reduce network congestion, since a node is interested only in signaling its presence to its neighbors, and with a high priority, in order to reduce the overall latency. Moreover, a low transmission power allows to reduce channel interference. The specific power and priority setting of a PP have to be tuned

[3] A given node elects itself as a CH for the i-th transmission domain if it is the farthest retransmitter of its transmission domain.

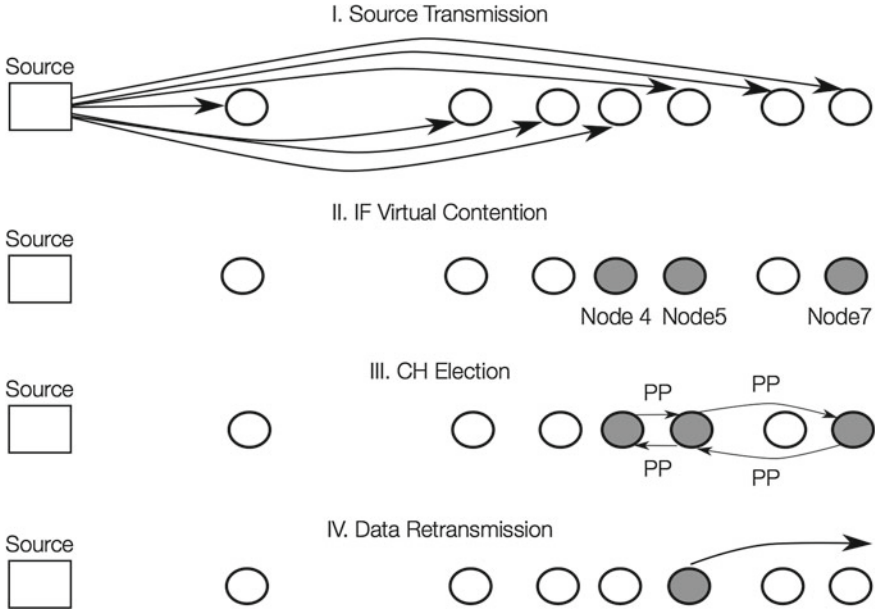


Fig. 3.14 CH election of the CHE-IF protocol

according to the used MAC protocol, as shown in [31]. After winning the virtual contention, every potential forwarder sends a PP. It then waits for a short interval, denoted as $T_{\mathrm{w}}^{\mathrm{PP}} \frac{d}{z}$, where $T_{\mathrm{w}}^{\mathrm{PP}}$ is a proper constant. If, within this interval, it receives at least a PP containing a value of distance larger than its own, it stops and discards the packet (in fact, there is an other better placed forwarder); conversely, it retransmits the CIP. In the worst case, when a collision between two or more PPs happens, this selection mechanism fails and no node of the cluster is elected. In this case, the retransmitter in the previous transmission domain will retransmit the CIP for restarting the CH designation procedure at the i-hop. This can happen until a maximum of 3 times, otherwise the whole designation procedure is considered failed.

The fourth step corresponds to the transmission of the CIP from the designated forwarding nodes at the i-th transmission domain.

3.8.2.2 Uplink

The uplink phase exploits the clustered structure created during the downlink phase. More precisely, during the uplink phase, the data acquired by the N vehicles of the VANET are transmitted to the final AP. Note that, unlike a regular sensor network, the created VANET can be used as long as its structure does not break, due to vehicle mobility. In other words, there is a maximum amount of data which can be collected [32].

The observed signal at the i-th vehicle can be expressed as

$$r_i = \begin{cases} 0 + w_i \text{ if } H = H_0 \\ s + w_i \text{ if } H = H_1 \end{cases} \qquad i = 1, \ldots, N \tag{3.17}$$

where $\{w_i\}$ are additive noise samples. Note that s is considered as a deterministic parameter. Assuming that the noise samples $\{w_i\}$ are independent random variables with the same Gaussian distribution $\mathcal{N}(0, \sigma^2)$, the common observation signal-to-noise ratio (SNR) at the vehicles, denoted as $\text{SNR}_{\text{vehicle}}$, can be defined as $\text{SNR}_{\text{vehicle}} \triangleq s^2/\sigma^2$ [33]. Each vehicle makes a decision comparing its observation r_i with a threshold value $\tau = s/2$ and computes a local decision $u_i = U(r_i - \tau)$, where $U(\cdot)$ is the unit step function. Note that a vehicle could transmit one single decision per packet or, by collecting consecutive phenomenon observations, it could transmit packets with more decisions. The strategy selection depends on the desired trade-off between data and overhead per transmitted packet. However, investigating this aspect goes beyond the scope of this Section.

Suppose that during the downlink phase the CHE-IF protocol has led to the creation of $n_c < N$ cluster. Each vehicle can communicate only with its local CH. Possible clustered topologies are represented in Figs. 3.15 and 3.16, according to the particular strategy for communications towards the AP.

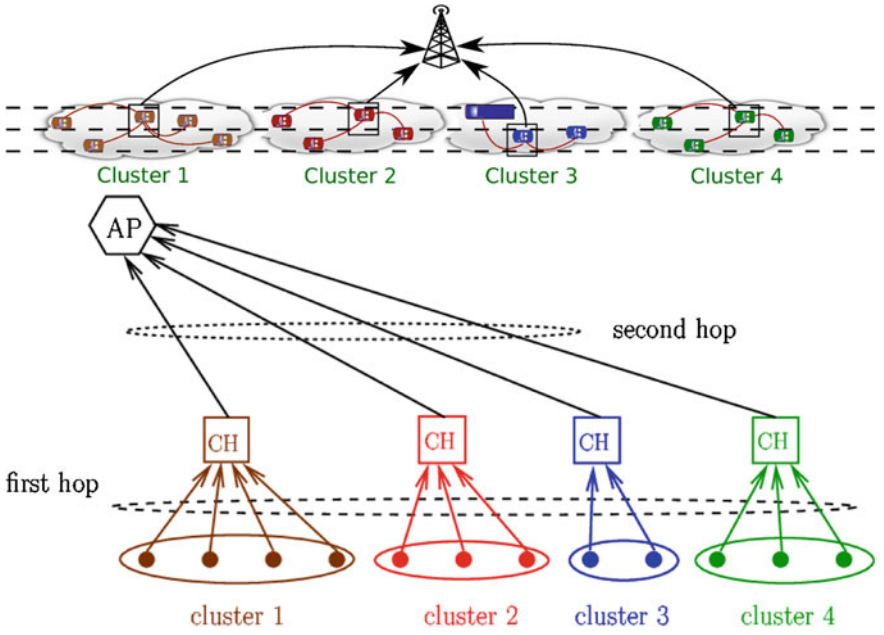


Fig. 3.15 Network topologies (*upper part*) and their logical representations (*lower part*): direct communications between CHs

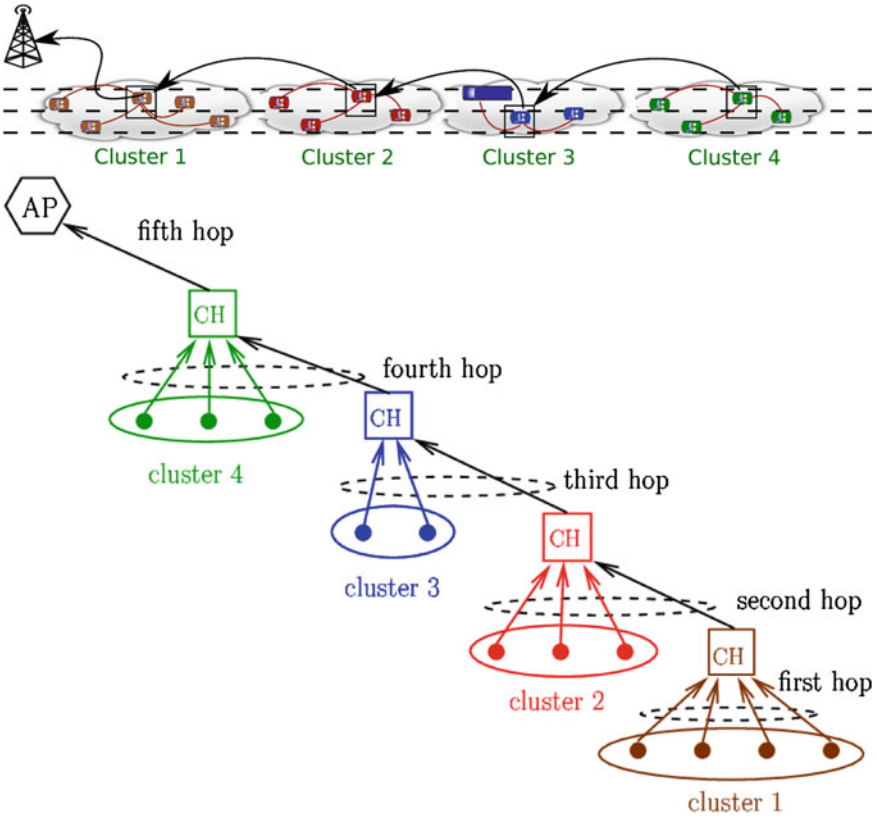


Fig. 3.16 Network topologies (*upper part*) and their logical representations (*lower part*): multi-hop communications between CHs and AP

In particular, when a sufficiently high transmit power is available, all CHs can communicate directly with the AP, as shown in Fig. 3.15. On the other hand, when the transmit power is not sufficiently high, multi-hop communications are required to transfer the information from the CHs towards the AP, as shown in Fig. 3.16.

The performance of the decentralized detection techniques presented here, has been analyzed and discussed in [32], considering realistic VANET clustered topologies.

References

1. Network Simulator 2 (ns-2): Available at: http://isi.edu/nsnam/ns/
2. Ni, S., Tseng, Y., Chen, Y., Sheu, J.: The broadcast storm problem in a mobile ad hoc network. In: Proceedings of ACM International Conference on Mobile Computer and Networking (MOBICOM), pp. 151–162. Seattle, WA, USA (1999)

3. Zanella, A., Pierobon, G., Merlin, S.: On the limiting performance of broadcast algorithms over unidimensional ad-hoc radio networks. In: Proceedings of IEEE International Conference on Wireless Personal Multimedia Communications (WMPC) pp. 165–169 (2004)
4. Torrent-Moreno, M., Mittag, J., Santi, P., Hartenstein, H.: Vehicle-to-vehicle communication: fair transmit power control for safety-critical information. IEEE Trans. Veh. Technol. **58**(7), 3684–3707 (2009)
5. Kihl, M., Sichitiu, M., Joshi, H.P.: Design and evaluation of two geocast protocols for vehicular ad-hoc networks. J. Internet Eng. Klidarithmos Press **2**(1), 127–135 (2008)
6. Korkmaz, G., Ekici, E., Özgüner, F., Özgüner, U.: Urban multi-hop broadcast protocol for inter-vehicle communication systems. In: Proceedings of ACM International Workshop on Vehicular Ad hoc Networks (VANET), pp. 76–85. ACM, Philadelphia, PA, USA (2004)
7. Korkmaz, G., Ekici, E., Ozguner, F.: Black-burst-based multihop broadcast protocols for vehicular networks. IEEE Trans. Veh. Technol. **56**(5), 3159–3167 (2007)
8. Sobrinho, J., Krishnakumar, A.: Quality-of-service in ad hoc carrier sense multiple access wireless networks. IEEE J. Select. Areas Commun. **17**(8), 1353–1368 (2002)
9. Fasolo, E., Zanella, A., Zorzi, M.: An effective broadcast scheme for alert message propagation in vehicular ad hoc networks. In: Proceedings of IEEE International Conference on Communication (ICC), vol. 9, pp. 3960–3965. Istanbul, Turkey (2006)
10. Sahoo, J., Wu, E., Sahu, P., Gerla, M.: BPAB: Binary partition assisted emergency broadcast protocol for vehicular ad hoc networks. In: Proceedings of International Conference on Computer Communications and Networks (ICCCN), pp. 1–6 (2009)
11. Busanelli, S., Ferrari, G., Panichpapiboon, S.: Efficient broadcasting in IEEE 802.11 networks through irresponsible forwarding. In: Proceedings of IEEE Global Telecommunication Conference (GLOBECOM). Honolulu, HA, USA (2009)
12. Hanashi, A.M., Siddique, A., Awan, I., Woodward, M.: Performance evaluation of dynamic probabilistic broadcasting for flooding in mobile ad hoc networks. Simul. Model. Pract. Theor. **17**(2), 364–375 (2009). Elsevier
13. Wisitpongphan, N., Tonguz, O., Parikh, J., Mudalige, P., Bai, F., Sadekar, V.: Broadcast storm mitigation techniques in vehicular ad hoc networks. IEEE Wirel. Commun. Mag. **14**(6), 84–94 (2007)
14. Busanelli, S., Ferrari, G., Giorgio, V.A.: On the Effects of Mobility for Efficient Broadcast Data Dissemination in I2V Networks. In: Proceedings of SWiM Workshop IEEE Global Telecommunication Conference (GLOBECOM). Miami, FL, USA (2010)
15. Wisitpongphan, N., Bai, F., Mudalige, P., Sadekar, V., Tonguz, O.K.: Routing in sparse vehicular ad hoc wireless networks. IEEE J. Select. Areas Commun. **25**(8), 1538–1556 (2007)
16. Papoulis, A.: Probability, Random Variables, and Stochastic Processes. Mcgraw-Hill College, New York (1991)
17. Insitute of Electrical and Electronics Engineers: IEEE Std 802.11TM-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications (2012)
18. IEEE Standard for Information technology-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 5: Enhancementss for Higher Throughput. IEEE Std 802.11n-2009, pp. 1–211 (2005)
19. Gast, M.: 802.11 Wireless Networks: The Definitive Guide, Second edn. O'Reilly Media, Sebastopol (2005)
20. IEEE Standard for Information technology-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements. IEEE Std 802.11e-2005, pp. 1–211 (2005)
21. Hartenstein, H., Laberteaux, K.: VANET Vehicular Applications and Inter-Networking Technologies. Wiley Press, Hoboken (2010)
22. IEEE: IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007), pp. 1–51 (2010)

23. Mohan, P., Padmanabhan, V., Ramjee, R.: Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In: 4th ACM Conference on Embedded Networked Sensor Systems (SenSys). Raleigh, North Carolina, USA (2008)
24. Ferrari, G., Busanelli, S., Iotti, N., Kaplan, Y.: Cross-network information dissemination in VANETs. In: Proceedings of IEEE International Conference on ITS Telecommunications (ITST). Saint-Petersburg, Russia (2011)
25. Panichpapiboon, S., Ferrari, G.: Irresponsbile forwarding. In: Proceedings of IEEE International Conference on Intelligent Transport System Telecommunication (ITST), pp. 311–316. Phuket, Thailand (2008)
26. Kosch, T., Kulp, I., Bechler, M., Strassberger, M., Weyl, B., Lasowski, R.: Communication architecture for cooperative systems in europe. IEEE Commun. Mag. **47**(5), 116–125 (2009)
27. Chen, Q., Schmidt-Eisenlohr, F., Jiang, D., Torrent-Moreno, M., Delgrossi, L., Hartenstein, H.: Overhaul of IEEE 802.11 modeling and simulation in ns-2. In: Proceedings of the ACM Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems (MSWiM), pp. 159–168. Chania, Crete Island, Greece (2007)
28. Fiore, M., Härri, J., Filali, F., Bonnet, C.: Vehicular mobility simulation for VANETs. In: Proceedings of SCS Annual Simulation Symposium (ANSS), pp. 301–309. Norfolk, VA, USA (2007)
29. VanetMobiSim Project: http://vanet.eurecom.fr/
30. Busanelli, S., Ferrari, G., Giorgio, V.A.: I2V highway and urban vehicular networks: a comparative analysis of the impact of mobility on broadcast data dissemination. J. Comm. SI Seamless Mobil. Wirel. Netw. **6**(1), 87–100 (2011)
31. Busanelli, S., Ferrari, G., Panichpapiboon, S.: Cluster-based irresponsible forwarding. In: Giusto, G.M.D., Iera, A., Atzori, L. (eds.) The Internet of Things, 20th Tyrrhenian International Workshop on Digital Communications. Springer, Berlin (2009)
32. Busanelli, S., Martalò, M., Ferrari, G.: Clustered vehicular networks: Decentralized detection "on the move". In: Proceedings of IEEE International Conference on ITS Telecommunications (ITST), pp. 744–749. Saint-Petersburg, Russia (2011)
33. Ferrari, G., Martalò, M., Pagliari, R.: Decentralized detection in clustered sensor networks. In: IEEE Transactions on Aerospace Electronic Systems (2009). To appear, 2010. Available: http://www.tlc.unipr.it/martalo/privfold/preprint.pdf

Chapter 4
Hierarchical Architecture for Cross Layer ITS Communications

4.1 The Big Picture

ITS are colonizing vehicles' cockpits in the form of in-car infotainment devices able to operate as stand-alone devices but also with the capabilities of interacting with smart devices, such as smartphones and tablets. With the exception of highly verticals solutions (e.g., tolling systems), all today vehicular communications are exclusively based on general purpose cellular networks, both through embedded dedicated cellular transceivers or through connections tethered from the smart-devices of the passengers. As a result of this situation, direct inter-vehicles communications are not currently possible, and all applications rely on communications mediated by a centralized entity, either a base station or a service center. The monopoly of cellular technologies is motivated by the difficulties that dedicated short-range communications (DSRCs) are facing in reaching the market, since they lack of a satisfactory business model. Moreover, the interest on DSRCs is decreasing because of the increasing diffusion of devices that passively interact with other vehicles (laser, laser scanner, cameras), in order to actively increase the safety of the passengers.

As a result of this analysis, this book has been conceived around the idea of reusing off-the-shelf technologies in order to reduce the time to market of ITS applications. Coherently with this vision, in the previous chapter we have focused on VANETs based on legacy IEEE 802.11 network interfaces, analyzing the performance of a class of multihop broadcast protocols. In this chapter, we move our attention on smartphones and related technologies, proposing them as key elements of an heterogeneous architecture. As of today, smartphones are gaining more and more popularity, and in the near future each vehicle will likely contain at least one of these devices. This new class of personal device assistants is typically equipped with a dual interface, for cellular networks (typically 3G-UMTS or LTE networks) and WiFi networks, and, therefore, should ideally be able to provide both continuous Internet connectivity and sporadic local connectivity. In this chapter, we present two ITS systems for information dissemination and collection in heterogeneous vehicular environment. The first system addresses the problem of effective information dissemination

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7_4

in vehicular environments and it has been proposed in the 2011–2012 Italy–Israel "Cross-Network Effective Traffic Alerts Dissemination" (X-NETAD) project [1]. As pointed out in [2], the goal of the X-NETAD project was the design and implementation of low cost, efficient and ubiquitous traffic alerts dissemination to drivers. In particular, it envisions the formation of *ephemeral* local VANETs, formed by one "primary" vehicle surrounded by "secondary" vehicles. The former vehicle acts as a gateway for traffic alerts coming from a 3G/4G network, which are broadcasted to the latter vehicles by means of WiFi broadcast communications. As aforementioned, the peculiar feature of X-NETAD consists in being entirely based on smartphones, without the need of dedicated OBUs. In particular, in this chapter we describe our implementation of the X-NETAD protocol on Android-based smartphones and the experimental results obtained in our test fields.

The X-NETAD application focus on real time traffic alerting, a core service of any ITS, due to its strong impact on a multitude of stakeholders. In fact: from the perspective of road operators, efficient traffic alerting allows congestion reduction and smoother traffic flow; from the perspective of drivers, the availability of reliable and updated information on traffic incidents means time and gas saving, increased safety and less stress; from the economic perspective, real time traffic alerting will save time and gas and decrease CO_2 emissions.

This chapter is structured as follows. In Sect. 4.2, related works on multihop broadcast protocols and VANET applications based on smartphones are discussed. In Sect. 4.3, the principle design behind X-NETAD is presented. The details of the dissemination protocol and its implementation on the Android platform are provided in Sect. 4.4. The experimental results are illustrated in Sect. 4.5.

4.2 Related Works

Historically, the systems for monitoring the traffic and detecting congestion and incidents have been based on police reports and dedicated road sensors (e.g., buried inductive loops, active infrared sensors, cameras, and radars). Over the last few years, the increasing pervasive diffusion of cellular communications has enabled the design of traffic estimation techniques based on the data provided from smartphones. In fact, nowadays it can be assumed that, with extremely high probability, there is (at least) a cellular phone inside each vehicle. Usually, cellular networks are used as source of anonymous geo-referenced raw data, which are stored and processed in a remote data center, in order to extract information on the traffic intensity. Relevant examples of this approach are: (i) the TrafficSense system developed by Cellint Traffic Solutions Ltd [3]; (ii) the Localizing and Handling Network Event Systems (LocHNESs) platform developed by Telecom Italia, which allows a real-time evaluation of urban dynamics based on the anonymous monitoring of mobile cellular networks [4]. Nevertheless, due to cellular networks' performance improvement and to the ever increasing diffusion of mobile devices, several ICT researchers designed

and analyzed distributed algorithms and overlays mainly for the implementation of decentralized location-based services. Distributed localization is a clear example of geo-collaboration service, where active users can efficiently discover and disseminate information about existing services (such as a video stream from a webcam, storage volunteers, and other), geo-referenced objects and information (e.g., positions of gasoline stations, accidents, bad surface conditions, traffic jams). These approaches, combined with mobile phone-based nodes, allow to build distributed applications to efficiently harvest measuring reports, traffic information, and travel time in the regions of interest.

Rybicki et al., with Peers on Wheels [5] and, more recently, with PeerTIS [6], have proposed P2P architectures where participating cars are peers organized in a Distributed Hash Table (DHT), to receive and distribute useful information to improve the vehicle travel time using dynamic route guidance. Another P2P approach, that we describe in Chap. 5 in depth, is called *Distributed Geographic Table* (DGT). The DGT is a structured overlay scheme where each participant can efficiently retrieve node or resource information (data or services) located near any chosen geographic position. In such a system, the responsibility for maintaining information about the positions of active peers is distributed among nodes, for which a change in the set of participants causes a minimal amount of disruption. The overlay is different from other P2P-based localization systems (DHT and tree-based overlays), where geographic information is routed, stored and retrieved among nodes, that are organized according to a structured overlay scheme. The DGT idea is to build up the overlay directly taking into account the geographic positions of the nodes and any other information required to build a network where overlay neighbors are also geographic neighbors and no additional messages are needed to obtain the closest neighborhood of a peer.

Hybrid vehicular networks, relying also on cellular connectivity, provide a good solution to overcome the coverage limitations of pure-VANET solutions when the vehicle spatial density is low, thus allowing to reach an increased number of potentially isolated vehicles. Open issues are (i) the data exchange cost over the cellular infrastructure and (ii) the latency of data delivery, which is higher than the pure-VANET approach. Therefore, cellular network-based distributed architectures are not suitable for short range safety-driving applications, but could be efficiently applied in scenarios where (i) latency is not an issue and (ii) the distance from the event or a generic geo-localized information is sufficient to guarantee proper data dissemination.

Regardless of the origin of the information, the issue of real-time disseminating this information to the interested vehicles is still an open problem, that we have tried to address with the IF protocol introduced in Sect. 3.5, which is used by X-NETAD as the main dissemination protocol.

4.3 Cross-network Information Flow

4.3.1 Information Dissemination Through Multihop Communications

The goal of the X-NETAD project was to design and implement an innovative traffic alerts dissemination system based on a cross-network (cellular and WiFi) software application, installable on dual-interface (UMTS and WiFi) smartphones [1]. An illustrative representation of the envisioned cross-network traffic alert dissemination system is shown in Fig. 4.1.

The key idea of the X-NETAD approach is that of leveraging on the spontaneous formation of WiFi local VANETs, with direct connections between neighboring vehicles, in order to disseminate traffic alerts and GPS location in the VANETs very quickly and inexpensively. It is important to remark that X-NETAD has been conceived of for unidirectional information exchange from a centralized data center to mobile vehicles, but it does not allow to collect information from the nodes.

The X-NETAD system has been designed in such a way that, at any given time, only a small percentage (e.g., 10 %) of the vehicles (i.e., the smartphones inside the vehicles) are assumed to be directly connected to the traffic alerts dissemination system through an Internet connection provided by a cellular network. More precisely, the X-NETAD architecture is based on three types of entities: (i) *Primary Users* (PUs); (ii) *Secondary Users* (SUs); and (iii) an Application Server (AS). The AS is in charge of maintaining, filtering and distributing traffic information, collected it from a series of third-party providers—in the specific case of the X-NETAD project, the Israeli company, Cellint, had this role. The AS supports the following minimum set of requirements and operations:

- to provide suitable access interfaces on the Internet;
- to provide a secure communication channel with the PUs;
- to possess authentication, authorization and accounting capabilities;

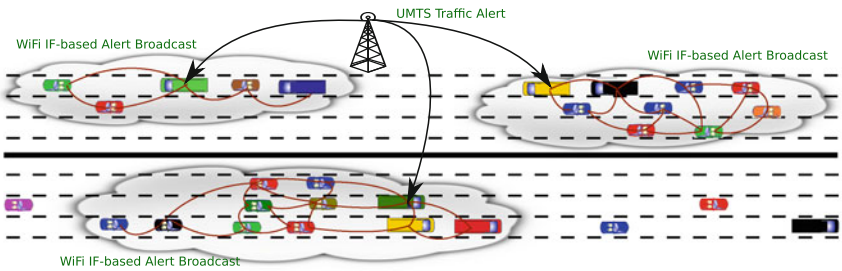


Fig. 4.1 Illustrative cross-network effective traffic alert dissemination (X-NETAD) traffic alerts dissemination scheme

- to support a basic set of operations to allow the PUs to register to the service and obtain localized traffic information (of a given geographical area or route), according to a pull-based distribution model.

PUs are privileged users that periodically retrieve the information of interest from the AS, through a 3G (in general, cellular) Internet connection. The SUs obtain information from the PUs via Vehicle-to-Vehicle (V2V) communications according to a push content distribution model (e.g., the PUs periodically disseminate it without any explicit query from the SUs), but they cannot receive the information directly from the AS. To this end, a very efficient broadcast technique for information dissemination in VANETs, denoted as Irresponsible Forwarding (IF), has recently been proposed [7, 8]. IF assigns to each vehicle an optimized rebroadcast probability, which takes into account the surrounding vehicle spatial density and can be easily tuned; this guarantees fast information dissemination, even in the presence of high vehicular traffic. According to theoretical and numerical analysis presented in [8], the IF protocol and the related Silencing IF protocol perform better than flooding in terms of latency and efficiency and slightly worse in terms of reliability. Furthermore, the advantages of IF with respect to flooding become more evident as the network congestion increases, as in the case of high average vehicular densities or high network loads. The IF protocol does not require nodes to exchange any additional message and, therefore, could easily replace the legacy flooding protocol that is still used by many Car-2-Car Communications Consortium (C2C-CC) [9] related projects. For instance, the geo-broadcasting protocols defined within the GeoNet project (e.g., GeoBroadcast and TopoBroadcast) rely on the flooding protocol for multihop broadcast communications [10]. Greater details concerning the IF protocol are provided in Sect. 3.5.2.

4.3.2 A Push/Pull Dissemination Approach

Generally speaking, the choice between pull-based and push-based approaches should be taken according to four main metrics and parameters: (i) the delay affecting the information distribution; (ii) the energy consumption of the device; (iii) the complexity of the client and server applications; and (iv) the unicast or broadcast nature of the communications. In fact, a push-based approach is suitable for broadcast communications, since it allows one to increase the network efficiency. As mentioned in Sect. 4.3.1, in X-NETAD, a hybrid push/pull paradigm is used. In fact, the communications between PUs and SUs use a push-based approach, while the communications between AS and PUs employ a pull-based approach.

A pull-based approach for the AS/PUs communications has been adopted for the following motivations.

- The communications between AS and PUs are unicast, since (i) 3G networks do not allow broadcast communications and (ii) each PU should receive different information according to its position.

- The information distributed in X-NETAD is not time-critical, and thus, there is no need to use a push-based approach in order to minimize the delay.
- The energy consumption is one of the main concerns with mobile applications, and it is well known that a 3G connection has a high energy consumption. By adopting a pull-based approach, the mobile application can schedule the request for traffic updates without the need of keeping a 3G connection active all the time. Moreover, the application can adapt the intervals between consecutive queries according to the behavior of the vehicles. For example, if the vehicle is not moving, there is no need to look for traffic updates. Similarly, if the battery of the smartphone is depleting, the application can decide to reduce the update frequency.
- A pull-based approach significantly increases the complexity of the server, since it has to cope with a potentially large number of concurrent requests—current techniques, however, should not make this aspect a limitation.

In the case of PUs/SUs communications, the scenario is totally different. In fact, in this case, the PU transmits the same information to all the SUs and, therefore, it is far more efficient to employ broadcast communications instead of a series of unicast communications. Moreover, in VANETs, the nodes cannot turn off their radio interfaces in order to save energy, because of the following critical issues that affect VANETs: (i) short-lived connections; (ii) highly dynamic topology; and (iii) harsh channel conditions. This prevents a pull-based approach from leading to significant energy savings.

The above reasons motivate the decision for adopting a push-based paradigm, which is also considered in recent works [11].

4.3.3 Securing X-NETAD

For X-NETAD, we have proposed a general architecture that is largely based on the one introduced by Papadimitratos et al. [12]. In particular, the node architecture is shown in Fig. 4.2. Coherently with the main goal of this book, we will not detail the characteristics of the security infrastructure, but we will only sketch it.

First of all, a PKI, constituted by several CAs, is established. The vehicles communicate with the CAs through either cellular-based or RSU-based communications. Each vehicle has a unique IDentification (ID), a pair of long-term private-public keys, and a long-term certificate issued by a CA upon node registration. Actually, long-term credentials are not directly used, but they are needed to issue second-level short-term credentials, denoted as "pseudonyms", which are actually used for securing V2V communications. The goal of using pseudonyms is twofold: (i) to reduce the risk of compromising the long-term credentials; (ii) to increase the privacy of the vehicle. However, it is important to remark that since short-term certificates need to be issued from CAs, then pseudonyms can be used only if Internet accesses are frequently available to the vehicles. In the case of the considered application, the PUs continuously have an Internet connection. To summarize, with respect to a legacy

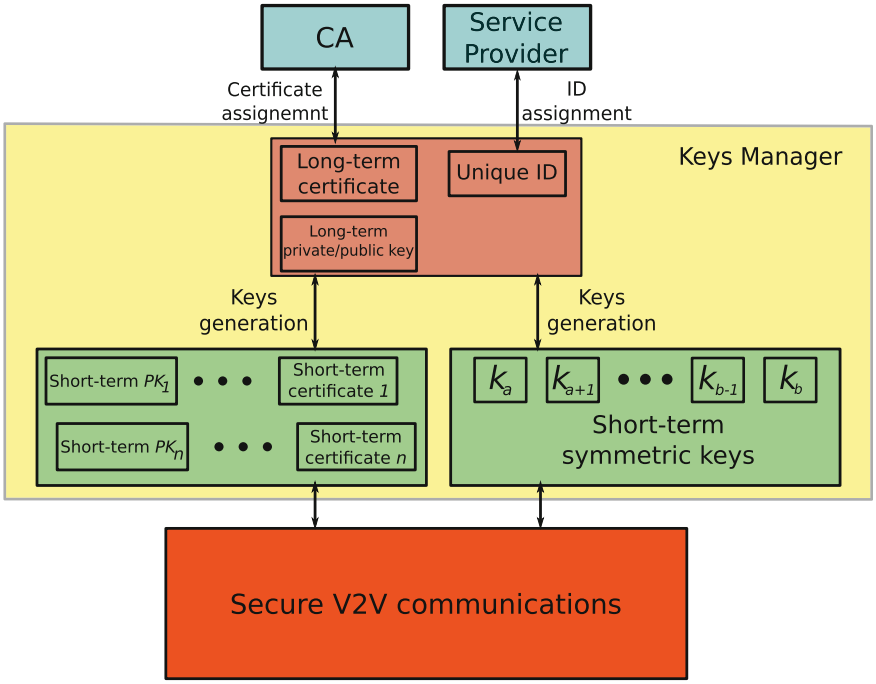


Fig. 4.2 Security architecture of the considered application

IEEE 1609.2 system, the proposed architecture provides (i) a higher level of privacy and (ii) key and identity management systems.

It should be now addressed the problem of deriving and distributing a common group key across all the authorized users. From a logical point of view, the problem of generating a common group key is the same addressed in detail in Appendix C. In fact, the service provider is the unique entity with the right to authorize network users. Therefore, it can act as a full-fledged KDC, reusing the technique described in Appendix C for managing user subscriptions and generating new group keys. Instead, the distribution of the new group keys is hindered by the lack of direct links between the SUs and the KDC. In order to solve this problem, we have adopted a two phases strategy. In the first phase, the KDC distributes the new group keys to the PUs by using the unicast links existent between them, which can be easily secured by using standard cryptographic tools. In the second phase, the PUs distribute the new group keys to the SUs belonging to their local clusters. The distribution is carried out by establishing temporary unicast links between PUs and SUs. The group keys can be encrypted, either with symmetric or asymmetric cryptographic methods, by using the encryption primitives provided by the IEEE 1609.2 stack and the cryptographic materials associated to the pseudonyms.

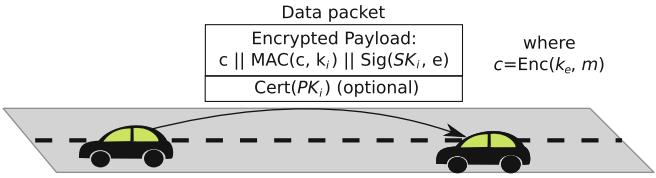


Fig. 4.3 Structure of the packet with emphasis on its encrypted payload

It is now possible to define the operations performed by a PU to broadcast an encrypted packet to the SUs of its local cluster. We remark that the symbol k denotes the common group key.

1. Generate two sub-keys k_i and k_e from the common secret key k.
2. Encrypt the message (denoted as m) with k_e: $c = \text{Enc}(k_e, m)$.
3. Compute the Message Authentication Code (MAC) of c with the key k_i: MAC (k_i, m).
4. Compute the signature of c with the private key Sk_i: $\text{Sig}(Sk_i, c)$.
5. Compose the packet as follows: $c||\text{MAC}(ki, c)||\text{Sig}(Sk_i, c)$.
6. If required, attach the certificate $\text{Cert}(PK_i)$ and, finally, send the message.

The structure of the packet built according to these operations is shown in Fig. 4.3.

Once the encrypted message has been correctly received, a SU should perform the following operations.

1. Obtain the public key PK_i from $\text{Cert}(PK_i)$ (if it is attached).
2. Generate two sub-keys k_i and k_e from the common secret key k.
3. Verify the integrity of the message, by computing the MAC of c with the key k_i, and compare it with the received MAC.
4. Verify the signature of c with the public key PK_i (if known). If the public key is not known, it may be decided not to verify the signature at all.
5. Finally, decrypt the message c with k_e, after having verified the authenticity and the integrity of the packet: $m = \text{Dec}(k_e, c)$.

4.4 Application Design and Implementation on Android Smartphones

This section proposes an overview on the design and implementation of the X-NETAD system. At first, we explain how the system operates, discussing also some challenges faced during the implementation process. Next, we describe the message exchange in the system, together with a description of the architecture blocks of X-NETAD nodes, focusing on the structural and logical differences between PUs and SUs.

4.4.1 System Overview and Challenges

The X-NETAD system is a network architecture conceived for the distribution of valuable traffic alerts to mobile vehicles, taking advantage of both cellular and *ad hoc* communications. A subset of nodes that form the system (namely, the PUs) should have the ability to simultaneously manage the cellular and the IEEE 802.11 interfaces (the latter being set in *ad hoc* mode), while the remaining nodes (i.e., the SUs) will only use the IEEE 802.11 interface. The X-NETAD application has been designed to be general enough for various mobile OSs and has been implemented on Android mobile devices.

Like other mobile OSs, Android is natively capable of connecting to IEEE 802.11 networks operating in the infrastructure mode, but it lacks any support for the *ad hoc* operating mode. However, it is possible to overcome this limitation, since most wireless network adapters inside Android devices are compliant with the full IEEE 802.11 standard and, thus, support the *ad hoc* mode. Such a functionality can be exploited running the OS as administrator with super-user rights, using a procedure called *rooting* and properly configuring the wireless adapter to connect to *ad hoc* networks using the classic *ifconfig* Linux command. Furthermore, the access to network interfaces is natively exclusive, i.e., only one between the cellular and the IEEE 802.11 interfaces is active at a time. In X-NETAD, we have introduced suitable software routines that overcome this limitation by properly altering the routing table of the devices, making it a true multi-homed device.

We designed X-NETAD in order to process and transfer large information amounts. In this way, the information exchanged—at first, between the AS and the PUs and, subsequently, between any PUs and the surrounding SUs—could include multimedia data, such as audio, video and map files. For instance, in our proof-of-concept application, the information issued by the AS is a compressed archive that includes a real-time traffic map in *jpg* format, an audio file in *mp3* format and an XML-formatted document. In the following, we will refer to the terms *traffic update*, *archive* and *information*, as the overall traffic information file created by the AS. The XML document contains information about the traffic update: its date and time of creation, the map's longitude and latitude coordinates expressed in degrees and, eventually, the information about errors that may have occurred in the network. An example of XML-formatted text file is shown in Fig. 4.4. We intentionally decided to focus on "complex" information, rather than on simple text messages, in order to be able to present a rich content to X-NETAD users. In this way, it is possible to inform the driver without distracting his/her eyes from the road, by simply playing the received audio alert on his/her mobile device.

The transmission of such a relatively large amount of data (the size of the archive could reach some MB) in the *ad hoc* domain becomes challenging, due to the limited range of the wireless interfaces and the vehicles' mobility. In addition, classical phenomena, such as fading, shadowing and the contention-based channel access mechanism employed by the IEEE 802.11 standard, actively contribute to limit the theoretical maximum throughput of the system. In order to reduce the impact of the

```
<report_info>
 <error/>
 <report_date>
  <date>2011-06-06</date>
  <time>18:25:00</time>
 </report_date>
 <map_top_left_point>
  <longitude>34.749871</longitude>
  <latitude>32.006862</latitude>
 </map_top_left_point>
 <map_bottom_right_point>
  <longitude>35.247987</longitude>
  <latitude>31.719802</latitude>
 </map_bottom_right_point>
</report_info>
```

Fig. 4.4 An example of XML-formatted document created by the application server (AS)

problems mentioned above, the compressed archive received by a PU through the cellular interface is not directly transmitted to SUs, but, instead, it is split in many small pieces that are disseminated independently.

On the basis of an analysis of a VANET's peculiar characteristics, such as high node mobility and limited transmission range, we adopted a broadcast (starting from a primary node) approach for our system. In X-NETAD, all vehicles act as independent routers and cooperate together, fostering the complete reception of the traffic update. All transmissions in the system are broadcast, in order to reach the largest number of nodes at a time and to avoid any overhead related to routing tables' update issues. The IF protocol controls message rebroadcasting, guaranteeing an efficient network-wide data dissemination.

4.4.2 Message Structure and Dissemination Protocol

In Fig. 4.5, an illustrative representation of the X-NETAD session of message exchange between the AS, one PU and a multitude of SUs is shown.

In particular, the pink right pane identifies the communication plane in UMTS network that will discussed in Sect. 4.4.2.1, while the gray left pane includes the communication plane on WiFi network and it is discussed in Sect. 4.4.2.2.

4.4.2.1 Cellular Domain

The UMTS plane only encompasses the messages exchanged between the PU and the AS according to a request/answer paradigm, requests are periodic but they can be also triggered by the PU. In particular, the PU periodically query the AS with an interval T^{q} (dimension: [s]) determined by the application according to many

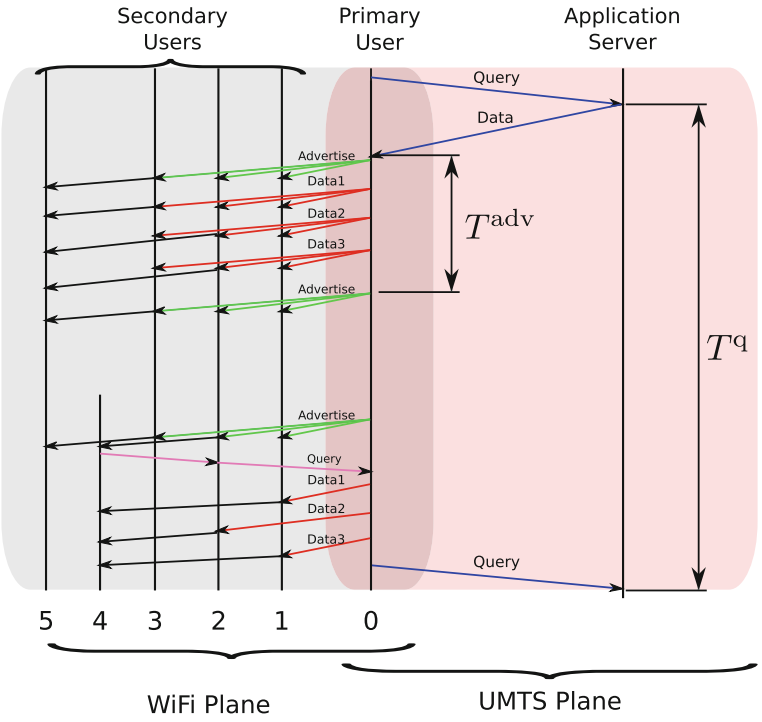


Fig. 4.5 Temporal diagram of a generic X-NETAD message flow

parameters, including traffic conditions and average speeds. Updated traffic information is retrieved by PUs via an HTTP GET request sent to the AS through the cellular network. Important parameters, such as the username, the password and the geographical coordinates (latitude and longitude), of the requesting node are included in the GET request by the X-NETAD client. The AS needs the usernames and passwords of the nodes for authentication purposes. The traffic information provided by the AS is geographically relevant, taking advantage of the PUs' coordinates. If the authentication information provided is correct, the AS replies to PUs by sending the archive containing the local traffic update.

4.4.2.2 *Ad Hoc* Domain

As previously mentioned, we designed the X-NETAD system in order to be able to disseminate a (relatively) large content, adapting to the constraints imposed by the VANET scenario. The strategy we adopted to enforce this feature splits the archive that comes from the cellular network into smaller pieces, called *chunks*, which are disseminated independently. Chunks are small data units of fixed size, exchanged with high probability during a single contact of limited duration. If a chunk is only

partially received from a node (e.g., due to a connection breakdown), it is discarded. If, on the one hand, splitting the traffic update into several chunks allows nodes to receive parts of the same traffic update from different nodes, on the other hand, this adds complexity to the system, because nodes should be able to recognize which chunks are missing and, if needed, should request them back. The system behavior can be described as follows. The original traffic update received from the AS through the cellular network is divided in several chunks by the PU, and every chunk is packed into different data-bearing messages denoted as *DATA*. A *DATA* message contains the actual data chunk and the chunk index, which is crucial, because it allows the nodes to identify which part of the update they are receiving. In fact, each SU maintains a local bit-vector in its memory, whose length equals the total number of chunks of the current traffic update, where the i-th bit indicates whether the i-th chunk is missing (bit set to zero) or present (bit set to one). After each *DATA* message is received by an SU, the local bit-vector is correspondingly updated. The payload of any *DATA* message, related to the same traffic update, has a fixed size, apart for the last chunk. This size depends on the dimension of the original traffic update and on the number of chunks in which it is divided. The chunk size is an important parameter that significantly affects the system performance. With large chunks, there is the risk of performance degradation when partially received chunks are discarded by the network interface. The choice of small chunks reduces the amount of wasted reception, but it also increases the overhead. It is worth recalling that before being able to decode a traffic update, a node must receive all the *DATA* messages associated with it. *DATA* messages are propagated and replicated network-wide using the IF protocol, thus providing redundancy in the system.

Whenever a PU receives a new traffic update from the AS using the cellular network, it should disseminate this information to SUs located in the *ad hoc* domain. To make SUs aware of the existence of a new traffic update, allowing them to create a local bit-vector for the update, the PU issues an *ADVERTISE* message. This message contains information on the update, such as its size and the number of chunks in which it is divided. Once a SU receives a new *ADVERTISE* message, it creates a new bit-vector for the update and becomes ready to receive its flux of *DATA* messages. Since, without this message, a SU cannot store the subsequent chunks of data, the goal is to provide a fast and reliable dissemination of the *ADVERTISE* message all over the network using the IF protocol. In order allow SUs, which join the network later, to get the traffic update, the *ADVERTISE* message is also periodically broadcast by PUs, with an interval denoted as T^{adv}.

Because of the probabilistic nature of IF and of the underlying contention mechanism, there still exists the opportunity that messages may be lost due to noise bursts on wireless channels or to message collisions. A simple solution to recover from packet losses is to use a feedback loop. To minimize the total number of packets sent in the network and, thus, to prevent congestion and additional collisions, SUs ask periodically for missing chunks using a *REQUEST* message. The payload of the *REQUEST* message is a bit-vector representing the reception status for each chunk. Periodically, each SU examines its local bit-vector, and missing chunks are requested back (in broadcast) to neighbors, using the *REQUEST* message. Regardless of the

nature of a node (PU or SU), upon the reception of a *REQUEST* message, a node checks its local bit-vector and tries to satisfy, if possible, the request. As opposed to other X-NETAD messages, the *REQUEST* message is intrinsically single hop and is never re-broadcast in order to maximize the reception probability of missing chunks and to avoid repeatedly saturating the network with useless packets. A node replies to a *REQUEST* message with one or more *DATA* messages if the requested chunks are locally available and according to a probabilistic function related to its distance from the sender. This replying probability is expressed according to the following "reverse" IF probability assignment function:

$$p = \exp\left\{-\frac{\rho_s d}{c}\right\} \tag{4.1}$$

where ρ_s is the spatial density of vehicles in the VANET; d is the distance between transmitter and receiver; and c is a shaping coefficient. Clearly, this function assigns a higher replying probability to nodes that are located near the original sender, namely those with a smaller packet loss probability. It is important to observe that the replied (e.g., transmitted upon reception of a *REQUEST* message) *DATA* messages are disseminated using the IF protocol.

4.4.3 System Architecture

In Fig. 4.6, a general illustration of the X-NETAD mobile application is shown. Traffic updates are transferred from the AS to the PUs via the cellular network. Content can also be exchanged directly between PUs and SUs when they are within the communication range of one another. Here we primarily focus on the system design with respect to content diffusion between nodes in the *ad hoc* domain.

The architectural differences between the two distinct types of nodes are restricted to the ability of PUs to connect to a cellular network, in order to communicate with the AS. For this reason, a PU's architecture exhibits a `Cellular Manager` block—drawn with a dotted line, to remark on its presence only at PUs—that is in charge of the communications taking place with the AS (see Sect. 4.4.2.1) using the cellular interface. The `Wireless Manager` block, on the other hand, is present at any node, as it drives the IEEE 802.11 interface and is fundamental in the dissemination process taking place in the *ad hoc* domain. The `X-NETAD Dissemination Manager` is the key component of the whole system, due to the fact that it is in charge of the logical system management. If a new update is received from the AS, its content is stored locally, split in smaller chunks and the dissemination process is started. Upon the reception of an X-NETAD message on the IEEE 802.11 interface, the `Dissemination Manager` performs all the actions required to handle the message (see Sect. 4.4.2.2). For example, when the system receives a *DATA* message, the `Dissemination Manager` checks if the received chunk is missing, updates

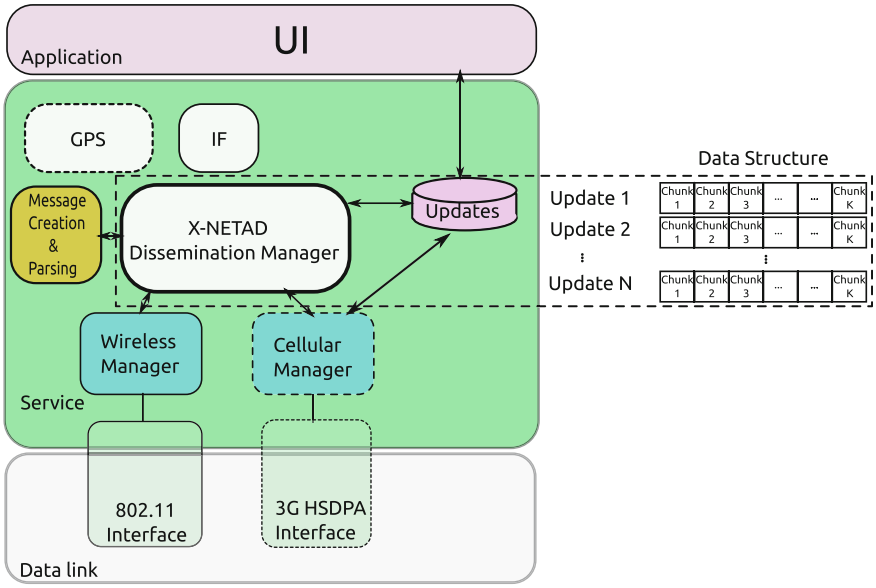


Fig. 4.6 System architecture of an X-NETAD node

(if required) its local bit-vector and, with the help of the `IF` and the `GPS` blocks, decides if the message has to be re-broadcast.

When a new traffic update is available, regardless of whether it was retrieved from the cellular or the IEEE 802.11 interface, it is presented to the user through the User Interface (UI) application. If the update contains an audio message, then such a message is played, and the additional content is presented according to the user's preferences.

4.5 Experimental Results

In this section, we analyze the experimental results in different environments, obtained with a test-bed composed by four LG Optimus One (P-500) smartphones and a Samsung Galaxy Tab tablet, all equipped with release 2.3.3 of the Android OS. The experimental campaign is conducted in three different phases. In the first phase, we preliminarily investigate the relationship between the *Packet Error Rate* (PER) and a couple of physical parameters, namely, the IEEE 802.11 transmission data rate (in Sect. 4.5.2.1) and the inter-node distance (in Sect. 4.5.2.2). In the second phase, described in Sect. 4.5.3, we evaluate the performance of the X-NETAD application in a couple of ideal static scenarios, where nodes are fixed. In the third phase, described in Sect. 4.5.4, we consider a more realistic scenario, where devices are positioned within distinct vehicles moving along a predefined path. In the second and third

phases, the behavior of the X-NETAD application is characterized by means of the performance metrics introduced in Sect. 4.5.1.

4.5.1 Metrics of Interest

Depending on its nature, the metrics of interest has been characterized in terms of average value, *Probability Mass Function* (PMF) or *Cumulative Distribution Function* (CDF). The performance metrics considered in our experimental campaign can be summarized as follows.

- The number of *DATA* messages retransmitted at each SU—the ratio between this quantity and the total number of transmitted packets is correlated to the Probability Assignment Function (PAF) of the IF protocol defined in Eq. (3.5).
- The PMF of the hop index at which the *DATA* messages are received. This indicator is strictly related to the distance between a PU and an SU.
- The number of missing *DATA* messages, before the first *REQUEST* message. This metric is a coarse indicator of the wireless channel quality. In ideal conditions, the PMF should concentrate only at zero. However, in realistic conditions, this statement is not valid, due to radio channel interference and fading.
- The number of *DATA* messages requested when receiving a *REQUEST* message. This metric is representative of the "attitude" of a node to receive requests for missing chunks from surrounding nodes.
- The PMF of the archive reception time. This metric indicates the amount of time it takes to a node to receive all the chunks of the archive sent by the PU.
- The CDF of the number of *DATA* messages received before each *REQUEST* message. This metric is an indicator of the reception status of the traffic update before each round of *REQUEST* messages.

4.5.2 Preliminary Results

4.5.2.1 PER as a Function of the Data Rate

Since the transmission power of IEEE 802.11 compliant transceivers is fixed by the regulatory authorities, the length of an IEEE 802.11 link (i.e., the transmission range of a node) depends only on the receiver sensitivity, which, in turns, is directly related to the selected data rate and modulation. The relationship between sensitivity and the allowed data rates of the IEEE 802.11 standard [13] is summarized in Table 4.1.

Furthermore, the IEEE 802.11 standard does not provide any automatic rate adaptation mechanisms for devices operating in the *ad hoc* mode. Therefore, the transmission rate must be set during the network initialization phase.

Table 4.1 Relationship between modulation, data rate and receiver sensitivity defined by the IEEE 802.11 b/g standard [13]

Data rate (Mb/s)	Scheme	Modulation	RX sensitivity (dBm)
1	DSSS	BPSK	−89
6	OFDM	BPSK	−82
9	OFDM	BPSK	−81
12	OFDM	QPSK	−79
18	OFDM	QPSK	−77
24	OFDM	16-QAM	−74
36	OFDM	16-QAM	−70
48	OFDM	64-QAM	−66
54	OFDM	64-QAM	−65

DSSS Direct Sequence Spread Spectrum; *BPSK* Binary Phase-Shift Keying; *OFDM* Orthogonal Frequency-Division Multiplexing; *QPSK* Quadrature Phase-Shift Keying; *QAM* Quadrature Amplitude modulation

For these reasons, it is interesting to preliminarily investigate the relationship between the transmission rate and the experimental Packet Error Rate (PER) observed in a direct IEEE 802.11 link. In this test, we consider two fixed smartphones, placed at a distance of 15 m: the first inside a car and the second inside a nearby building with large windows. For each value of the data rate, the first smartphone broadcasts 100 streams composed of 1, 000 User Data Protocol (UDP) packets with an application payload set at 100 bytes. In this setup, packet losses are mainly caused by: (i) multipath fading, due to the wireless propagation inside the room and the vehicle; (ii) shadowing, due to the random movement of persons and vehicle. We consider several transmission data rate values lying in the range, [1, 54] Mb/s. From the obtained experimental results, shown in Fig. 4.7, it emerges that, as expected, lowering the data rate can help to reduce the PER and extend the transmission range of X-NETAD nodes. However, we found that the minimum experimental PER is obtained when the data rate is set at 11 Mb/s, rather than at 1 Mb/s (i.e., at the lowest possible data rate). Such a behavior can be explained by the fact that when the data rate is set to 11 Mb/s, the modulation scheme switches to the Direct Sequence Spread Spectrum (DSSS), which is more robust in terms of PER, than the Orthogonal Frequency-Division Multiplexing (OFDM) used at lower rates. Further reduction in data rate does not lead to an additional PER reduction, since the transmission time becomes longer, increasing the likelihood of transmitting during a noise burst, thus resulting in a slight PER increase. According to the results obtained in our tests, in the remaining part of this analysis, we set the transmission rate to 11 Mb/s, in order to have a good trade-off between the transmission range and the PER.

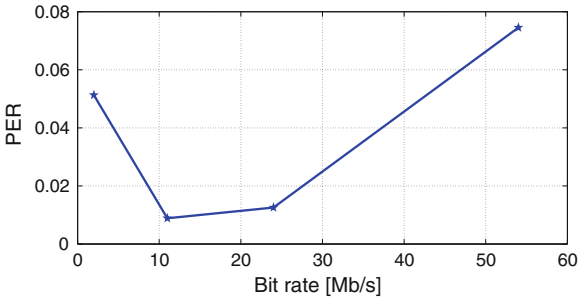


Fig. 4.7 Experimental packet error rate (PER) as a function of the data rate, by considering $P_t = 32$ mW and $d = 15$ m

4.5.2.2 PER as a Function of the Distance

Besides the data rate, the distance between communicating nodes in the system has a major impact on the performance of the X-NETAD dissemination protocol. Even if the IF protocol assigns higher rebroadcasting probabilities to the farthest nodes, packet loss probability is an increasing function of the distance. Propagation losses affect all packets, depending on a multitude of factors (e.g., fading, shadowing, absorption, multipath interference, distance). Unlike other factors that are strictly related to the surrounding environment, one can estimate the impact of the distance between sender and receiver nodes, through the *Friis* transmission formula [14]:

$$\frac{P_r}{P_t} \propto \left(\frac{\lambda}{d}\right)^n \tag{4.2}$$

where P_t represents the transmission power [dimension: (mW)]; P_r is the received power [dimension: (mW)], λ is the wavelength [dimension: (m)]; d is the distance between receiver and sender [dimension: (m)]; and n is an experimental coefficient that can vary (typically, it lies in the range [2, 4]—$n = 2$ for free space transmission). Equation (4.2) relates the received power with the transmission power and has an impact on the average number of packets received by a node in our system, because IEEE 802.11-compliant devices have a maximum transmission power (32 mW). Therefore, the longer the distance between the nodes, the higher the probability that a packet is lost. The *Friis* formula gives a characterization of the average received power due to propagation losses over an ideal channel, but does not take into account all the space-time fluctuations, due to phenomena, like *fading*, *shadowing* or *Doppler* effects. Despite these limitations, Eq. (4.2) remains a useful tool for understanding the impact that propagation losses have on the application performance.

In order to better understand the impact of the transmission distance on the PER, we carry out trials in an ideal static scenario, where the wireless propagation can be considered as in free space. We employ two smartphones, configured to send a stream of 100 UDP packets—each packet with an application-level payload

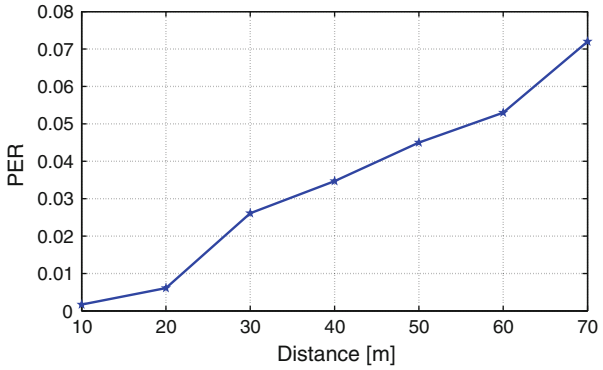


Fig. 4.8 Experimental PER as a function of the inter-node distance, by considering $P\text{tx} = 32$ mW and $R = 11$ Mb/s

equal to 100 bytes—on the WiFi interface. For each run, we repeat 100 times the transmission of the packet stream, and the experimental PER is computed by averaging over the total number of packets sent. Test devices are fixed at a height of about 1.5 m from the ground, in order to avoid as much as possible ground absorptions and reflections—moreover, this value represents, also, the approximate height of a smartphone positioned in a vehicle. We considered different distances between the smartphones. The transmit power is set to the maximum allowed level (32 mW) and the data rate to 11 Mb/s, as indicated in Sect. 4.5.2.1.

As expected, according to the experimental results shown in Fig. 4.8, the PER is an increasing function of the inter-node distance. Furthermore, the shape of the experimental curve appears to be approximately linear. According to the results in Fig. 4.8, it emerges that for $d = 60$ m, the PER is equal to 0.054. We consider this value of d as the experimental transmission range, z, inside the IF PAF defined in Eq. (3.5)—this experimental value is lower than the open space transmission range claimed by the IEEE 802.11 standard [13]. Note that, with respect to the tests performed in Sect. 4.5.2.1, the PER in Fig. 4.8 is much lower, as in this case the test are carried out in more favorable conditions (almost free space).

4.5.3 Tests in Ideal Static Scenarios

As mentioned before, in the second phase of our experimental campaign, we perform a number of field tests in ideal scenarios, where the nodes are static and positioned in an environment not affected by phenomena like shadowing and fading. In the considered experimental conditions, the wireless medium access problem cannot be extensively analyzed, and the IF protocol does not perform at its best. The number of retransmission choices is, in a probabilistic sense, very small. Although a larger number of nodes should be considered (e.g., at least 15–20 devices) in order to have

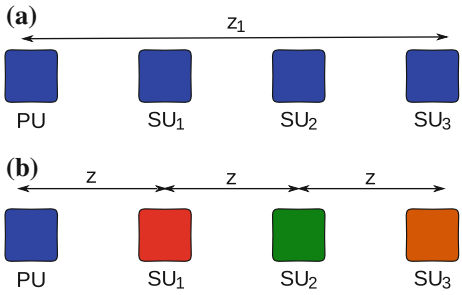


Fig. 4.9 Static X-NETAD application testbed: (**a**) Scenario 1, all secondary users (SUs) are located in the transmission range of the PU; (**b**) Scenario 2, each SU is at the edge of the transmission range of previous nodes

higher node spatial density, the obtained results shed light on the performance of the proposed information dissemination system.

4.5.3.1 Experimental Setup and Results

We evaluate the performance of the X-NETAD application in two different static scenarios, which are illustrated in Fig. 4.9. For each test, the transmit power, P_t, of each device was set to its maximum value, equal to 32 mW, and the data rate, R, was fixed at 11 Mb/s—the rate that guarantees the minimum experimental PER, as shown in Sect. 4.5.2.1. According to the results presented in Sect. 4.5.2.2, we use the value, 60 m, as the device's transmit range, z, to be used in the IF protocol. The linear node spatial density, ρ_s, is related to the considered scenario, since it depends on the (physical) spatial distribution of the nodes. The shaping coefficient, c, is set to 5 to balance the small number of nodes present in the system and, thus, to increase the rebroadcasting probability. In each test, the application disseminates the same archive of roughly 100 KB, divided in 128 chunks. During tests, we considered, as an expiration date for each content, $T^{\mathrm{q}} = 60$ s (the choice of this value allows significant information propagation). Finally, the waiting time before sending a *REQUEST* message is set to 5 s.

Scenario 1

This scenario presents the most intense level of channel access contention, since all the SUs are within the transmission range of the PU, as shown in Fig. 4.9a. The linear spatial density, ρ_s, is equal to 0.05 veh/m, and the total number of SUs is given by $\rho_s z = 3$ veh. In Fig. 4.10, the experimental PMF of the number of rebroadcast *DATA* messages at each SU is shown. As one can observe, the rebroadcasting probability of SUs is strictly correlated to their distances from the PU (see Eq. (3.5)). Fluctuations around average values of the rebroadcasting probability depend on GPS localization errors.

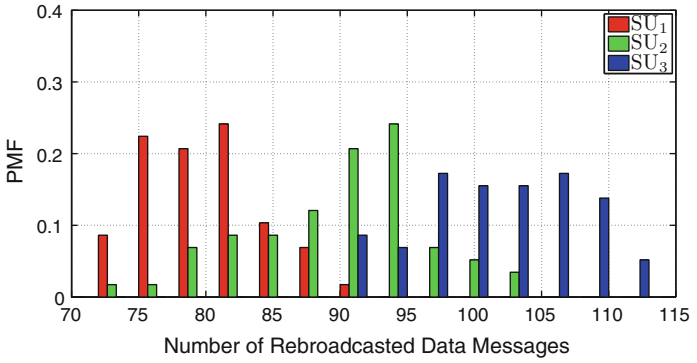


Fig. 4.10 Probability mass function (PMF) of rebroadcast *DATA* messages in Scenario 1

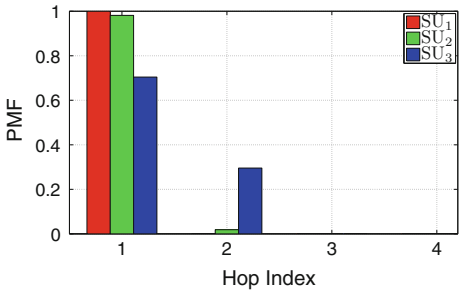


Fig. 4.11 Hop index PMF for received *DATA* messages in Scenario 1

In Fig. 4.11, the PMF of the hop reception index of *DATA* messages at each SU is shown. In principle, all SUs should receive all messages directly from the PU, because they are within its transmission range. Nevertheless, even though some messages might be lost because of channel impairments, they can still be received, as information propagation is guaranteed by the rebroadcasting nature of the IF protocol. For this reason, we found that some *DATA* messages are received with a hop index equal to 2. This proves that probabilistic rebroadcasting adds redundancy, which, in turn, increases the system reliability.

Despite this, there is a chance that a certain message may be lost. In Fig. 4.12, the PMF of the number of missing chunks at a given node, before the first *REQUEST* round, is shown. It can be pointed out that in this scenario, both SU_1 and SU_2 experience a small number of lost *DATA* messages (smaller than the 10 % of *DATA* messages sent), while SU_3 experiences a higher loss ratio, due to the fact that its distance from PU is longer. Moreover, from Fig. 4.12, one can see that SU_3 always misses at least one chunk, while SU_1 and SU_2 sometimes (with a probability higher than 10 %) do not need to send any feedback message. At this point, it is worth investigating which nodes receive and satisfy the largest number of *REQUEST* messages. For this purpose, in Fig. 4.13, we show the total number of missing *DATA* chunks

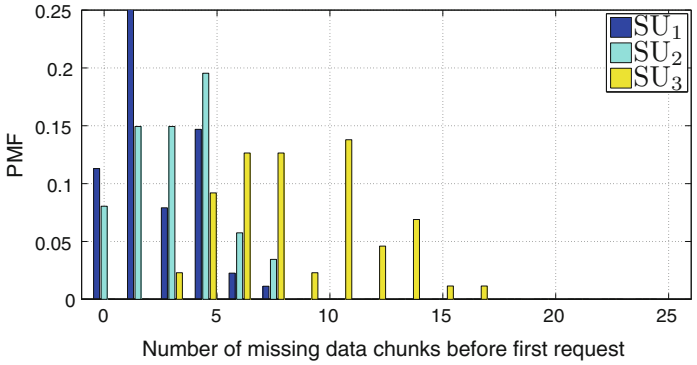


Fig. 4.12 PMF of missing *DATA* chunks, before the first *REQUEST* round in Scenario 1

requests, received by a given node, and the total number of *DATA* messages replied by the node as a result of previous requests. According to these results, SU_1 and SU_2 receive the largest number of packet requests. However, the larger number of replies is given by PU_1, since it does not adopt a probabilistic broadcasting protocol (in fact, it satisfies the 100 % of requests). SUs have a much lower satisfaction ratio, which reduces as the distance from the PU_1 rises: this is due to Eq. (4.1) and because a node cannot satisfy a *REQUEST* for a packet that has not yet been received. Finally, in Fig. 4.14, the PMF of the traffic updates reception time is shown. It can be observed that, on average, all nodes have a completion time (i.e., all the *DATA* messages of a given archive are received) lying in the interval $[5 - -7]$ s. This means that only one *REQUEST* message is sent back (we recall that $W_{\text{req}} = 5$ s). The exception is represented by SU_3, which completes the reception of all *DATA* messages in more than 10 s with a probability around 0.18. In order to have a better comprehension of this phenomenon, in Fig. 4.15, the CDF of received *DATA* messages is shown as a function of the number of *REQUEST* messages. One can see that before the first *REQUEST*, SU_1 typically has a completion rate equal to 0.98, while SU_2 and SU_3 achieve slightly smaller values (equal to 0.97 and 0.93, respectively). Hence, in this scenario, all the SUs can complete the reception before the second *REQUEST* with a high probability.

Scenario 2

In this scenario, shown in Fig. 4.9b, it is assumed that each node is located at the edge of the transmission range of the preceding node. For this reason, multi-hop communications are needed to disseminate traffic updates throughout the network, and this degrades the performance. In particular, the PMF of the retransmitted *DATA* messages is presented in Fig. 4.16. We note that, with respect to Scenario 1, the rebroadcasting probability is (obviously) higher for all SUs ($d = z$). The number of

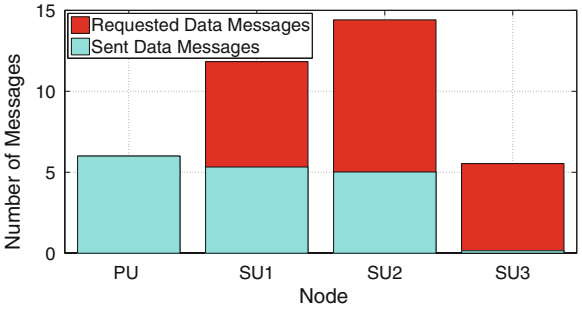


Fig. 4.13 Average number of *DATA* chunks requested and average number of *DATA* messages sent following a *REQUEST*, in Scenario 1

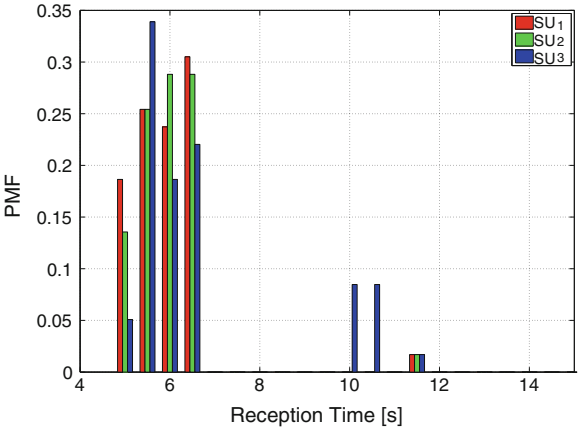


Fig. 4.14 PMF of the archive reception time in Scenario 1

retransmissions is upper bounded by 128, which represents the number of chunks in which the archive is fragmented.

In Fig. 4.17, the PMF of the received *DATA* messages is shown. It can be observed that, for each SU, there exists, also, the possibility to receive messages from other SUs, due to the feedback mechanism detailed in Sect. 4.4.2.2. For instance, some *DATA* messages received by SU_3 are one-hop messages, because they come from SU_2 and are sent after the reception of a *REQUEST* message. Nevertheless, we note that for all SUs, there is a dominant hop value centered at their distance (in terms of hops number) with respect to the PU.

As can be seen from Fig. 4.18, the number of missing chunks before the generation of the first *REQUEST* message is larger than in the previous case, thus significantly penalizing Scenario 2 with respect to Scenario 1. In particular, it can be noted that: (i) each SU needs to use at least one *REQUEST* message; and (ii) the number of missing

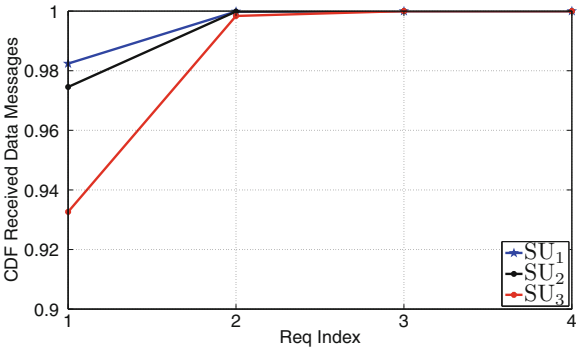


Fig. 4.15 Cumulative distribution function (CDF) of received *DATA* messages before *REQUEST* messages in Scenario 1

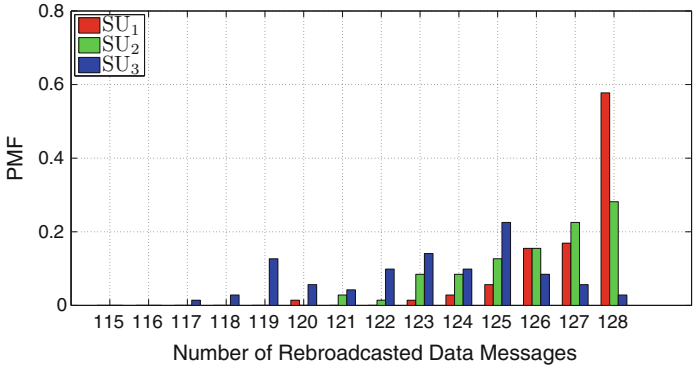


Fig. 4.16 PMF of the number of rebroadcast *DATA* messages in Scenario 2

chunks is, for each run, larger than 10. It can also be noted that SU_3 is particularly penalized, because link losses generate cumulative effects.

In Fig. 4.19, the PMF of the archive completion time is shown. As expected, the reception time is longer than in Scenario 1. The difficulty of receiving all the chunks is evidenced by the larger number of *REQUEST* rounds for missing chunks that contribute to increase the completion time.

Finally, in Fig. 4.20, the PMF of the completion time before the first *REQUEST* message is shown, which is around 0.7, 0.65 and 0.55, respectively, for SU_1, SU_2 and SU_3. The use of a single feedback *REQUEST* message raises the completion rate to a value of about 0.9 for each node. Hence, on average, at least one or even more *REQUEST* messages are required to complete the reception of the archive.

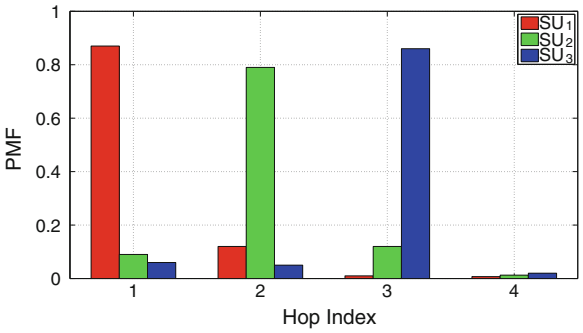


Fig. 4.17 PMF of the hop index for received *DATA* messages in Scenario 2

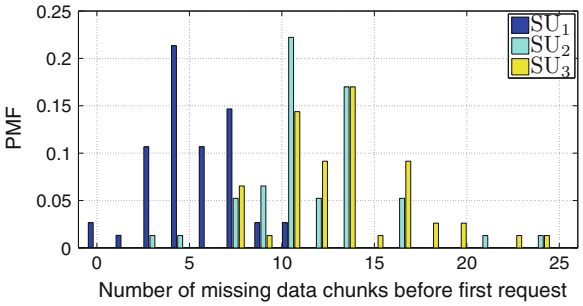


Fig. 4.18 PMF of missing *DATA* chunks, before first *REQUEST* round in Scenario 2

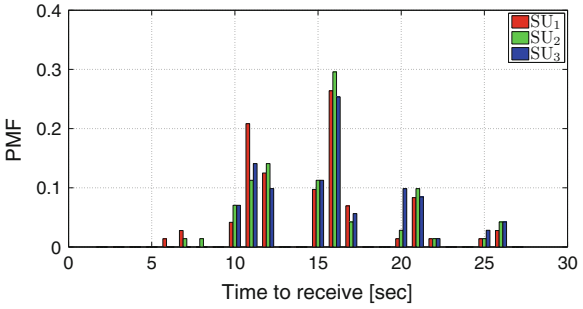


Fig. 4.19 PMF of the archive reception time in Scenario 2

4.5.4 Tests in a Mobile Scenario

In the third phase of our field tests, we considered a more realistic scenario (denoted as Scenario 3, in the following), where the smartphones are positioned within vehicles moving along a predetermined path. The experimental setup is described in Sect. 4.5.4.1, while the obtained results are discussed in Sect. 4.5.4.2.

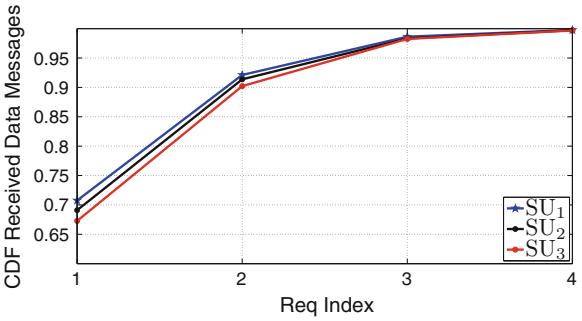


Fig. 4.20 CDF of received *DATA* messages before *REQUEST* messages in Scenario 2

4.5.4.1 Scenario Description

We consider a single PU and four smartphones acting as SUs. The PU is placed in a given car, while the remaining smartphones (SU_1, SU_2, SU_3 and SU_4) are positioned in distinct vehicles. The vehicles move assuming a platoon configuration led by the PU. For the whole duration of the experiment, the vehicle with SU_1 remains behind the car containing the PU, while the other vehicles periodically exchange their positions. During the test, the smartphones are held by the driver or positioned within the cockpit (e.g., over the seats or over the dashboard). The five vehicles involved in the tests complete 20 laps of the circuit shown in Fig. 4.21, designed within the campus of the University of Parma. The circuit path length is 1.72 km, and the vehicles completed the test with an average speed of roughly 40 km/h (11.1 m/s) and maintaining an inter-vehicle distance within the interval, [10–30] m.[1] Since the test has been conducted on a public road, sporadically, other cars have disturbed the platoon configuration. At every lap, the PU sends two distinct archives: one at the beginning of the lap and the second at the middle of the lap. Therefore, the results shown here have been obtained averaging over 40 transmission acts conducted in 20 laps of the circuit.

4.5.4.2 Experimental Results

In Fig. 4.22, the PMF of the hop index, at which messages are received, is shown. From the results in the figure, it can be observed that the "average" network topology emerges from the test. In particular, it can be observed that for all the SUs, 95 % of the fragments are received within the first two communication hops. In other words, in the majority of the cases, the SUs have been in direct visibility (or at least two hops away) from the PU. Therefore, we expect a performance comparable to that observed in Scenario 1.

[1] The average speeds and the inter-node distances are obtained by means of GPS.

Fig. 4.21 Circuit followed by the vehicles during the tests. Image taken from Open Street Map

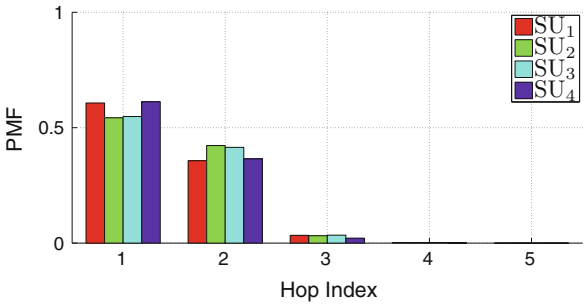


Fig. 4.22 PMF of the hop index at which *DATA* messages are received in Scenario 3

From the point of view of the number of missing packets after the first transmission of the PU, from Fig. 4.23, it can be observed that the performance is comparable with that obtained by SU_2 and SU_3 in Scenario 1. A remarkable exception is represented by the node, SU_1, which exhibits excellent performance in Scenario 3.

Similarly, also from Fig. 4.24, it can be observed that the behavior, in terms of requested messages and satisfied messages, is comparable to that observed in Scenario 1. However, there is a significant difference with respect to Scenario 1. In the latter, the larger fraction of requests is satisfied by the PU, while in Scenario 3, the SUs play a more important role.

By observing Fig. 4.25, it is possible to get some interesting insights, concerning the main differences between Scenario 1 and Scenario 3. First of all, it can be observed

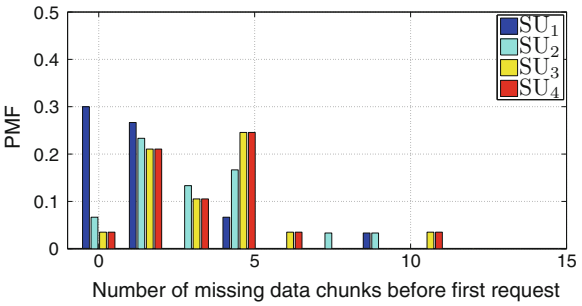


Fig. 4.23 PMF of the number of missing *DATA* fragments before the transmission of the first *REQUEST* message in Scenario 3

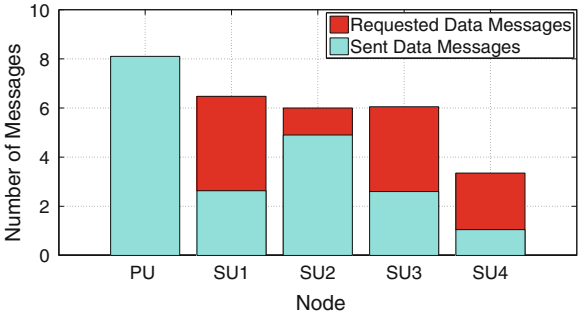


Fig. 4.24 Number of requested *DATA* messages and number of replica *DATA* messages sent upon the reception of a request, in Scenario 3

that the archive is received only after a first round of *REQUEST* messages and subsequent retransmissions. Moreover, a network bimodal behavior can be observed:

- the archive is received in a very short time (after 1 or 2 *REQUEST* message rounds)—this happens when the network conditions are good and stationary;
- the archive is received after a large number of rounds (5 or 6)—this happens when the network homogeneity disappears because of phenomena that break the network topology, such as the presence of other vehicles fragmenting the platoon or an excessive inter-vehicle distance.

Finally, in Fig. 4.26, the CDF of the received *DATA* messages, before a *REQUEST*, is shown. The obtained results justify the conclusions reached in the previous paragraph (relative to the results in Fig. 4.25). In fact, it can be observed that a small number of rounds is often sufficient to collect the majority of fragments. However, a significantly longer time is required for completing the reception of the entire archive.

To summarize, the obtained results show that the performance on a realistic scenario, even if simplified by keeping an almost constant vehicle speed and a limited inter-vehicle distance, has been shown to be similar to the one obtained in Scenario 1 and even better than the one obtained in Scenario 2.

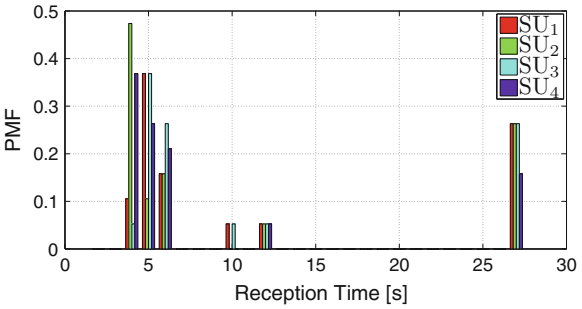


Fig. 4.25 PMF of the archive reception time in Scenario 3

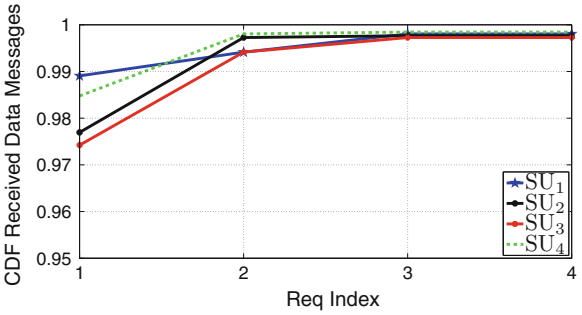


Fig. 4.26 CDF of the received *DATA* messages before a *REQUEST* in Scenario 3

4.5.5 Discussion

The results obtained from the experimental campaign have confirmed the good performance exhibited by the data dissemination protocol adopted in X-NETAD. In fact, it has been shown that a small number of rounds of requests (at most five) is sufficient to disseminate the whole information archive to all network nodes. This happens also in the case of unfavorable network conditions (as in the case of Scenario 2), where SUs are sparse and distant from the PU (up to three hops away). Remarkably, these results have been obtained while limiting the overall number of transmissions in the network.

The core retransmission mechanism of X-NETAD is the IF broadcast protocol, which is designed to reduce the channel congestion and, thus, scales well with the number of nodes, as shown in [7]. The presented results have been obtained in scenarios with a small number of nodes, where the impairments of the wireless channel are the principal cause of the packet losses, while the losses due to channel congestion are negligible. However, we expect that the overall performance of the X-NETAD protocol should scale well with the number of nodes, owing to the equivalent property of the IF protocol.

Finally, we comment on a comparison between the performance of the X-NETAD protocol with that of other dissemination protocols. Even if we do not have conducted experimental campaigns using alternative protocols, it is possible to get some insights from some of our previous works [8]. In particular, according to the observations in Sect. 4.1, the IF protocol exhibits better performance than the flooding protocol, especially under heavily loaded networks. Unfortunately, as our testbed is composed of a small number of smartphones, we were not able to evaluate scenarios with high spatial densities. Therefore, in our specific experimental scenario, it is reasonable to expect that a flooding-based dissemination strategy will exhibit a performance similar to that of IF.

References

1. X-Netad, E.P.: http://www.eurekanetwork.org/project/-/id/6252
2. Ferrari, G., Busanelli, S., Iotti, N., Kaplan, Y.: Cross-network information dissemination in VANETs. In: Proceedings of IEEE International Conference on ITS Telecommunications (ITST), Saint-Petersburg, Russia (2011)
3. Avni, O.: Performance and limitations of cellular-based traffic monitoring technologies: based on case study of benchmarking cellint's trafficsense with road sensors. In: Institute of Transportation Engineers (ITE) Technical Conference and Exhibit, San Diego, CA, USA (2007)
4. Calabrese, F., Colonna, M., Lovisolo, P., Parata, D., Ratti, C.: Real-time urban monitoring using cell phones: a case study in rome. IEEE Trans. Intell. Transp. Syst. **12**(1), 141–151 (2011)
5. Rybicki, J., Scheuermann, B., Kiess, W., Lochert, C., Fallahi, P., Mauve, M.: Challenge: peers on wheels—a road to new traffic information systems. In: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (2007)
6. Rybicki, J., Scheuermann, B., Koegel, M., Mauve, M.: Peertis: a peer-to-peer traffic information system. In: Proceedings of the Sixth ACM International Workshop on VehiculAr InterNETworking (2009)
7. Busanelli, S., Ferrari, G., Panichpapiboon, S.: Efficient broadcasting in IEEE 802.11 networks through irresponsible forwarding. In: Proceedings IEEE Global Telecommunications Conference (GLOBECOM), Honolulu, HA, USA (2009)
8. Busanelli, S., Ferrari, G., Gruppini, R.: Recursive analytical performance evaluation of broadcast protocols with silencing: application to VANETs. EURASIP J. Wireless Commun. Netw. **2012**(10), 1–21 (2012)
9. Car-to-Car Communication Consortium: http://www.car-2-car.org/
10. Final GeoNet Specification, G.D.D.: (2010). http://www.geonet-project.eu/
11. Baiocchi, A., Cuomo, F.: Infotainment services based on push-mode dissemination in an integrated vanet and 3g architecture. J. Commun. Netw. **15**(2), 179–190 (2013). doi:10.1109/JCN.2013.000031
12. Papadimitratos, P., Buttyan, L., Holczer, T., Schoch, E., Freudiger, J., Raya, M., Ma, Z., Kargl, F., Kung, A., Hubaux, J.: Secure vehicular communication systems: design and architecture. IEEE Commun. Mag. **46**(11), 100–109 (2008)
13. Insitute of Electrical and Electronics Engineers.: IEEE Std 802.11TM-2007. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications (2012)
14. Rappaport, T.S.: Wireless Communications. Principles & Practice, 2nd edn. Prentice-Hall, Upper Saddle River (2002)

Chapter 5
Novel Distributed Algorithms for Intelligent Transportation Systems

5.1 Introduction

While hybrid communication schemes, i.e., combining V2V with V2I capabilities, would inherently provide the most effective and robust solution, we remark that purely infrastructure-based communication does not limit the application level of data dissemination and processing to a specific centralized architecture. In fact, a V2I infrastructure does not necessarily imply a centralized organization, which would inevitably lead to scalability issues—for example, to cope with the information requirements of thousands or millions of vehicles moving around in a large metropolitan area. While multiple distributed (e.g., hierarchical) subsystems can be deployed to achieve better scalability, a completely decentralized peer-to-peer (P2P) approach is more appealing. Initially exploited within V2V schemes [1], P2P approaches have been recently followed also for implementing decentralized TIS [2]. In fact, P2P strategies allow responsibility decentralization, as well as computational and communication load balancing, which can be beneficial for smartphone-based Vehicular Sensor Networks (VSNs) [3].

In the context of P2P TISs, we have implemented the D4V architecture, based on opportunistic mechanisms for the dissemination of data generated by vehicle sensors and drivers. D4V requires no dedicated hardware and leverages upon COTS and worldwide available devices (such as smartphones), rather than dedicated devices. To the best of our knowledge, D4V is the only TIS providing, at the same time, massive scalability (because of its P2P nature), deployability (because of the light hardware requirements), and message configurability. D4V is based on a P2P overlay scheme denoted as Distributed Geographical Table (DGT) [4, 5]—indeed, D4V stands for *DGT for VSNs*. The DGT overlay scheme represents a scalable and robust infrastructure for application-level services, and relies on the unification of the concepts of geographical and virtual neighborhoods [4]. In the following chapters, we first illustrate the DGT, with a detailed performance evaluation. Then, we present the D4V.

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7_5

5.2 Distributed Geographic Table

A *structured* decentralized P2P overlay is characterized by a controlled overlay, shaped in a way that resources (or resource advertisements) are placed at appropriate locations [6]. Moreover, a globally consistent protocol ensures that any node can efficiently route a search to some peer that has the desired resource, even if the resource is extremely rare. Beyond basic routing correctness, two important topology constraints guaranteeing that (i) the maximum number of hops in any route (route length) is small, so that requests complete quickly, and (ii) the maximum number of neighbors of any node (maximum node degree) is small, so that maintenance overhead is not excessive. Of course, having shorter routes requires higher maximum degree.

The Distributed Geographic Table (DGT) is a structured overlay scheme where each participant can efficiently retrieve node or resource information (data or services) located near any chosen geographic position. In such a system, the responsibility for maintaining information about the position of active peers is distributed among nodes, for which a change in the set of participants causes a minimal amount of disruption.

The DGT is different from others P2P-base localization systems, where geographic information is routed, stored and retrieved among nodes, that are organized according to a structured overlay scheme. The DGT idea is to build up the overlay directly taking in account the geographic position of node and information allowing to build a network where overlay neighbors are also geographic neighbors and no additional messages are needed to obtain the closest neighborhood of a peer (Fig. 5.1). In the following sections, we present the DGT approach, using the P2P system notation introduced by Aberer et al. [7].

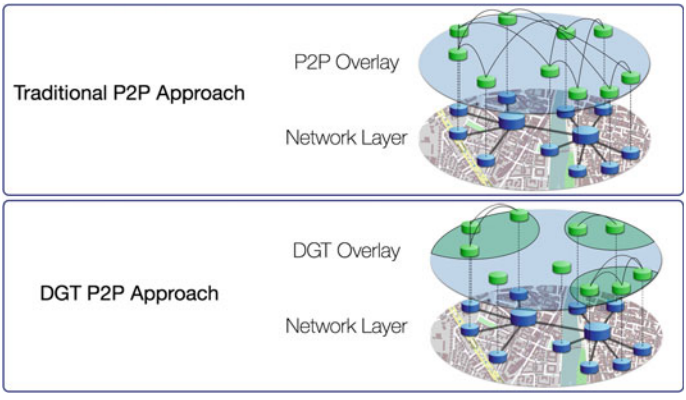


Fig. 5.1 Comparison between traditional P2P approaches and the distributed geographic table

5.3 Conceptual Framework

In a generic DGT overlay, the set of peers is called $\mathscr{P}$, each peer being characterized a unique $id \in \mathscr{I}$ (where $\mathscr{I}$ is the space of identifiers). The association between a peer and an identifier is established by a function $F_p : \mathscr{P} \rightarrow \mathscr{I}$.

The space of world's coordinates is called $\mathscr{W}$ and $w \in \mathscr{W}$, $w = \langle latitude, longitude \rangle$ is the generic location. Thus, a peer $p \in \mathscr{P}$ may be identified by the pair $\langle id_p, w_p \rangle$, where $id_p \in \mathscr{I}$ and $w_p \in \mathscr{W}$.

In a DGT, the distance between two nodes is defined as the actual geographic distance between their locations in the world (also known as great-circle distance or orthodromic distance):

$$d : \mathscr{W} \times \mathscr{W} \rightarrow \mathbb{R}. \tag{5.1}$$

The *neighborhood* of a geographic location is the group of nodes located inside a given region surrounding that location (as illustrated in Fig. 5.2). More precisely, given the set of all geographic regions delimited by a closed curve $\mathscr{A}$, the neighborhood is defined as

$$\mathscr{N} : \mathscr{W} \times \mathscr{A} \rightarrow 2^{\mathscr{P}} \tag{5.2}$$

where $2^{\mathscr{P}}$ is the set of all possible connections between peers. In order to evaluate the neighborhood of a target geographic point $t \in \mathscr{W}$ using a region $A_t \in \mathscr{A}$ centered in t, let us define:

$$\mathscr{N} = \{p \in \mathscr{P} | w_p \in A_t\} \quad t \in \mathscr{W}, A_t \in \mathscr{A} \tag{5.3}$$

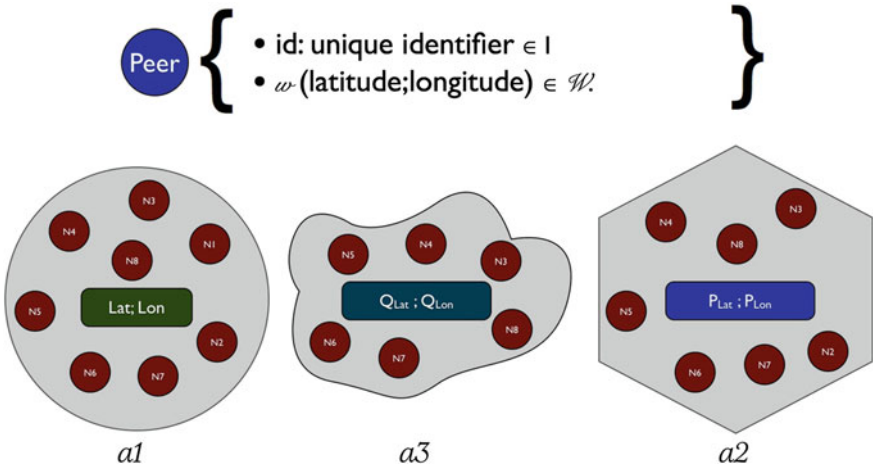


Fig. 5.2 The DGT routing strategy used to build and maintain a local or a remote neighborhood

where, as earlier, w_p is the geographic position of peer $p \in \mathscr{P}$. By selecting, for example, a circular region $C_t \in \mathscr{A}$, with a radius $c_r \in \mathbb{R}^+$, it is quite simple to evaluate the node's neighborhood. In fact:

$$C_t = \{w \in \mathscr{W} | d(w,t) \leq c_r\} \quad t \in \mathscr{W} \tag{5.4}$$

$$\mathscr{N} = \{p \in \mathscr{P} | w_p \in C_t\} \quad t \in \mathscr{W}, a \in \mathscr{A}. \tag{5.5}$$

If p is moving, then $\mathscr{N}$ dynamically changes accordingly.

5.3.1 Routing Strategy

The main service provided by the DGT overlay is the routing of requests for finding available peers in a specific area, i.e., to determine the neighborhood of a generic global position $w \in \mathscr{W}$.

Routing is a distributed process implemented as asynchronous message passing. By executing the $route(p, w, a)$ operation, a peer forwards to another peer $p \in \mathscr{P}$ a request for the list of nodes that peer p knows to be located in the region $a \in \mathscr{A}$, whose center is $w \in \mathscr{W}$. Thus, a routing strategy can be described by a possibly non-deterministic function:

$$\mathscr{R} : \mathscr{P} \times \mathscr{W} \times \mathscr{A} \rightarrow 2^{\mathscr{P}} \tag{5.6}$$

that returns the neighborhood $\mathscr{N}(w, a)$, around the geographic position w and within region a, known by peer p.

The routing process is based on the evaluation of the region of interest centered in the target position (local or remote). The idea, depicted in Fig. 5.3, is that each

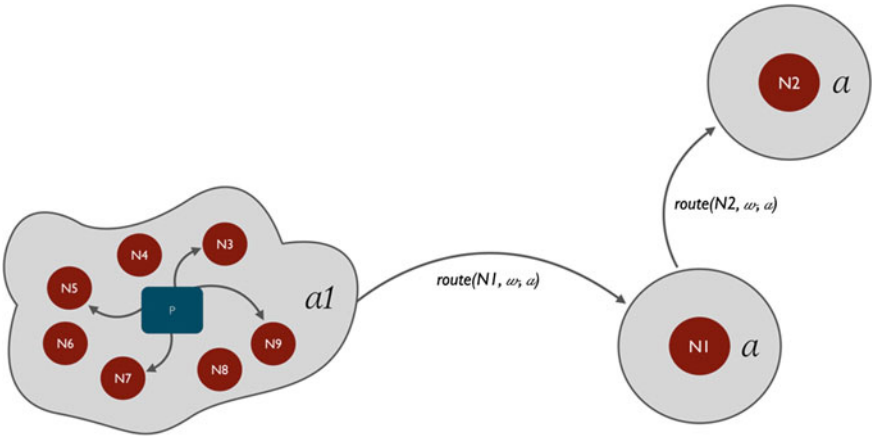


Fig. 5.3 The DGT routing strategy used to build and maintain a local or a remote neighborhood

peer involved in the routing process selects, among its known neighbors, those that presumably know a large number of peers located inside or close to the chosen area centered in the target point. If a contacted node cannot find a match for the request, it does return a list of closest nodes, taken from its routing table. This procedure can be used both to maintain the peer's local neighborhood $\mathcal{N}$ and to find available nodes close to a generic target.

Regarding the local neighborhood, the general aim of the approach is to have accurate knowledge of nodes that are close to the peer and of a gradually reduced number of known nodes that will be used to forward long range geographic queries. This idea recalls Granovetter's theory of weak ties [8], stating that human society is formed by small complete graphs whose nodes are strongly connected (friends, colleagues, etc.). These clusters are weakly connected between each other, e.g., a member of a group superficially knows a member of another group. The most important fact is that weak ties are those which make human society an *egalitarian small world network*, i.e., a giant cluster with small separation degree and no hubs.

Whenever a single active node in the system wants to contact other peers in its area (e.g., to provide or search for a service), it does not need to route additional and specific discovery messages to its neighbors (or to a supernode responsible for a specific zone) in order to find peers that are geographically close. Instead, it simply reads its neighbor list, that is proactively filled with "geographic neighbors".

Our peer neighborhood construction protocol has been inspired by Kademlia [9], used, for example, in recent versions of the eMule client (as an alternative to the traditional eDonkey protocol) [10]. Many of Kademlia's benefits result from its use of the XOR metric for distance between points in the key space. XOR is symmetric, allowing Kademlia participants to receive lookup queries from precisely the same distribution of nodes contained in their routing tables, that are organized as sets of "k-buckets". Every k-bucket is a list having up to k entries: in other words, each node in the network has lists containing up to k nodes, each list being associated to a given distance from the node itself. To locate nodes near a particular ID, Kademlia uses a single routing algorithm from start to finish. In contrast, other systems use one algorithm to get near the target ID and another for the final hops.

Peer neighborhood construction in DGT uses the geographic metric, instead of Kademlia's XOR metric. Each node knows its *global position* (*GP*) retrieved with a GPS system or with other localization technologies, and knows a set of real neighbors organized in a specific structure based on the distance that these nodes have with respect to the node's position.

The main goal of the DGT protocol is to build and maintain an overlay where each node knows all the active nodes that are available in a geographic region, in order to implement and provide specific applications and services. An example of application based on such a protocol may be a city monitoring system that uses decentralized nodes to monitor the traffic status of the city. By using this system, there is no need to deploy powerful servers: light peers can be activated in strategic locations, in order to cover the whole city area. Each of them can analyze its region of interest, monitor traffic conditions in real-time, and evaluate the position of peers in order to inform them about accidents and traffic jams, suggesting alternative paths.

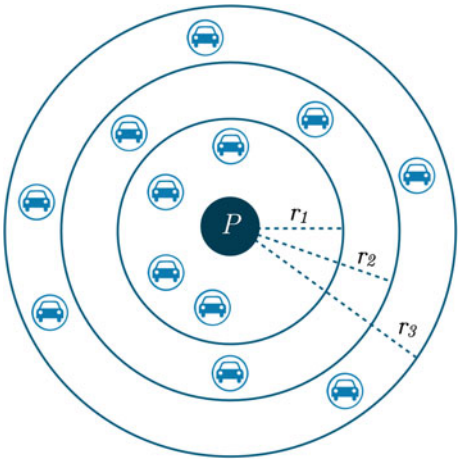


Fig. 5.4 GeoBucket data structure

5.3.2 Data Structure

Every peer maintains a set of *GeoBuckets* (GBs), each one being a (regularly updated) list of known peers sorted by their distance from the GP of the peer itself (Fig. 5.4). GBs can be represented as K concentric circles, with increasing (application-specific) radii $\{R_i\}_{i=1}^{K}$ and thickness $\{r_i\}_{i=1}^{K}$, with $R_i = \sum_{j=1}^{i} r_j$. If there is a known node whose distance from the peer is larger than the radius of the outmost circle R_K, it is inserted in another list that contains the nodes outside the circle model.

Each peer in the GB set is characterized by:

- *Unique ID*—univocally identifies the peer within the DGT;
- *Global Position* (*GP*)—latitude and longitude retrieved with a GPS system or with other systems (e.g., GSM cell-based localization);
- *IP Address*—allowing to identify the node in Internet—if the peer is behind NAT, the IP address may be that of a relay;
- *UDP Port*—on which the peer listens, waiting for connection attempts;
- *Number of known nodes*—used to compare two nodes that have the same distance.

Moreover, each peer has a set of message types on which it is interested.

5.3.3 Network Join

The bootstrapping procedure is a common and crucial issue both in small and wide-spread P2P networks and several approaches have been studied and tested during last decade [11]. Essentially the basic bootstrapping process composed by to two phases: find a remote peer, and connecting to it in order to receive a first group of nodes already active in the overlay. In a DGT overlay the idea is to use two different approaches (depicted in Fig. 5.5) according to the status of the nodes and the number

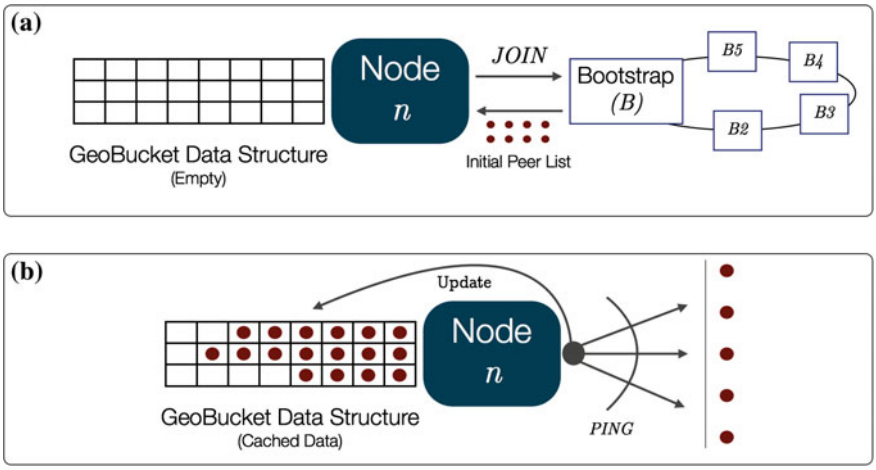


Fig. 5.5 Schematic representation of a single step of the DGT Lookup procedure

of known nodes or cached data. When a new node (First case in Fig. 5.5) wants to access the DGT overlay, it sends a join request, together with its GP, to a *bootstrap node*, that returns a list of references to peers that are geographically close to the joining one. It is important to emphasize that this information is not updated: referenced peers may have moved away from their initial locations. It is up to the joining peer to check for the availability of listed peers. This operation is performed not only during the first join of the peer, but also when the peer finds itself to be almost or completely isolated. In these situations (that typically arise when peers enter low density areas), the node may send a new join request to the bootstrap node, in order to obtain a list of recently connected peers that may become neighbors. The second case is associated to a nodes that wants to re-join the DGT overlay after an offline period (case b in Fig. 5.5). In this situation it is highly reasonable that the node already has some cached data about its last GBs. In order to avoid the use of bootstrap node and, according to the delta between the current time the disconnection instant, the node could try to recover its distributed knowledge by pinging cached peer references and refresh its data structure. If after this refreshing procedure the reconnecting node is not able to use its information it will re-join using the bootstrap support cleaning its history.

5.3.4 Peer Lookup

The main procedure used during peer discovery is *FIND_NODES*(*GP*), that returns the β peers that are nearest to the specified GP. Peer n keeps up-to-date its neighborhood awareness by periodically applying *FIND_NODES*( ) to its global position own GP_n. Such a procedure (with any target GP) may also be executed upon request from another peer.

Algorithm 1 Periodic Lookup Algorithm

$i \leftarrow 0$
get α nodes from geo-buckets (nearest to GP): $C_i = \{n_{1i}, .., n_{\alpha i}\}$
repeat
  $j \leftarrow 1$
  while $j \leq \alpha$ **do**
    if n_{ji} not yet queried **then**
      n_{ji}.*FIND_NODES(GP)*
    end if
    $j \leftarrow j + 1$
  end while
  get α nodes (nearest to GP) from the $\alpha\beta$ results: C_{i+1}
  $i \leftarrow i + 1$
until $C_{i+1} == C_i$
$f \leftarrow i$
get K nodes (nearest to GP) from geo-buckets, not already in C_f
$j \leftarrow 1$
while $j \leq K$ **do**
  if n_{ji} not yet queried **then**
    n_{ji}.*FIND_NODES(GP)*
  end if
  $j \leftarrow j + 1$
end while

Node n searches in the GB associated to the requested GP. The final objective of the lookup (summarized in Algorithm 1 and schematically depicted in Fig. 5.6) is to find the $\alpha \leq K$ peers that are nearest to the selected GP, including newly connected nodes, as well as mobile peers that have entered the visibility zone. The lookup

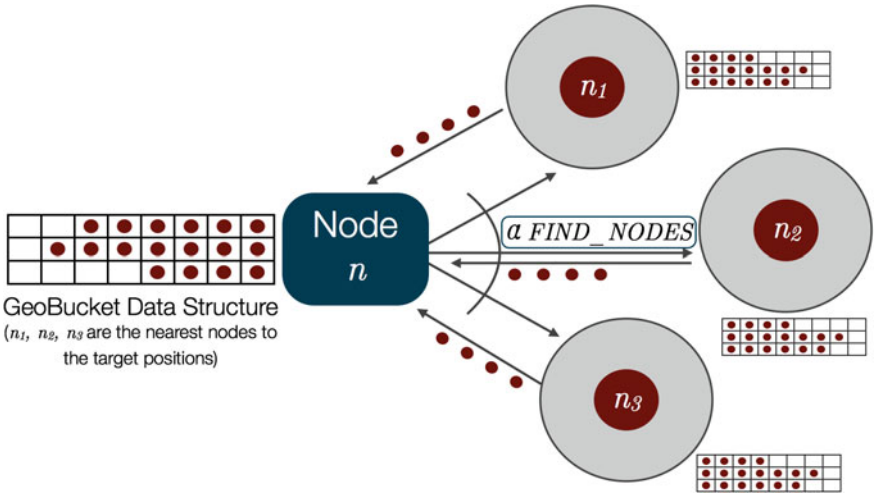


Fig. 5.6 Schematic representation of a single step of the DGT Lookup procedure

initiator starts by picking α nodes from its closest non-empty GB—or, if that bucket has less than α entries, it just takes the α closest nodes, by extending the lookup to all its GBs. Such a peer set is denoted as $\mathscr{C}_i = \{n_{1i}, \ldots, n_{\alpha i}\}_{i=1}^{K}$, where i is an integer index. The initiator sends parallel *FIND_NODES* requests, using its GP as target, to the α peers in $\mathscr{C}_i$. Each questioned peer responds with β references. The initiator sorts the result list according to the distance from the target position, then picks up α peers that it has not yet queried and re-sends the *FIND_NODES* request (with the same target) to them. If a round of *FIND_NODES* fails to return a peer closer than the closest already known, the initiator re-sends the *FIND_NODES* to K closest nodes not already queried. The lookup terminates when the initiator has obtained responses from the K closest nodes, or after f cycles, each cycle resulting with an updated set of nearest neighbors C_i. Thus, the number of sent *FIND_NODES*(*GP*) messages is in the worst case $f \cdot \alpha + K$, that depends on the spatial density of peers in the area of interest. A peer is allowed to run a new lookup procedure only if the previous one is completed, in order to reduce the number of exchanged messages and to avoid the overlapping of the same type of operations.

The general idea is that soon after the bootstrap or when neighbor peers are highly dynamic, the period of the discovery process may be very small and may increase when the knowledge becomes sufficiently stable among active peers. We set a lower and an upper bound for the discovery period, i.e., respectively, $T_{\min}$ and $T_{\max}$.

5.3.5 Position Update

Any active peer in the network can change its geographic position for many reasons (the user may be walking, driving, etc.).

To preserve the consistency of the DGT, each peer needs to periodically schedule a maintenance procedure that compensates network topology changes. The practical usability of a DGT critically depends on the messaging and computational overhead introduced by this maintenance procedure, whose features and frequency of execution are application-dependent.

When an active peer in the network changes its geographic position, it has to send updates of its GP to neighbors, in order to improve the accuracy of their knowledge. To avoid excessive bandwidth consumption, every peer communicates its position update to neighbors only if the displacement is higher than ε (km) (Fig. 5.8). If during this message exchange a peer receives a node's update confirming that the new position is out of its area of interest, the neighbor's reference is removed from the appropriate GB and a *REMOVE* message is sent to the peer (Fig. 5.7).

The DGT allows peers to have accurate knowledge of geographically close neighbors and a limited view of the outer world. However, whenever necessary, and with limited incremental computational and transmission costs, peers are able to find new connected nodes that are entering the target area. The described P2P localization scheme represents maybe the core layer of a vehicular network able to discover and inform drivers that are potentially interested in specific traffic messages or to data acquired by vehicle sensors.

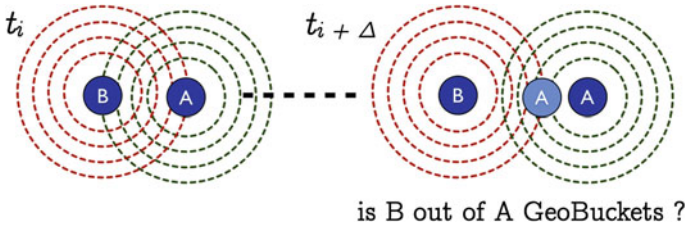


Fig. 5.7 DGT position update

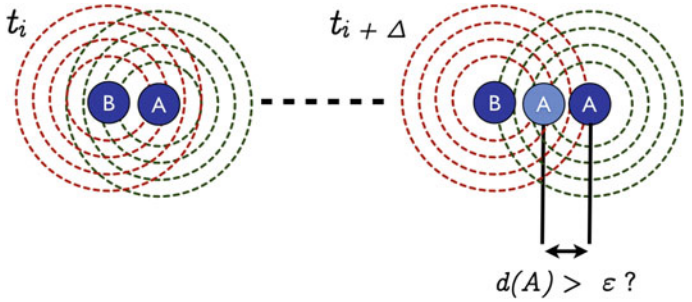


Fig. 5.8 DGT neighbor remove process

5.4 Analytical Model for Performance Evaluation

In this section, we first present an analytical performance evaluation framework of the DGT-based proactive neighbor localization algorithm. Furthermore, we carry out an extensive simulative analysis of our DGT implementation in several scenarios in order to evaluate algorithm performance, parameters influence and robustness.

The performance evaluation of our DGT-based peer neighborhood construction protocol will be carried using the following metrics:

- *PMN*: Percentage of Missing Nodes in the geo-buckets of a peer, with respect to those really present in the area.
- *MR* (msg/s): Message Rate, i.e., the average number of messages received per second by each node.
- *NPE* (km): Node Position Error, i.e., the average distance between a peer's position reference in a GeoBucket and its actual position.

Assuming that N peers are distributed within a square surface with side of length L, the corresponding node spatial density, denoted as ρ, is N/L^2. If nodes are static and uniformly distributed over the square surface, ρ is also the local node spatial density. In the presence of node mobility, the node distribution is likely to be non-uniform: the corresponding node spatial density can be heuristically estimated as $\delta N/L^2$, where $\delta \in \mathbb{R}^+$ is a compensation factor which takes into account the fact that the nodes could be locally denser ($\delta > 1$) or sparser ($\delta < 1$) than the average value ρ.

At a specific time, a peer wants to identify available geographic neighbors within a circular region of interest with radius r. This region, centered at the peer, is denoted as R and its area is $A = \pi R^2$. In general, within the region of interest of a node there are two classes of neighbors: detectable (i.e., nodes which can be detected by one or more nodes) and non-detectable (i.e., nodes which cannot be detected by any node).

Assuming that peers are distributed according to a two-dimensional Poisson distribution[1] with parameter ρ, the average number of nodes in the region R is $\overline{N}_{\text{tot}}^{(R)} = \rho \cdot A$. Let us denote by $x \in (0, 1)$ the percentage of non-detectable nodes in the region R (i.e., there are, on average, $x \cdot \overline{N}_{\text{tot}}^{(R)}$ non-detectable peers). Assuming further that the number of detectable peers in R has a Poisson distribution with parameter $\rho \cdot A \cdot (1-x)$, it follows that their average value is $\overline{N}_{D}^{(R)} = \rho \cdot A \cdot (1 - x)$.

As described in Sect. 5.3, during each step of the discovery procedure a peer picks the closest α known neighbors (if available) and sends them simultaneous `FIND_NODES` requests centered in its geographic location. The goal of the interrogating peer is to retrieve detectable nodes in its area of interest. If, at the end of an iteration, no new node is retrieved, the discovery process ends and will be rescheduled according to a specific strategy.

In order to evaluate the number of discovered peers at each discovery iteration (without counting the same node more than once), the α `FIND_NODES` requests, scheduled at each discovery step, must be taken into account considering not only the single intersection between two peers but the multiple overlapped regions between the α contacted nodes. In Fig. 5.9, an illustrative scenario with $\alpha = 3$ overlapping circular areas is shown.

Since the intersection of α circular regions can be highly varying (depending on their relative positions), we simplify the analysis assuming that adjacent contacted peers are spaced by an angle $2\pi/\alpha$ and are positioned in the center of the corresponding radius of the circular region of interest of the reference peer. We denote as A_j the sum of the areas of the intersection region shared only by the requesting peer and j contacted peers. In Fig. 5.9, the areas $\{A_1, A_2, A_3\}$ are indicated. These areas will be computed using the Circles Intersection library for MATLAB [2]. Explicit expressions (not shown here for the sake of conciseness) can be derived by means of the analytical method proposed by Fewell [12].

Under the above assumptions, the average number of new peers discovered after s steps can be written as

$$\overline{n}(s) = \begin{cases} 0 & s = 0 \\ l_0 & s = 1 \\ \overline{n}(s-1) + \sum_{j=1}^{\alpha} \overline{d}_j \, (\overline{n}(s-1)) & s \geq 2 \end{cases} \tag{5.7}$$

[1] This is an approximation. In fact, owing to node mobility, the local distribution is likely to be not Poisson. However, as we will consider only average values, the Poisson approximation will shown to be accurate.

[2] http://www.mathworks.com/matlabcentral/fileexchange/5313.

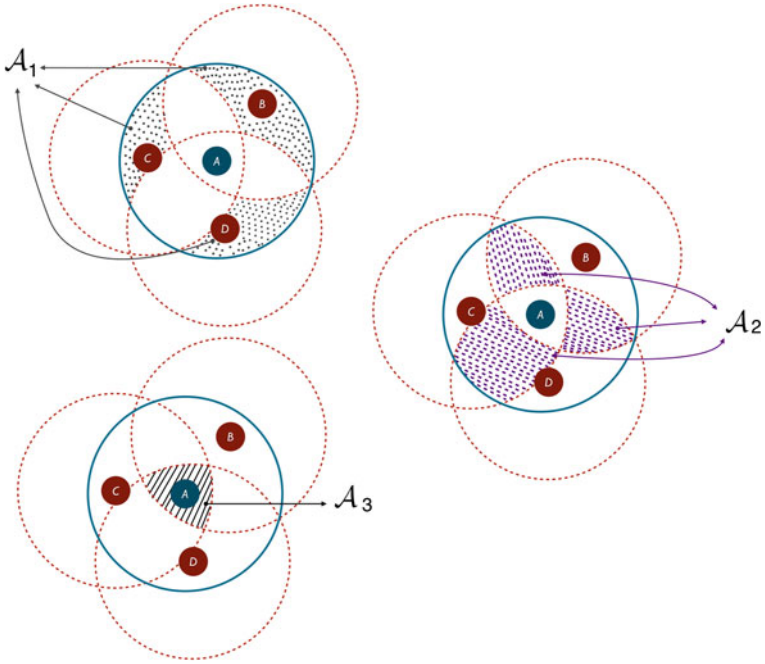


Fig. 5.9 Intersection regions (with corresponding areas $\{A_j\}$) between $\alpha = 3$ overlapping circular areas of interest

where l_0 is the initial size of the peer list; $\overline{n}(1)$ is the average number of initial peers (transferred to the peer of interest); $\overline{n}(s-1)$ is the number of new peers discovered up to the $(s-1)$-th step ($s \geq 2$); $\overline{d}_j$ represents the average number of new peers discovered in the region of area $\mathscr{A}_j$ and can be expressed as follows:

$$\overline{d}_j\left(\overline{n}(s-1)\right) = \rho \cdot A_j \cdot (1-x) \cdot b_j\left(\overline{n}(s-1)\right). \tag{5.8}$$

In (5.9), $b_j\left(\overline{n}(s-1)\right)$ is a heuristic function used to model the number of replicas obtained in the j-th intersection between the applicant's region of interest and the regions of interest of the queried peers. This parameter depends on (i) the number of nodes that share the same zone (i.e., j) and can answer with the same peer references and (ii) the the average number of known nodes at step $s-1$—in fact, the number of known nodes at each step needs to be taken into account to evaluate potential replicas. Taking into account the fact that if a node has knowledge of its neighbors, the probability of discovering an already known peer is higher, the following heuristic expression for b_j allows to derive accurate performance results:

$$b_j\left(\overline{n}(s-1)\right) = \left[1 - \frac{\overline{n}(s-1)}{\overline{N}_D^{(R)}}\right]^j. \tag{5.9}$$

Finally, the average number of newly discovered nodes up to step s can be expressed as follows:

$$\overline{n}(s) = \overline{n}(s-1) + \sum_{j=1}^{\alpha} \rho \cdot A_j \cdot (1-x) \cdot \left[1 - \frac{\overline{n}(s-1)}{\overline{N}_D^{(R)}}\right]^j. \tag{5.10}$$

Note that the recursive analytical computation of $\{\overline{n}(s)\}$ stops when a pre-set peer discovery limiting number is reached.

In Fig. 5.10, the performance results predicted by the analytical model proposed above are compared with simulation results (obtained by means of the DEUS simulation platform, described in Appendix A), considering scenarios with (a) 500 peers

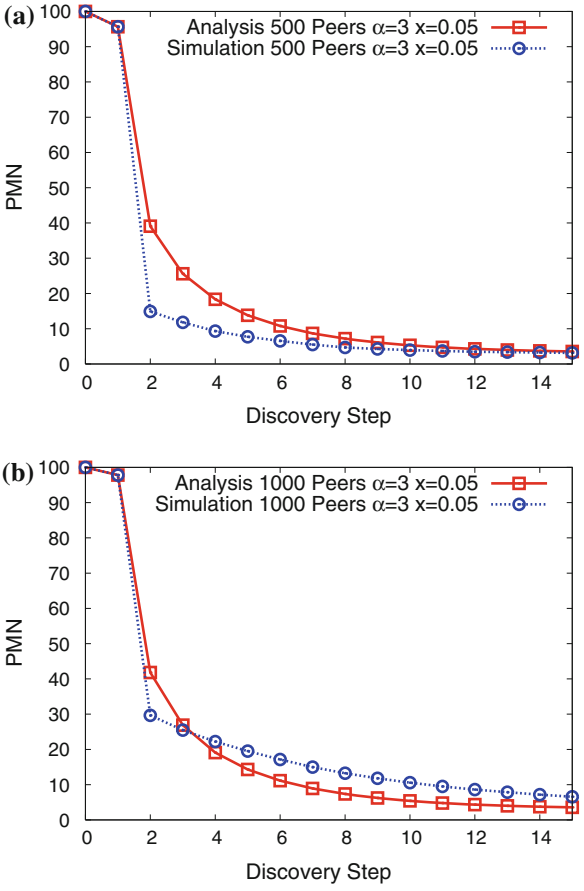


Fig. 5.10 PMN as a function of the discovery step, considering **a** 500 peers and **b** 1,000 peers. In all cases, $\alpha = 3$

and (b) 1,000 peers. In both cases (a) and (b), peers are distributed within a square surface with side of length $L = 6.53$ km, with an initial peer list size with $\overline{n}(1) = 10$ peers, a discovery limiting number of 100, and $x = 0.05$. It can be observed that analytical performance results are very close to simulation results, so that we can conclude that the accuracy of the analytical framework is satisfactory.

In order to investigate the impact of α, in Fig. 5.11 the PMN is shown, as a function of the discovery step, considering (a) $\alpha = 1$ and (b) $\alpha = 2$. In both cases, the number of active peers is set to 200. It can be observed that the agreement between simulations and analysis is even stronger than in Fig. 5.10. By observing the results in Figs. 5.10 and 5.11, it can be concluded that a small number of discovery steps is sufficient, regardless of the value of α, to significantly reduce the PMN.

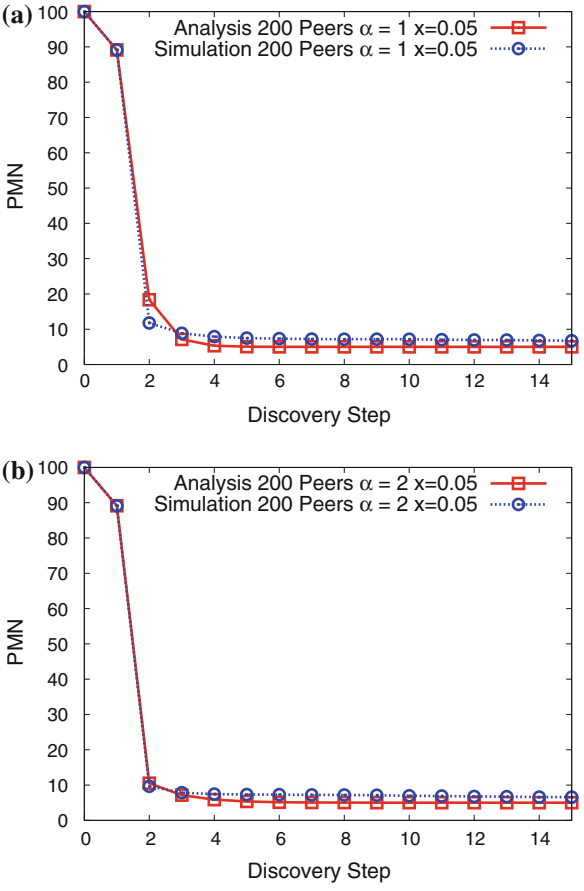


Fig. 5.11 PMN as a function of the discovery step, considering **a** $\alpha = 1$ and **b** $\alpha = 2$. In all cases, 200 peers

5.5 DGT and Mobility

Mobility models represent the movement of mobile users, and how their location, velocity and acceleration change over time. Such models are frequently used for simulation purposes when new communication or navigation techniques are investigated. Mobility management schemes for mobile communication systems make use of mobility models for predicting future user positions.

The mobility model is one of the fundamental elements in the performance evaluation of simulated network with mobile users such as V2V and V2I applications aiming at realistic mobility patterns.

We focus on the study of vehicular mobility models in order to evaluate the DGT approach in a dynamic scenario such as a Smart City. In this context multiple vehicles and user are moving at the same time querying about a location of interest and generating location based information or data such as traffic related messages or sensed data along the streets.

Figure 5.12 illustrates five major mobility model categories, which were defined by Hartenstein [13]:

- *Random Models*: Vehicular mobility is considered random and mobility parameters, such as speed, heading and destination are sampled from random processes. A very limited interaction between vehicles is considered in this category.
- *Flow Models*: Single and multi-line mobility models based on flow theory are considered from a microscopic or macroscopic point of view. The literature considers the following threes different classes for flow models:

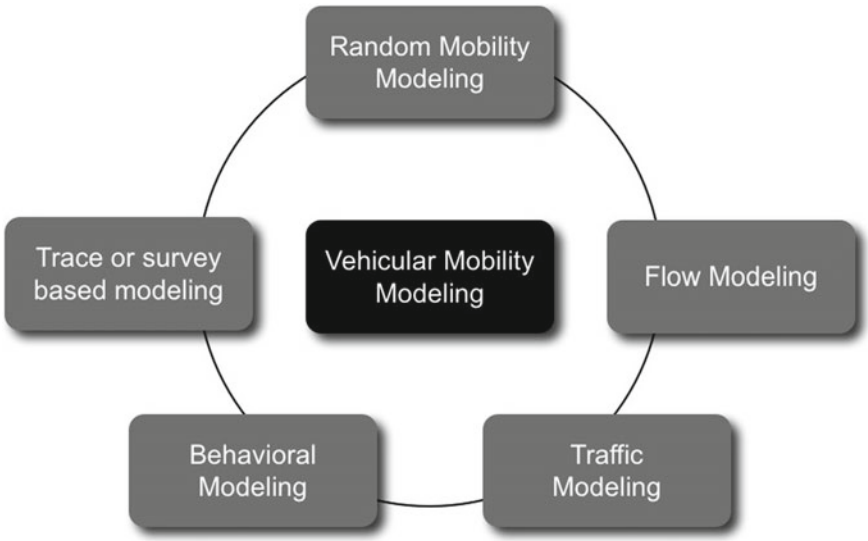


Fig. 5.12 Classification of vehicular mobility modeling approaches

  - *Microscopic*: describe the mobility parameters of a specific car with respect to other cars in detail. Usually takes in account acceleration/deceleration, safe distance, reaction time or safe speed. Its high level of precision is reflected by high computational complexity.
  - *Macroscopic*: don't consider the mobility parameters of a specific car, but instead quantity of macroscopic meaning such as flow, speed or density are modeled
  - *Mesoscopic*: describes traffic flows at an intermediate level of detail. Individual parameters can be modeled yet of a macroscopic meaning. The aim is to benefit from the scalability of the macroscopic approach but still providing a detailed modeling close to microscopic models.

- *Traffic Models*: Trip and path models are described in this category, where each car has an individual trip or a path, or a flow of cars is assigned to trips or paths.
- *Behavioral Models*: They are not based on predefined rules but instead dynamically adapt to a particular situation by mimicking human behaviors, such as social aspects, dynamic learning, or following AI concepts.
- *Trace-Based Models*: Mobility traces may also be used in order to extract motion patterns and either create or calibrate models.

According to the concept map in Fig. 5.12, and to the ideas of Harri et al. [14, 15], mobility models intended to generate realistic vehicular motion patterns should include the following features (Fig. 5.13).

- *Accurate and realistic topological maps*: street topologies should manage different densities of roads, should contain multiple lanes, different categories of streets and associated speed limitations.
- *Obstacles*: obstacles should be intended as both constraints to car mobility and hurdles to wireless communications.

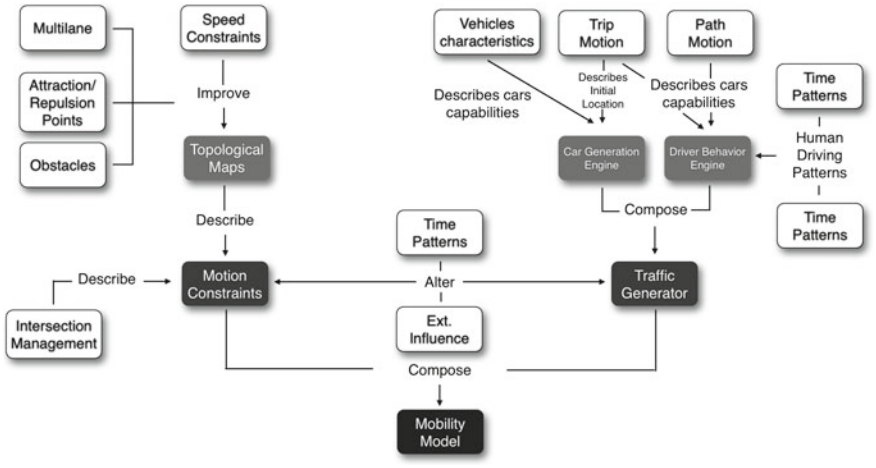


Fig. 5.13 Concept map of for the design of vehicular mobility models

- *Attraction/repulsion points*: initial and final destinations of road trips are not random. Most of the time, many drivers are driving toward similar final destinations or attraction points, or from similar initial locations or repulsion points, typically creating bottlenecks.
- *Vehicles characteristics*: each category of vehicle has its own characteristics, which has an impact on a set of traffic parameters. For example, macroscopically speaking, some urban streets and highways are prohibited to trucks depending on the time of the day. Microscopically speaking, acceleration, deceleration, and speed capabilities of cars and trucks are different. The accounting of these characteristics alters the traffic generator engine when modeling realistic vehicular motion.
- *Trip motion*: a trip is macroscopically seen as a set of source and destination points in the urban area. Different drivers may have diverse interests which affect their trip selection.
- *Path motion*: a path is macroscopically seen as the set of road segments taken by a car on its travel between an initial and a destination point. As in real life, drivers do not randomly choose the next heading when reaching an intersection as is the case in most vehicular networking traffic simulations. Instead, they choose their paths according to a set of constraints such as speed limitations, time of the day, road congestion, distance, and even the driver's own habits.
- *Smooth deceleration and acceleration*: vehicles do not abruptly break and move deceleration and acceleration models should be considered.
- *Human driving patterns*: drivers interact with their environments, not only with respect to static obstacles but also to dynamic obstacles, such as neighboring cars and pedestrians. Accordingly, the mobility model should control vehicles mutual interactions such as overtaking, traffic jams, or preferred paths.
- *Intersection management*: this corresponds to the process of controlling an intersection and may either be modeled as a static obstacle (stop signs), a conditional obstacle (yield sign), or a time dependent obstacle (traffic lights). It is a key part in this framework that however only has an influence on the motion constraint block, as the traffic generator block cannot not see the difference between a stop sign or high density traffic. Both are interpreted as a motion constraint.
- *Time patterns*: traffic density is not identical during the day. A heterogeneous traffic density is always observed at peak times, such as rush hours or during special events.
- *External influence*: some motion patterns cannot be proactively configured by vehicular mobility models as they are externally influenced. This category models the impact of accidents, temporary road works or real-time knowledge of the traffic status on the motion constraints and the traffic generator blocks. Communication systems are the primary source of information about these external influences.

Figure 5.14 and Table 5.1 illustrate the notation usually employed for the formal description of a vehicular mobility model where vehicle i will be considered as the reference vehicle. At time t, $x_i(t)$ and $v_i(t)$ represent respectively the position and speed of vehicle i. Indexes $i+1$ and $i-1$ represent the vehicle immediately in front and behind vehicle i with position $x_{i+1}(t)$ and $x_{i-1}(t)$, and with speed $v_{i+1}(t)$ and $v_{i-1}(t)$. We can additionally consider $\theta_i(t)$ as the heading of a vehicle i at time t.

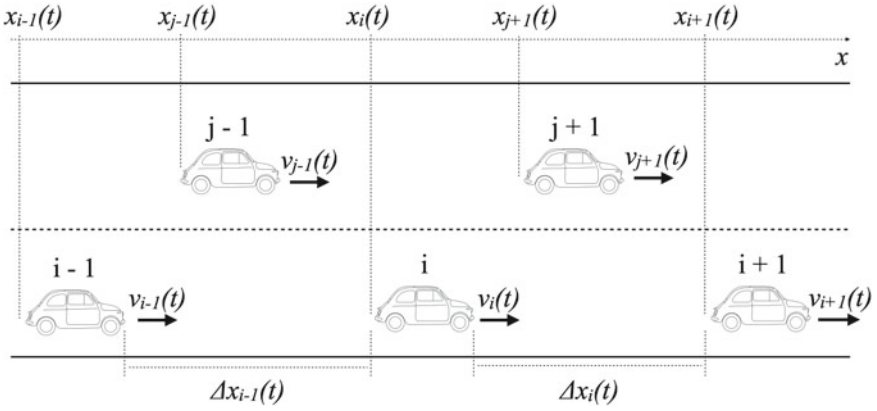


Fig. 5.14 Vehicular mobility model notation

Table 5.1 Vehicular mobility model description

Symbol	Definition	Dimension
Δt	Time step	s
a	Maximum acceleration	m/s^2
b	Maximum deceleration	m/s^2
L	Car length	m
$v^{\min}$	Minimum velocity	m/s
$v^{\max}$	Maximum velocity	m/s
v^{des}	Desired or targeted velocity	m/s
$\theta^{\max}$	Maximum heading	rad
$\theta^{\min}$	Minimum heading	rad
T	Safe time headway	s
Δx^{safe}	Safe distance headway	m
v^{safe}	Safe velocity	m/s
τ	Driver reaction time	s
μ	Stochastic parameter	[0;1]

Our model partially follows the approach of Zhou et al. [16] where the key idea is to use *switch stations* (SSs) connected via virtual tracks to model the dynamics of vehicle and group mobility. For example, our simulative analysis considers a square area around the city of Parma adding 20 SSs inside and outside the city district (see for example Fig. 5.15). Stations are connected to each other through virtual paths that have one lane for every direction, speed limitation associated with the street category and specific road density limit to model vehicle speed in traffic jam conditions. When a new car joins the network, it first associates with a random SS, then it selects a new destination station and starts moving on the connection path between them. This

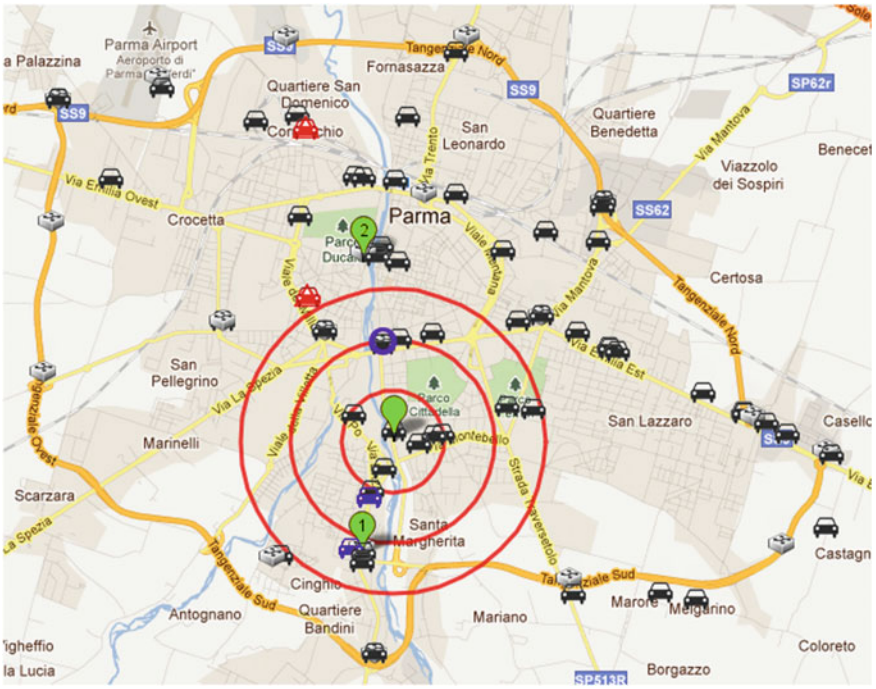


Fig. 5.15 Example of simulated DGT-based VSN (in the city of Parma)

procedure is repeated every time the car reaches a new SS and has to decide its next destination.

Each switch station has an attraction/repulsion value that influences the user's choice for the next destination station. This value may be the same for each path in order to allow for random trip selection.

A set of parameters is associated with each car, thus affecting macroscopic and microscopic aspects of traffic circulation, like street and highway limitations (i.e., some types of vehicles are forbidden on particular paths) as well as acceleration, deceleration, and speed constraints.

We modeled different external events that may happen during the traffic simulation and alter drivers' behavior, such as accidents, temporary road works or bad conditions of road surface like ice, snow or potholes that can be detected by vehicle sensors.

Drivers not only interact with obstacles, but also adapt their behavior according to their knowledge about car surroundings. For example, they may try to change their path if they are informed about a traffic jam or an accident slowing or blocking, and they reduce their speed in proximity of locations characterized by bad surface conditions.

We consider microscopic flow modeling where mobility parameters of a specific car are described with respect to other cars. Several approaches take into account for example the presence of nearby vehicles when modeling the speed of the car

(e.g., FTM [17], Krauss [18], and IDM [19]). In particular, the FTM model has been implemented in our simulator because it is the most accurate for our scenario, with different speed limits for each virtual path, without high computational requirements. FTM describes speed as a monotonically decreasing function of vehicular density, forcing lower values when the traffic congestion reaches a critical point. In our case, the desired speed of a car moving along the points of a path p is computed according to the following equation:

$$v^{\text{des}} = \max\left\{v_{\min}, v^{p}_{\max}\left(1 - \frac{k}{k_{\text{jam}}}\right)\right\} \tag{5.11}$$

where $v_{\min}$ is the minimum car speed (depending on vehicle characteristics), $v^{p}_{\max}$ is the speed limit related to the path, k is the current density of the road, given by n/l (n represents the number of cars on the road and l its length), and k_{jam} is the vehicular density for which a traffic jam is detected. As mentioned before, we also want to model the behavior of a driver in proximity of a road point with bad surface condition. The idea it that a conscientious driver, knowing that along his/her road there is a potential dangerous location, reduces the car speed according to the distance from that point. The safe speed v^{safe} is defined by the following equation:

$$\begin{aligned} v^{\text{safe}} &= \frac{d^2}{k_1} + k_2 \\ k_1 &= \frac{d^2_{\text{limit}}}{v^{\text{des}} - v_{\min}} \\ k_2 &= v_{\min} \end{aligned} \tag{5.12}$$

where d is the distance between the vehicle and path location with bad surface condition, d_{limit} is the limit from which the evaluation of safe speed starts, k_1 and k_2 are two constants depending on v_{des} and $v_{\min}$ that are set to have the desired speed at limit distance and the minimum one near the dangerous location.

5.5.1 Mobility Model with Vertical Handover

Since we want to evaluate the robustness of our DGT-based localization algorithm in a dynamic urban scenario with mobile devices, considering several (possibly overlapping) regions characterized by different types of network coverage (see Fig. 5.16). To this purpose, we use one model to describe the mobility of vehicles, and another model for taking into account vertical handover.

Vertical handover or *vertical handoff* refers to a network node that change the type of connectivity it uses to access a supporting infrastructure, usually to support node mobility. For example, a laptop might be able to use both a high speed wireless LAN and a cellular technology for Internet access. Wireless LAN connections generally provide higher speeds, while cellular technologies generally provide more ubiquitous coverage. Thus the laptop user might want to use a wireless LAN connection whenever one is available, and to 'fall over' to a cellular connection when

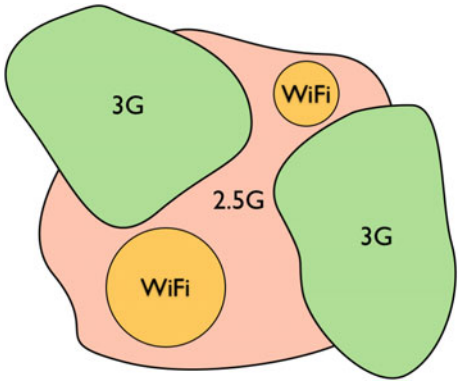


Fig. 5.16 Example of area with different, partially overlapping network coverages

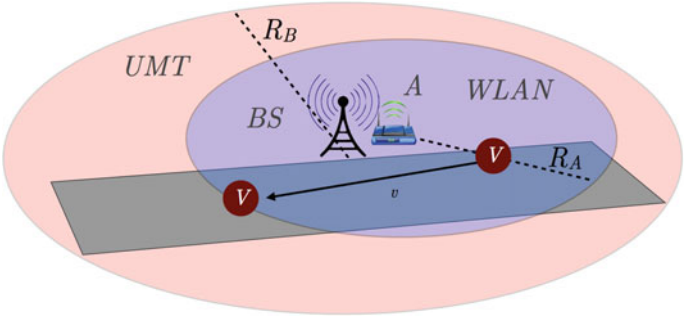


Fig. 5.17 Vertical handover scenario considering two overlapping regions (cellular base station and WiFi access point)

the wireless LAN is unavailable. Vertical handovers refer to the automatic fallover from one technology to another in order to maintain communication. This is different from a *horizontal handover* between different wireless access points that use the same technology in that a vertical handover involves changing the data link layer technology used to access the network.

Figure 5.17 shows a typical vertical handover scenario where a vehicle (or a generic mobile node) is moving with speed v under the coverage of two different network schematically represented as two circular overlapping regions respectively associated to a UMTS Base Station (BS) and to a WiFi Access Point (AP).

Regarding vertical handover, different algorithms for reducing the delay and the packet loss rate have been proposed. The Always Best Connected (ABC) concept was introduced by Gustafsson et al. [20, 21] to achieve seamless connectivity between WLAN and UMTS. The idea of using the vehicle speed as assessment criterion for vertical handover has been presented in [22, 23].

The vertical handover algorithm we adopt in our analysis is based on the approach presented by Esposito et al. [24], that bases the handover decision both on vehicle speed and handover latency. Our version of the model considers a vehicle V moving

with speed v^{des} in an environment characterized by several heterogeneous and overlapping access network at the same time. Defining SN (with bitrate B_{SN}) as the *serving network* to which the user is connected, CN (with bitrate B_{CN}) the *candidate network* and L as the handover latency (the time interval during which the peer does not receive any data due to the socket switching), then the network switch is performed only if the time that the vehicle will spend in the area covered by the cell with higher bitrate (ΔT) is long enough to compensate for the data loss due to the switch overhead ($L < \Delta T$). The handover condition is defined as:

$$B_{CN} > \frac{B_{SN}}{1 - \frac{L}{\Delta T}} + \delta \tag{5.13}$$

where $\delta \in \mathbb{R}^+$ is an hysteresis factor used to avoid handover if the two competing networks have negligible bitrate difference. Since as previously described we are using a different mobility model and real vehicular city traces, instead of the Manhattan mobility model road composed of straight lanes used by the original authors, we need to redefine ΔT as follows:

$$\Delta T = \frac{\Delta x}{|v^{des}|} = \frac{R - d(V, CN)}{|v^{des}|} \tag{5.14}$$

where R is the radius candidate network station and $d(V, CN)$ is the geographic distance between the vehicle and the candidate network. The more the user is close to a cell site, the more he/she will stay within the coverage region of that cell site.

After each handover execution, the algorithm enters in idle mode for an inter-switch waiting period T_w, in order to avoid a high handover frequency that may happen when the vehicles travel on a border line between two different cells (ping-pong effects [25]) .

The types of wireless connection we take into account are 2.5G and 3G mobile telephone technologies, as well as WiFi and WiMAX. In the first case we consider that connectivity is provided through horizontal handover where available cells located in the area allow the communication, and do not involve changing the technology used to access the network at the data link layer.

5.6 DGT Simulation

Evaluating the performance of our DGT-based protocol in significant, dynamic scenarios cannot be done analytically, because of the high complexity (non-linearity) of the problem. The interactions of the nodes determine their future state and that of the system [26]. Moreover, they usually exhibit high levels of concurrency and asynchrony, and their performance may be highly influenced by the changing environmental conditions of the environment, if they move.

For the qualitative and quantitative analysis of such systems, discrete event modeling and simulation (in which time jumps from event to event) are usually adopted [27]. In order to choose the proper simulation environment, the following criteria

should be taken into account: simulation architecture (the operation and the design of the simulator), usability (how easy the simulator is to learn and use), extensibility (the possibility to modify the standard behavior of the simulator in order to support specific protocols), configurability (how easily the simulator can be configured and with which level of detail), scalability (the ability to simulate how a P2P protocol scales with thousands, or more, nodes), statistics (how much the results are expressive and easy to manipulate), reusability (the possibility to use the simulation code to write the real application).

By looking at the state of the art, it is evident that almost every simulation tool targets a specific class of problems. Only few of them may be considered general-purpose. Among these, the most advanced, in our opinion, is CD++ [28], which is a modeling environment that allows to define and execute DEVS models [27]. OMNeT++ is another well-known general purpose discrete event simulation tool, which has been publicly available since 1997 [29]. Like CD++, also OMNeT++ is based on the concept of simple and compound modules. The user defines the structure of the model (the modules and their interconnection) using a topology description language called NED. OMNeT++ has been used in numerous domains from queueing network simulations to wireless and ad-hoc network simulations, from business process simulation to peer-to-peer network, optical switch and storage area network simulations.

Unfortunately, such tools are not particularly suitable for the simulation of distributed systems with thousands nodes, characterized by a high level of churn (node joins and departures), and reconfiguration of connections among nodes. To fill this gap, in 2009 we started a project for the development of an open source, Java-based, general-purpose discrete event simulation tool, called DEUS [30]. To simulate a distributed system at the application level, DEUS is particularly convenient, because of its extreme ease of use and flexibility. However, it does not provide packages for simulating networking layers, and we do not foresee to implement them. For this reason, until now the scheduling of application-level events to simulate the exchange of messages among nodes has been necessarily configured by the user, using reasonable values—which can be considered as a naive approach.

In order to implement realistic simulations of DGT-based systems, we integrated DEUS with Google Maps API. With the features provided by Google Maps API we created a simple HTML/Javascript control page which allows to monitor any simulated node, following it from starting to final position, and all the neighbors in its geo-buckets. This solution allows to study the protocol not only with specific P2P metrics—like message rate, miss ratio, number of peers—but also with a direct monitoring of peer behaviors during the simulation.

Recently, we introduced a general methodology to further improve our DEUS-based simulations, leveraging on a highly reliable and complete open source tool for the discrete event simulation of Internet systems, namely ns-3.[3] The latter relies on high-quality contributions of the community to develop new models, debug or maintain existing ones, and share results. In Appendix A, we provide more details

[3] http://www.nsnam.org.

about DEUS, and we describe our positive experience in integrating ns-3's LTE-EPC package [31] to support the network-aware simulation of the DGT. In next section we illustrate the packet delay model, after which we present the DEUS-based simulations of the DGT.

5.6.1 Packet Delay Model

To better characterize the communication among DGT peers in the urban environment, we defined the sub-model illustrated in Fig. 5.18, using ns-3 with the Lena LTE-EPC package[4] [31]. The latter provides (1) the E-UTRA part of the Long Term Evolution (LTE) technology, dealing with physical and MAC layers, as well as Scheduler functionalities, and (2) support for the LTE RLC and PDCP protocol, together with EPC data plane features, such as the S1-U interface and the SGW and PGW entities. Shortly, such a ns-3 package supports the detailed simulation of end-to-end IP connectivity over LTE-EPC.

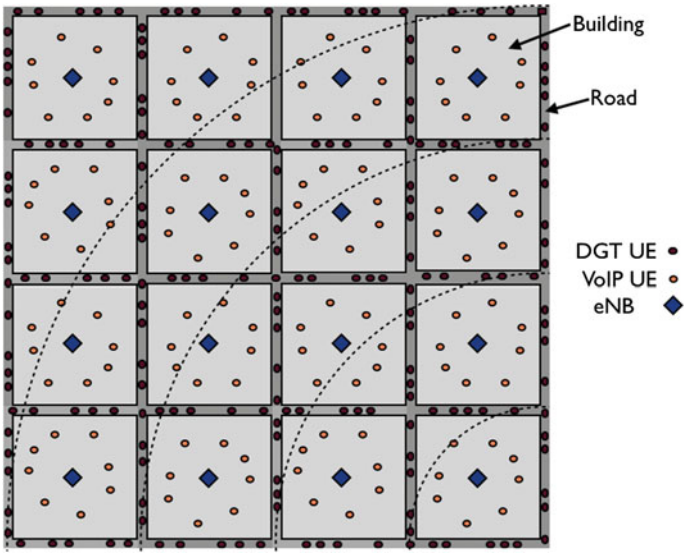


Fig. 5.18 Bird's-eye view of the simulated scenario, with $n = 200$ DGT nodes and $v = 96$ other UEs randomly placed within the buildings. The geo-buckets of the DGT node in the bottom right corner of the map are also drawn, to show that the side length of the considered area equals the GB radius

We considered DGT peers having GBs with radius of 2 km. Thus, we defined a square area having side length $l = 2$ km, with a grid of $r = 10$ roads (5 in N–S direction, and 5 in W–E direction) and vehicles running over them (with linear density

[4] We used the version released the 23rd of January 2013.

Table 5.2 Spatial characterization of the buildings

Building #	Parameters	Values
1...16	x_{min} [m]	100
	x_{max} [m]	300
	y_{min} [m]	100
	y_{max} [m]	300
	z_{min} [m]	0
	z_{max} [m]	21
	# floors	7
	# walls type	ConcreteWithWindows

$\delta = 10$ vehicles/km). Drivers are provided with User Equipments (UEs), which execute a DGT-based application. The total amount of DGT UEs is $n = r\delta l = 200$.

Parallel roads are spaced by $l/4 = 0.5$ km. Between each pair of parallel roads, there are four large buildings with squared area, each one having seven floors. Table 5.2 describes such buildings in detail. Randomly located within each building, there are $v/16$ other UEs, where v is their total amount. The pathloss model is ns3::BuildingsPropagationLossModel.

On top of each building, exactly in the middle, there is an E-UTRAN Node B, also known as Evolved Node B, (abbreviated as eNodeB or eNB), i.e., a base station which serves a subset of the $n+v$ UEs. Table 5.3 reports the configuration parameters for the eNBs and the UEs. Regarding the eNBs, they have FDD paired spectrum, with 50 Resource Blocks (RBs) for the uplink, which means a nominal transmission rate of 50 Mbps, and 50 RBs for the downlink (50 Mbps)—like currently deployed LTE systems.

DGT UEs use UDP to send four types of DGT packets to each other. The first type, called Descriptor, is for neighborhood consistency maintenance purposes. Such a packet has the following structure:

- key: int (4 bytes)
- timestamp: float (4 bytes)
- lat: double (8 bytes)
- lng: double (8 bytes)

Table 5.3 LTE model: eNB and UE settings

Device type #	Parameters	Values
eNB	UlBandwidth [RB]	100
	DlBandwidth [RB]	50
	DlEarfcn	50
	UlEarfcn	18,100
	z [m]	23
	Tx Power	49
	Noise Figure	5
UE	Tx Power	23
	Noise Figure	9

where key is the identifier of the peer in the DGT overlay network, timestamp is the current time, and lat/lng indicate the location of the node. Considering also the 12 bytes header, the size of the DGT packet is 36 bytes.

The second type of packet is the Lookup Request, which is used to search for remote nodes placed around a specified location. Its structure is the following:

- senderKey: int (4 bytes)
- lat: double (8 bytes)
- lng: double (8 bytes)

where senderKey is the identifier of the peer in the DGT overlay network that sends the request, and lat/lng indicate the location of interest. With the 12 bytes header, the size of such a DGT packet is 32 bytes.

The third packet type is the Lookup Response, which is sent by a DGT node as a reply to a lookup request, if the node owns the searched resource/information. The structure of the packet is the following:

- senderKey: int (4 bytes)
- lat: double (8 bytes)
- lng: double (8 bytes)
- descriptors: Descriptor[20] ($\leq$480 bytes)

where senderKey is the identifier of the peer in the DGT overlay network that sends the response, lat/lng indicate its location, and descriptors is a list of maximum 20 node descriptors. Considering the 12 bytes header, the maximum size of such a DGT packet is 512 bytes.

Finally, traffic information packets have the following structure:

- trafficMessage: String (30 bytes)
- senderDescr: Descriptor (24 bytes)
- ttl: float (4 bytes)
- range: double (8 bytes)

where trafficMessage is the message to be transmitted (e.g., "traffic jam"), senderDescr is the descriptor of the sender DGT node, ttl is the time to live of the message, i.e., the number of re-propagations it can be subject to, and range indicates the radius of the dissemination circle, which spatially limits the forwarding process.

We set an inter-packet interval of 50 ms for all types of DGT messages. Thus, the maximum rate is $512 \times 20 \simeq 10$ kB/s, while the minimum is $32 \times 20 = 0.64$ kB/s. In a dynamic DGT (the one we simulate with DEUS), packets are not sent periodically. For example, descriptors are sent only every ε meters. Lookup requests are sent only when necessary, as well as lookup responses. Traffic information messages are sent only when something interesting can be communicated to the other nodes (for example, a traffic jam or an incident).

The other UEs transmit and receive VoIP packets (using UDP) with a remote host located in the Internet. Such packets have a 12 bytes header and a 13 bytes payload, and inter-packet interval of 20 ms (we considered the AMR 4.75 kbps codec).

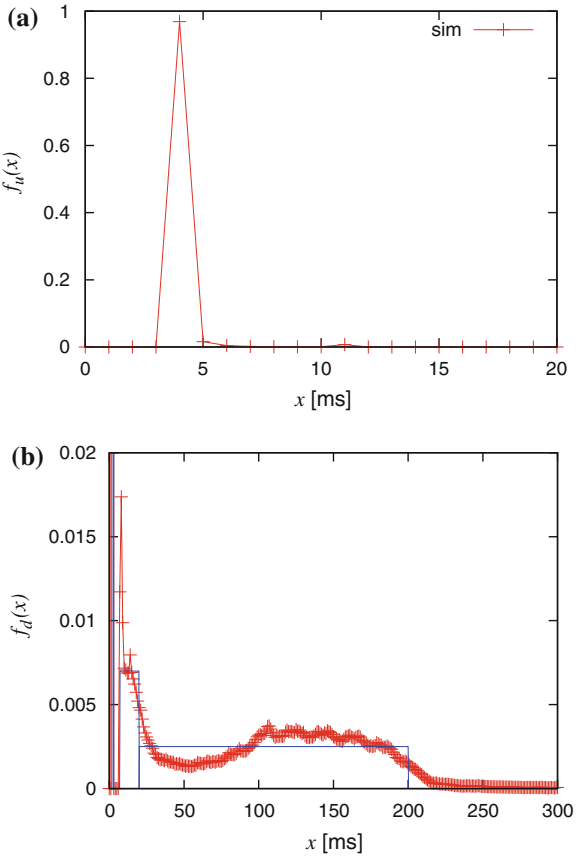


Fig. 5.19 PDFs of the uplink (**a**) and downlink (**b**) delays for DGT packets (for the case with $\delta = 5$ vehicles/km), obtained with ns-3

Each eNB has a scheduler which allocates RBs (which are the smallest elements of resource allocation) to users for predetermined amount of time. In these simulations, the Proportional Fair scheduler is used (ns3::PfFfMacScheduler), which tries to maintain a balance between two competing interests: trying to maximize total wireless network throughput while at the same time allowing all users at least a minimal level of service.

As previously stated, every DGT UE sends a DGT packet (with randomly chosen type), with inter-packet interval of 50 ms. The ns-3 simulations were executed on a Ubuntu Linux 11.10 x86_64 machine with 16 GB of RAM and double quad core processor Intel(R) Xeon(R) Intel Xeon E5504 2.00 GHz. Each simulation was repeated with 20 different seeds for the random number generator.

For the DGT packet flow, we analyzed also the uplink and downlink delays—to this purpose, we modified the logger of the LTE LENA package in ns-3, to obtain a discretized probability density function (PDF) of the RLC packet delay. The PDF of

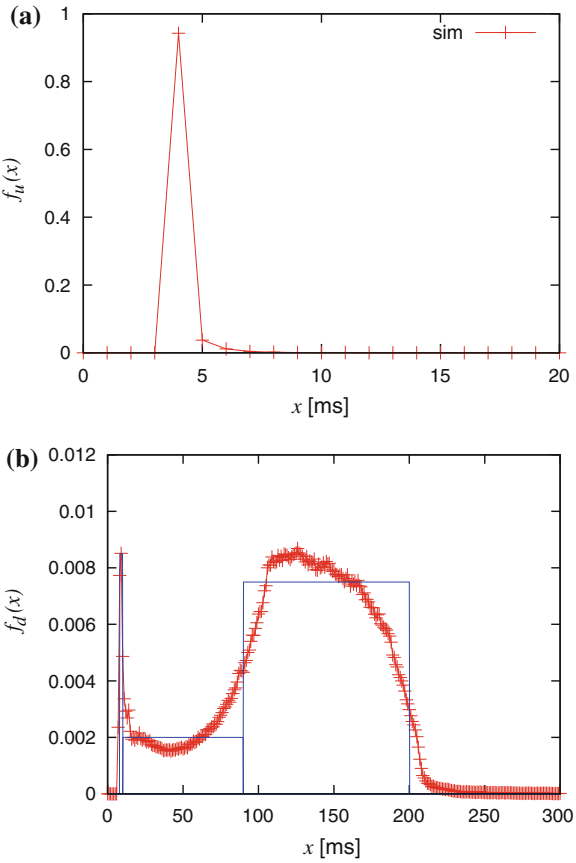


Fig. 5.20 PDFs of the uplink (**a**) and downlink (**b**) delays for DGT packets (for the case with $\delta = 10$ vehicles/km), obtained with ns-3

the uplink delay is basically a delta function, always centered in 4 ms, independently on the linear density of the vehicles (Figs. 5.19a, 5.20a, 5.21a). Instead, the PDF of the downlink delay is more complex, and is highly affected by the linear density of the vehicles (Figs. 5.19b, 5.20b, 5.21b). The PDF is then used to generate realistic packet delays in the DEUS-based simulations, using the well-known *inversion method* [32], which is based on the inverse probability theorem:

- choose the cumulative distribution function $F(x)$ of the random variable to be sampled;
- generate a set of uniform random numbers such that $R \sim U(0, 1)$;
- compute the random variate $X_i = F^{-1}(R_i)$.

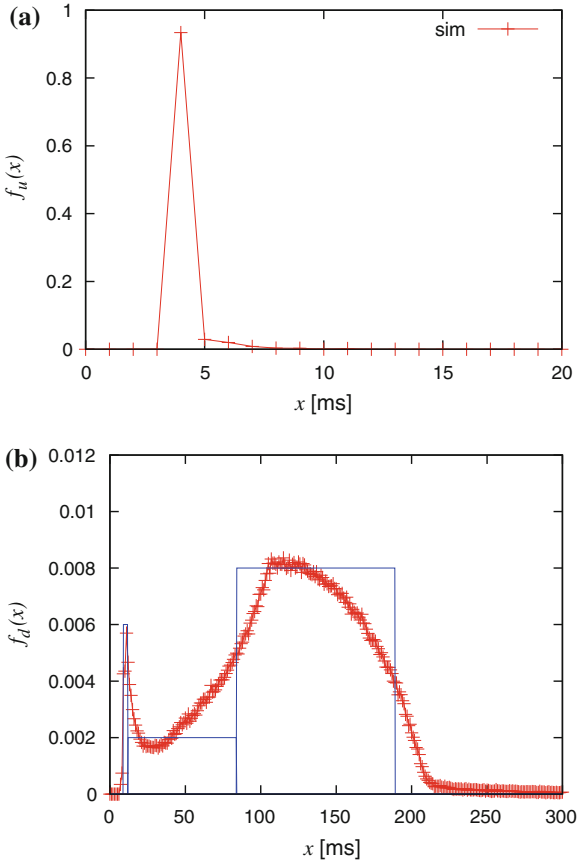


Fig. 5.21 PDFs of the uplink (**a**) and downlink (**b**) delays for DGT packets (for the case with $\delta = 20$ vehicles/km), obtained with ns-3

For practical implementation purposes, the discretized PDF of the downlink RLC packet delay is approximated by a piecewise constant function, whose numerical inversion is straightforward due to the reduced number of pieces (3).

5.6.2 DEUS Model

With the packet delay model described above, we obtained an improved DEUS simulation model of the DGT, with respect to the previous ones which used, for every transmission, an exponential delay with mean value obtained by considering the nominal uplink and downlink.

We simulated a DGT overlay with 1,000 mobile vehicles, over a period of 10 h. In the first half of such a period, the network grows from 0 to 1,000 nodes. In the

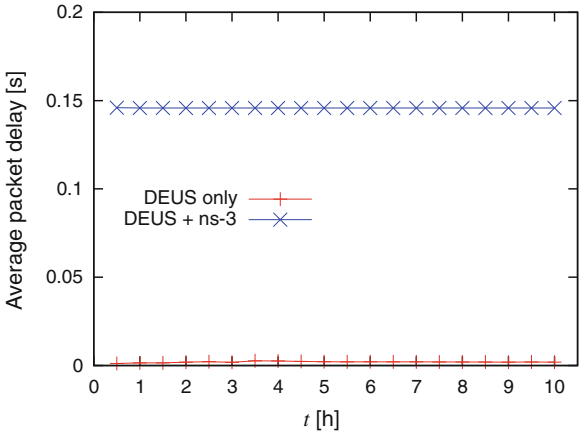


Fig. 5.22 Average packet delay, measured with DEUS, for the simulated DGT overlay network with 1,000 vehicles

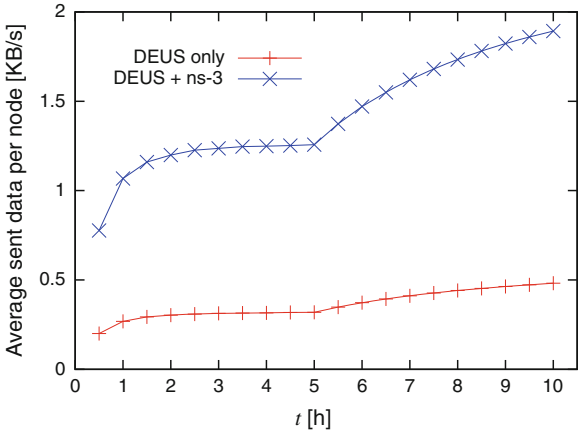


Fig. 5.23 Average amount of sent data per node, measured with DEUS, for the simulated DGT overlay network with 1,000 vehicles

second half, the size of the network remains stable. We logged the average packet delay and amount of sent data per node, computed on the whole overlay network. Figs. 5.22 and 5.23 compare the results obtained with the old simulation model, and those obtained with the refined one.

As we expected, in the refined model the average delay is higher than the one obtained with the naive model, which is based on nominal uplink and downlink values. Also the average amount of sent data is higher, because in the refined model we take into account also the header of the packets (12 bytes are 1/3 of Descriptor packets, which are the most frequently sent).

In order to evaluate the performance and behavior of our DGT-based protocol, we considered two different mobility models, namely a very generic one and another specific to street vehicles. Two different metropolitan areas have been chosen for the simulation, the first one around Frankfurt (square area with side of 20 km) and the other surrounding Parma (square area with side of 7 km). In both cases a list of real road paths have been generated offline (using the GoogleMaps API and a refining algorithm) from an initial set of potential points of interest.

The first mobility model is a random model where an active peer in the system selects one of the available paths and starts moving over it segment after segment. Each peer is a mobile node with a random base speed (v_b) between 5 and 100 km/h that can be associated to pedestrians, bikers and vehicles. For each segment, peer speed is randomly selected according to an exponentially distributed random variable with mean value v_b.

The second mobility model we considered is more complex also from a computational point of view, since it takes into account characteristics and parameters of inter-vehicular networks and is the FTM model presented in the previous section.

5.6.2.1 Evaluation of Geo-bucket Configuration

The first part of our simulation analysis aims at outlining how the choice of the geo-bucket configuration, in terms of number of GBs and their thickness, influences DGT performance as expressed by the PMN and the MR. The following results refer to the generic mobility model, unless otherwise specified.

Two different peer systems are simulated over a significant time span. The first one includes 1,000 peers for a virtual time of 10 h (corresponding to 10,000 virtual time units) while the second one has doubled size (2,000 peers) and time span (20 h, that is 20,000 virtual time units). In both cases the node set grows to full size during the first half of the simulation, after which only an insignificant number of peers enters and leaves the network.

Table 5.4 presents all the considered cases.

Figure 5.24 shows the PMN for cases 1, 2 and 3, which refer to the first peer system (1,000 nodes over 10 h). We remark that the PMN remains moderate over all the simulated period, for all geo-bucket configurations. In particular for the first half of simulation, where many new nodes enter the system, the PMN value only grows up to around 5 %, while it decreases significantly when the peer network reaches a more stable state.

Table 5.4 Considered GB configurations

Case	#geo-buckets	GB thickness (km)	#Peer	Final VT
1	10	1.5	1,000	10,000
2	5	3	1,000	10,000
3	10	0.5	1,000	10,000
4	10	1.5	2,000	20,000
5	5	1.5	2,000	20,000

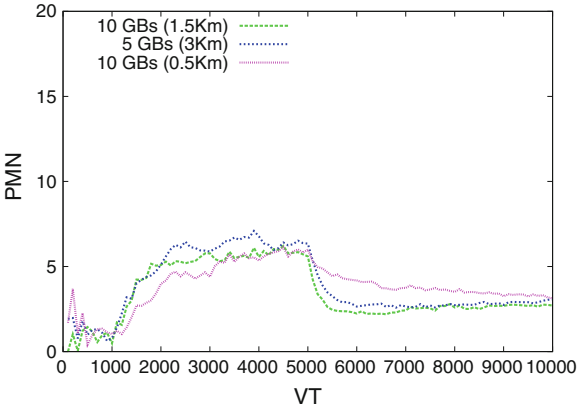


Fig. 5.24 PMN—GB configuration evaluation (cases 1, 2 and 3) with 1,000 nodes

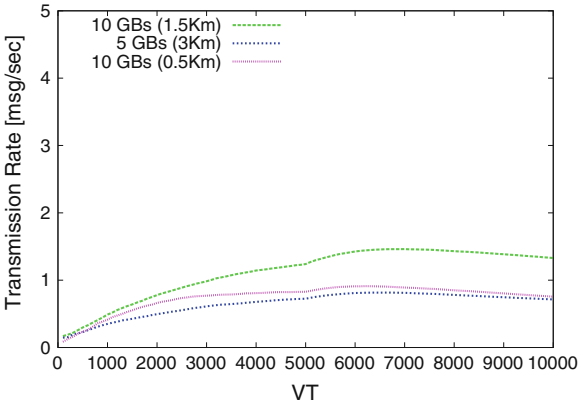


Fig. 5.25 Global MR—GB configuration evaluation with 1000 nodes

Another important metric that we must take in account to evaluate these different configurations is the MR. Figure 5.25 shows results for cases 1, 2 and 3. Given that a larger covered area ($\pi \cdot r_{GB}^2$) is potentially associated to a higher number of active peers, an increased number of known nodes must be contacted to obtain GP updates. For this reason, simulation results show that cases 1 and 2 have an increased MR value compared with case 3 where the covered area is smaller. In any case, the number of exchanged messages is very low, notwithstanding the fact this is a fully decentralized system where knowledge is maintained cooperatively by all available peers.

The same analysis was carried out on the larger (2,000 active peers) network using two configuration of geo-buckets (cases 4 and 5) in order to assess the protocol's behavior with a different distribution of nodes.

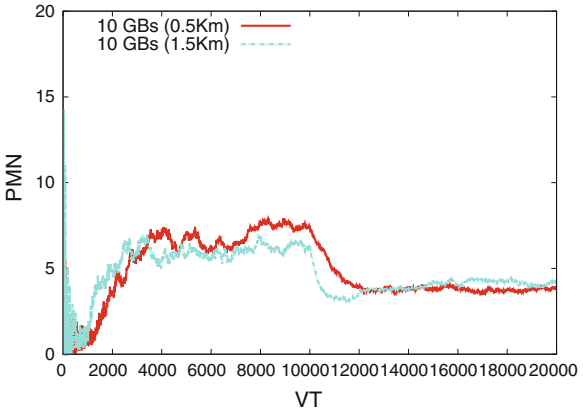


Fig. 5.26 PMN—GB configuration evaluation (cases 4 and 5) with 10 GBs and 2,000 nodes

Results in Fig. 5.26 show that also with a higher number of available peers and using two GB configurations the PMN is very small (under 10 and around 5 %). In the 4th case, 10 geo-buckets with 1.5 km thickness are used, which means a covered area of 706 km^2, whereas in case 5 we have only 5 geo-buckets with the same thickness for a covered area of 176 km^2. There an evident difference in the covered area but the performance is very good in both cases. The little amount of missing nodes depends on the dynamics created by new incoming peers and by the high rate of movements generated by nodes traveling on their paths.

In order to provide this level of performance with different configurations and covered areas, the protocol needs to route messages to users in the target zone. The denser scenario, as described for the smaller network, implies a different amount of exchanged messages (Fig. 5.25) that depends on peer density in the analyzed area.

We observe that the accuracy of the protocol shows little dependance on the configuration of geo-buckets (number and thickness). Results show that different parameter setups still obtain very low PMN, given the highly dynamic context where all peers are mobile users that change their position very often. The other important aspect that comes from this analysis is the relationship between the covered area and the MR value, that we must take into account when designing an application based on this protocol, in order to find the right compromise between the size of analyzed zone and the number of exchanged messages.

Another important issue related to the PMN is to understand the distribution of missing nodes across available GBs in order to verify the knowledge evolution of active peers (Fig. 5.27).

Figure 5.28 is related to case 1, with 10 geo-buckets having a 1.5 km thickness. We already showed the associated PMN in Fig. 5.24 which stays very low for all the simulation at about 5 % or less. We analyze now how this value is split across different GBs. For the inner geo-bucket the percentage of missing node is around 0 % for the whole simulation's time. Predictably, the largest amount of missing peers is located in the external GBs that cover areas even very far from the peer. The outmost

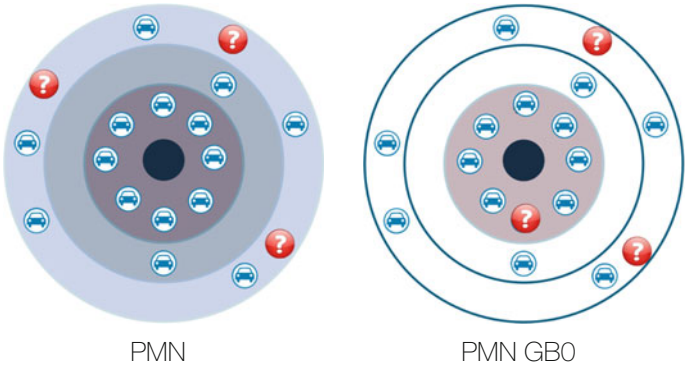


Fig. 5.27 Visual representation of the global PMN (for all GBs) and the same metric for a single GeoBucket (e.g., GB0)

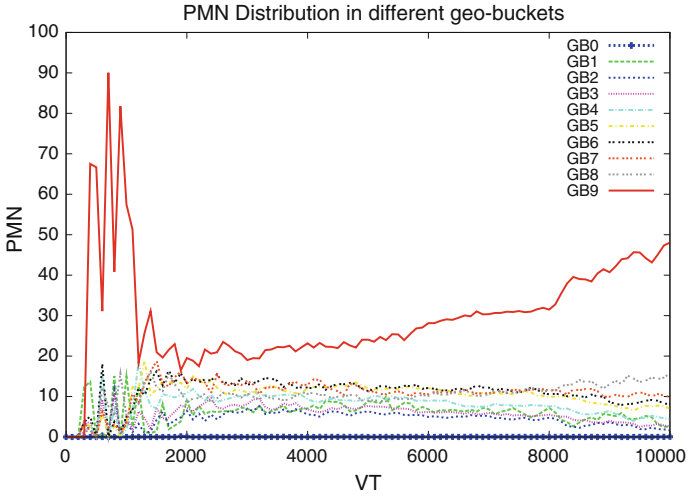


Fig. 5.28 PMN distribution in each geo-bucket—GB configuration evaluation (case 1)

GB, i.e., GB9 has the highest percentage of missing nodes and other geo-buckets limit the PMN under the 20 %. This is a very important result that shows how the protocol is very accurate and reliable and how it fulfills the DGT goal of having a high percentage of known peers that are very close to a node's position. This result was obtained with a 1.5 km GB thickness and may be very useful for example in vehicular networks, where it is very important to have the best knowledge of active users in a specific area of interest around the car.

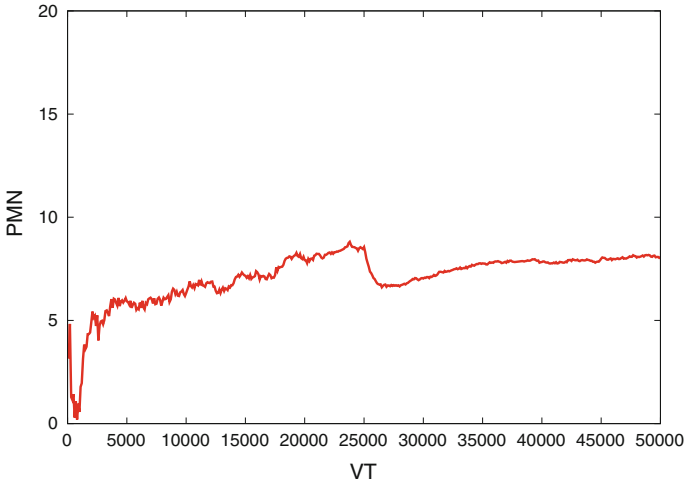


Fig. 5.29 PMN—larger network with ≈5,400 peers

5.6.2.2 Larger Network

The performance of our protocol was also evaluated in the context of an increased number of peers and with high dynamics due to the numerous joins. The simulation considered ≈5,400 peers over a virtual time of 50 h using 10 geo-buckets with 1.5 km thickness.

Figure 5.29 shows the achieved percentage of missing peers that appears slightly increased if compared with the results of the first scenario, although in any case it is reasonably under the 10 %.

The cost in terms of exchanged messages (Fig. 5.30) is still very low, if we consider that the geo-bucket covered area is large and the high density of active peers. We can see that in the first half of simulation there is an increase of the analyzed parameter because there are a lot of new joins over a short time and in the same area. This behavior causes new activities related to joins and position updates that require additional message exchange among peers. In the second half, when the number of new incoming users is decreased, the resulting MR is reduced.

Considering this larger network scenario, we show the results related to the average node position error (NPE) (Fig. 5.31). The ε parameter is crucial in this regard: a very low value of 0.5 km—if compared with the target area of each peer (10 GB $\times$ 1.5 km)—was set. Results confirm that on average the error is around ε for the duration of the simulation. The optimal choice for this parameter can be related to the requirements of the particular application. For example, there may be a need for high accuracy across a very large covered area, e.g., road/highway monitoring system.

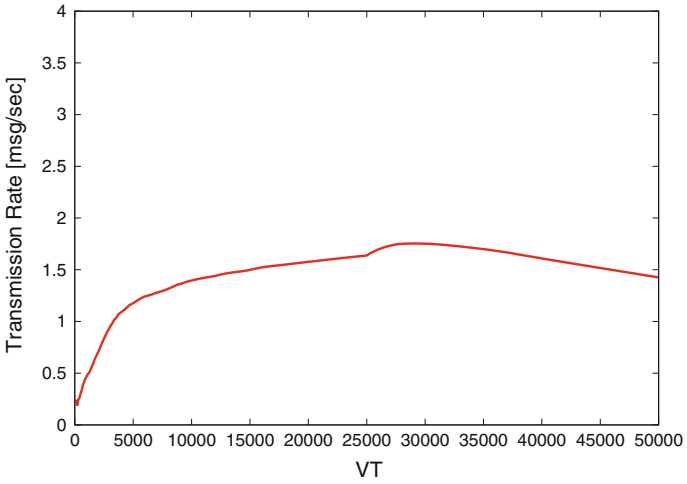


Fig. 5.30 MR—larger network with ≈5,400 peers

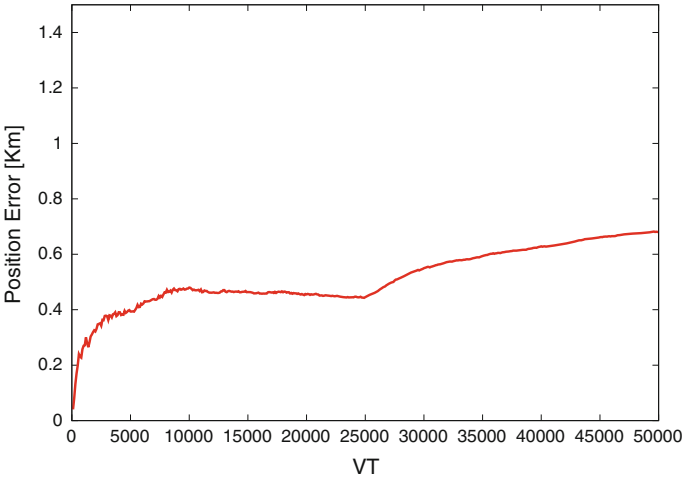


Fig. 5.31 NPE—larger network with ≈5,400 peers

5.6.2.3 Evaluation of the Position Update Mechanism

This scenario was created to assess the effects of ε variations on protocol performance. Using a network of 2,000 nodes (10 GBs and a thickness of 1.5 km), we ran multiple simulations by varying ε between 0.1 and 1.35 km in 0.25 km steps. As previously stated, ε represents a displacement threshold, used in the Position Update procedure. A low value means that updates of peer positions are performed very often when users change their locations, whereas a high value causes infrequent updates.

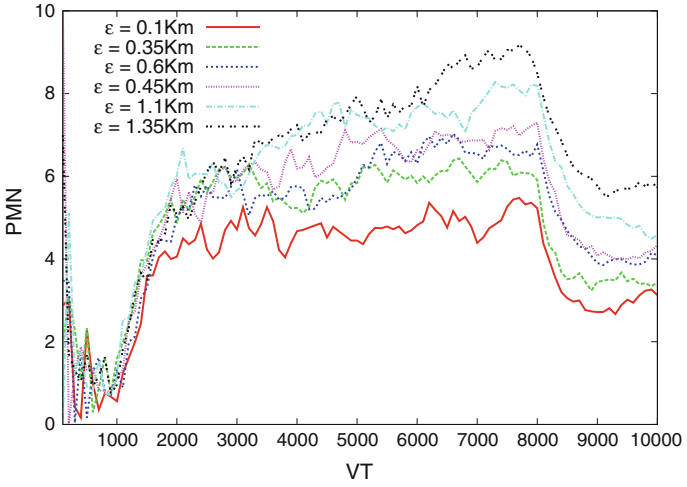


Fig. 5.32 PMN—position update evaluation for several values of ε

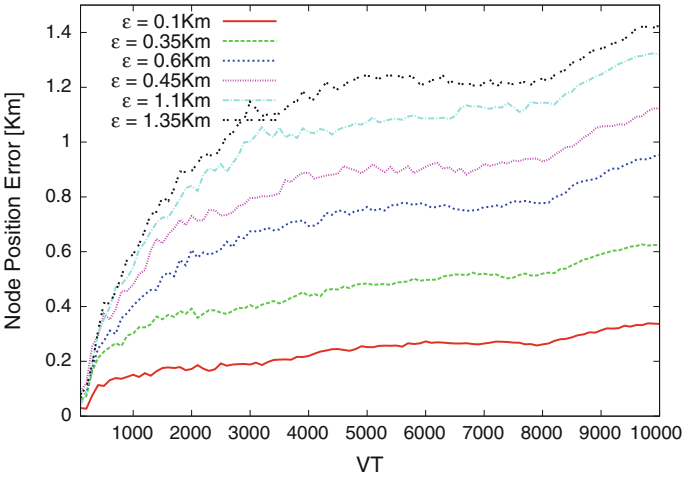


Fig. 5.33 NPE—position update evaluation for several values of ε

The accuracy of information stored in GBs is clearly related to the value of ε. Fig. 5.32 shows the percentage of missing nodes with multiple ε values and we can see that there is a noticeable spread in the PMN results. This behavior is justified by the fact that a large ε value may lead to the erroneous exclusion or removal of a peer from the GBs, resulting into accuracy loss and inconsistency.

The analysis of the NPE (Fig. 5.33) shows that the average error is slightly larger than the threshold as there is an additional little variation introduced by peers' mobility and information's distribution among available nodes.

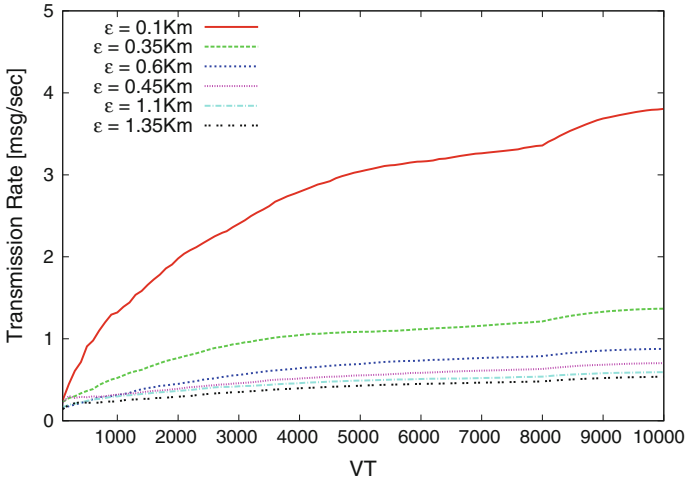


Fig. 5.34 MR—position update evaluation for several values of ε

Another important aspect related to these analysis is the number of exchanged messages. A small value of ε that results in a reduced error of position is strongly correlated with an increased value of MR. Figure 5.34 shows the results of different configurations and suggests that a value between 0.35 and 0.6 km can be a good compromise in terms of messages and accuracy for the chosen set of parameters.

This scenario is useful to understand the importance of the ε parameter and how we can make a better use of it. Clearly, this parameter is strongly related to application requirements, for which a careful analysis during the design phase gives the opportunity to reduce the number of exchanged messages without a great impact on global accuracy.

5.6.2.4 Robustness Evaluation

A common element of all peer-to-peer systems that affects their performance is the high node dynamics due to churn. This section reports DGT results related to a very pessimistic scenario where the initial growth of the network is followed by a stabilization interval without new joins, and finally by a high churn phase. In the latter, a predefined portion of active peers (evenly distributed over the simulated area) disconnects at once to let us evaluate the overall robustness of the DGT system. Simulation are based on a vehicular mobility model, with paths located in the city of Parma. The network is characterized by 1,000 active peers with the same growing behavior of previous described scenarios and using dynamic discovery period with a range of [1.5; 6] min depending on the number of new found nodes at the previous discovery iteration. The number of geo-bucket is 5 with a thickness of 0.5 km and $\varepsilon = 0.1$. A varying degree of node disconnection and the related PMN distribution

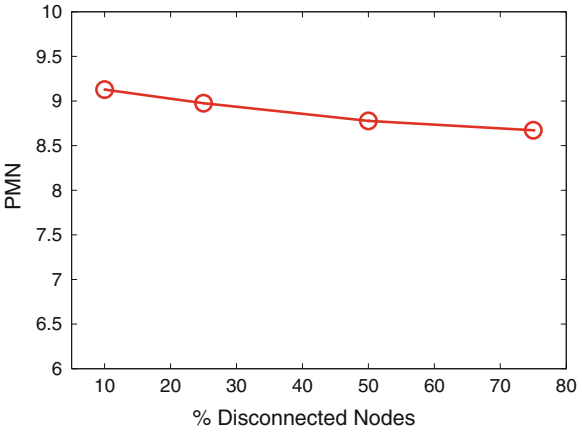


Fig. 5.35 PMN for different percentages of disconnecting nodes

have been analyzed. Figure 5.35 shows that the percentage of missing nodes for different fractions of disconnected peers maintains the same value (between 8 and 10 %) without any significant variation. To understand the reason of such a good result, consider a peer that is aware of N neighbors. If M of the N neighbors leave the network unexpectedly, the peer may incur in false positives. Fortunately, the period during which the peer is not aware of the changes in its neighborhood is usually very short, because the peer sends maintenance messages, whose frequency increases with mobility. It is an interesting and important result that validates and confirms the robustness of the implemented DGT overlay which can efficiently manage abruptly and massive disconnections and consequently will handle at ease normal behaviors of active users in P2P networks. This results is also supported by the graph in Fig. 5.36 that illustrates how the PMN is distributed in GB_0 revealing that is always very low and not significantly affected by peer disconnections.

5.6.2.5 Urban Environment Analysis

After the encouraging results shown in previous scenarios, the system behavior has been evaluated using the same mobility model of the previous section.

The analysis is divided in two different parts, with the first one focused to confirm previous results in a better modelled mobility scenario. The second part aims at evaluating how the size of the peer's local region of interest (as expressed by the number of geo-buckets—K) affects DGT performance, that is the PMN, as also in this scenario we are interested in finding all active nodes in the region of interest of a generic peer. Both simulation types refer to a square region surrounding the city of Parma, having a GB thickness of 0.5 km, $\varepsilon = 0.1$ as well as a dynamic discovery period with a range of [1.5; 6] min as in previous analysis.

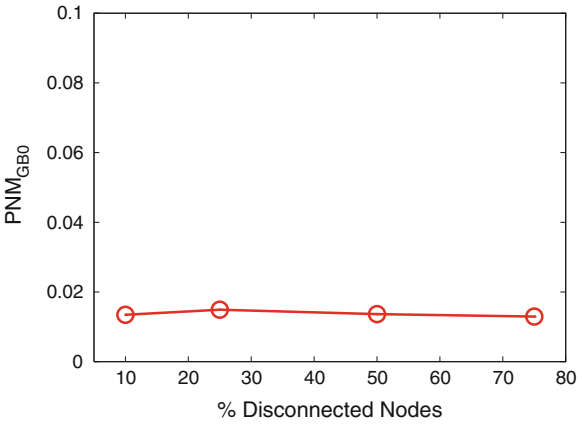


Fig. 5.36 PMN_{GB0} for different percentages of disconnecting nodes

The first analysis considers a constant number of geo-buckets equal to 5 (covering a a region of interest of $\cong$ 19 km^2) and monitors the variation of the overall percentage of missing nodes to different peer distribution in the network.

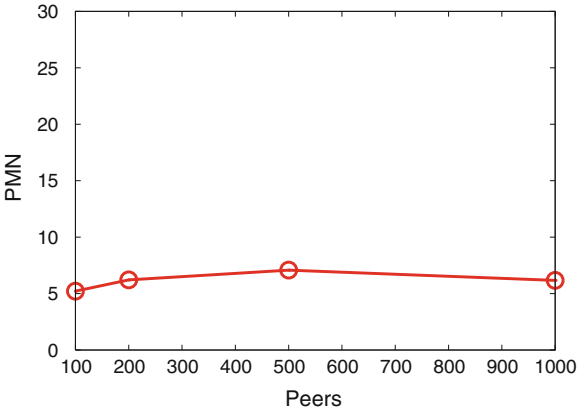


Fig. 5.37 *PMN* value for different network sizes

Simulation life is initially characterized by a growing number of active users that step by step join the network, start moving, exchanging messages and discovering their neighbors. This phase is followed by a stable period without new joins or disconnections where the activities of the system proceed normally according to nodes movement and behaviors. Figure 5.37 reports the PMN value along all the simulation and confirms that the number of missing nodes is really low and around 5 % for different sizes of the peer network. The second evaluation takes into account the effects

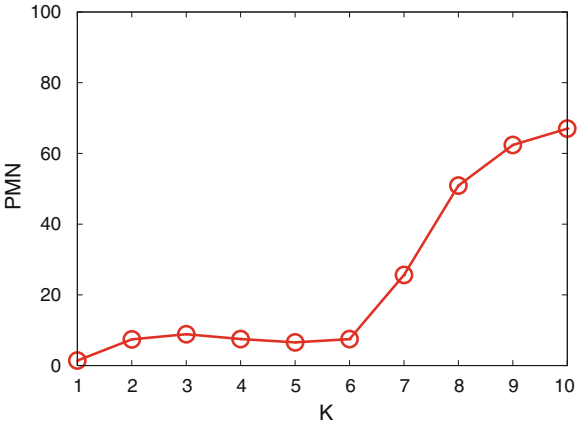


Fig. 5.38 PMN results for different K values

of the variation of the number of GBs (K) and the related covered area for a DGT node. Considering, as previously described, a reasonable high dynamic discovery period, it is possible to see in Fig. 5.38 how the PMN value evolves according to the growth of K. The selected time interval allows to maintain a low percentage of missing nodes until the number of GBs is equal to 6 (area $\cong 28\,\text{km}^2$) otherwise the value grow very fast. This is of course related to the discovery time because a larger area implies, on average, an increased number of nodes that change their position, and consequently requires the search procedure to be scheduled more frequently. To have a complete picture of the situation, it is very important to investigate how this PMN value is distributed among available GBs and in particular in the first one that contains the knowledge about the closest neighborhood for a peer. Figure 5.39 confirms also in this case the good performance of the DGT approach that allows to keep the PMN_{GB_0} near to zero for all tested GBs configurations.

5.6.2.6 Vertical Handover Analysis

The analysis of the robustness of DGT-based localization, considering the vertical handover model defined in previous section has been performed considering the same region of 10 km^2-squared area centered on the city of Parma and the mobility model previously explained.

We have considered a DGT overlay where available nodes have $K = 3$ different GeoBuckets, with a thickness of 1Km and a dynamic discovery period ranging from 1.5 to 6 min, depending on the number of discovered nodes (if the latter decreases, then the period increases). Simulations cover ten hours of system life (10, 000 virtual time units) and have been averaged over several execution runs with different seeds. Additional parameter values are $T_w = 20$ (s) to reduce pingpong effects and

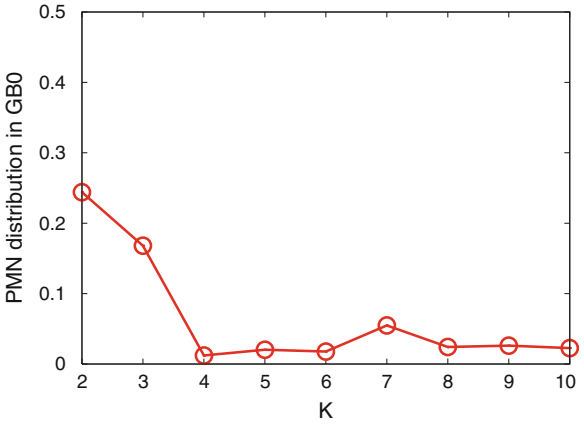


Fig. 5.39 PMN distribution in GB0 for different K values

$\delta = 1$ (Mbps) as hysteresis value to avoid handover if the two competing networks have negligible data rate difference. Road accident events are scheduled during the simulation according to a Poisson process with mean inter-arrival value of 1,000 VTs. These and other events are sensed by vehicles and disseminated over the DGT through different message types.

In the evaluation, we have considered the following additional performance metrics:

- $RT(x)$ [dimension: (s)]: Reconnection Time, i.e., the average time required by a temporary disconnected peer to recover the knowledge of its neighborhood and minimize the PMN value under x %.
- *Packet Loss/min*: average number of packets that fail to reach the destination per minute per peer. It takes in account both DGT and content dissemination packets.
- % *Coverage*: Estimated coverage percentage of traffic information messages at a certain time of the simulation. It is evaluated as the number of peers that actually received a specific message over those that should have it.

With the aim of improving the accuracy and the realism of simulated models, we performed multiple field measurements using Android smartphones (HTC Desire and Samsung Galaxy) on a vehicle moving along several Parma streets. In this way, we obtained experimental data about:

- *Uplink and Downlink Rates*: Real-world communication rates for uplink and downlink channels.
- *Cell Tower Information*: Information about cell towers located in the area of interest, such as geographic location, provider, connection type, measured distance to cell tower and RSSI value.

All measurements have been carried out with different smartphones and SIM modules of three italian providers (TIM, 3 ITA, Vodafone). Tower locations were

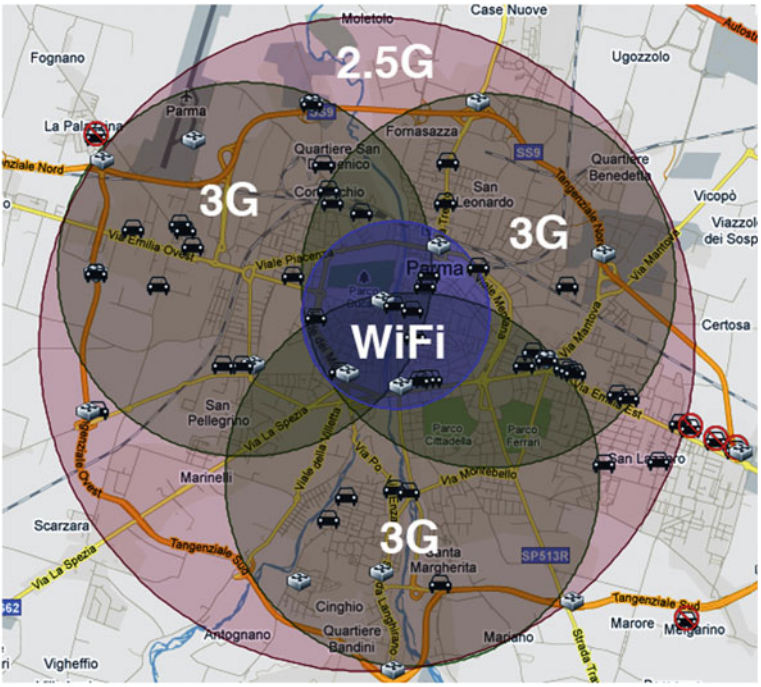


Fig. 5.40 Simulated network regions in Parma urban area

Table 5.5 Types and performance of the available network regions in the simulated urban area

#	Type	Uplink min–max (kbit/s)	Downlink min–max (kbit/s)
1	2.5G	30–90	60–170
2	3G	35–1,150	91–2,650
3	WiFi	100–2,000	2,000–10,000

used to build a map of available cell towers (the overall coverage is schematically shown in Fig. 5.40), each one characterized by a specific connectivity type and a coverage area with a 1.5 km radius. A WiFi region with a radius of 2.0 km is also located in the city center (which likewise approximates the coverage that is actually available in Parma center) to provide higher data rates where the density of peers is very high and consequently larger messages are exchanged among nodes to share the information about neighborhood. For each type of connectivity Table 5.5 reports the ranges of data rates experimentally obtained on the field and thus the limits for the values considered by the simulation.

A first simulative analysis has been carried out to evaluate the robustness of the DGT overlay with respect to vertical handover in five scenarios with different network coverage, considering the vertical handover latencies reported in Table 5.6. Simulated

Table 5.6 Vertical handover latency timetable

	Disc. (s)	2.5G (s)	3G (s)	WiFi (s)
Disc.	0	1	1	5
2.5G	1	0	0	5
3G	1	1	0	5
WiFi	5	5	5	0

Table 5.7 Simulated scenarios with different connectivity coverage of the urban area

Scenario	2.5G regions (%)	3G regions (%)	WiFi regions (%)
1	100	100	100
2	100	75	0
3	100	50	0
4	100	25	0
5	100	0	0

coverage distributions are summarized in Table 5.7, starting from a fully operational Scenario 1 and proceeding with a progressive decrease of available connection types for WiFi and 3G networks.

Graph (c) in Fig. 5.41 reveals how significantly the presence of a WiFi region in the first scenario and the related expensive vertical handover effect influences the reconnection time needed by a peer to recover the PNM under the 10 % of missing nodes. In scenarios where there is no WiFi area the required period is quite smaller. This behavior affects only marginally the number of packet lost per minute (b) and the global PMN distribution (a)—the latter results slightly higher in the first scenario compared to the others. Furthermore, the percentage of missing node in the inner GeoBucket (b) remains really low in all simulated scenarios, allowing for a high coverage of traffic information disseminated among nodes. Those results are a consequence of the efficiency and robustness of the DGT approach that allows to quickly identify new available nodes close to peer's geographic location, by means of periodic discovery and maintenance procedures, and at the same time to detect disconnected nodes.

A second simulative analysis aimed at measuring the robustness of the DGT overlay with respect to increasing values of vertical handover WiFi latency. We considered a rather pessimistic range ($L \in [1s; 8.5s]$) of latency values (if compared to the ones used in other papers [24]), in order to heavily test our approach, also taking into account that the latency is constant for all the duration of the simulation and for all peers. In terms of network coverage the simulated scenario is again Scenario 1, where all types of connectivity are available at the same time in the area of interest. As expected, an increased value for the WiFi vertical handover latency heavily affects RT values (as shown in Fig. 5.42c), and consequently the global percentage of missing nodes (Fig. 5.42a), that proportionally grows with the latency. The same behavior can be observed for the number of lost packets per minute (Fig. 5.42d) that however remains globally small. As presented in the analysis related

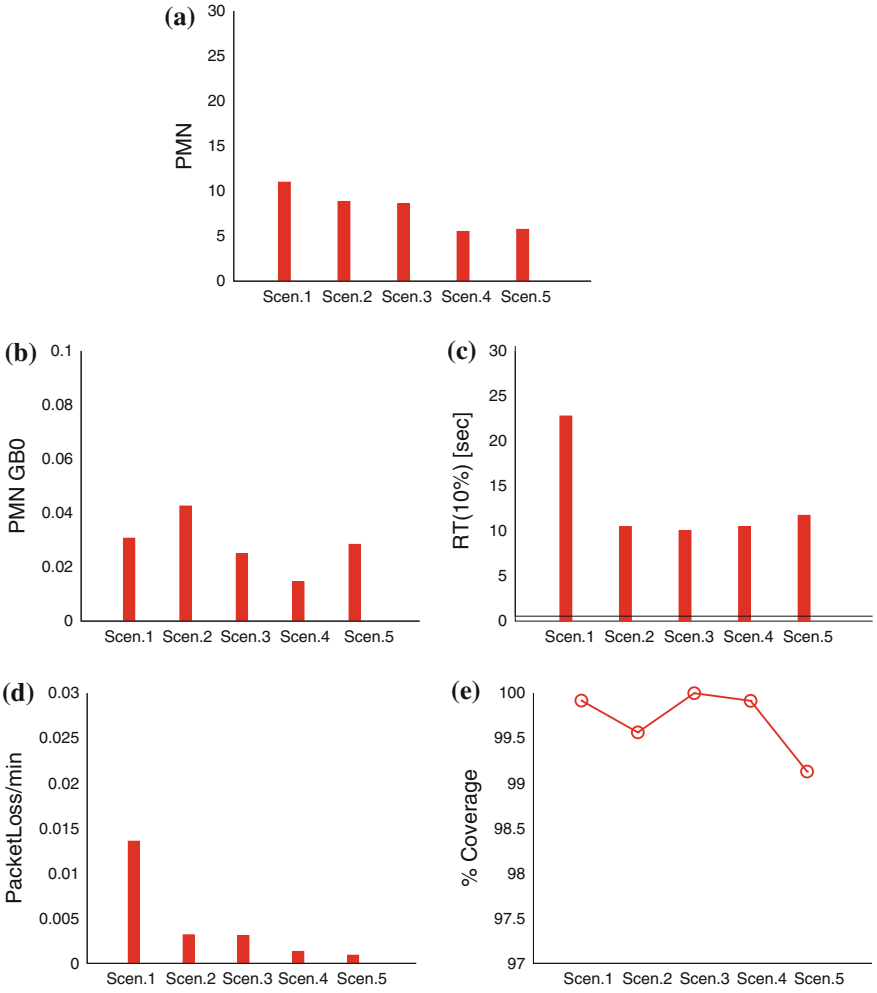


Fig. 5.41 Results related to different simulated scenarios. **a** PMN. **b** PMN in GB0. **c** RT(10 %). **d** PacketLoss/min. **e** % coverage of traffic information messages

to the variation of the connectivity coverage, one of the main important metrics for evaluating the robustness of the DGT approach is the PMN evaluated in the first GeoBucket(s). In fact, a high knowledge in the inner container and a gradually reduced value in the others mean that in any case the peer can perform successfully the discovery procedure, keeping the neighborhood updated. Most importantly, the peer can properly disseminate traffic information messages to its neighbors. Results presented in Fig. 5.42b, c confirm how the design of this peer-to-peer inter-vehicular DGT network is robust also in presence of a high values of latency, being able to correctly deliver messages with a percentage always higher than 99 %.

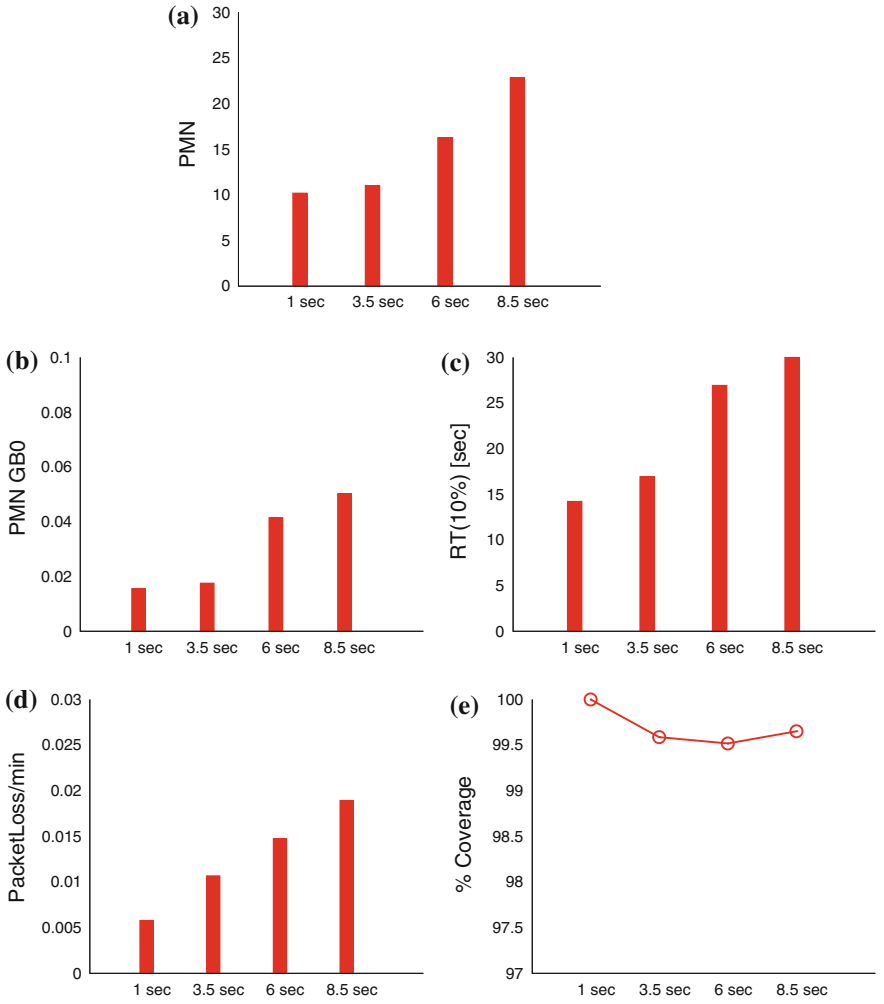


Fig. 5.42 Results related to different latency values. **a** PMN. **b** PMN in GB0. **c** RT(10 %). **d** PacketLoss/min. **e** % coverage of traffic information messages

5.6.2.7 Discussion

Previously illustrated scenarios have shown that the effectiveness and efficiency of the protocol depend on the following system parameters:

- Target Area A (as covered by the geo-buckets),
- Discovery Period (T_d),
- Position Update Threshold (ε).

How to configure optimally these parameters ultimately depends on the application. In this first analysis we focused on a very "extreme" situation, where the objective of each peer was to discover all of its surrounding peers within a quite large area. As a matter of fact, some types of applications may require such a tight constraint, while others may have less stringent requirements.

An example application based on the DGT overlay may be a totally decentralized traffic information system that allows to efficiently disseminate information about urban traffic status—such as accidents, jams, potholes or bad surface conditions. This application would not need that each peer knows all its neighbors. Potentially, a limited but well distributed knowledge would be sufficient to properly send messages and notifications to interested users (in terms of events and locations). Key parameters of such an application would be T_d and ε, and their values should be chosen to keep the neighborhood updated in highly dynamic scenarios—e.g., a vehicular network where location changes are very frequent and often fast. At the same time, parameter A could be selected according to the distance that published localized information needs to cover.

A radically different application may be a Social Advertisement System, where shops and city offices would propose products and services to new or already registered users, according to their profile or to suggestions and feedbacks taken from their friend relationships. This system may be designed using two different instances of DGT protocols, one for nodes representing providers and another (completely different) for users. The former would catch all available peers in a target area A_p, which would become the key parameter of the architecture. The latter, instead, would build limited user lists, associated to a small area A_u used to route incoming messages and build awareness about the location of friends and services. In such a system, the frequency of the discovery procedure and the ε parameter may be set to obtain looser position updates, since—for the sake of advertising—misplacing users a few hundred meters does not make any practical difference.

We also presented an investigation on the robustness of our DGT-based localization protocol to vertical handover in highly serviced urban areas. We have illustrated some significant simulative scenarios whose results evidence the independence of the percentage of missing nodes in the inner GeoBucket from peer disconnections due to vertical handovers as well as the short time required subsequently to recover knowledge about most neighbors consequently allowing to correctly deliver

Presented results are relevant also to show how a traffic information system and/or a vehicular network based on mobile devices and a peer-to-peer approach is feasible, and how in the near future those kinds of systems could be really and massively utilized by end users.

5.7 DGT for Vehicular Networks: The D4V Architecture

In this section we present the application of the DGT approach in the context of a completely distribute intelligent transportation system. We used the designed overlay to build a distributed traffic information system (TIS) for the dissemination of traffic

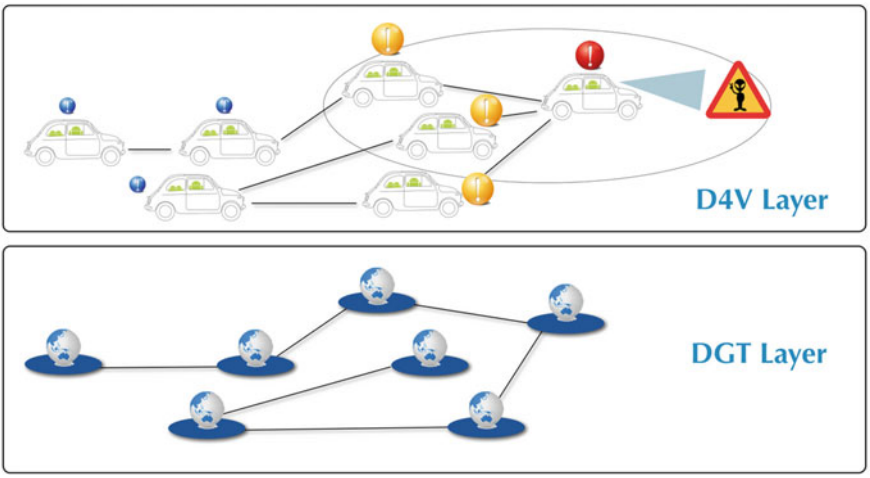


Fig. 5.43 DGT and D4V layers used to realize a distributed TIS

alert messages. In such a system (depicted in Fig. 5.43) users can participate using their smartphone to send and receive real-time information about traffic conditions or potentially dangerous situations. Furthermore, we present the implementation of the first DGT prototype to evaluate the performance of the protocol in a actual smartphone-based vehicular network.

5.7.1 Traffic Information System and Vehicular Sensor Networks

Driving safely, efficiently and comfortably does not depends only on the vehicle, but also on a large number of external factors that are difficult to predict without the support of IT. Among others, Vehicular Inter-networking [13] has a prominent role, paving the way to several valuable applications, namely geocasting, mobile data sensing and storage, street-level traffic flow estimation, etc. [33]. Vehicular networks builds upon Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) connection, as well as on hybrid variants [13].

Vehicular Sensor Networks (VSNs) are emerging as an appealing technology for monitoring the physical world, especially in urban areas where a high concentration of vehicles equipped with onboard sensors is expected in the near future. One reason of the interest for VSNs is that vehicles can be easily equipped with powerful processing units, wireless transmitters, and sensing devices. The latter may even be somehow complex, costly, and heavy like GPS, cameras, vibration sensors, acoustic detectors, etc.

Recently, the VSN research community has started investigating the possibility of using smartphones as V2V and V2I communication nodes, but also as portable sens-

ing platforms [34]. Smartphones are characterized by ever increasing technology—in terms of computational, networking and storage capabilities—and good connectivity. Users often carry such powerful handheld devices in their cars to take advantage of multimedia playback, navigation assistance as well as Internet connectivity. In the near future, many vehicles may be exploited as mobile sensors to gather, process and transmit data harvested along the roads in urban and extra-urban environments, potentially encompassing multiple types of information ranging from traffic/road conditions to pollution data and others.

As a matter of fact, until support for ad-hoc WiFi connectivity or for the new Wi-Fi-Direct standard[5] will be widespread, smartphone-based VSNs will require the presence of a communication infrastructure (e.g., 3G/LTE cellular networks, WiMax, etc.). Thus, they will share the advantages of V2I schemes over V2V technologies, namely a better support in Commercial Off-the-Shelf (COTS) equipment, native long-range communication capabilities as well as support for broadcast or multicast communication (at least at the application level). At the same time, cellular-network VSNs exhibit some disadvantages w.r.t. V2V schemes, such as higher latency at short distances, local communication obtained only indirectly and by adding overhead, and the need for service coverage along with the associated data traffic costs.

While hybrid communication schemes, i.e., combining V2V with V2I capabilities, would inherently provide the best overall solution (robust, efficient Internet-capable networking), we remark that the choice of an infrastructure-based communication does not constrain the upper level of data dissemination and processing, as well as the application level of the service organization of traffic information systems. In fact, a V2I infrastructure does not necessarily imply a centralized organization, which would inevitably lead to scalability issues, for example to cope with the information requirements of thousands or millions of vehicles moving around in a large metropolitan area. While multiple distributed organizations (e.g., hierarchical) can be deployed to achieve better scalability, a completely decentralized P2P approach is highly appealing. Initially exploited within V2V schemes [1], P2P approaches have been recently followed also for realizing decentralized traffic information system [2]. In fact, the P2P application layer supports the decentralization of the responsibility as well as of the computational and communication loads, which can also be beneficial for smartphone-based VSNs [3].

In this context, we introduce the D4V (DGT 4 VANET) architecture, whose objective is to define a scalable architecture for opportunistic dissemination of data provided by vehicle sensors and drivers, relying on commercial smartphones, rather than dedicated devices. D4V is based on the concept of Distributed Geographical Table (DGT), to properly manage and update information about the neighborhood of a vehicle involved in the system.

An almost complete overview of existing and emerging technologies and solutions for distributing and aggregating sensor data in vehicular networks has been proposed by Uichin and Gerla [33]. An example is MobEyes [35] that proposes a strategy for harvesting, aggregating, and distributing sensed data by means of proactive urban

[5] http://www.wi-fi.org/Wi-Fi_Direct.php.

monitoring services provided by vehicles that continuously discover, maintain, and process information about events of the urban scenario. Messages and summaries are routed to vehicles in the proximity, to achieve common goals, such as providing police cars with the trajectories of specific "target" cars.

A fundamental issue for VSNs is connectivity: different wireless access and communication methods have been evaluated, including Dedicated Short-Range Communication (DSRC) [36], WiMax/802.16e [37], WLAN [38], as well as cellular systems [39]. The use of a cellular communication network reduces the problem of implementing a working traffic information system (TIS), but introduces, on the other side, the issue of collecting data and distributing them to interested users.

A common approach is based on the client/server paradigm, where all data generated by vehicles are stored in a central server or a server farm on the Internet. Hull et al. [40] pointed out the major technical challenges of this solution, that are mostly related with the huge amount of simultaneous updates and queries generated by movements and requests of users (each car is a source of queries and sends it own measurements regularly).

For these reasons, in recent years researcher started investigating architectures based on the P2P paradigm, to build a distributed TIS where cars are not only consumers but also producers of information. Rybicki et al., with Peers on Wheels [3] and, more recently, with PeerTIS [41], have shown P2P architectures where participating cars are peers organized in a Distributed Hash Table (DHT). Roads are divided into road segments, each with a unique ID that is used as key in the DHT. The main idea is that each node is responsible for a certain part of the ID space and, consequently, for a certain number of road segments. Up to now one of the troubling issues is the fact that obtaining full information about planned and alternative routes is expensive in terms of bandwidth consumption. The work of Santa et al. [2] shows another P2P approach based on cellular networks (CNs) and on the JXTA middleware [42], to enable the transmission of information among vehicles and between vehicles and infrastructure, bounding the propagation of messages. CNs are also used not only in P2P solutions but also in participatory platforms, for participatory vehicular sensing [43–45], allowing applications such as ride quality monitoring, street-level traffic flow estimation, and proactive urban surveillance.

5.7.2 D4V

Most vehicular network safety applications need information from a very limited geographic area around the vehicle's current position. This may not be the case for driving comfort applications such as traffic intensity or traffic jam monitoring, as well as parking discovery [46] or guidance systems that distribute information about the traffic or road state for the entire city or for those regions where the car is located or is moving towards. Our goal is to design and build a reliable and scalable system capable of disseminating in an opportunistic way information coming from

driver's inputs or directly from one of the vehicle sensors, e.g., active shock absorber, cameras, engine, temperature sensors, etc.

Generally speaking, distributing information over long ranges in vehicular applications is a very challenging task in terms of how to gather, transport and aggregate such information. In this Section, we consider the case of network and user interfaced uniquely by a mobile device. The information that enriches the knowledge base of the car will be collected from internal and external data sources, namely vehicle or roadside infrastructure sensors. The on-board intelligence of the car extends, maintains, and disseminates this information by creating a local view of the car surroundings.

In the literature different techniques for content dissemination in VSN are described, like flooding and geocasting [47, 48], request/reply [49, 50], broadcasting, *sharing* [51] and beaconing [52, 53]. In the D4V (DGT for Vehicular Networks) architecture, we adopt the DGT scheme and the opportunistic and spatio-temporal dissemination approach proposed by Leontiadis and Mascolo [54], which is based on the publish/subscribe paradigm and allows message distribution to all interested receivers in an area by keeping messages alive in that zone for a specified period of time. On account of its properties, we believe that such an integrated solution copes well with a very dynamic scenario where users can easily and frequently change their subscription interests according to their planned path, current season, city neighborhood, etc.

D4V's basic message has been defined with: a *type*, for the notification category (for example, the class of traffic events or sensor data); a *location*, associated with the information; a *range*, that represents the area around message's location that the notification should reach; a *time to live* of the event; and a message *payload* containing—whenever necessary—additional and detailed information about the event.

Different types of messages can thus be distributed by the same dissemination protocol. It is possible to create, for example, a message to warn approaching users about a traffic queue or a dangerous situation, to distribute data extracted from the different sensors of the vehicle or to notify other users about a free space in a parking area. Each user selects the list of message types for which she/he is interested and adds this information to her/his peer descriptor, allowing other peers to send only appropriate messages according to the receiver's preferences.

When a new message is generated, the publisher picks up from its GBs the closest known nodes within the notification's range that are interested in the particular information type (by reading the peer descriptor), and sends them the new message, trying to avoid duplications. When a notification is received, the system checks if it still matches the user interests or if it does not (in the presence of dynamic subscription), or if it is already known. In the case of a new information, the node adds it to its knowledge and distributes it again to known interested peers.

When a peer receives the references about a new node in its area of interest, it checks if in its knowledge base there are notifications not yet expired that may be useful for that peer. If the target peer has not yet been contacted for the same reason, the node sends the message. During this dissemination processes, it is necessary to check if some messages have expired, and, consequently, to remove them and their

references from the vehicle knowledge base, thus avoiding the distribution of an obsolete notification.

5.8 D4V Simulation

In this section, we present the simulative analysis of a D4V-based application that allows vehicles to adapt their routes according to traffic information gathered from other vehicles in the area. The following set of performance metrics are considered:

- *Bandwidth* [(dimension: (kbyte/peer · s)]: Average message rate sent per peer per second.
- *CP*: Estimated coverage percentage of D4V messages (*TrafficInformation* and *SensorData*) at a certain time of the simulation. It is evaluated as the ratio between the number of peers that actually received a specific message and the number of those which should have it.
- *TJCP*: Average percentage of cars involved in a traffic jam.
- *DFE* [dimension (km)]: the Distance From Event is the average distance between the geographic location of a vehicle that did not receive a traffic jam message and the position of that event. This metric improves the information provided by CP. Indeed, a high DFE value (compared with the message range) means that drivers who do not receive the message are far from the dangerous situation and probably will receive the information shortly from the other neighbors that have been already informed.

By means of DEUS [30], with the packet delay model presented in Sect. 5.6.1, we have simulated a VSN deployed across the city of Parma, considering a number of vehicles that move over 100 km of realistic paths generated using the Google Maps API. Each simulated vehicle selects a different path and starts moving over it. Using the features provided by the Google API we have created a simple HTML and Javascript control page that allows the monitoring of the temporal progression of the simulated system, in which any node can be selected to view its neighborhood (videos are available at http://dsg.ce.unipr.it/d4v). The simulated D4V covers ten hours of system life (10,000 virtual time units) with 20 switch stations, 5 virtual tracks with bad road surface (either ice, water, snow, or pothole), accident events scheduled during the simulation according to a Poisson stochastic process and with different message types to disseminate information about sensed data and traffic situation. Simulations have been repeated with 5 different seeds for the random number generator, that are sufficient to obtain a narrow I_{95} confidence interval (± 10 % of the steady state value, in the worst case).

This simulation takes into account the configuration of GBs that was shown to obtain the best DGT performance in [4]. Each node has 4 GBs with a thickness of 0.5 km and a peer limit of 10 nodes, covering a region of interest of 12.5 km^2 and a dynamic discovery period ranging from 1.5 to 6 min depending on the number of discovered nodes.

The first step of the DEUS-based evaluation aims at analyzing the effect of varying the ε threshold ($\varepsilon \in [0.1; 1.0]$ km with a step equal to 0.3 km) considering two different vehicle densities $\delta = 10$ vehicles/km and $\delta = 20$ vehicles/km and a range of interest, for the disseminated messages, equal to 4 km. As defined earlier, ε represents the minimum displacement threshold considered by a peer to notify its geographic position update to nodes in its neighborhood. The analysis aimed at evaluating the effects of the variation of the update frequency on system performance and on information dissemination. Graphs in Fig. 5.44 show simulation results for the considered main metrics, in particular Fig. 5.44a illustrates the global CP as a function of ε values showing that traffic information messages are highly distributed to active peers in the configured range of interest. A higher peer density contributes to increase knowledge sharing, supporting the dissemination process given that more nodes receive and forward messages to interested drivers. In Fig. 5.44b the percentage of vehicles involved in a traffic jam is shown as a function of ε. The results confirms the robustness of the system which, even in the presence of a reduced update frequency, is able to properly distribute traffic information, leaving unchanged the percentage of drivers involved in the jam. The effectiveness of the approach with different position update thresholds can be observed also by analyzing the DFE value (Fig. 5.44c) that remains constant and very close to the dissemination range value of 4 km, confirming that peers that do not receive a traffic message are those located very far from the traffic event, thus having a good margin to receive the alert on time. This analysis suggests that vehicles involved in the queue are those that were really close to the traffic jam and had not enough time to react and change direction. Data traffic as a function of ε values is illustrated in Fig. 5.44d which confirms that finer position updates to neighbors yield to an increased network usage.

The second step of the evaluation has been driven by the goal of studying the D4V performance with respect to the variation of the dissemination range which is representative of the circular area around the message origin the notification should reach. Thus, in this scenario we vary it from 1 to 10 km to understand how it influences the dissemination process and its cost. Figure 5.45a, b illustrate the coverage percentage and the total number of vehicles involved in traffic jams, respectively. Both graphs show how a range of interest as small as 1 km affects the message distribution process due to a lower margin between the traffic jam and the drivers. In this situation, peers may receive alert messages when they are too close to dangerous situations, thus becoming involved in the queue. In particular, we remark how a lower vehicle density worsens such phenomenon because of the smaller number of nodes which can redistribute their knowledge about the traffic conditions. At the same time, it can be observed how there is no significant gap using range values larger than 4 km for both peer density curves. In Fig. 5.45c, the DFE value for the considered configurations is shown as a function of the range. For comparison, the optimal distance from the event is also shown. The latter coincides with the value of the dissemination range, because, ideally, the minimum distance of peers which did not receive the traffic information message yet is clearly the range of interest.

Results show that within a 4 km range the DFE remains close to the optimal bound, while higher values of the dissemination range decrease the DFE albeit still quite

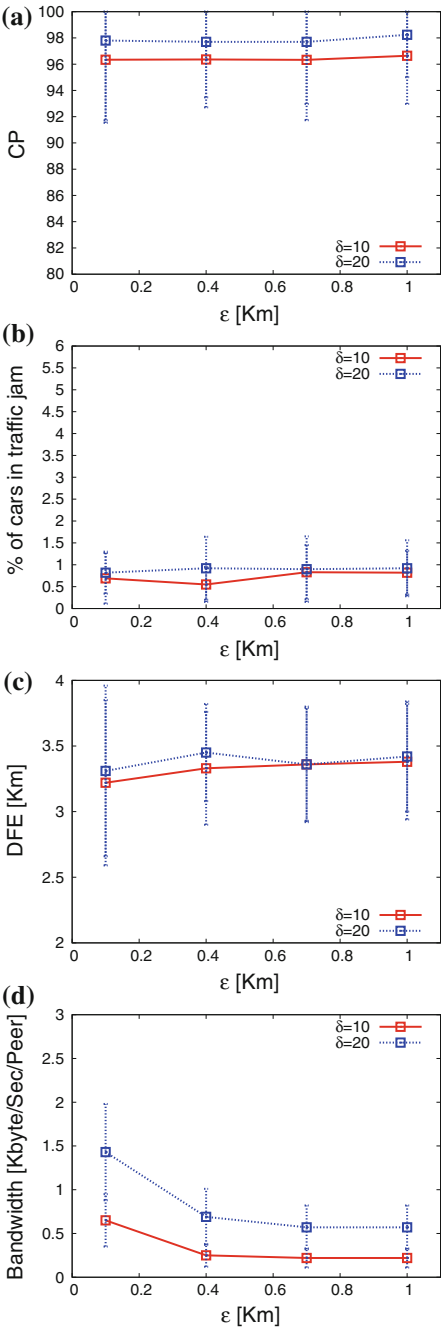


Fig. 5.44 Simulation results for different ε values: Coverage % (**a**). Number of vehicles in traffic jam (**b**). Distance from event (km) (**c**). Bandwidth (kbyte/peer/s) (**d**)

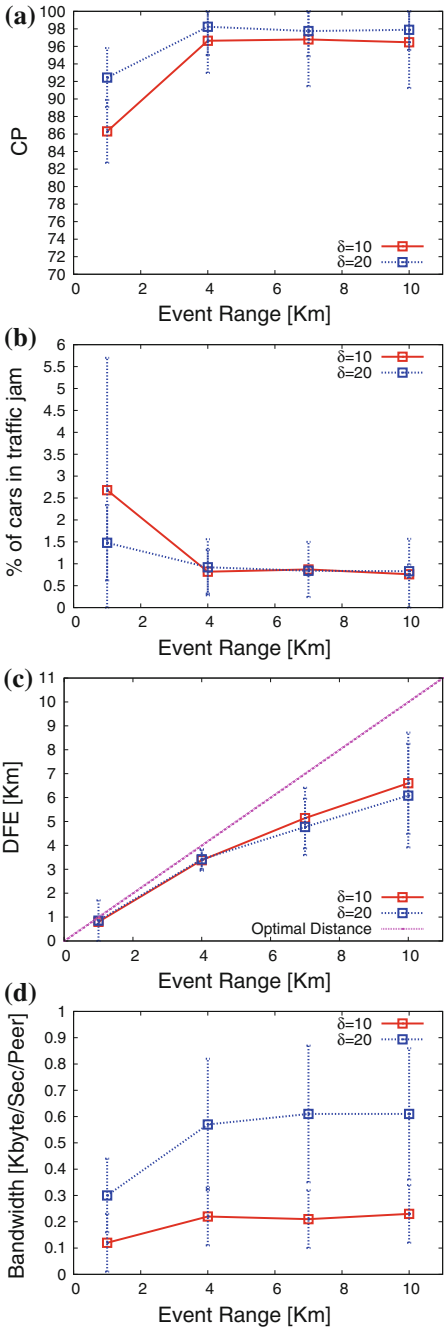


Fig. 5.45 Results for different values of the dissemination range: Coverage % (**a**). Number of vehicles in traffic jam (**b**). Distance from event (km) (**c**). Bandwidth (kbyte/peer/s) (**d**)

close to the bound. Hence, drivers who do not receive the alert for a specific event are in any case sufficiently far from it and will receive the alert with enough time to react. Finally, an extended event range corresponds to an increased notification area and, consequently, to a larger number of interested drivers that may be contacted. However—as shown in Fig. 5.45d—this slightly affects the amount of exchanged messages.

The third stage of the simulative analysis aims at evaluating the system performance with respect to the variation of the peer density in the vehicular network. The dissemination range of each node is set to 4 km (chosen according to previous results) and the same dynamic discovery period presented earlier. The scenario is characterized by an initially growing number of active vehicles, followed by a stable phase without new joins or disconnections. The results in Fig. 5.46a confirm that the proposed solution is able to cope with different node densities with no performance degradation, keeping the Coverage Percentage value significantly high (between 98 and 100 %)—even in the case of very low density (5 peers/km) which could be quite critical for VANET-based applications. We recall that, if a mobile peer finds itself in a desert area, it will still be able to fill its external geobucket with remote peers, by requesting their contacts to the bootstrap node. This distributed knowledge provides appropriate support to efficiently disseminate messages about traffic jams or sensed data. As in the second experiment, the results in Fig. 5.46c show that an increasing number of active peers maintains the DFE high and close to the dissemination range. This results into an accurate dissemination of traffic information messages that allows drivers to receive alert information on time, still sufficiently far from the dangerous location.

In Fig. 5.46b, the percentages of vehicles blocked in a traffic jam, with and without D4V content dissemination, are directly compared. This confirms that the D4V approach drastically reduces the number of involved vehicles that would otherwise grow significantly for increasing density.

Figure 5.46d shows the average data traffic per peer (in KB/sec/peer) which is necessary to maintain the DGT overlay and disseminate traffic information messages to other active neighbors, as a function of the density. Since we assume to use UDP as transmission protocol, there are no retransmissions in case of lost packets. Here we show the average bandwidth—estimated from simulative results, corrected considering the cost of the headers—consumed in the best case (when the transmitted message is much higher than the IP header) and in the worst case (when the transmitted message has a size that is comparable to the IP header, e.g., a location update, that contains only a peer descriptor and a location). Even if there is a moderate and natural increase associated with the growth of nodes, the amount of data exchanged by each peer remains very limited. This behavior is associated with the fact that, as described in the previous section, D4V is based on an opportunistic content dissemination strategy. D4V tries to minimize the amount of sent packets, by forwarding them only to interested users, trying, at the same time, to reduce the number of duplicated messages. The density values we considered are 5/10/20 vehicles per km. Higher values would be neither realistic nor interesting, as they would mean that all vehicles on the roads are running the DGT.

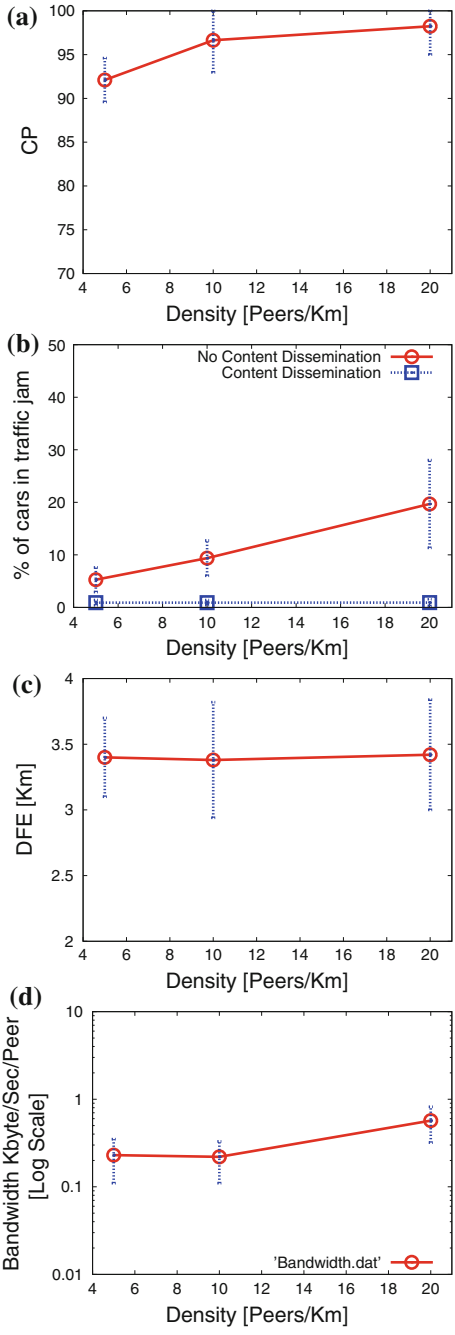


Fig. 5.46 Simulation results for different peer densities: Coverage % (**a**). Number of vehicles in traffic jam (**b**). Distance from event (km) (**c**). Bandwidth (kbyte/peer/s) (**d**)

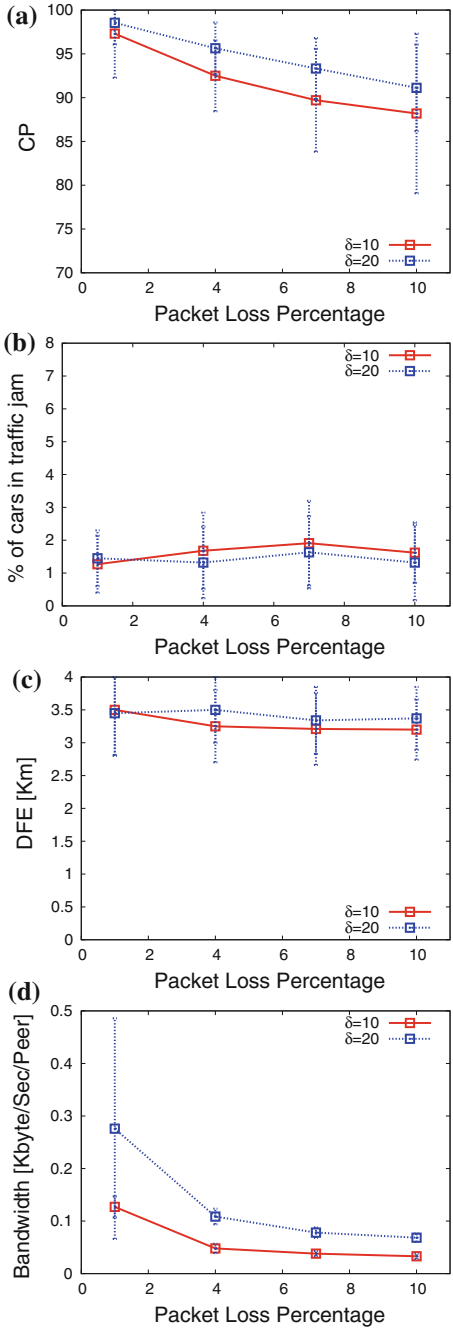


Fig. 5.47 Results for different Packet Loss Percentage: Coverage % (**a**). Number of vehicles in traffic jam (**b**). Distance from event (km) (**c**). Bandwidth (kB/peer/s) (**d**)

Figure 5.47 is dedicated to the analysis of the system robustness, in terms of packet loss percentage P, in a scenario with $\delta = 10$ peer/km and $\delta = 20$ peer/km, 4 GBs with thickness of 0.5 km, and a message dissemination range of 4 km. In the current form of the simulation code, there is no recovery procedure to verify whether a transmitted message is correctly delivered and, if it is necessary, to retransmit it. This is really important to properly interpret presented results, in particular for the dissemination of traffic information messages and the global robustness of the DGT approach. In Fig. 5.47a the global CP is shown as a function of the packet loss percentage, confirming that on average peers maintain a detailed knowledge of traffic events (>90 %) in the first GB.

In Fig. 5.47b, the percentage of vehicles involved in a traffic jam is investigated; it is a slightly increasing function of P, given that some peers may not receive alerts on the dangerous event and could be involved in the queue. The design and the distributed knowledge provided and maintained by the DGT allows to inform a large number of drivers keeping the number of queued vehicle really low. D4V robustness is also confirmed by the results in Fig. 5.47c, showing that the nodes that do not receive traffic information messages are considerably distant from the event location. Moreover, the DFE is almost independent of P. Figure 5.47d reports the data consumption, which is unavoidably lower than in the other scenarios due to the lack of a recovery procedure for lost packets.

Finally, considering the behavioral model of a driver in proximity of a road stretch with a bad surface condition, that we have introduced in Sect. 5.5, in Fig. 5.48 we show the monitored speed for five virtual tracks with bad surface conditions for all drivers (including both the informed ones and those not informed) that drive across the street during the simulation. The observed results clearly show that a decreased speed is measured near the critical location (at distance zero), along with an increasing velocity while moving away from it. Because of that, we can say that the deployment of D4V would probably reduce the risk of accidents and nuisances along troubled roads on account of the achieved information sharing among drivers, including those still approaching the dangerous point.

5.9 D4V Prototype

The simulative analysis of sample scenarios based on experimental measurements of coverage and connection throughput, carried out across/around Parma urban area, gave us valuable insights to start the development of the first release of a DGT Library and the first prototype of the D4V system. The library implements the base functionalities and policies of a DGT overlay such as the discovery procedure, the management of the neighborhood and the GBs maintenance. The D4V application layer uses such features to implement the content dissemination algorithm and the user interface to get the input from the drivers related to a specific traffic event, and to show approaching dangerous situations. The development of the DGT library started from our novel peer-to-peer middleware called *Sip2Peer* [55], which is an open-source SIP-based

middleware for the implementation of any peer-to-peer application or overlay without constrains on peer nature (traditional PC or mobile nodes) and specific architecture. At this moment sip2peer is available for Java SE and Android platforms, but we are working on the iOS release. The Java and Android implementations of the library are based on the Java SIP stack called *MjSip* [56] that allows to manage the exchange of SIP messages to control multimedia streams. Sip2Peer supports two message formats. It is possible to manage simple text message containing any kind of information like raw data or XML, and it is also possible to natively use the JSON format. Following this scalable approach, the main class of the Sip2Peer API, i.e., Peer, provides all necessary methods for sending and receiving messages, allowing the developer to select the best solution according to his/her protocol and overlay and solve problems related to NAT traversal.

Figure 5.49 groups together DGT and D4V modules, to present all involved elements whose integration defines the behavior of a peer. Such modules are presented and described in the following.

- *Sip2Peer Layer* (*SL*): Represents the communication module providing methods to receive and send messages from and to other active peers in the network. This layer interacts with the DMH (illustrated below) to route and forward outgoing and incoming messages, providing proper notifications when a packet has been correctly delivered or not.
- *DGT Message Handler/Dispacher* (*DMH*): Conveys and manages all DGT messages. It does notify neighbors position updates and redirect subscriptions and event messages from and to the Subscription Manager (detailed below).
- *Geo-Bucket Manager* (*GM*): Manages the data structures of the peer according to its geographic location. This module interacts with the Location Manager to be notified for a position update, and thanks to DMH notifications it is able to add new discovered peers or remove nodes out of the region of interest. Since it is a

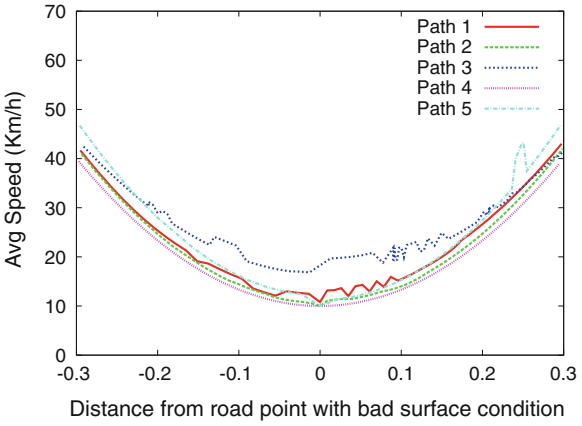


Fig. 5.48 Average of driver speed near road points with bad surface condition

base module of the DGT Library, it is configurable allowing to define the number of buckets and their thickness (target area) and the maximum number of nodes that a single GB can maintain during node life.

- *Location Manager* (*LM*): Subscribes to information and updates about device location through GPS, WiFi or cellular network, trying to minimize the energy consumption according to the context application characteristics. In detail in a urban scenario where the mobile is used inside a car where the energy consumption is not a constrain the location could be detected used external localization devices such as a navigator providing that potentially could provide additional and useful information about the planned route and the target location.
- *DGT Kernel* (*DK*): Is the core of a DGT node, implementing the routing strategy and the discovery procedure for neighborhood maintenance, as well as short/long range queries triggered by the user (through the User Interface) or periodically efficiently scheduled by the DK thread according to the application purpose and users settings. It also allows the interaction with DMH and GM to properly disseminate messages and alerts coming from UI or other external inputs.
- *Subscription Manager* (*SM*): It is related to the D4V prototype and has been designed to manage the subscription system of the node, allowing to add or remove subscriptions and handling an filtering incoming events or user queries. It does interact with the UI to notify relevant incoming alerts or messages, and sets preferences about the subscriptions.

The User Interface (UI) allows to present to the user all required information and interface elements, to control DGT functionalities, like dissemination an alert messages about traffic jams or to schedule a query concerning a region of interest. It allows to visualize on a map (or in a dedicated list view) peer/vehicle and neighbor locations, as well as scheduled query results. Our first prototype and the associated UI has been designed and developed on the Android platform.

When the he/she runs the application, the user watch a map displaying the updated vehicle location, and configure through a specific menu the ip of the bootstrapping node used to join the DGT network (Fig. 5.50a). When the first DGT discovery has been completed and the neighborhood is formed the user can see neighbor vehicles on the map view and alert messages near its geographic position (Fig. 5.50b). For this first prototype we gave the possibility to the user to generate information related to four different type of event through a dedicated and simple user interface (Fig. 5.50c). By clicking on the associated button the user can generate and distribute an alert message related to Traffic Jam, Car Accident, Man at Work and Bad Surface Condition. The generated message contains information about the geographic location, the event generation time, the source user and the expiration time (set by default to one hour), after which the DGT peer stop disseminating it, unless a new user refreshes the information. While the user is driving, changing its location, the application checks if one or more received traffic messages are close to the car in a range of 200 m, and notify with a dialog message the alert type and its location (Fig. 5.50d, e). Furthermore, the user can review in any moment the list of received

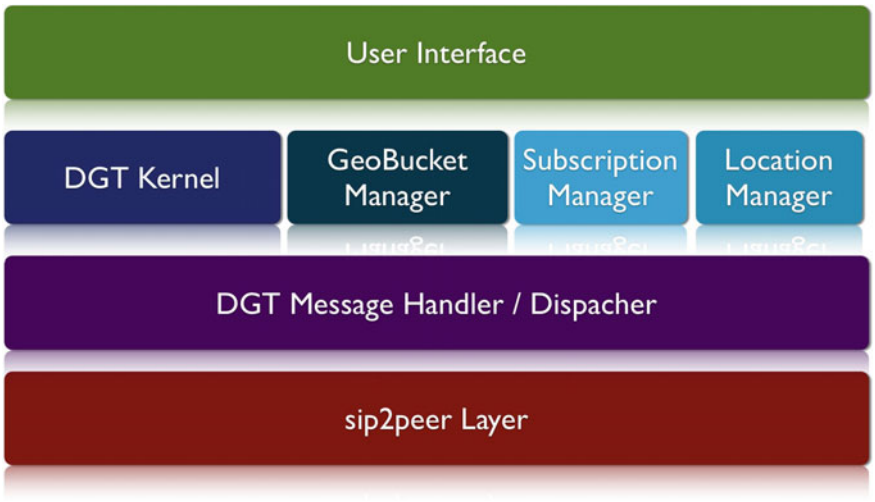


Fig. 5.49 D4V prototype modules

messages (Fig. 5.50f) sorted by distance, visualize them on the map, renew expired content or report an abuse or false information generated by a user.

Our plan was to evaluate the designed application during the development phase in a scenario with a relevant number of users before releasing the prototype to a group of alpha tester. To this purpose, an additional module has been created to emulate the behavior of vehicle moving along city streets. This module implements an FTM mobility model to evaluate the car speed, based on the switch station model presented in the previous chapter, and provides a web-based tool (Fig. 5.51) to monitor peer movements during experiments. Using this module has possible to create a complete and autonomous D4V node (called D4V-Bot) able to join the DGT network and generate a traffic message if required by the experiment setup.

5.9.1 Performance Evaluation of the D4V Prototype

A first evaluation phase has been conducted with a hybrid group of 50 nodes composed by real Android devices and D4V-Bots. The experiments have been conducted initially in a controlled environment of our laboratory (Fig. 5.52a, b). Peers were able to join the network, build their neighborhood and maintain it during the experiment, according to the changes of their geographic location. When a node (Android or Bot) generates a new message related to a traffic event, the latter was correctly distributed and shown on smartphones of user inside the region of interest of the event. Starting from the results of this first preliminary evaluation it was important to properly measure the network performance in order to understand if the results of the simulation

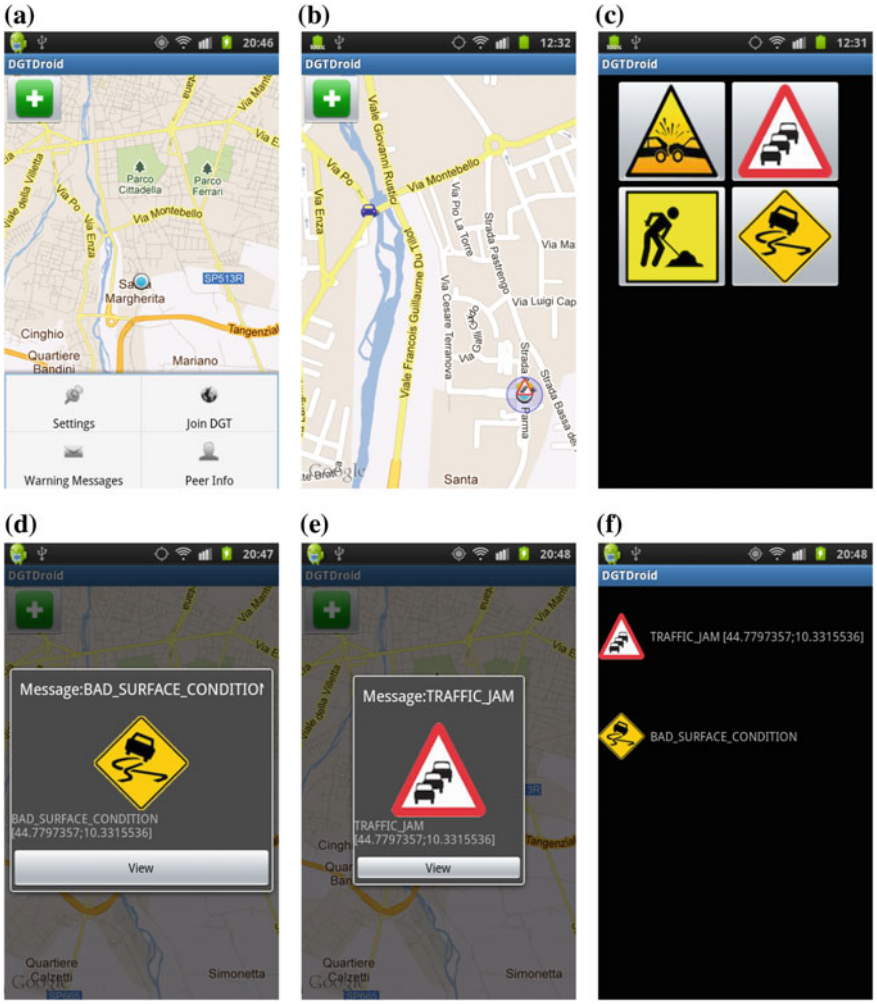


Fig. 5.50 Android DGT Prototype. **a** Settings menu. **b** Map view. **c** Traffic message creation view. **d**, **e** Bad surface condition and traffic jam messages popup view. **f** Incoming message list

analysis are confirmed by the results of the prototype test, also with an heterogeneity in terms of access network. For this purpose we deployed our D4V-Bot experiment on PlanetLab.

PlanetLab is a global research network that supports the development of new network services. Since the beginning of 2003, more than 1,000 researchers at top academic institutions and industrial research labs have used PlanetLab to develop new technologies for distributed storage, network mapping, peer-to-peer systems, DHTs, and query processing. PlanetLab currently consists of 1,089 nodes at 532 sites and

the University Of Parma contributes with 2 nodes in the world and european network. We deployed 50 D4V-Bots on 13 different PL servers in 13 different countries. Each node every 30 s logs on file a JSON string containing all the needed information to analyze the behavior the peer such as geographic location, exchanged kbytes, received and sent messages. At the end of the experiment a dedicated tool parses all available log files to build a time line of the experiment made by steps of 30 s containing all the required statistic for the performance evaluation.

Experiments results are based on five different runs of 26 min. The performance metrics that have been taken in account are:

- *Bandwidth* [dimension: (kbyte/peer · s)]: Average message rate sent per peer per second.
- *CP*: Estimated coverage percentage of D4V messages (*TrafficInformation* and *SensorData*) at a certain time of the simulation. It is evaluated as the ratio between the number of peers that actually received a specific message, and the number of those which should have it.
- *DFE* [dimension (km)]: the Distance From Event is the average distance between the geographic location of a vehicle that did not receive a traffic jam message, and

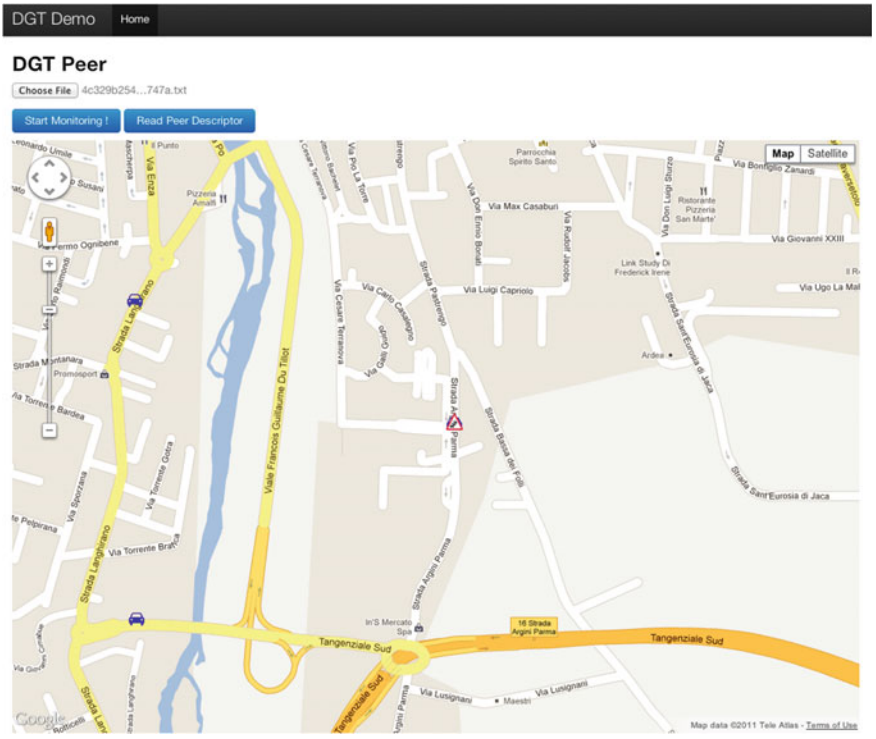


Fig. 5.51 D4V prototype web monitoring tool

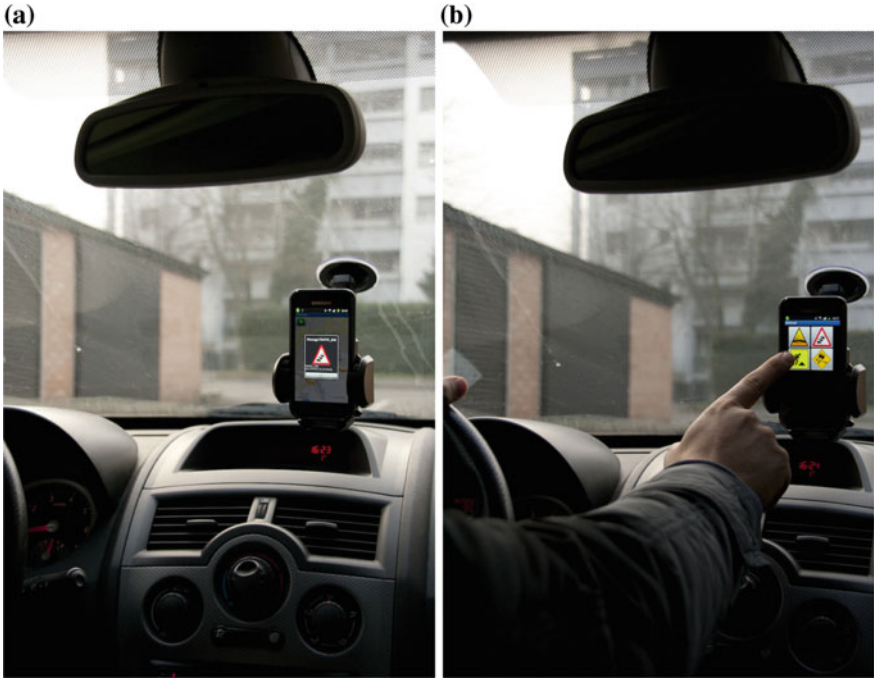


Fig. 5.52 Car setup. **a** Incoming warning. **b** Traffic message creation

the position of that event. This metric improves the information provided by CP. Indeed, a high DFE value (compared with the message range) means that drivers who do not receive the message are far from the dangerous situation, and probably will receive the information shortly from the neighbors that have been already informed.

- *Delay*: Represents the round-trip delay time (RTD). It is the length of time it takes for a signal to be sent, plus the length of time it takes for an acknowledgment of that signal to be received. This time delay therefore consists of the transmission time of a signal between the two points of a signal.
- *%Packet Loss*: Average % of Packet Loss for a peer during the experiment.

The graph in Fig. 5.53 shows the trend of the Coverage Percentage with the average value for the PlanetLab experiment and the same results obtained in the simulation analysis of our previous evaluation. The generation of traffic messages starts 400 seconds the D4V-Bots have started running, in order to give them enough time to build the DGT network. Results show that the average value of CP is really high close to 97 % and in particular significantly near the average value of our simulations (≅98 %). The CP curve shows that when new messages are generated the coverage percentage goes slightly down to lower value (≅88 %) but after one or two time line steps (30/60 s) recovers to an high coverage percentage confirming

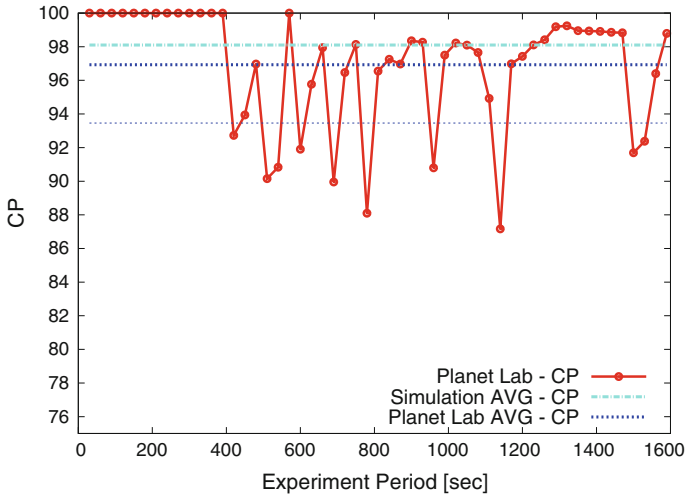


Fig. 5.53 PlanetLab—coverage percentage

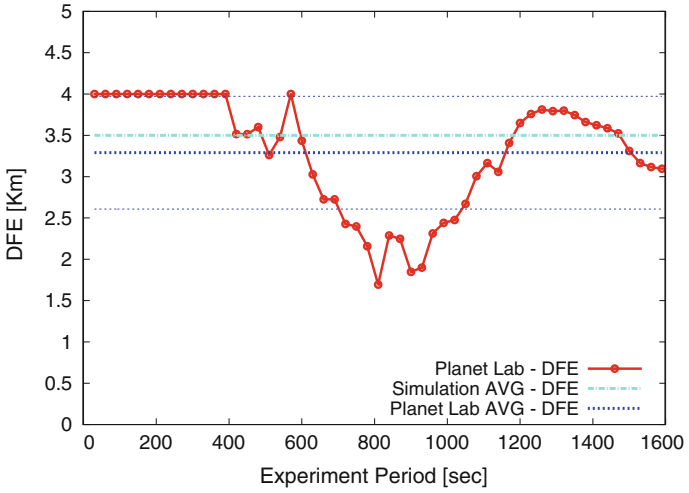


Fig. 5.54 PlanetLab—distance from event (DFE)

that the dissemination process and the neighborhood knowledge allow to efficiently distribute messages.

An additional performance metrics that allows to better understand the behavior of the protocol is the DFE values. For such an experiment we have considered a range of interest for disseminated message of 4 km. Figure 5.54 illustrates the DFE trend comparing PlanetLab and simulation average, that also in this case are really close and around ($\cong$3.5 km). The graph confirms that vehicles that did not receive the message

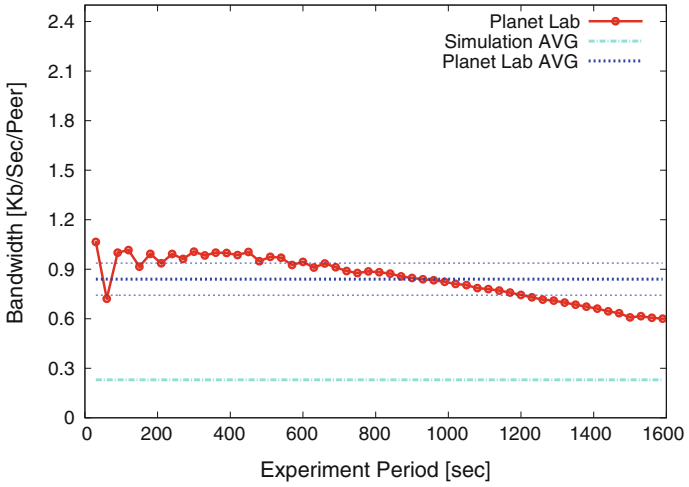


Fig. 5.55 PlanetLab—average bandwidth

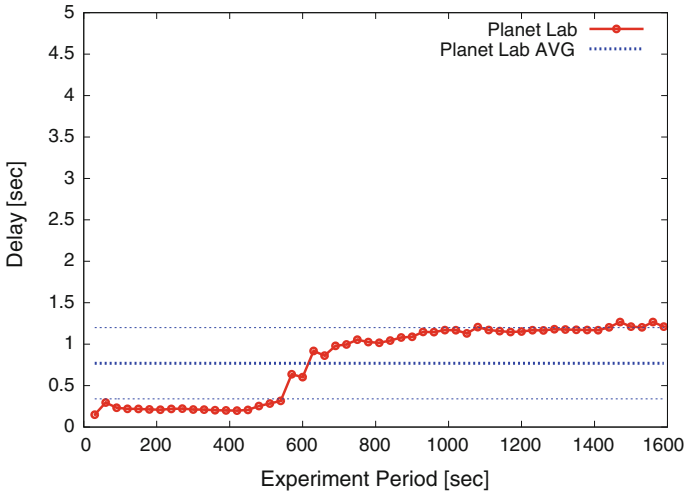


Fig. 5.56 PlanetLab—average delay

are on average really far from the dangerous event, and have a sufficient margin to receive the message before approaching the potentially dangerous location changing their direction to reach their destination using a different route or just adapting their vehicle speed for example near a portion of damaged road surface.

The graph in Fig. 5.55 reports the bandwidth in terms of kb/s/Peer during the experiments conducted on PlanetLab. In this case the average value (≅0.3 kb/s/Peer) is different from the same value obtained during the simulation (≅0.9 kb/s/Peer).

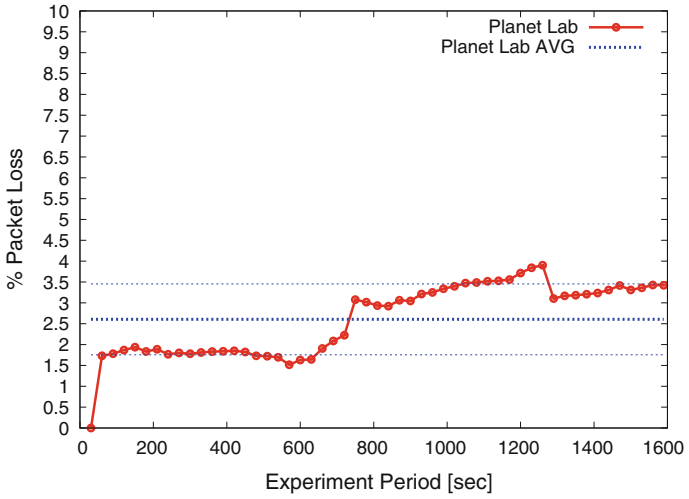


Fig. 5.57 PlanetLab—average packet loss

This difference is can be related to the fact that in the real implementation there is an additional overhead due to packet header and additional exchanged information, that initially was not considered during the modeling of the communication in our simulator.

The graphs in Figs. 5.56 and 5.57 finally illustrate the trend and the average of delay and the percentage of packet loss measured during the experiments. The delay is reasonable for the designed application, considering that the small package size and that the experiments have been done on 13 servers located in different research institutions, with different network capabilities and load during the experiment (PlanetLab nodes are used at the same time by several application and the condition could change in the course of the experiment). After the evaluation on the PlanetLab network the next phase has been dedicated to implement and optimize the basic implemented library trying to reduce the amount of messages exchanged among connected nodes. The first optimization is related to the number of exchanged messages during the publication and discovery phases. Each DGTEvent message is attached with a list of size L containing the descriptors of peers which have already received that message. In this way, the message is forwarded only to nodes interested to it *and* not already receiving it. Moreover, the messages are forwarded by a peer to its neighborhood with a given probability p. It is straightforward to observe that the forwarding probability has to be properly set. In fact, if this parameter is too low, a given message may be stopped somewhere in the network, and no more propagated. Another implemented modification is a method to inform when a node is disconnected from the network. In the original solution, a node is recognized as disconnected if it does not reply to a presence message. In the modified solution, instead, if a node wants to disconnect from the network, it informs its neighborhood.

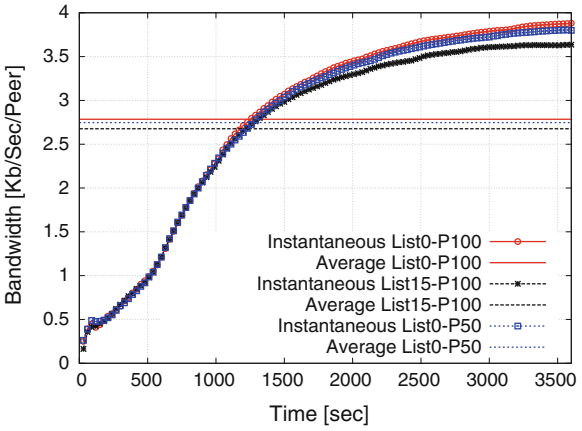


Fig. 5.58 Comparison of bandwidth consumptions for analyzed configurations

Finally, the DGT messages have been provided with a time to live (TTL), as well as the possibility to reduce the list size from the bootstrap node. Also these optimizations have been made to reduce the bandwidth usage. A final important feature, inherited from Sip2Peer, is the SBC-based NAT traversal mechanism, which allows the peers to communicate in presence of NATs—a very relevant problem especially in mobile scenarios. The SBC (Session Border Controller) is a node provided with a public IP address. It allows a generic peer to check if it is in a private network with a NAT, also requesting (if needed) a IP address/port pair to communicate with other peers.

The updated library has been tested both on traditional PCs and on real Android devices to validate the designed behaviors with different mobility patterns and network coverages. Consecutively, an additional evaluation of the DGT Java Library has been carried out by deploying 200 peers on a cluster at our Department. Each peer is randomly placed in the area of the city of Parma. All peers are connected to the DGT overlay network and can move according to the FTM mobility model described by (5.11). Performance evaluation is performed by letting each peer logging every 30 s its speed, geographical position, and amount and type of exchanged data. During network operations, another peer is generated and selected as the bootstrap node which generates random events, e.g., accidents or road works. Three scenarios have been analyzed, changing L and p. In the first case, $L = 0$ and $p = 1$; in the second scenario, $L = 15$ and $p = 1$; in the third scenario, $L = 0$ and $p = 0.5$. Finally, 5 independent runs have been performed to average over statistical fluctuations.

In Fig. 5.58, the bandwidth [dimension: (kB/s)] is shown, as a function of time, for the three considered scenarios. Both time-instantaneous and average values are shown.

One can observe that the bandwidth increases with time, since more and more packets and messages are generated during the experiments. For a fixed value of p,

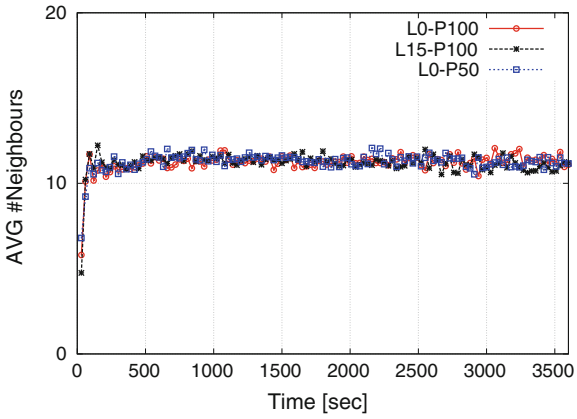


Fig. 5.59 The average number of neighbours for each active peer

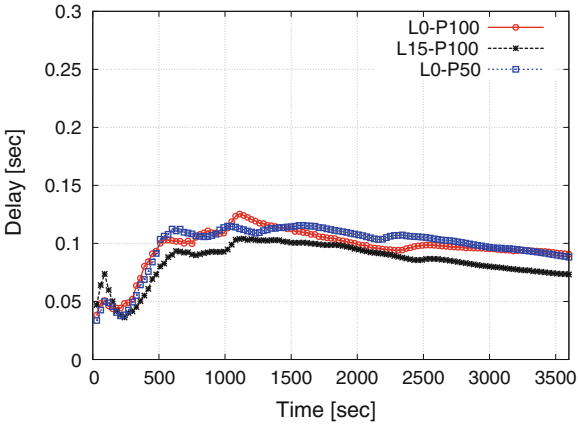


Fig. 5.60 The average delay measured during the experiment

increasing the value of L leads to better performance. In fact, each forwarding node is aware of more nodes which have already received a given message. Moreover, for a fixed value of L, decreasing the forwarding probability p reduces the bandwidth usage, due to the fact that less packets are transmitted across the network.

Since from Fig. 5.58 the lowest bandwidth usage can be obtained with $L = 0$ and $p = 0.5$, we now analyze the delay and neighboring performance in this scenario. In Figs. 5.59 and 5.60, the delay (D) and the number of neighbors (NN) are reported, as functions of time, for $L = 0$ and $p = 0.5$. Both time-instantaneous and average values are shown.

Note that both D and NN are stable around their mean value, thus showing that the effectiveness of the proposed DGT approach is confirmed in the experimental setting as well.

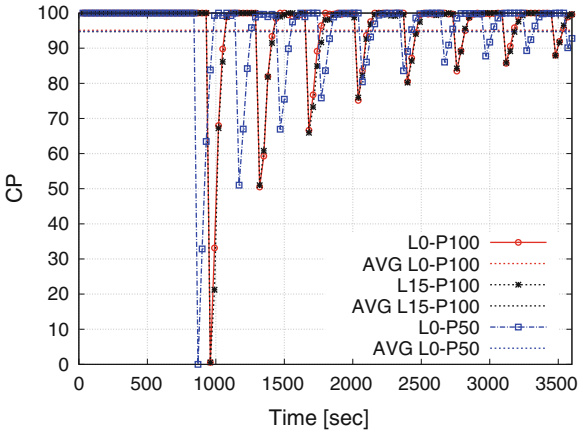


Fig. 5.61 Coverage percentage trends of evaluated scenarios

Graph in Fig. 5.61 shows the trend analysis of the Coverage Percentage during the experiment for the three considered scenarios in terms of p and L. At the beginning of the evaluation the measured metric is always at 100 % since there are not events in the system and the peers of the DGT network are just maintaining the geographic routing tables while while they are moving according the implemented behavior and model. After 15 min a new node called Event Generator (EG), responsible for the generation of traffic events (selected among previously introduced types), joins the P2P network and when has built its initial neighborhood starts a periodic (with an average time $T_d = 4$ min) internal process to disseminate random event along the streets. Lower peaks in the graph trends are related to the instant of the generation of the event when the percentage of nodes that have received the alert in the area of interest is naturally low. At the beginning of the EG life the peaks are higher since the peer is still building and improving its geographic knowledge. Nevertheless the recovery time needed to bring the coverage percentage over the 90 % is significantly low and in the vicinity of a few number of time slots of 30 s (on average around 1.5 min to recover over the 90 %).

As previously illustrated, the coverage percentage is only the first aspect of the D4V analysis. In order to understand which is the distance from the event (DFE) of those peers that did not received yet the alerts generated by EG. The trends of the DFE metric in the evaluated scenarios illustrated in Fig. 5.62 confirms the excellent performance of the DGT overlay and its capability to properly disseminate geo-referenced messages according to D4V specifications. Reported values show how the average distance from the event during the overall experiment is significantly high (around 3.7 km) with only a lower peak of 1 km a the beginning of EG file (when its geographic knowledge is lower). Both presented graphs illustrate how the use of different values of L and p affect as expected only the bandwidth consumption and not the D4V behavior. CP and DFE have the same trend with negligible differences

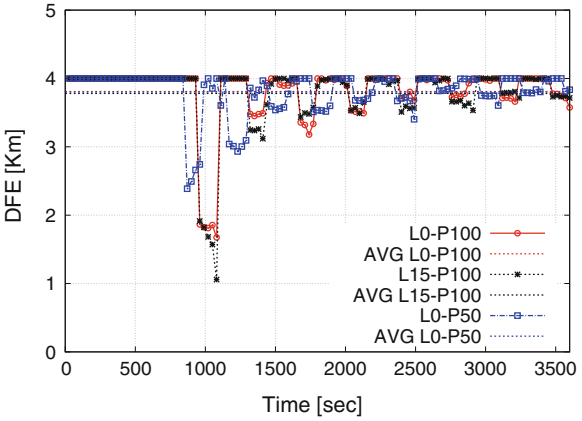


Fig. 5.62 Distance from event trends of evaluated scenarios

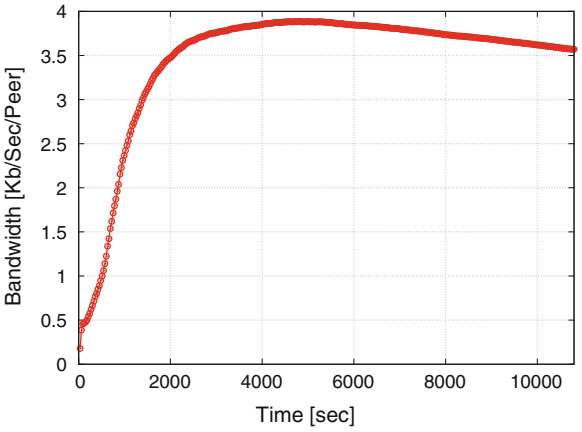


Fig. 5.63 Bandwidth consumption for an experiment of three hours with L = 15, p = 100 and 200 active nodes

attributable to the randomness of the mobility patterns and of the events generation that in a real evaluation are not easy to control.

At a later stage and in order to evaluate the developed library during a longer experiment, we have performed a new evaluation using the better configuration in terms of bandwidth consumption (with $L = 15$ and $p = 100$) and a duration of three hours with 200 nodes and a single event generator EG that joins and generates traffic related events after the first 15 min.

The first graph showed in Fig. 5.63 reports the average bandwidth during the experiment showing how it grows at the beginning while the DGT/D4V network is the initial phase and active nodes are improving and building their distributed geographic knowledge. Step by step the network reaches a stable state converging to

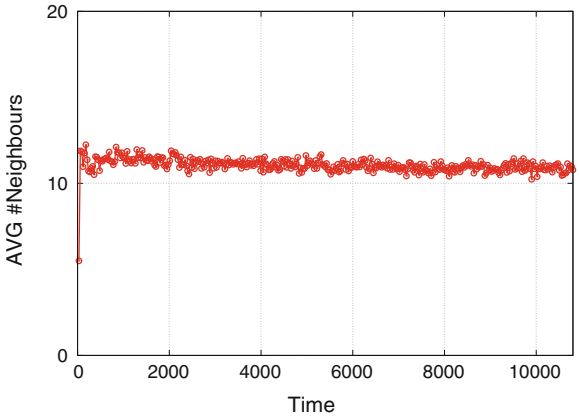
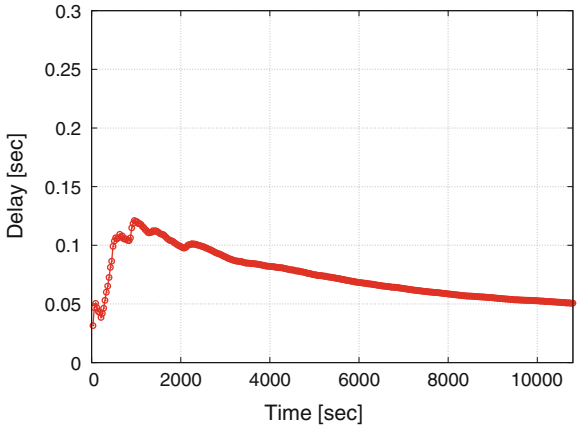


Fig. 5.64 The average number of neighbours for each active peer

Fig. 5.65 The average delay measured during the experiment of 3 h with LA $= 15$, p $= 100$ and 200 active nodes



a lower average bandwidth value. In any case showed bandwidth consumption are absolutely feasible for existing GPRS, UTMS and 3G cellular networks consuming only a small amount of the available capacity.

Graphs in Figs. 5.64 and 5.65, as in previous analysis, show respectively how the average delay and the number of neighbors are stable around their mean value also during a longer simulation confirming the effectiveness of the proposed DGT/D4V solution.

Figure 5.66 reports the trend of the average global coverage percentage during the performed evaluation and allows to confirm the behaviour identified with the previous analysis also during a significantly longer experiment. The graph shows how the CP has some lower peaks only at the beginning while the EG knowledge is still in an initial phase and step by step when the global distributed knowledge

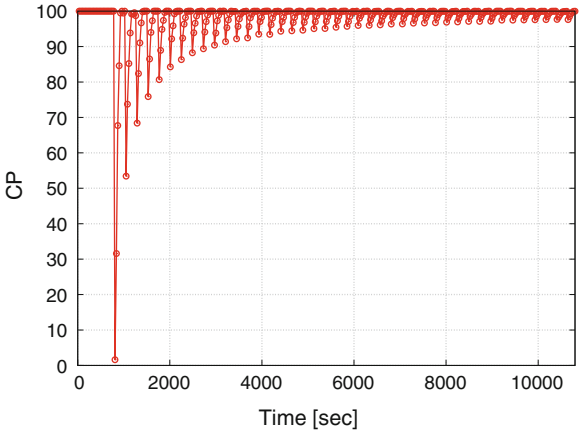


Fig. 5.66 Coverage percentage trends during the experiment of 3 h with L = 15, p = 100 and 200 active nodes

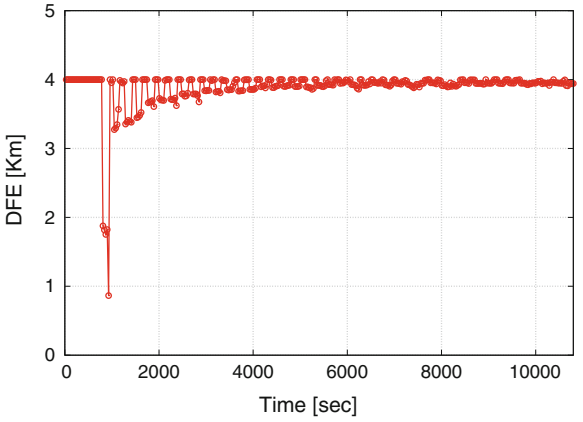


Fig. 5.67 Distance from event trends during an experiment of 3 h with L = 15, p = 100 and 200 active nodes

is higher the required time to recover the percentage over the 90 % is dramatically reduced and on average under 30 s.

This suitable and expected behavior is replicated in Fig. 5.67 showing the average distance from event of active peers in the network. It confirms the excellent performance to properly disseminate alert messages among interested and close neighbors and shows how the network is able to increase its effectiveness while the distributed knowledge is growing up. After an initial phase where the DFE metric has some lower peaks the convergence reaches remarkable values really close to the

limit of 4 km of the region of interest. Presented results completely depicts the performance and the behaviour of the designed and implemented overlay where active nodes can efficiently disseminate and receive traffic alert messages (through their smartphones) with reduced dissemination periods and with an high average distance from the potentially dangerous location giving them the time to change driving direction, reduce the vehicle speed and generally react and be focused on their driving.

5.10 Concluding Remarks

In this Chapter, we introduced and presented the design, implementation and evaluation of a novel structured P2P overlay scheme, called *Distributed Geographic Table* (DGT), which allows its (mobile) participants to efficiently retrieve node and resource information (data or services) located near any chosen geographic position. In a DGT network the responsibility for maintaining information about position of active users is properly distributed among nodes, for which a change in the set of participants causes only a minimal amount of disruption, without reducing the quality of provided services.

The following objectives guided the design of DGT:

- avoiding that a central node, or a particular peer, could become a bottleneck, or even a single point of failure, for the whole system because of its responsibility on a large subset of the network or of a specific geographic region.
- reducing the risk of disseminating obsolete information in the network;
- managing, by means of a distributed strategy, a high amount of simultaneous updates and queries, while limiting the platform cost—thus, making easier the access to the market;
- to build an overlay where neighbors in the P2P overlay are at the same time real geographic neighbors, thus allowing to avoid the need to forward additional messages to discover closest nodes;
- to properly evaluate and take into account the heterogeneity and mobility of user devices.

We evaluated the discovery procedure of DGT by means of analytical methods, and state-of-art simulation techniques, which enabled a high level of realism for the modeled scenarios. The DGT algorithms achieve convincing performance for extended ranges of system parameters, and properly maintain the neighborhood knowledge of each peer, discover new nodes and updated information with reduced overhead in terms of exchanged messages. All parameters in a DGT overlay can be tuned in order to achieve the most suitable trade-offs and performance, for the target application.

In the second part of the chapter, we illustrated a DGT-based TIS called D4V. In such a system, user nodes can be installed on smartphones, or in car equipment, to

send and receive real-time information about traffic conditions or potentially dangerous situations. The D4V architecture was evaluated by means of extensive simulations based on established vehicular mobility models, and through the deployment of a mixed network including both virtual and real users with Android devices.

A D4V prototype, based on the Sip2Peer open source middleware developed in our laboratory, has been deployed on the PlanetLab testbed, to verify network performance in a heterogeneous environment. Simulative and real measurements are very close, thus confirming the reliability of the simulation model, as well as the good performance of the DGT overlay. The scheme guarantees a high coverage, in terms of vehicular notification, over a wide range of system parameter values, whilst generating limited data traffic and well coping with significant packet losses. Hence, we are confident that D4V could be effectively used on the road to reduce the number of drivers involved in traffic jams, as well as to disseminate alert messages about potentially dangerous road stretches, thus allowing drivers to reduce risks and nuisances along their paths.

References

1. Abuelela, M., Olariu, S.: Zipper: A zero-infrastructure peer-to-peer system for vanet. In: International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM). Chania, Crete Island, Greece (2007)
2. Santa, J., Moragon, A., Gomez-Skarmeta, A.F.: Experimental evaluation of a novel vehicular communication paradigm based on cellular networks. In: IEEE Intelligent Vehicles Symposium. Eindhoven, Netherlands (2008)
3. Rybicki, J., Scheuermann, B., Kiess, W., Lochert, C., Fallahi, P., Mauve, M.: Challenge: Peers on wheels-a road to new traffic information systems. In: Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking (2007)
4. Picone, M., Amoretti, M., Zanichelli, F.: Proactive neighbor localization based on distributed geographic table. In: 8th ACM-SIGMM International Conference on Advances in Mobile Computing and Multimedia (MoMM 2010). Paris, France (2010)
5. Picone, M., Amoretti, M., Zanichelli, F.: Proactive neighbor localization based on distributed geographic table. Int. J. Pervasive Comput. Commun. **7**(3), 240–263 (2011)
6. Amoretti, M.: A survey of peer-to-peer overlay schemes: Effectiveness, efficiency and security. Recent Pat. Comput. Sci. **2**(3), 195–213 (2009)
7. Aberer, K., Alima, L., Ghodsi, A., Girdzijauskas, S., Haridi, S., Hauswirth, M.: The essence of p2p: A reference architecture for overlay networks. In: 5th IEEE International Conference on Peer-to-Peer Computing (P2P 2005). Konstanz, Germany (2005)
8. Granovetter, M.: The strength of weak ties. Am. J. Sociol. **78**(6), 1360–1380 (1973)
9. Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the xor metric. In: First International Workshop on Peer-to-peer Systems (2002)
10. eMule Team: eMule project homepage. http://www.emule-project.net
11. Wolinsky, D.I., St Juste, P., Boykin, P.O., Figueiredo, R.: Addressing the p2p bootstrap problem for small overlay networks. In: IEEE Tenth International Conference on Peer-to-Peer Computing (P2P), pp. 1–10. IEEE (2010)
12. Fewell, M.: Area of common overlap of three circles. http://www.dtic.mil/cgi-bin/GetTRDoc?Location=U2&doc=GetTRDoc.pdf&AD=ADA463920 (2006)

13. Hartenstein, H., Laberteaux, K.: VANET Vehicular Applications and Inter-Networking Technologies. Wiley, New York (2010)
14. Harri, J., Filali, F., Bonnet, C.: A framework for mobility models generation and its application to inter-vehicular networks. In: 3rd IEEE International Workshop on Mobility Management and Wireless Access (MobiWac‡05). Cologne, Germany (2005)
15. Fiore, M., Harri, J., Filali, F., Bonnet, C.: Vehicular mobility simulation with vanetmobisim. Simulation **87**(4), 275–300 (2009)
16. Zhou, B., Xu, K., Gerla, M.: Group and swarm mobility models for ad hoc network scenarios using virtual tracks. In: IEEE Military Communications Conference (MILCOM). Monterey, California, USA (2004)
17. Seskar, I., Marie, S., Holtzman, J., Wasserman, J.: Rate of location area updates in cellular systems. In: IEEE Vehicular Technology Conference (VTC'92). Denver, Colorado, USA (1992)
18. Krauss, S., Wagner, P., Gawron, C.: Metastable states in a microscopic model of traffic flow. Phys. Rev. E **55**(1), 55–97 (1997)
19. Krauss, S., Wagner, P., Gawron, C.: Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E **62**(2), 1805–1824 (2000)
20. Gustafsson, E., Jonsson, A.: Always best connected. Phy. Rev. E **10**(1), 49–55 (2003)
21. Olivera, J., Cortazar, I., Pinart, C., Los Santos, A., Lequerica, I.: Vanba: A simple handover mechanism for transparent, always-on v2v communications. In: 69th IEEE Vehicular Technology Conference (VTC2009-Spring). Barcelona, Spain (2009)
22. Yan, Z., Zhou, H., Zhang, H., Zhang, S.: Speed-based probability-driven seamless handover scheme between WLAN and UMTS. In: 4th International Conference on Mobile Ad Hoc and Sensor Networks. Wuhan, China (2008)
23. Kwak, D., Mo, J., Kang, M.: Investigation of handoffs for IEEE 802.11 networks in vehicular environment. In: 1st International Conference on Ubiquitous and Future Networks (ICUFN). Hong Kong (2009)
24. Esposito, F., Vegni, A., Matta, I., Neri, A.: On modeling speed-based vertical handovers in vehicular networks. In: IEEE Global Telecommunications Conference (GLOBECOM'10). Miami, Florida, USA (2010)
25. Balasubramaniam, S., Indulska, J.: Vertical handover supporting pervasive computing in future wireless networks. Comput. Commun. J. **27**(8), 708–719 (2003)
26. Gershenson, C., Heylighen, F.: How can we think the complex? In: Richardson, K. (ed.) Managing Organizational Complexity: Philosophy, Theory and Application, pp. 47–71. Information Age Publishing, Charlotte (2005)
27. Zeigler, B.P., Praehofer, H., Kim, T.G.: Theory of Modeling and Simulation, 2nd Edn. Academic Press, Massachusetts (2000)
28. Wainer, G.: Cd++: A toolkit to develop DEVS models. Softw.: Pract. Experience **32**(13), 1–46 (2002)
29. Varga, A., Hornig, R.: An overview of the OMNeT++ simulation environment. In: 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools 2008). Marseille, France (2008)
30. Amoretti, M., Agosti, M., Zanichelli, F.: Deus: A discrete event universal simulator. In: 2nd ICST/ACM International Conference on Simulation Tools and Techniques (SIMUTools 2009). Rome, Italy (2009)
31. Baldo, N., Requena-Esteso, M., Nin-Guerreo, J., Miozzo, M.: A new model for the simulation of the LTE-EPC data plane. In: 5th ICST/ACM International Conference on Simulation Tools and Techniques (SIMUTools 2009). Sirmione, Italy (2012)
32. Papoulis, A.: Probability, Random Variables, and Stochastic Processes. Mcgraw-Hill College (1991)
33. Lee, U., Gerla, M.: A survey of urban vehicular sensing platforms. Comput. Netw. **54**(4), 527–544 (2010)
34. Gerla, M., Kleinrock, L.: Vehicular networks and the future of the mobile internet. Comput. Netw. **55**(2), 457–469 (2011)

35. Lee, U., Magistretti, E., Zhou, B., Gerla, M., Bellavista, P., Corradi, A.: Mobeyes: Smart mobs for urban monitoring with vehicular sensor networks. IEEE Wireless Commun. **13**(5), 52–57 (2006)
36. Jiang, D., Taliwal, V., Meier, A., Holfelder, W., Herrtwich, R.: Design of 5.9 Ghz DSRC-based vehicular safety communication. IEEE Wireless Commun. **13**(5), 36–43 (2006)
37. Han, M., Moon, S., Lee, Y., Jang, K., Lee, D.: Evaluation of VoIP quality over WiBro. In: Passive and Active Measurement Conference (PAM). Cleveland, Ohio, USA (2008)
38. Hadaller, D., Keshav, S., Brecht, T., Agarwal, S.: Vehicular opportunistic communication under the microscope. In: Internatioal Conference on Mobile Systems, Applications and Services (MobiSys). San Juan, Puerto Rico (2007)
39. Qureshi, A., Carlisle, J., Guttag, J.: Tavarua: Video streaming with wwan striping. In: 14th Annual ACM International Conference on Multimedia. Santa Barbara, California, USA (2006)
40. EHull, B., Bychkovsky, V., Zhang, Y., Goraczko, M., Miu, A., Shih, E., Balakrishnan, H., Madden, S.: Cartel: A distributed mobile sensor computing system. In: 4th ACM Conference on Embedded Networked Sensor Systems (SenSys). Boulder, Colorado, USA (2006)
41. Rybicki, J., Scheuermann, B., Koegel, M., Mauve, M.: Peertis: a peer-to-peer traffic information system. In: Proceedings of the Sixth ACM International Workshop on VehiculAr InterNETworking (2009)
42. Team, J.: JXTA technology: Creating connected communities. Tech. rep, Sun Microsystems (2004)
43. Mohan, P., Padmanabhan, V., Ramjee, R.: Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In: 4th ACM Conference on Embedded Networked Sensor Systems (SenSys). Raleigh, North Carolina, USA (2008)
44. Burke, J., Estrin, D., Hansen, M., Parker, A., Ramanathan, N., Reddy, S., Srivastava, M.: Participatory sensing. In: WSW'06. Boulder, Colorado, USA (2006)
45. Mun, M., Reddy, S., Shilton, K., Yau, N., Boda, P., Burke, J., Estrin, D., Hansen, M., Howard, E., West, R.: Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In: International Conference on Mobile Systems, Applications and Services (MobiSys). Krakow, Poland (2009)
46. Caliskan, M., Graupner, D., Mauve, M.: Decentralized discovery of free parking places. In: 3rd International Workshop on Vehicular Ad Hoc Networks. Los Angeles, California (2006)
47. Tonguz, O., Wisitpongphan, N., Bai, F., Mudalige, P., Sadekar, V.: Broadcasting in VANET. In: Workshop on Vehicular Ad Hoc Networks (Move'07). Montreal, Canada (2007)
48. Bronsted, J., Kristensen, L.: Specification and performance evaluation of two zone dissemination protocols for vehicular ad-hoc networks. In: 39th Annual Simulation Symposium. Huntsville, Alabama (2006)
49. Zhao, J., Cao, G.: Vadd: Vehicle-assisted data delivery in vehicular ad hoc networks. IEEE Trans. Veh. Technol. **57**(3), 1910–1922 (2007)
50. Wegener, A., Hellbruck, H., Fischer, S., Schmidt, C., Fekete, S.: Autocast: An adaptive data dissemination protocol for traffic information systems. In: 66th IEEE Vehicular Technology Conference (VTC '07). Dublin, Ireland (2007)
51. Shinkawa, T., Terauchi, T., Kitani, T., Shibata, N., Yasumoto, K., Ito, M., Higashino, T.: A technique for information sharing using inter-vehicle communication with message ferrying. In: IEEE International Conference on Mobile Data Management (MDM '06). Nara, Japan (2006)
52. Xu, H., Barth, M.: An adaptive dissemination mechanism form intervechicle communication-based decentralized traffic information system. In: 9th International IEEE Conference on Intelligent Transportation Systems (ITCS '06). Toronto, Canada (2006)
53. Fujiki, T., Kirimura, M., Umedu, T., Higashino, T.: Efficient acquisition of local traffic information using inter-vehicle communication with queries. In: 10th International IEEE Conference on Intelligent Transportation Systems (ITCS '07). Seattle, Washington, USA (2007)

54. Leontiadis, I., Mascolo, C.: Opportunistic spatio-temporal dissemination system for vehicular networks. In: International Conference on Mobile Systems, Applications and Services (MobiSys). San Juan, Puerto Rico (2007)
55. Distributed Systems Group: ip2 peer project home page. http://code.google.com/p/sip2peer/ (2013)
56. MjSip Team: Mjsip project official page. http://www.mjsip.org (2013)

Appendix A
DEUS: A Simple Tool for Complex Simulations

Multi-scale modeling and simulation (M&S) are necessary for the design and analysis of smart cities, that are particularly complex systems. The discrete event approach is particularly suited, allowing to focus on state changes in correspondence of meaningful events, thus reducing the execution time of simulations. To support this claim, we show how we can model a fundamental component of smart cities, namely the traffic information system, with our discrete event simulation environment, called DEUS [1]. A holistic simulation platform for emergent phenomena which include dependent systems does not exist yet. However, DEUS is flexible and general-purpose enough, to enable the effective simulation of several types of actors, devices, infrastructures, etc.

In the Sect. A.1 we recall the main features of DEUS, in particular those never presented before. In Sect. A.3 we present the package we have implemented to model mobile entities, such as cars, and the integration of DEUS core engine with Google Maps API.

A.1 DEUS in a Nutshell

DEUS is multi-platform, being developed in Java. Its API (illustrated in Sect. A.2) allows developers to implement (by sub-classing) (i) *nodes*, i.e., the entities which interact in a complex system, leading to emergent behaviors such as humans, pets, cells, robots or intelligent agents; (ii) *events*, e.g., node births and deaths, interactions among nodes, interactions with the environment, logs and so on; and (iii) *processes*, either stochastic or deterministic ones, constraining the timeliness of events.

Once specific Java classes have been implemented, it is possible to configure a simulation with the DEUS graphical user interface, which includes:

- the Visual Editor, for the generation of XML documents describing the simulations;
- the Automator, for the execution of parametric simulations and the automatic generation of statistics (in a Gnuplot-compliant format).

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7

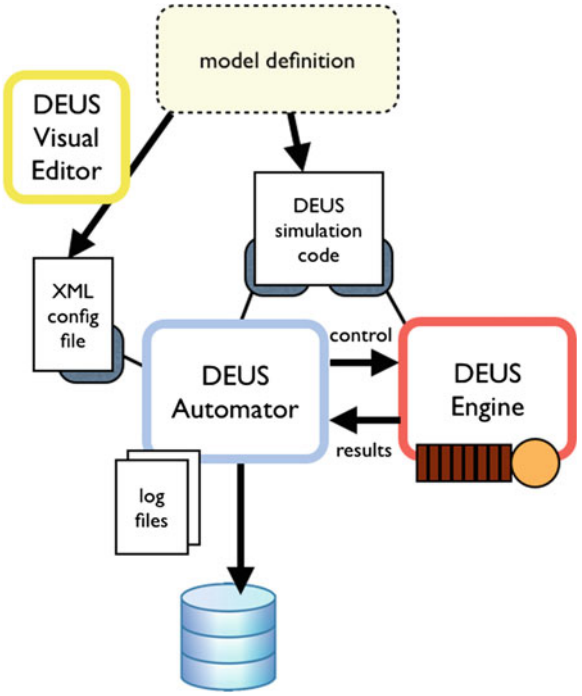


Fig. A.1 Discrete event simulation with DEUS

Figure A.1 illustrates how DEUS simulation models, in terms of XML configuration files and Java code, are created (using also a Visual Editor), and then executed by means of the Automator and the Engine. The former allows to perform sensitivity analysis, by setting ranges for node and process parameters. The Engine is the core of DEUS, managing the event queue and the simulation loop.

A node may represent a dynamic system characterized by a set of possible states, whose transition functions may be implemented either in the source code of the events associated to the node, or in the source code of the node itself. Multi-scale modeling of complex system can be achieved by defining nodes of different complexity and connecting them. DEUS comes with a library of predefined, common processes, and many others can be implemented by the user.

Sample code is available on the web site of project DEUS.[1] Recently we have published an "USN resilience example", which simulates an Unstructured Supernode Network (USN) [2], i.e., a peer-to-peer overlay network characterized by a group of peers, denoted as "supernodes", which have the responsibility of routing messages. Conversely, other peers ("leaf" nodes) are only resource providers and consumers, and need to connect to the supernode layer in order to publish and discover resources. In the considered scenario, if the lifetime L of a leaf node is longer than d, then at

[1] http://code.google.com/p/deus/.

time d the leaf node becomes a supernode and connects to m other randomly selected supernodes. The simulation allows to evaluate the node degree distribution and the probability of isolations of the supernodes, as functions of m. On a MacBook Pro with 2.4 GHz Intel Core 2 Duo and 4 GB 1067 MHz DDR3, simulating a network with 1000 nodes takes a couple of minutes. With 10000 nodes it takes, on average, half an hour. Such an appreciable performance is due to the fact that only the application-level logic is being simulated, considering stochastic variables such as the inter-arrival time and the lifetime of the nodes, and neglecting communication delays. To simulate data exchange among peers, it would be necessary to define message delivery events with realistic timing processes. Of course it would be possible to implement, using the API of DEUS, the detailed simulation of the networking layers. In the following section, we describe a NoC example in which the simulation of communication delays plays a major role.

A.2 DEUS API Structure and Features

Being DEUS a general purpose simulator, we kept basic interfaces and classes separated from more specific ones. By means of subclassing, it is possible to create specific modules for the simulation of any kind of complex system. Moreover, we developed a first extension package related to peer-to-peer resource sharing networks.

The experience we acquired during the development of other simulation code (mainly using ns-2 [3] and PeerSim [4]) showed us how difficult is to manage memory when it comes to the simulation of systems with a large number of interacting parts (*nodes*, if systems are described as a graph). Java is an extremely powerful language and the flexibility of its object orientation plus the reflection mechanism make it a prolific field upon which build this kind of project; however the difficulties in managing the garbage collection mechanism requires a good design in the memory management. For these reasons, as we describe in more details later on, DEUS relies on an efficient cloning mechanism: the initial process load configuration objects into memories and new instances of those objects are obtained through deep cloning. This features allows the deletion of objects by invoking their class destructor.

A.2.1 Simulation Objects and Behavior

The development of DEUS started from the definition of the basic simulation objects and the design of the configuration procedure, having in mind all the dynamics of complex systems that one may need to simulate. The goal was to achieve high flexibility and usability, allowing developers to specify a section with simulation objects and another one with simulation behavior, maximizing the possibility to reuse components and providing self-validation constraints so that the engine could process the configuration file through reflection and without any further validation. Simulation

objects are *events*, *nodes*, *resources*, while simulation behavior is managed through *processes* and *engine* objects.

An *event* represents the base simulation unit, i.e., the piece of code that is going to be scheduled by the system. Moreover, as complex systems are made by interacting components, we introduced the concept of *node*, which also corresponds to a data structure collector the event could relies on. Each node can have a set of *resources*, a structured way to represent objects the node can share or use through the event code. The association between events and nodes is given by *process* objects that are responsible for event schedule timing calculation. The *engine* object puts everything together by linking events that are scheduled at the beginning of the simulation.

The simulation behavior follows the standard model of discrete event simulations: initialization of system state variables and clock, scheduling of initial events and, until the ending condition is true, calculation of next clock time and processing of the next event in the scheduling queue. However, few additions have been made to make the model more flexible. For each event is possible to specify whether its execution is *one-shot*, so that the event will be removed from the schedule after its completion, or not, so that the event will be rescheduled according to the timing given by its associated process. Moreover, each event is provided with a listening mechanism over the scheduling process so that the latter will be able to schedule other events, namely *referenced events*, right after the event's execution. The ending condition of the simulator happens once the maximum simulation time has been reached or the scheduling queue is empty.

Unlike other simulation frameworks that allow the time of an event to be specified as an interval, giving the start time and the end time of each event, DEUS allows only the specification of the start time. The engine is currently single-threaded, so it has only one current event, but the parallelism in simulation is given by the maintenance of system state according to the virtual time. In the near future we will provide support for multi-threaded and network-distributed simulation engine.

A.2.2 DEUS Core

DEUS has been divided into packages, each one addressing a specific aspect of the simulation. The root package is

```
it.unipr.ce.dsg.deus
```

and contains the following sub-packages:

- `core` base system components including simulation object interfaces, configuration parser and engine;
- `schema` object model representing the configuration file;
- `util` support classes for simulation engine;
- `impl.event` reference implementations of the event object;
- `impl.node` reference implementations of the node object;

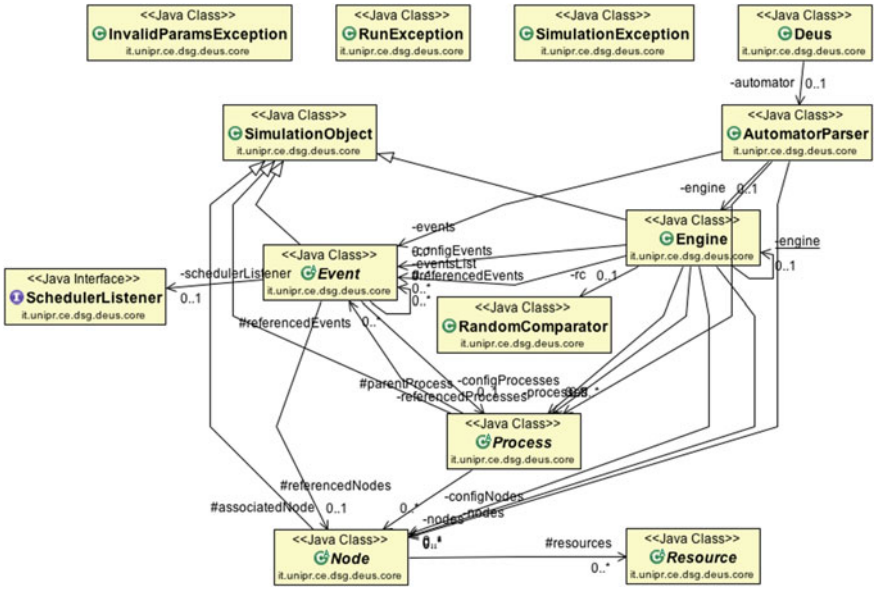


Fig. A.2 Class diagram of DEUS `core` package

- `impl.resource` reference implementations of the resource object;
- `impl.process` reference implementations of the process object.

In the next sections we will provide a detailed description of the main classes contained in each package, besides the `scheme` and `util` packages that are just used for supporting features. A class diagram including the `core` and `impl` packages is provided in Fig. A.2.

A.2.2.1 Core Package

The `Event` class represents the simulation object being scheduled by the Engine. Each event is identified by a configuration id, a set of properties, a flag indicating if the event should be executed only once, a set of referenced events, a parent process, the triggering time and a listener to handle the execution of referenced events. In order to keep the simulation memory area as small as possible, each event is created by cloning the original event obtained from the simulation configuration parser, therefore each implementing class should provide the code for cloning the event ensuring that its internal state is consistent, by re-initializing the event members that do not have to be cloned.

The `Node` class represents a generic data structure collector inside the simulation, so the main use is to store, read and delete information useful for simulation state. Each node is identified by a configuration id, a set of properties and a set of resources.

Similarly to the `Event` class there is the same cloning mechanism to keep the memory requirements small for the simulation execution.

The `Resource` class represents a generic resource associated to a node. The class itself does not provide any method, is just used to force implementors to use this node/resource model representation.

The `Process` class represents the simulation object responsible to determine the timing of events scheduling. Each process is identified by a configuration id, a set of properties, a set of referenced nodes and a set of referenced events.

The `Engine` class represents the simulation engine of DEUS. After the configuration file is parsed, the obtained configured simulation objects (nodes, events and processes) are passed to the Engine that will properly initialize the queue of events to be run. The simulation is a standard discrete event simulation where each event has an associated triggering time, used as a sorting criteria. The events inserted into the simulation queue are processed individually one after each other, each time updating the current simulation virtual time. The run method of the engine will process each event in the event queue until a maximum virtual time is reached or the queue is empty. In each cycle the first event of the queue is removed (the one with the lowest triggering time), the virtual time of the simulation is updated and the event is executed. If the event has some referenced events, those will be scheduled right after the event execution, in the same order used to define them in the configuration file. If the event is not one-shot and it has a parent process, then it will be scheduled for a next execution with a triggering time calculated according to the parent process' strategy.

The `AutomatorParser` class is responsible for handling the simulation configuration file according to the *DEUS XML schema*. The configuration can be seen as a set of nodes, resources, events, processes and engine parameters. This class handles the configuration of each simulation object and stores them in a set of array data structures. Each simulation object has a set of base features, plus references to other simulation objects: nodes can have a set of resources, events can have a set of referenced events, processes can have references to both nodes and events. At the end of the configuration file parsing process, this class initializes the Engine object enabling the simulation execution.

A.2.2.2 impl.event Package

The `BirthEvent` class represents the birth of a simulated node. During its execution an instance of the node associated to the event will be created.

The `DeathEvent` class represents the death of a simulated node. During the execution of the event the associated node will be killed or, in case nothing is specified, a random node will be chosen instead.

The `LogPopulationSizeEvent` class is used to simulate a logging event that stores the number of nodes in the simulation, each time it is scheduled. It demonstrates that an event can be really anything, in the context of the complex system to be simulated.

A.2.2.3 impl.node Package

The `BasicNode` class is the default implementation of the node abstract class, without any specific properties. A specific implementation is provided in the p2p package, which is described later in the Appendix.

A.2.2.4 impl.resource package

The `AllocableResource` class represents a generic allocable resource. This kind of resource has a type/amount pair parameter which must be specified through the configuration file.

The `ResourceAdv` class represents a resource advertisement, i.e., a document that describes a `ConsumableResource` (with a name and an amount), and the interested node. Once the resource described by a `ResourceAdv` has been discovered, the owner of the resource should be registered into the `ResourceAdv`, and the found flag set to true.

A.2.2.5 impl.process package

The `PeriodicProcess` represents a generic periodic process. It has a parameter called *period* that is used to generate the triggering time. Each time the process receives a request for generating a new triggering time, it computes it by adding the period value to the current simulation virtual time. An extension of this class is provided through the `TwoSpeedsPeriodicProcess` class that allows the specification of two different periods; the switch between first period and second period is made using a virtual time threshold.

The `PoissonProcess` represents a generic Poisson process. It has one parameter called *meanArrival*that is used to generate the triggering time. Each time the process receives a request for generating a new triggering time, it computes it by adding the current simulation virtual the value of an Homogeneous Poisson Process with the rate parameter calculated as 1/*meanArrival* time. Similarly to the `TwoSpeedPeriodicProcess`, there is the `TwoSpeedPoissonProcess` class to provide a Poisson Process that changes its speed after a virtual time threshold has been reached.

A.2.3 Extension Package for the Simulation of Peer-to-Peer Systems

To simulate a particular kind of complex system, namely peer-to-peer resource sharing networks, we implemented the

```
it.unipr.ce.dsg.deus.p2p
```

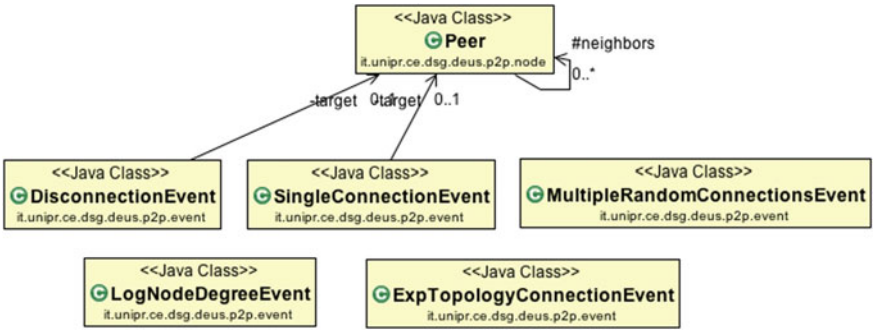


Fig. A.3 Class diagram of the `p2p` package

package, which contains the following sub-packages:

- `node` the model of peer.
- `event` the events characterizing a P2P network;

In the following we provide a detailed description of the main classes contained in each package. The related class diagram is illustrated in Fig. A.3.

A.2.3.1 Node Package

The `Peer` class is an extension of the `Node` class that represents the concept of peer in a network. Each peer is identified by a unique key generated by the engine (in the given key space) and is characterized by a list of neighbors, peers with whom it has an active link connection, and a status regarding peer connection to the network (whether is connected or not). Some methods have been implemented to manage neighborhood and notification messages.

A.2.3.2 Event Package

The `SingleConnectionEvent` class simulates the connection event of a peer in the network. The peer can choose a randomly node to which connect or starts from a well-known one. An extension of this class is provided through the class called `MultipleRandomConnectionsEvent`, which enables the connection to more than one node, randomly chosen in the network.

The `DisconnectionEvent` class is used to disconnect a specific node from the network. Alternatively it can be used to disconnect a random node from the network.

The `LogNodeDegreeEvent` class represents a logger that works out on Peer nodes. It calculates the node degree distribution for each peer of the network. The results is a list of degree starting from 1 up to the maximum degree inside the network; for each degree the number of nodes having it is computed.

A.3 Simulation of Mobile Nodes

Mobility models are massively used by mobile communication systems for predicting future user positions. They are crucial to test and evaluate for example vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I) or smart cities networking applications in real testbed environments. Those aspects are becoming highly relevant, considering the growing market of smartphones and mobile devices that are changing and reshaping the application design.

Starting from these motivations we decided to natively support mobility in our general purpose simulator extending the base node class and introducing the concept of Mobile Peer with a dedicated API that provides all necessary classes and methods to support mobility. This approach allows the developer to design and implement protocols and applications based on user mobility, starting from the classes offered by DEUS mobility package or extending them to be customized on specific requirements.

A.3.1 Mobility Model

Mobility Models (MM) describe the movement of mobile users, and how their location, velocity and acceleration change over time. Such Models are frequently used for several simulation purposes when new communication or navigation techniques are evaluated.

Designed mobility model follows the approach of Zhou et al. [5] where the key idea is to use switch stations (SSs) connected via virtual tracks to model the dynamics of vehicle and group mobility. Stations are connected to each other through virtual paths that have one lane for every direction, speed limitation associated with the street category and specific road density limit to model user/vehicle speed in jam conditions. When a new node joins the network, it first associates with a random SS, then it selects a new destination station and starts moving on the connection path between them. This procedure is repeated every time the user reaches a new SS and has to decide its next destination. Each switch station has an attraction/repulsion value that influences the user's choice for the next destination station. This value may be the same for each path in order to allow for random trip selection or could be configured by the developer according to the application field. Figure A.4 shows a schema of the SS Model and an example of simulative analysis that considers a square area around the city of Parma, with 20 switch stations inside and outside the city district.

When SSs and virtual paths between them have been configured for the simulation each node extending MobileNode class starts moving on its selected path. Each movement is associated to a specific DEUS event (MovePeerEvent) that simulates the movement of a peer to the next location.

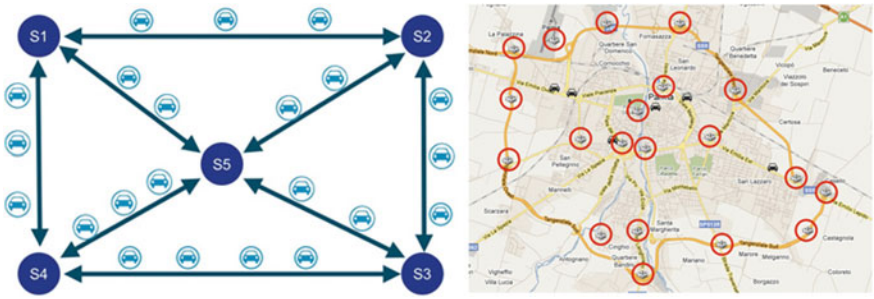


Fig. A.4 A schema of the SS Model (*on the left*), and an example of simulative analysis that considers a square area around the city of Parma, with 20 switch stations (*on the right*)

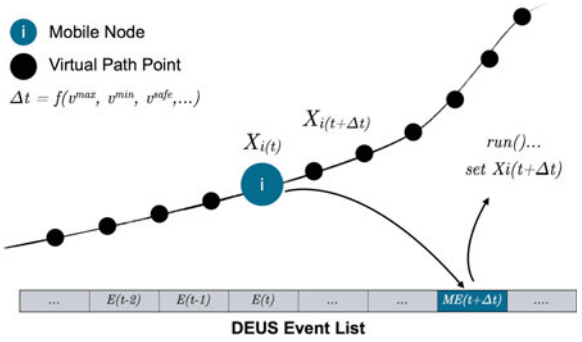


Fig. A.5 By knowing the node velocity at a certain instant, it is possible to insert an event in the simulator queue that describes the node position changes along the path

By defining $X_i(t)$ and $X_i(t + \Delta t)$ as node i's locations at time t and $t + \Delta t$, respectively, we can say that Δt is the time needed by i to move from the first to the second location and it is evaluated according to the distance between those points and to node speed. Given that, by knowing the node velocity at a certain instant, it is possible to put an event in the simulator queue that describes the node position changes along the path (Fig. A.5). When the event is picked up from the queue by the DEUS Engine, the location of the associated node is finally updated and a dedicated method is called to allow the developer to implement specific behaviors related to the position change of the node.

Within the presented approach, a general definition is needed for the speed of a Mobile Node, which may represent for example a pedestrian or a vehicle moving along a path. In order to do that, we rely on the concept of Speed Model, that computes the speed according to a set of parameters (the latter could be variable according to the model) describing the node characteristics and the state of the environment, such as the position of vehicles in the neighborhood, the actual speed, the maximum/minimum velocity, the maximum/minimum acceleration, the path length, and the node density on the track. DEUS provides an interface called ISpeedModel,

that allows to define the Speed Model that the user wants to use in its simulation. By default we provide the implementation of Fluid Traffic Model (FTM) [6]. It can be seen as an hybrid model, adopting a traffic system approach on a microscopic level. FTM describes the speed as a monotonically decreasing function of the vehicular density, forcing a lower bound on speed when the traffic congestion reaches a critical states.

A.3.2 MobDEUS Architecture

In Sect. A.3.1 we introduced the concept of mobility in DEUS. In order to offer a complete tool set, we have developed some external applications that are useful to configure and monitor the simulations.

Figure A.6 schematically illustrates the design of MobDEUS. Involved elements can be divided in two categories: the first one contains two external tools used to generate mobility scenario files needed by the DEUS Engine, and the second one is related to real-time monitoring of simulations.

The **Switch Station Web Editor** (**SSWE**) is a web-based tool written using Javascript and the Google Maps API, that allows to create and manage Switch Station configurations. The user can place on the map each SS according to its latitude and longitude, and generate, load and edit new or existing configuration files in order

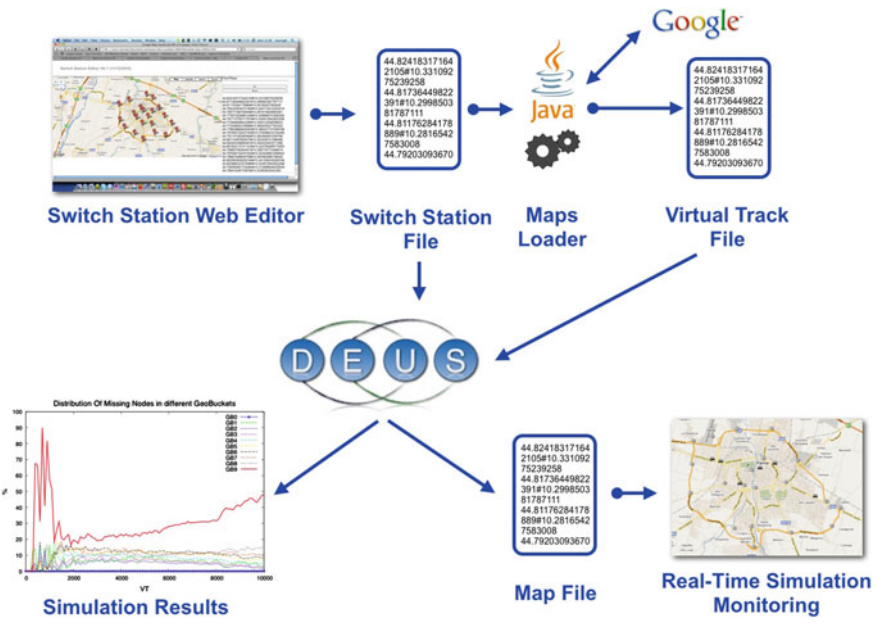


Fig. A.6 Simulation design of traffic information system for smart cities with MobDEUS

to be used with the simulator. DEUS provides a dedicated class called SwitchStationController to read, parse and load in a accessible list the configured SSs for the simulation.

The **Map Loader** (**ML**) is a Java application that communicates with Google Servers to retrieve paths between a list of Switch Stations. It takes as input file the SS list generated by the SSWE. Since the original received paths contain sparse geographic coordinates, each track is locally enriched by ML by computing coordinates between original points. The precision of this procedure can be defined by the user, by specifying the desired accuracy level (called *mobility precision* (*MP*), expressed in meters) that he/she wants between two available geographic locations in the resulting file. The generated output contains the list of all virtual tracks and their coordinates and can be automatically read by DEUS Engine using the SwitchStationController class. In such a way the user is able to properly configure the simulation scenario and when the SwitchStationController instance has been created and has imported the user files, each MobileNode in the simulation uses such information for its mobility.

The **Real-Time Simulation Monitoring** (**RTSM**) is a web-based tool written using Javascript and Google Maps API, that allows to monitor active mobile nodes during a simulation. The DEUS mobile module provides a dedicated log event called LogNodeMapEvent, that periodically generates an XML file containing node

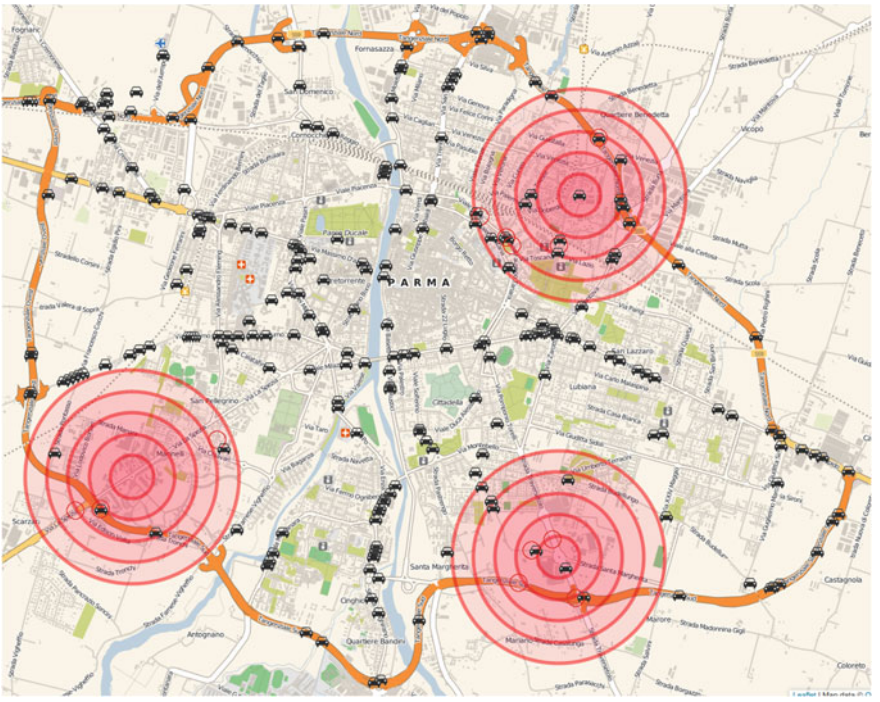


Fig. A.7 Real-time simulation monitoring

positions and additional details about each node, like start/end Switch Station, neighbors information, actual speed, etc. Those data are placed on a map and each marker can be associated to a different icon, according to node type or status. By clicking on each marker, the developer obtains additional information about the actual status of the node, by visualizing the node profile that is written in the map file. Map data are automatically updated, thus allowing to monitor node behaviors during the simulation. Since DEUS is a completely open project, the Javascript code of the RTSM may be extended according to user preferences, for example to enrich visualized information (as shown in Fig. A.7). The frequency of updates of our GMaps-based visualization tool may be lower than the frequency of simulated events (in this case we show only a subset of the events). In order to have full control over visualization, we plan to develop a standalone tool based on Open Street Map which will work offline using downloaded map portions.

A.4 Integration with ns-3

To simulate a distributed system with DEUS, it is necessary to write the classes which represent nodes, events and processes. Node may represent devices, servers, virtual machines, applications, etc. Events may be associated to specific nodes (e.g., start, connection, disconnection, internally/externally triggered state change, stop, etc.), or involving several nodes (it is the case of logging events). To simulate a message delivery from one node to another, it is necessary to define the sender, the destination and to schedule a "delivered message" event in the future (in terms of virtual time of the simulation). The scheduling time of such an event must be set using a suitable process, selected among those that are provided by the DEUS API, or defined by the user, possibly.

For example, if the purpose of the simulation is to measure the average delay of propagating multi-hop messages within a network of nodes (e.g., a peer-to-peer network), the value of each link's delay must be realistic, taking into account the underlying networking infrastructure. In particular, if the communication is wireless, estimating the delay of point-to-point communication is a challenging task.

The direct integration of DEUS with ns-3, with the former that "calls" the latter to compute a delay value every time a node must send a message to another node, taking into account current surrounding conditions, is unpractical and would highly increase the simulation time. Instead, a more effective and efficient solution (illustrated in Fig. A.8) includes the following steps:

1. given a complex system to be simulated, identify the main sub-system types, each one being characterized by specific networking parameters;
2. with ns-3: create detailed simulation models of the sub-systems (i.e., sub-models) and measure their characteristic transmission delays, taking into account both message payloads and proper headers;

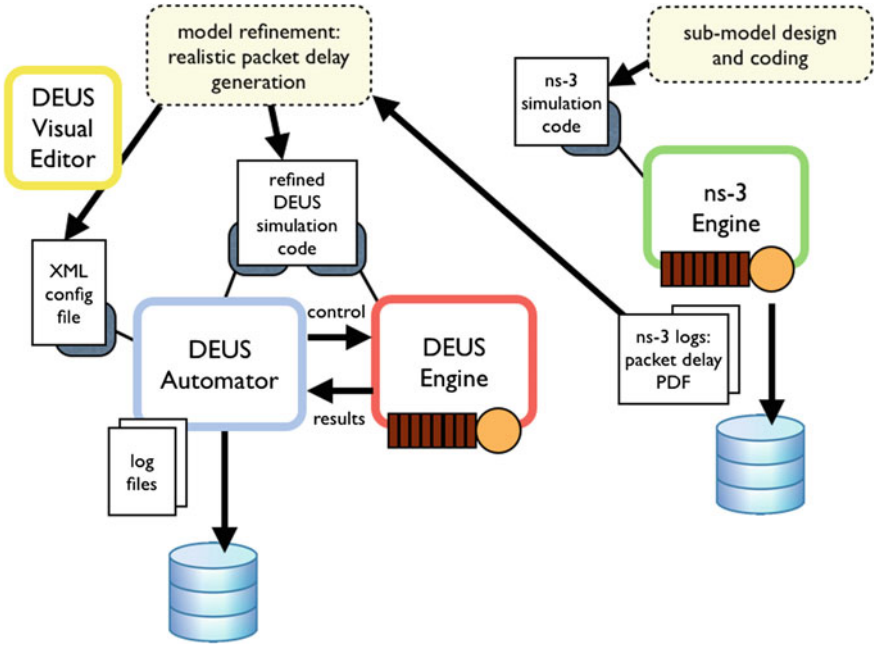


Fig. A.8 Discrete event simulation with DEUS and ns-3

3. with DEUS: simulate the whole distributed system, with refined scheduling of communication events, taking into account the transmission delays computed at step 2.

For example, if the overlay network relies on a cellular network, the sub-model to be characterized with ns-3 could be a set of cells. Its size should be significantly large, with respect to the system to be simulated with DEUS. If such a system is a peer-to-peer network, the end-to-end communication among couples of peers could span few or many cells, depending on the overlay scheme. Multi-cell communication may be very fast, in case base stations are connected by optical fibers [7]. However, inter-cell interference and horizontal handover could be taken into account, when simulating mobile nodes. Moreover, the simulation of each cell should take into account the presence of other mobile nodes, that are not directly involved in the distributed application of interest, but consume significant resources. Finally, the same sub-system could be simulated with different geographic conditions, e.g., in a city (with small cells, buildings, and noisy channel), or in a rural area (with larger cells and a less disturbed channel).

Appendix B
Mathematical Frameworks

B.1 Derivation of the Average Nodes Positions

In this appendix, we derive the average value of the positions vector $\mathbf{R}^{(n)}$ ($n \in \mathbb{N}$) of n Poisson points falling in the finite interval $\mathscr{I} = (0, z)$. The average values can be computed by first deriving the joint PDF of the vector $\mathbf{R}^{(n)}$, denoted as $f_{\mathbf{R}}^{(n)}(\mathbf{r})$, and defined over a proper n-dimensional domain $\mathscr{D}_n$. From $f_{\mathbf{R}}^{(n)}(\mathbf{r})$, it is then possible to derive the marginal PDF of R_j ($j = 1, 2, \ldots, n$), denoted as $f_{R_j}^{(n)}(r_j)$ and, from this, the average value $\overline{R}_j^{(n)}$.

B.1.1 A Single Point in $\mathscr{I}$

In this case, $n = 1$ and $\mathscr{D}_n = \mathscr{I}$. In this case, R_1 has a uniform distribution in $\mathscr{I}$ and its (marginal) PDF is given by:

$$f_{R_1}^{(1)}(r_1) = \begin{cases} \frac{1}{z} & r_1 \in \mathscr{D}_1 \\ 0 & \text{otherwise.} \end{cases}$$

The average value is:

$$\overline{R}_1^{(1)} = \frac{z}{2}.$$

B.1.2 Two Points in $\mathscr{I}$

Without loss of generality, it is possible to order the points by imposing that $r_2 > r_1$. Thanks to this assumption, $\mathscr{D}_2$ can be expressed as follows:

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7

$$\mathscr{D}_2 = \left\{(r_1, r_2) \in \mathbb{R}^2 : r_1 \in (0, z), r_2 \in (0, z), r_1 < r_2\right\}.$$

The joint PDF is uniform over $\mathscr{D}_2$ and can be expressed as follows:

$$f_{R_1R_2}(r_1, r_2) = \begin{cases} \frac{1}{\text{Area}(\mathscr{D}_2)} & (r_1, r_2) \in \mathscr{D}_2 \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \frac{2}{z^2} & (r_1, r_2) \in \mathscr{D}_2 \\ 0 & \text{otherwise.} \end{cases}$$

From the joint PDF, the marginal PDFs of R_1 and R_2 can be obtained:

$$f_{R_1}^{(2)}(r_1) = \int_0^\infty f_{R_1R_2}(r_1, r_2)\,\mathrm{d}r_2 = \begin{cases} \int_{r_1}^{z} \frac{2}{z^2}\,\mathrm{d}r_2 & r_1 \in (0, z) \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \frac{2(z-r_1)}{z^2} & r_1 \in (0, z) \\ 0 & \text{otherwise} \end{cases} \tag{B.1}$$

$$f_{R_2}^{(2)}(r_2) = \int_0^\infty f_{R_1R_2}(r_1, r_2)\,\mathrm{d}r_1 = \begin{cases} \int_{0}^{r_2} \frac{2}{z^2}\,\mathrm{d}r_1 & r_2 \in (0, z) \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \frac{2(z-r_2)}{z^2} & r_2 \in (0, z) \\ 0 & \text{otherwise.} \end{cases} \tag{B.2}$$

Using Eqs. (B.1) and (B.2), the average values of R_1 and R_2 can be expressed:

$$\overline{R}_1^{(2)} = \int_0^z r_1 \frac{2(z - r_1)}{z^2}\,\mathrm{d}r_1 = \frac{z}{3}$$
$$\overline{R}_2^{(2)} = \int_0^z r_2 \frac{2(z - r_2)}{z^2}\,\mathrm{d}r_2 = \frac{2}{3}z.$$

B.1.3 A Generic Number of n Points in $\mathscr{I}$

As in the case with $n = 2$, it is possible to order the points as that $r_1 < r_2 < \cdots < r_n$, without losing any generality. Hence, the n-dimensional domain can be expressed as follows:

$$\mathscr{D}_n = \left\{(r_1, \ldots, r_n) \in \mathbb{R}^n : r_i \in (0, z) \forall i \in \{1, \ldots, n\}, r_1 < r_2 < \cdots < r_n\right\}.$$

The joint PDF of the n Poisson points has the following expression:

$$f_{R_1 \ldots R_n}(r_1, \ldots, r_n) = \begin{cases} \frac{1}{\text{Volume}(\mathscr{D}_n)} & (r_1, \ldots, r_n) \in \mathscr{D}_n \\ 0 & \text{otherwise} \end{cases} = \begin{cases} \frac{n!}{z^n} & (r_1, \ldots, r_n) \in \mathscr{D}_n \\ 0 & \text{otherwise.} \end{cases}$$

where

$$
\begin{aligned}
\text{Volume}(\mathscr{D}_n) &= \int_0^z \int_{r_1}^z \cdots \int_{r_{n-1}}^z 1 \, \mathrm{d}r_n \cdots \mathrm{d}r_2 \, \mathrm{d}r_1 \\
&= \int_0^z \int_{r_1}^z \cdots \underbrace{\int_{r_{n-2}}^z (z - r_{n-1}) \, \mathrm{d}r_{n-1}}_{\frac{(z-r_{n-2})^2}{2}} \cdots \mathrm{d}r_2 \, \mathrm{d}r_1 \\
&\vdots \\
&= \tfrac{z^n}{n \cdot (n-1) \cdot (n-2) \cdots 3 \times 2} \\
&= \tfrac{z^n}{n!}.
\end{aligned}
$$

The marginal PDF of the position of the ith point is given by:

$$
\begin{aligned}
f_{R_i}^{(n)}(r_i) &= \underbrace{\int_0^\infty \cdots \int_0^\infty}_{n-1 \text{ times}} f_{R_1 \ldots R_n}(r_1, \ldots, r_n) \, \mathrm{d}r_n \cdots \mathrm{d}r_{i+1} \, \mathrm{d}r_{i-1} \cdots \mathrm{d}r_1 \\
&= \int_0^{r_i} \int_{r_1}^{r_i} \cdots \int_{r_{i-2}}^{r_i} \int_{r_i}^{z} \cdots \int_{r_{n-1}}^{z} \frac{n!}{z^n} \, \mathrm{d}r_n \cdots \mathrm{d}r_{i+1} \, \mathrm{d}r_{i-1} \cdots \mathrm{d}r_1 \\
&= \frac{n!}{z^n} \int_0^{r_i} \int_{r_1}^{r_i} \cdots \int_{r_{i-2}}^{r_i} \int_{r_i}^{z} \cdots \underbrace{\int_{r_{n-2}}^{z} (z - r_{n-1}) \, \mathrm{d}r_{n-1}}_{\frac{(z-r_{n-2})^2}{2}} \cdots \mathrm{d}r_{i+1} \, \mathrm{d}r_{i-1} \cdots \mathrm{d}r_1 \\
&\vdots \\
&= \frac{n!}{z^n} \int_0^{r_i} \int_{r_1}^{r_i} \cdots \int_{r_{i-2}}^{r_i} \frac{(z - r_i)^{n-i}}{(n-i) \cdot (n-i-1) \cdots 3 \times 2} \, \mathrm{d}r_{i-1} \cdots \mathrm{d}r_1 \\
&= \frac{n!}{z^n} \frac{(z - r_i)^{n-i}}{(n-i)!} \int_0^{r_i} \int_{r_1}^{r_i} \cdots \int_{r_{i-2}}^{r_i} \mathrm{d}r_{i-1} \cdots \mathrm{d}r_1 \\
&= \frac{n!}{z^n} \frac{(z - r_i)^{n-i}}{(n-i)!} \int_0^{r_i} \int_{r_1}^{r_i} \cdots \underbrace{\int_{r_{i-3}}^{r_i} (r_i - r_{i-2}) \, \mathrm{d}r_{i-2}}_{\frac{(r_i - r_{i-3})^2}{2}} \cdots \mathrm{d}r_1 \\
&\vdots
\end{aligned}
$$

$$= \frac{n!}{z^n} \frac{(z-r_i)^{n-i}}{(n-i)!} \underbrace{\int_0^{r_i} \frac{(r_i - r_1)^{i-2}}{(i-2)\cdots 3\times 2} \mathrm{d}r_1}_{\frac{r_i^{i-1}}{(i-1)\cdots 3\times 2}}$$

$$= \frac{n!}{z^n} \frac{(z-r_i)^{n-i}}{(n-i)!} \frac{r_i^{i-1}}{(i-1)!} \quad i = 1, \ldots, n. \tag{B.3}$$

On the basis of Eq. (B.3), it is straightforward to derive the marginal PDF of R_i ($i = 1, 2, \ldots, n$), given the presence of n points in the interval $\mathscr{I}$:

$$f_{R_i}^{(n)}(r_i) = \begin{cases} \frac{n!}{z^n} \frac{(z-r_i)^{n-i}}{(n-i)!} \frac{r_i^{i-1}}{(i-1)!} & r_i \in (0, z) \quad i = 1, \ldots, n \\ 0 & \text{otherwise.} \end{cases} \tag{B.4}$$

Finally, from Eq. (B.4) the average value of R_i can be expressed as follows:

$$\overline{R}_i^{(n)} = \int_0^z r_i \frac{n!}{z^n} \frac{(z-r_i)^{n-i}}{(n-i)!} \frac{r_i^{i-1}}{(i-1)!} \mathrm{d}r_i = i \frac{z}{n+1} \quad i = 1, \ldots, n.$$

B.2 Per-node Retransmission Probability in a Network with Equally Spaced Nodes

We consider the deterministic scenario introduced in Sect. 3.6.1, composed by a fixed number n of nodes equally spaced in the interval $\mathscr{I} = (0, z) \subset \mathbb{R}$, with the positions vector $\mathbf{R}^{(n)}$ defined in Eq. (3.1). In this appendix, we derive the following probabilities:

$$p_{\text{rtx}}^{(n)}(i) = \mathrm{P}\{S_i\} \quad i = 1, 2, \ldots, n$$

where the event $\mathscr{S}_i$ was defined in Sect. 3.6.1. In order to derive $p_{\text{rtx}}^{(n)}(i)$, it is helpful to introduce the following auxiliary events:

- $\mathscr{B}_i \triangleq$ {the node i is designated as a relay};
- $\mathscr{C}_i \triangleq$ {the node i wins the contention among a set of n nodes};
- $\mathscr{D}_i^{(m)} \triangleq$ {the node i wins the contention among a set of m contending nodes};
- $\mathscr{W}_k \triangleq$ {the value BC=k is chosen by a single node} $\quad k = 0, \ldots, cw - 1$;
- $\mathscr{W} \triangleq$ {at least a value of BC $\in$ [0, cw−1] is chosen by a single node}.

The event $\mathscr{S}_i$, defined in Sect. 3.6.1, is verified if both the events $\mathscr{B}_i$ and $\mathscr{C}_i$ happen. Therefore, $p_{\text{rtx}}^{(n)}(i)$ can be expressed as:

$$p_{\text{rtx}}^{(n)}(i) = \mathrm{P}\{\mathscr{S}_i\} = \mathrm{P}\{\mathscr{B}_i \cap \mathscr{C}_i\} = \mathrm{P}\{\mathscr{B}_i\}\mathrm{P}\{\mathscr{C}_i\}$$

where the last equality is motivated by the independence of the events $\mathscr{B}_i$ and $\mathscr{C}_i$. The probability $\mathrm{P}\{\mathscr{B}_i\}$ is known, since it should be replaced with one of the PAF used by the protocols considered in this Appendix [defined in Eqs. (3.4) and (3.5)]. On the opposite, the unknown probability $\mathrm{P}\{\mathscr{C}_i\}$ can be derived by applying the total probability theorem, thus obtaining:

$$\begin{aligned} p_{\text{rtx}}^{(n)}(i) &= \mathrm{P}\{\mathscr{B}_i\}\mathrm{P}\{\mathscr{C}_i\} \\ &= p_i \sum_{m=1}^{n} \mathrm{P}\{\mathscr{D}_i^{(m)}\} p_{V_i^{(n)}}(m-1) \\ &= p_i \sum_{m=1}^{n} q^{(m)}(i)\, p_{V_i^{(n)}}(m-1) \end{aligned}$$

where $q^{(m)}(i) \triangleq \mathrm{P}\{\mathscr{D}_i^{(m)}\}$ and $V_i^{(n)} \in \{0, \ldots, n-1\}$ is a discrete random variable defined as:

$$V_i^{(n)} \triangleq \{\text{the number of nodes competing with the node } i \text{ given } n \text{ nodes}\}.$$

It can be shown that the PMF of $V_i^{(n)}$ can be expressed as follows:

$$p_{V_i^{(n)}}(v) = \begin{cases} \prod_{s\in\mathbb{L}}(1-p_s) & v = 0 \\ \sum_{j_1\in\mathbb{J}_{1,i,v}} \sum_{j_2\in\mathbb{J}_{2,i,v}} \cdots \sum_{j_v\in\mathbb{J}_{v,i,v}} & \\ \left(\prod_{s\in\{j_1,j_2,\ldots,j_v\}} p_s \prod_{t\in\mathbb{L}\setminus\{j_1,j_2,\ldots,j_v\}} \bar{p}_t\right) & 0 < v < N-1 \end{cases}$$

where $\mathbb{L} \triangleq \{1, 2, \ldots, N\}\backslash\{i\}$ and the sets $\{\mathbb{J}_{k,i,v}\}$ are defined as follows:

$$\mathbb{J}_{k,i,v} = \begin{cases} \{k+1, k+2, \ldots, n-v+k\} & i \le k \\ \{k, k+1, \ldots, i-1, i+1, \ldots, n-v+k\} & k < i \le n-v+k-1 \\ \{k, k+1, \ldots, n-v+k-1\} & i > n-v+k-1. \end{cases}$$

The probability $q^{(m)}(i)$ can be computed by analyzing the BA mechanism of the IEEE 802.11b standard. In particular, it emerges that $q^{(m)}(i)$ is independent of i and can be expressed as follows:

$$q^{(m)}(i) = q^{(m)} = \frac{\mathrm{P}\{\mathscr{W}\}}{n} = \frac{\mathrm{P}\left\{\bigcup_{k=0}^{cw-1} \mathscr{W}_k\right\}}{n}. \tag{B.5}$$

Since the events $\{\mathscr{W}_k\}$ are not disjoint, it is necessary to use the generalized union probability formula [8, Chap. 4] to compute $\mathrm{P}\{\mathscr{W}\}$:

$$\begin{aligned}\mathrm{P}\{\mathscr{W}\} = \mathrm{P}\left\{\bigcup_{k=0}^{cw-1} \mathscr{W}_k\right\} = &\sum_{k_1=0}^{cw-1} \mathrm{P}\left\{\mathscr{W}_{k_1}\right\} \\ &- \sum_{k_1<k_2} \mathrm{P}\left\{\mathscr{W}_{k_1} \cap \mathscr{W}_{k_2}\right\} \\ &+ \sum_{k_1<k_2<k_3} \mathrm{P}\left\{\mathscr{W}_{k_1} \cap \mathscr{W}_{k_2} \cap \mathscr{W}_{k_3}\right\} \\ &+ \cdots + \\ &(-1)^{cw+1} \mathrm{P}\left\{\mathscr{W}_0 \cap \mathscr{W}_1 \cap \cdots \cap \mathscr{W}_{cw-1}\right\}.\end{aligned} \tag{B.6}$$

Since the addenda of each single sum of the right-hand side of (B.6) are the same, taking into account the number of possible combinations, the generic right-hand side of (B.6) can be then expressed as follows:

$$(-1)^{r+1} \sum_{k_1<k_2<\cdots<k_r} \mathrm{P}\left\{\mathscr{W}_{k_1} \cap \mathscr{W}_{k_2} \cap \cdots \cap \mathscr{W}_{k_r}\right\} = (-1)^{r+1} \binom{cw}{r} \frac{(cw-r)^{n-r} \prod_{j=0}^{r-1}(n-j)}{(cw)^n}.$$

Thanks to Eq. (B.6), $q^{(n)}$ can be finally given the following expression:

$$q^{(n)} = \frac{\sum_{r=1}^{\min(n,cw)} (-1)^{r+1} \binom{cw}{r} (cw-r)^{n-k} \prod_{j=0}^{r-1}(n-j)}{n(cw)^n}$$

where the term $\min(n, cw)$ is introduced to deal with the case $n < cw$.

B.3 Per-node Delay in a Network with Equally Spaced Nodes

In this appendix, we derive the number of slots spent by the ith node during the *backoff* conditioned to the event $\mathscr{S}_i$, denoted as $N_{i|i}^{\mathrm{bo}}$. By analyzing the BA mechanism of the IEEE 802.11b standard, one obtains:

$$\overline{N}_{i|i}^{\mathrm{bo}} = \mathbb{E}\left[N_{i|i}^{\mathrm{bo}}\right] = \sum_{v=0}^{N-1} \mathbb{E}\left[N_i^{\mathrm{bo}} | V_i^{(N)} = v, \mathscr{S}_i\right] \mathrm{P}\left\{V_i^{(N)} = v | \mathscr{S}_i\right\} \tag{B.7}$$

where $P\{V_i^{(N)} = v|\mathscr{S}_i\}$ can be derived by means of the Bayes theorem as follows:

$$\mathrm{P}\left\{V_i^{(N)} = v|\mathscr{S}_i\right\} = \frac{\overbrace{p_i\, q^{(v+1)}}^{\mathrm{P}\left\{\mathscr{S}_i | V_i^{(N)} = v\right\}} \overbrace{p_{V_i^{(N)}}(v)}^{\mathrm{P}\left\{V_i^{(N)} = v\right\}}}{\underbrace{\mathrm{P}\{\mathscr{S}_i\}}_{p_{\mathrm{rtx}}^{(N)}(i)}} = \frac{p_i\, q^{(v+1)}\, p_{V_i^{(N)}}(v)}{p_{\mathrm{rtx}}^{(N)}(i)}. \tag{B.8}$$

Instead, $\mathbb{E}[N_i^{\text{bo}}|V_i^{(N)} = v, \mathscr{S}_i]$ can be derived by observing that the delay associated with the event {the node i transmits with success given v contending nodes} depends on two factors: (i) the slot $BC_i \in \{0, \ldots, cw-1\}$ selected by the node i for transmitting; (ii) the number of collisions occurred in the slots $0, \ldots, k-1$, which, given that $BC_i = k$, corresponds to the following random variable:

$$N_{k,v}^{\text{col}} = \{\text{number of collisions in slots } 0, \ldots, k-1\} \qquad N_{k,v}^{\text{col}} \in \{0, J_{k,v}\}$$

where $J_{k,v} \triangleq \min(k, \lfloor (v/2) \rfloor)$ denotes the maximum number of collisions that can happen in slots $0, \ldots, k-1$. On the basis of these considerations it can be shown that:

$$\begin{aligned}\mathbb{E}[N_i^{\text{bo}}|V_i^{(N)} = v, \mathscr{S}_i] &= \sum_{k=0}^{cw-1} \sum_{j=0}^{J_{k,v}} \mathbb{E}[N_i^{\text{bo}}|\{V_i^{(N)} = v, \mathscr{S}_i, BC_i = k, N_{k,v}^{\text{col}} = j\}] \cdot \\ &\quad \mathrm{P}\left\{BC_i = k \wedge N_{k,v}^{\text{col}} = j|V_i^{(N)} = v, \mathscr{S}_i\right\} \\ &= \sum_{k=0}^{cw-1} \sum_{j=0}^{J_{k,v}} (k + jT^{\text{tx}}) P_v(k, j) \qquad v = 0, \ldots, N-1\end{aligned}$$

where $P_v(k, j) \triangleq \mathrm{P}\left\{BC_i = k \wedge N_{k,v}^{\text{col}} = j|V_i^{(N)} = v, \mathscr{S}_i\right\}$ is the (k, j)th element of the matrix $\mathbf{P}_v$, of dimension $cw \times (J_{cw-1,v} + 1)$. There exist N matrices $\mathbf{P}_v$, one for each value of $\{V_i^{(N)} = v\},\ v \in \{0, N-1\}$.

In order to derive $\mathbf{P}_v$ it is necessary to define thefollowing random variables:

$$\begin{aligned} H_{k,v} &= \{\text{number of nodes with } BC < k\} & H_{k,v} &\in \{0, \ldots, v\} \\ &= \begin{cases} \sum_{m=1}^{v} I_{m,k} & k > 0 \wedge v > 0 \\ 0 & \text{otherwise} \end{cases} & I_{m,k} &= \begin{cases} 1 & BC_m < k \\ 0 & BC_m \end{cases} \\ N_{k,v|h}^{\text{opp}} &= \{\text{number of nodes with } BC = k|H_{k,v} = h\} & N_{k,v|h}^{\text{opp}} &\in \{0, \ldots, v-h\} \\ &= \begin{cases} \sum_{m=1}^{v-h} L_{m,k} & k \geq 0 \wedge v > 0 \\ 0 & \text{otherwise} \end{cases} & L_{m,k} &= \begin{cases} 1 & BC_m = k \\ 0 & BC_m > k. \end{cases} \end{aligned}$$

$$\begin{aligned} N_{k,v|h}^{\text{col}} &= \{\text{number of collisions in the } 0, \ldots, k-1|H_{k,v} = h\} \\ &\qquad N_{k,v|h}^{\text{col}} \in \{0, \ldots, J_{k,h}\} \\ N_{k,v|h}^{\text{win}} &= \{\text{number of slots } 0, \ldots, k-1 \text{ chosen by a single node } |H_{k,v} = h\} \\ &\qquad N_{k,v|h}^{\text{win}} \in \{0, \ldots, k\} \end{aligned}$$

It is then possible to compute $P_v(k, j)$ using Bayes theorem and the total probability theorem:

$$
\begin{aligned}
P_v(k,j) &= \mathrm{P}\left\{BC_i = k \wedge N^{\mathrm{col}}_{k,v} = j | V_i^{(N)} = v, \mathscr{S}_i\right\} \\
&= \frac{\mathrm{P}\left\{\mathscr{S}_i \wedge BC_i = k \wedge N^{\mathrm{col}}_{k,v} = j | V_i^{(N)} = v\right\}}{\mathrm{P}\left\{\mathscr{S}_i | V_i^{(N)} = v\right\}} \\
&= \frac{\mathrm{P}\left\{\mathscr{S}_i \wedge N^{\mathrm{col}}_{k,v} = j | V_i^{(N)} = v, BC_i = k, \right\} \mathrm{P}\left\{BC_i = k | V_i^{(N)} = v\right\}}{p_i\, q^{(v+1)}} \\
&= \frac{1}{cwp_i q^{(v+1)}} \sum_{h=0}^{v} \mathrm{P}\left\{\mathscr{S}_i \wedge N^{\mathrm{col}}_{k,v|h} = j | \{V_i^{(N)} = v, BC_i = k, H_{k,v} = h\}\right\} \cdot \\
&\quad \mathrm{P}\left\{H_{k,v} = h | \{V_i^{(N)} = v, BC_i = k\}\right\} \\
&= \cdots \\
&= \frac{1}{cwq^{(v+1)}} P'_v(k,j) \\
&= \frac{1}{cwq^{(v+1)}} \begin{cases} \left(\frac{cw-1}{cw}\right)^v & k = 0 \\ \sum_{h=0}^{v} M_{k,v}(j,h) N_{k,v}(0,h) & k = 1, \ldots, cw-1 \end{cases} \\
&= \frac{1}{cwq^{(v+1)}} \begin{array}{cc} \left(\frac{cw-1}{cw}\right)^v & k = 0 \\ \sum_{h=0}^{v} M_{k,v}(j,h) N_{k,v}(0,h) & k = 1, \ldots, cw-1 \end{array}
\end{aligned}
$$

where the (j,h)th elements of matrix $\mathbf{N}_{k,v}$ (with dimension $(v+1) \times (v+1)$) are defined as:

$$
\begin{aligned}
N_{k,v}(n,h) &= \mathrm{P}\left\{N^{\mathrm{opp}}_{k,v|h} = n | \{V_i^{(N)} = v, BC_i = k, H_{k,v} = h\}\right\} \\
&= \binom{v-h}{n} \frac{(cw-k-1)^{v-h-n}}{(cw-k)^{v-h}} \\
&\quad v = 0, \ldots, N-1 \qquad k = 1, \ldots, cw-1 \qquad h, n = 0, \ldots, v
\end{aligned}
$$

while the (j,h)th elements of matrix $\mathbf{M}_{k,v}$ (with dimension $(J_{k,v}+1) \times (v+1)$) are defined as:

$$
\begin{aligned}
M_{k,v}(j,h) &= \mathrm{P}\left\{N^{\mathrm{win}}_{k,v|h} = 0 \wedge N^{\mathrm{col}}_{k,v|h} = j | \{V_i^{(N)} = v, BC_i = k, H_{k,v} = h\}\right\} \cdot \\
&\quad \mathrm{P}\left\{H_{k,v} = h | \{V_i^{(N)} = v, BC_i = k\}\right\} \\
&\quad v = 0, \ldots, N-1 \qquad k = 1, \ldots, cw-1 \\
&\quad j = 0, \ldots, J_{k,v} \qquad h = 0, \ldots, v
\end{aligned}
$$

In order to reduce the computational burden, the matrix $\mathbf{M}_{k,v}$ can be derived by means of a recursive strategy. In particular, it can be observed that the number of collisions at the kth hop is identical to (if nobody selects the value $\mathrm{BC} = k-1$) or greater than 1 (if at least two nodes select that value). Hence, once derived $\mathbf{M}_{1,v}$, it is

possible to determine $\mathbf{M}_{k,v}$ for all the remaining values of k. In particular, the direct formulation for $k = 1$ is the following:

$$M_{1,v}(j,h) = \begin{cases} \left(\frac{cw-k}{cw}\right)^v & j = 0,\ h = 0 \\ 0 & j = 0,\ h > 0 \\ 0 & j = 1,\ h \leq 1 \\ \binom{v}{h} \frac{(cw-k)^{v-h}}{(cw)^v} & j = 1,\ 1 < h < v \end{cases}$$

from which it is possible to derive $\mathbf{M}_{k+1,v}$ for any values of k:

$$M_{k+1,v}(j,h) = \begin{cases} 0 & h = 1 \\ \left(\frac{cw-k+1}{cw}\right)^v & j = 0, h = 0 \\ I_{j,k} M_{k,v}(j,h) N_{k,v}(0,h) & \\ + \sum_{t=2j-2}^{h-2} M_{k,v}(j-1,t) N_{k,v}(h-t,t) & j \in \mathscr{J}_{k,v}, h \in \{2j, v\} \\ 0 & \text{otherwise} \end{cases}$$

where the indicator function $I_{j,k}$ is defined as

$$I_{j,k} \triangleq \begin{cases} 1 & j \neq k \\ 0 & j = k. \end{cases}$$

Finally, using Eqs. (B.8) and (B.9) in (B.7), one obtains the final expression:

$$\begin{aligned} \overline{N}_{i|i}^{\text{bo}} &= \frac{p_i}{p_{\text{rtx}}^{(N)}(i)} \sum_{v=0}^{N-1} q^{(v+1)}\, p_{V_i^{(N)}}(v) \sum_{k=0}^{cw-1} \sum_{j=0}^{J_{k,v}} \left(k + jT^{\text{tx}}\right) P_v(k,j) \\ &= \frac{p_i}{cw\, p_{\text{rtx}}^{(N)}(i)} \sum_{v=0}^{N-1} p_{V_i^{(N)}}(v) \sum_{k=0}^{cw-1} \sum_{j=0}^{J_{k,v}} \left(k + jT^{\text{tx}}\right) P'_v(k,j) \\ &= \frac{p_i}{cw\, p_{\text{rtx}}^{(N)}(i)} \sum_{v=0}^{N-1} p_{V_i^{(N)}}(v) \sum_{k=1}^{cw-1} \left[k \sum_{j=0}^{J_{k,v}} P'_v(k,j) + T^{\text{tx}} \sum_{j=1}^{J_{k,v}} j\, P'_v(k,j) \right] \end{aligned}$$

This allows to determine $D_{i|i}$ for every node of a given TD.

B.4 Recursive Approach for the Evaluation of the Performance Global Metrics

In this appendix, we outline the recursive approach which, coherently with an inductive principle, allows to derive the average global performance metrics (namely, RE, D, and TE), on the basis of the average local performance metrics of a generic TD.

We recall that, thanks to the assumptions of the deterministic approach, all the TDs are identical and composed of n nodes. The recursive approach is detailed by considering the computation of D, but with the same approach it is also possible to derive RE and TE. The computation of the average D is carried out taking into account only the packets successfully arriving at the last reachable node, ignoring the unsuccessful retransmissions.

For all the values of m such that $m \leq n$, all the n nodes within the first TD are reached by the source. Therefore, the average delay coincides with the average transmission time of the source, given by Eq. (3.11), i.e.,

$$\overline{D}^{(m)} = \overline{T}^{\text{tx}}_{\text{src}}, \qquad 1 < m \leq n.$$

However, for all the values $m > n$, at least a retransmission is necessary to reach the mth node. In particular, if we consider the case $m = n+1$, the $(n+1)$th node can

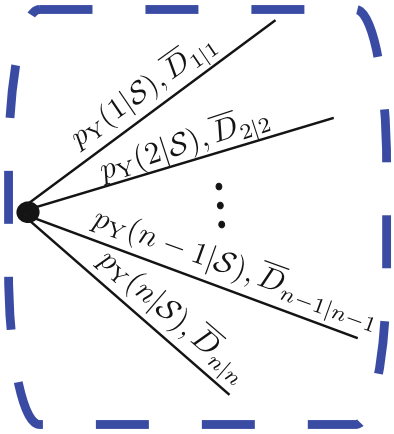


Fig. B.1 Tree-based computation of the average delay $\overline{D}^{(m)}$, when $m = n+1$

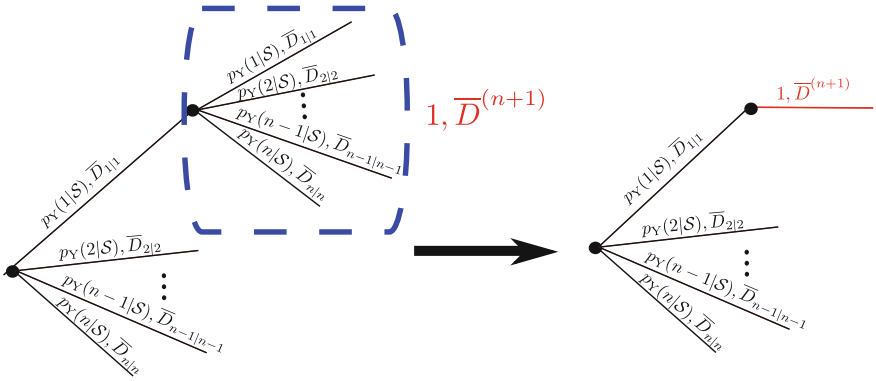


Fig. B.2 Tree-based computation of the average delay $\overline{D}^{(m)}$, when $m = n+2$

be reached only and only if a successful transmission is carried out by a node of the first TD. This event can happen in n different ways, each associated with a different delay. The tree of the possible decisions is represented in Fig. B.1, where every branch is labeled with the associated probability and with the corresponding value of delay. Since we are conditioning to the fact of having a successful transmission, the probability of the event {the ith node transmits} is given by $p_Y(i|\mathscr{S})$. Therefore, the average delay $\overline{D}^{(n+1)}$ can be obtained as follows:

$$\overline{D}^{(n+1)} = T_{\text{src}}^{\text{tx}} + \sum_{i=1}^{n} \overline{D}_{i|i}\, p_Y(i|\mathscr{S}).$$

When $m = n + 2$ the situation is slightly more complicated, since when the first node is selected in the first TD, two transmissions are needed to reach the $(n + 2)$th node. In this case, a second TD, identical to the first, is formed, thus leading to the addition of n branches to the tree, as shown in Fig. B.2. However, since the two TDs are identical, the branches following the event $\{y = 1|\mathscr{S}\}$, can be replaced by the

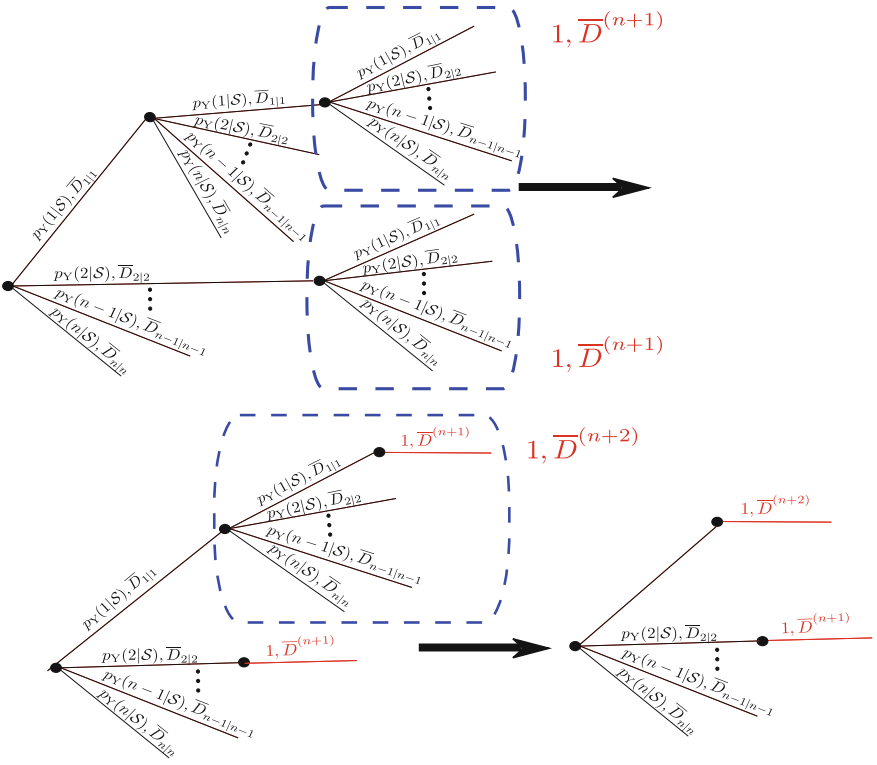


Fig. B.3 Tree-based computation of the average delay $\overline{D}^{(m)}$, when $m = n + 3$

average delay computed for $m = n + 1$. Therefore, one obtains:

$$\overline{D}^{(n+2)} = T_{\text{src}}^{\text{tx}} + (\overline{D}^{(n+1)} + \overline{D}_{i|i})\, p_Y(1|\mathscr{S}) + \sum_{i=2}^{n} \overline{D}_{i|i}\, p_Y(i|\mathscr{S}).$$

Similar considerations can be drawn in the case with $m = n+3$: the corresponding tree is shown in Fig. B.3. In this case, the two circled branches in the left figure, can be replaced by $\overline{D}^{(n+1)}$, obtaining the tree in the central figure that can be further simplifying by using $\overline{D}^{(n+2)}$, thus leading to the following expression:

$$\overline{D}^{(n+3)} = T_{\text{src}}^{\text{tx}} + \left(\overline{D}^{(n+2)} + \overline{D}_{1|1}\right) p_Y(1|\mathscr{S}) + \left(\overline{D}^{(n+1)} + \overline{D}_{2|2}\right) p_Y(2|\mathscr{S}) + \sum_{i=3}^{n} \overline{D}_{i|i}\, p_Y(i|\mathscr{S}).$$

Now, by induction it is possible to derive the formulation of $\overline{D}^{(N)}$ given in (3.10).

Appendix C
Batch-Based Group Key Management

In this Appendix, a new group key distribution protocol is presented. The proposed protocol allows a server (KDC) to efficiently distribute a group key to all members of a multicast group dealing with dynamic joins and leaves of users as group members. The proposed solution is summarized in Sect. C.2, and detailed in Sects. C.3 and C.4.

C.1 Related Works

According to the group communication paradigm, a single member can originate and deliver a message to the whole group of nodes, through multicast (or broadcast) communication services [9], and thus in a more efficient manner than an equivalent unicast-based solution. The first applications taking benefit of the group communications model, such as online gaming and audio/video streaming [10], have historically operated on the Internet. In recent years, the ever increasing diffusion of ad hoc (mostly wireless) networks has offered a new fertile ground for the development of new types of group-based applications. In scenarios such as wireless sensor networks [11], mobile ad hoc networks [12], and Internet of Things (IoT), a large number of applications (e.g., data dissemination, data gathering, peer-to-peer communications) need an underlying multicast data delivery service.

Securing group communications consists in providing confidentiality, authenticity, and integrity of messages exchanged within the group, through suitable cryptography services [13], and without interfering with the data path of the multicast data flow[2] [15]. The achievement of this goal in an efficient and scalable manner is a challenging task, as it requires that a large and dynamically varying number of users share cryptographic materials, even in the presence of unpredictable group membership changes, due to new users entering (joining) the network and to old users leaving the network. In fact, after any membership change, the shared cryptographic

[2] Iolus [14], for example, is a scheme that interferes with the normal packet stream, since a group security intermediary has to decrypt and encrypt all the packets transiting in its own group.

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7

materials should be refreshed through a suitable *rekeying* operation, so that a former group member has no access to current communications (*forward secrecy*) and a new member has no access to previous communications (*backward secrecy*) [16, 17].

While authenticity and integrity protection in group communications can be easily achieved through asymmetric cryptography, like in traditional point-to-point communications (e.g., through digital signatures), the simplest and most scalable way to provide data confidentiality within a multicast group is to encrypt the data through symmetric cryptography, with a secret key shared (only) by all users belonging to the group. Such a symmetric key is normally referred to as *group key*.

Under the assumption of using security primitives unbreakable for an attacker with limited computational power, the main issue in group communications consists in distributing the secret group key to all the legitimated users and updating it at any group membership change. Such a problem is known as *Group Key Distribution* (GKD) and it can be tackled by following two different models [18]: (i) Broadcast Encryption (BE) [19], which assumes that current data be decipherable independently of past transmissions (the receivers are *stateless*); (ii) Multicast Key Distribution (MKD), which allows the users to maintain state of the past cryptography material [20] (*stateful*). There are two main categories of MKD protocols: centralized [21] or distributed [22]. According to the former, the keys' distribution task is assigned to a single entity, denoted as Key Distribution Center (KDC). In the case of the distributed approach instead, the group key is established and maintained by the users themselves, in a distributed fashion. The centralized MKD approach has several advantages: (i) simplicity; (ii) a small number of exchanged messages compared to other methods; (iii) the possibility of operating on intrinsically broadcast channels, where the source (which also acts as MKD server) sends data to all the possible destinations. Distributed methods typically offer greater reliability, since they do not require any centralized entity to trust, but they have higher communication and computational costs and are not applicable to asymmetric communication scenarios, where data cannot be exchanged between any pair of nodes. For these reasons, in the rest of this Appendix we focus on centralized approaches.

In a centralized MKD protocol, among the different methods to achieve secure communications in a group of n members, one of the simplest consists in having: (i) a group key, shared by group members only and changed every time a user joins or leave the group [23]; (ii) n individual long-term keys shared (pairwise) between the KDC and every group member. A message sent by a member to the whole group is encrypted with the group key, so that only the remaining members can decrypt it. Instead, the individual keys are used for securing unicast communications and for reeking. The management of the group keys has a cost, in terms of number message exchanges, that varies according to the protocol used to update the group key to all members (rekeying). In the simplest case, the KDC separately sends the new group key, encrypted with the member's long-term key, to all group number, thus determining a number of exchanged messages proportional to n. The overall number of communications also depends on the average number of rekeyings. In some protocols, the group key is refreshed suddenly after any join or leave events. In these cases the number of rekeyings is directly proportional to the number of change of

membership events. In other cases, the group key is programmatically refreshed, according to a slotted schedule. These protocols have a constant number of rekeying operations, but they reduce the freedom of the nodes, and make impossible to perform instant evictions of malicious or dangerous users. Besides this classification, it is possible to define hybrid protocols, where join and leaves are performed programmatically, while evictions are performed immediately.

The technique described in this Appendix is based on a key derivation scheme that has been properly extended in order to deal with both unpredictable leave events and collusive attacks. In particular, we present a MKD protocol tailored for very dynamic ad-hoc networks, either wired or wireless. Time is partitioned in fixed-length intervals, each of them associated with a different group key. Even if a user can join anytime (asynchronously), it shall wait until the beginning of the next slot before becoming a group member. This introduces a delay, on average equal to half of the slot interval, but allows to reduce the number of rekeying acts. Similarly, the planned leave of a legitimate member shall also happen at the beginning of a slot period. In other words, the protocol is slotted and adopts a synchronous *batch rekeying* mechanism [23], which improves the efficiency without posing security threats. The protocol also provides proper mechanisms to deal with unpredictable leave events and to resist against collusive attacks.

The aim of the protocol is to minimize the computational burden of group members and the overhead, expressed in terms of number of exchanged messages, while achieving a sufficiently high security level. The proposed protocol can operate on very dynamic scenarios with a large number (thousands) of nodes and offers excellent performance under the assumption of a low evictions rate.

C.2 Protocol Overview

Let us consider a multicast group communication scenario in which the same data has to be securely sent to a group of destinations. In order to guarantee data confidentiality, the message has to be encrypted with a secret (group) key shared by, and only by, all group members. We consider a dynamic scenario in which, at any time, a new user may join the system as new group member, and an old user may leave the group. As described in the previous sections, this requires a suitable group key distribution protocol, able to distribute a new key to all members upon every change of group membership. We consider a key distribution scenario based on a trusted KDC that takes care of: (i) maintaining a secure association with all the users belonging to the system; (ii) generating a new group key every time the group membership changes; (iii) efficiently managing the distribution of the new group key to all group members, guaranteeing both forward and backward secrecy.

In a more general scenario, join and leave operations occur unpredictably, in a completely asynchronous and dynamic way. However, in order to optimize and significantly reduce the complexity and the number of exchanged messages required to handle group member changes and group key re-distribution (rekeying), a more

practical method is to allow the KDC to handle simultaneously a number of membership changes. This can be achieved by splitting time into intervals (sometimes referred to as "time slots" or, simply, "slots") and letting the KDC handle all membership changes that occur in the same time interval. Key distribution mechanisms that work in this way are often referred to as "batch" methods. Note that our proposed method applies when these time intervals have the same length or different lengths. However, very common scenarios are those in which membership changes are handled, for practical reasons, in a daily or monthly manner: this is the case, for example, of applications that consider service subscriptions with specific durations (expressed exactly in days or months). Other common possible time slot units can be minutes, seconds, or years.

Although the time slot in which a new user wants to join the system is in general difficult (or impossible) to predict (as it can apply at any time), there are many application scenarios in which the duration of the membership of a user is specified at the moment when the user joins the system, possibly further extended on the basis of a renewal strategy. Service subscriptions are often handled by applications in this way, with the possibility (in a limited subset of cases) of considering some form of revocation mechanisms in order to handle situations (often seen as exceptions) in which a membership has to be revoked in advance before its natural expiration time (for example, if a user unexpectedly leaves the system or if he/she is removed due to a misuse or for administrative reasons).

In spite of the above considerations, the majority of the proposed key management mechanisms do not take advantage of this operation and simply consider any leave event as not pre-determined, as it always occurs randomly.

On the opposite, we explicitly consider two different kinds of leave events: (i) "pre-determined" leave events, where the leave time is selected in advance when the user joins the network, or when it refreshes his/her membership, as in the case of a natural membership expiration; (ii) "unpredictable" leave events, when the time of leave does not coincide with the one selected at the time of joining or refreshing, for example in the case of explicit membership revocation. In our method, like in [24], both kinds of leave events are explicitly considered, taking the advantage of the balance of the former leaving strategy with respect to the latter.

We consider a different group key K_i for each time slot i with $i = 0, 1, 2, \ldots, N$. In order to efficiently handle both types of leave events, the group key is obtained through a one-way function of two sub-keys $K1_i$ and $K2$:

$$K_i \doteq f(K1_i, K2) \qquad i = 0, 1, 2, \ldots$$

with $K1_i$ and $K2$ properly managed in order to handle both types of leave events. In particular, the values $K1_i$ are associated to every time slots Δt_i (with $i = 0, 1, 2, \ldots, N$); they are pre-determined and provided to group members according to their assigned membership duration. The values of $K1_i$ are generated in an intelligent and secure manner in order to simplify the assignment to joining users, by providing only some root secret materials that can be used by the member to further

derive all $K1_i$ values associated with all time slots he/she subscribed for. $K1_i$ are then used to handle all new join and "pre-determined" leave events.

On the other hand, $K2$ is used to handle all "unpredictable" leave events. It is changed and re-distributed by the KDC to all (and only) group members, in a scalable way, similarly to other mechanisms already proposed in the literature.

Since the amount of operations and exchanged messages differ for managing of the subkeys $K1_i$ and $K2$, the total amount of operations and exchanged messages is a function of the rate of the "unpredictable" leave events over join and "pre-determined" leave events.

Details of how $K1_i$ and $K2$ are derived and managed are described hereafter.

C.3 Protocol Details

The objective of the proposed key management protocol is to provide a group key that can be securely shared by (and only by) all group members, taking into account and properly handling:

1. regular membership changes, that are due to new users that join the group and active members that leave the group for "clean" membership expiration ("pre-determined" leaves);
2. exceptional active member leaves, e.g., in the case of explicit membership revocation ("unpredictable" leaves).

In order to take into account membership changes of the first type, the overall time span is considered divided into a sequence of N time slots Δt_i with $i = 0, 1, 2, \ldots, N$ and in general, $\Delta t_i \neq \Delta t_j$ for $i \neq j$. In practice, however, it will be common to have $\Delta t_i = \Delta t_j = \Delta t \ \forall i, j$, with Δt equal to standard time units, such as a minute, a second, a month, etc. For each time interval Δt_i, in the following referred as "slot" i, a different group key K_i is determined. Consider now a user member x that will belong to the group from time t_a to time $t_b + 1$, i.e., from time slot Δt_a to time slot Δt_b: he/she will receive the subset of keys $S_X = \{K_i, \text{with } i = a, a + 1, a + 2, \ldots, b\}$.

According to the above approach, as far as only membership changes of the first type are considered, the KDC is requested to generate all keys K_i and give to each new incoming member only the subset of keys corresponding to the time slots over which he/she will belong to the group. If the member will stay for a total of m time slots, this will require the KDC to give to the new member m different keys. In order to limit the total amount of cryptographic material that the KDC has to send to each new member, a proper distribution protocol is adopted.

However, regular membership changes (first type) are not the only events that require the assignment and distribution of a new group key (i.e., a rekeying operation). In the case of an unpredictable leave event (second type) in time slot Δt_h of member y that negotiated with KDC a membership from time slot Δt_a to time slot Δt_b, at least all previously assigned keys (from $K_h + 1$ to K_b) must be re-assigned and distributed

to all valid group members. This is needed in order to prevent y to decrypt messages that are sent after time slot Δt_h with valid keys that he/she received by the KDC in joining the group.

To handle both types of membership changes in a secure and flexible way, the following key derivation and distribution protocol is proposed.

Let us consider N time slots, with $N = 2^D$. Each time slot Δt_i is associated with a key K_i defined as:

$$K_i \doteq f(K1_i, K2) \qquad i = 0, 1, 2, \ldots, N-1$$

where K_i, $K1_i$, and $K2$ are fixed or variable-length bit strings, and $f(.)$ is a cryptographic one-way function that returns a bit string of length equal to or greater than K_i. If $f(.)$ returns a bit string of length greater than K_i, a truncation can be applied. A cryptographic hash function $H()$ (for example SHA-1 [25] or MD5 [26]) can be used in place of $f(.)$:

$$K_i \doteq f(K1_i, K2) = H(K1_i \| K2) \qquad i = 0, 1, 2, \ldots, N-1$$

The subkey $K1_i$ is defined as follows. Consider a binary tree with depth equal to $D+1$, including the root node (level 0). At any level h, starting form 0, the binary tree has 2^h nodes. The last level is D, leading to $2^D = N$ leaves. Let's indicate with (h, j) the node j of level h, with $0 \leq h \leq D$ and $0 \leq j \leq 2^h - 1$. Each node (h, j) of the tree, excluding the last level D, has two child nodes that are respectively: left child $(h+1, 2j)$ and right child $(h+1, 2j+1)$. Each node (h, j) is associated to a value $x_{h,j}$ that is derived by the value of parent node as follows:

$$\begin{aligned} x_{h+1,2i} &\doteq f_0(x_h, i) \\ x_{h+1,2i+1} &\doteq f_1(x_h, i) \end{aligned}$$

or equivalently:

$$x_{h,i} \doteq \begin{cases} f_0(x_{h-1,i/2}) & i = 0, 2, 4, \ldots, 2^h - 2 \\ f_1(x_{h-1,(i-1)/2}) & i = 1, 3, 5, \ldots, 2^h - 1 \end{cases}$$

where $f_0()$ and $f_1()$ are two different cryptographic one-way functions. They could be also defined based on the same function $f()$ as follows:

$$\begin{aligned} f_0(x) &\doteq f(x) \\ f_1(x) &\doteq f(x+1) \end{aligned}$$

In this case we can write $x_{h,i}$ (recursively) as:

$$x_{h,i} \doteq f(x_{h-1,\lfloor i/2 \rfloor} + (i \bmod 2))$$

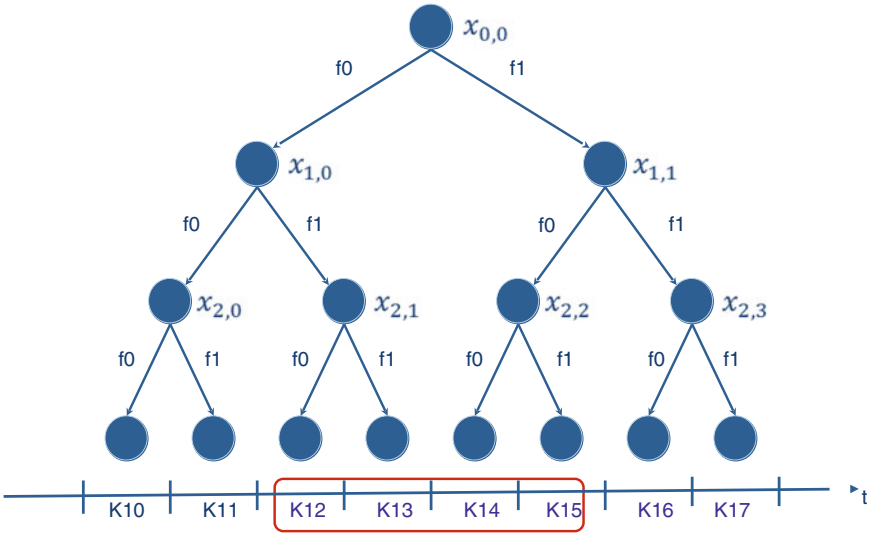


Fig. C.1 Deriving all $K1$ subkeys by applying functions f_0 and f_1

By repeatedly applying the previous equations, starting from the value $x_{h,i}$ of node (h, i), it is possible to generate all values associated to the nodes of the subgraph that has (h, i) as root. At the same time the value $x_{h,i}$ of node (h, i) can be obtained from the value associated to any node along the path from the $x_{h,i}$ to the tree's root $(0, 0)$.

Given such a binary tree, we define the subkey $K1_i$ equal to the value of the leaf i, that is:

$$K1_i \doteq x_{D,i}$$

Then, $K1_i$ can be obtained from the value associated to any node along the path from the leaf i to the tree's root, or equivalently from any values from $x_{D,i}$ to $x_{0,0}$. Figure C.1 shows the $K1$ subkeys derivation process described above. At the same time, starting from the value $x_{h,i}$ of node (h, i), it is possible to obtain all subkey values in the interval from $2^{D-h} \cdot i$ to $2^{D-h} \cdot (i + 1) - 1$ included, that is all subkeys from $K1_{2^{D-h} \cdot i}$ to $K1_{2^{D-h} \cdot (i+1)-1}$. Note that, as a special case, the value $x_{0,0}$ can generate all subkeys from $K1_0$ to $K1_{N-1}$. Figure C.2 shows how to obtain backward and forward secrecy by distributing the minimum set of values $x_{h,i}$ that cover the time period of a member's subscription.

This property can be used by the KDC to distribute the $K1_i$ subkeys to new members in a very efficient way, reducing from $O(N)$ to $O(\log(N))$ the number of values that the KDC has to pass to a new member in order to set subkeys for all the temporal period that the new member will belong to the group.

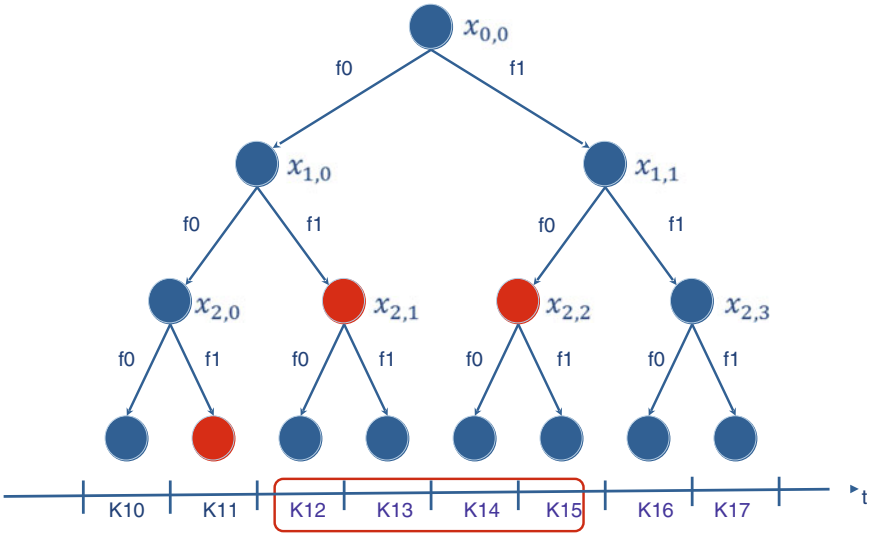


Fig. C.2 Achieving backward and forward secrecy

The worst case occurs when the node joins the group from time slot 1 to time slot N-1, included. In this case, $2 \cdot (\log_2(N) - 1)$ keys need to be distributed: $x_{D,1}$, $x_{D,N-1}$, $x_{D-1,1}$, $x_{D-1,\frac{N}{2}-1}$, $\ldots x_{2,1}$, $x_{2,2}$.

Let's now consider the subkey $K2$. Its value is maintained constant as far as only regular membership changes happen. As soon as a unpredictable leave event occurs, all un-expired keys of the leaving member must be revoked and replaced by new ones. This objective is reached by replacing the $K2$ that in turn will change all successive group keys K_i that are generated by the values of $K1_i$ and $K2$.

When a new $K2$ value is generated, this has to be distributed by the KDC to all remaining valid group members. This operation is very similar to the one faced by current centralized key distribution protocols: for example LKH [16] can be used.

C.4 Managing Keys for Unlimited Time Intervals

The described protocol assumes that the number of time slots is fixed and equal to $n = 2^D$. Therefore, it is inevitable that the key distribution protocol is doomed to come to an end eventually. In this section, we sketch a simple extension of the protocol to allow the KDC to handle time intervals that might last beyond the one covered by a single tree (i.e., more than n time slots). The basic idea is to instantiate as many trees as required in order to manage a time interval of arbitrary length. This extension makes it possible to manage time intervals of any length with just a slightly increased computational effort and memory consumption.

Time is split into intervals I_k of length ΔT and periods Π_i of length $n \cdot \Delta T$. Each period Π_i is associated with a given seed s_i, which is equal to the value of the root $x^i_{0,0}$ of a tree. Within a given period, the protocol works exactly as described above. If a subscription lasts beyond the end of a period, it is necessary to distribute keys from more then one tree. Each tree can be computed "on the fly" in the following way:

$$x^i_{0,0} \doteq g(s_i) \doteq g(h(s_{i-1}))$$

where s is a seed, $h(.)$ is a "one-way" function (i.e., hashing function), and $g(.)$ is a "blinding" function (i.e., XOR function). For instance, possible choices for $h(.)$ and $g(.)$ are

$$h(s_i) = H(s_{i-1}) = H^{i+1}(s)$$
$$g(s_i) = s \oplus h(s_i) = s \oplus H^{i+1}(iv)$$

where H is a hashing function and iv is an initial vector. Therefore:

$$s_0 = s \oplus H(iv)$$
$$s_1 = s \oplus H(H(iv)) = s \oplus H^2(iv)$$
$$\ldots$$
$$s_i = s \oplus H^{i+1}(iv)$$

The key $x_{h,i}$ can be calculated as

$$x_{h,i} = x^P_{h,I} = \underbrace{f(f(\ldots f(}_{h\text{times}} g(s_{P-1}) + \underbrace{a_0) + a_1) \ldots) + a_{h-1})}_{h \text{ bits}}$$

where $P = \lfloor i/2^h \rfloor$ is the index of the period, $I = i \bmod 2^h$ is the index of the period's interval, and $a_0, a_1, \ldots a_{h-1}$ are the h bits of the binary representation of I.

This mechanism makes it possible to extend the functioning of the protocol to cover an unlimited time interval without extra memory requirements as keys can be computed on the fly with no particular computational effort since operations like hashing and XORing are very lightweight.

References

1. Amoretti, M., Agosti, M., Zanichelli, F.: Deus: a discrete event universal simulator. In: 2nd ICST/ACM International Conference on Simulation Tools and Techniques (SIMUTools 2009). Rome, Italy (2009)
2. Amoretti, M.: A modeling framework for unstructured supernode networks. IEEE Commun. Lett. **16**(10), 1707–1710 (2012)

3. Network Simulator 2 (ns-2). http://isi.edu/nsnam/ns/
4. Montresor, A., Jelasity, M.: PeerSim: a scalable P2P simulator. In: 9th International Conference on Peer-to-Peer (P2P'09), pp. 99–100. Seattle, WA (2009)
5. Zhou, B., Xu, K., Gerla, M.: Group and swarm mobility models for ad hoc network scenarios using virtual tracks. In: IEEE Military Communications Conference (MILCOM). Monterey, California, USA (2004)
6. Seskar, I., Marie, S., Holtzman, J., Wasserman, J.: Rate of location area updates in cellular systems. In: IEEE Vehicular Technology Conference (VTC'92). Denver, Colorado, USA (1992)
7. Nagate, A., Hoshino, K., Mikami, M., Fujii, T.: A field trial of multi-cell cooperative transmission over lte system. In: IEEE International Conference on Communications (ICC 2011). Kyoto, Japan (2011)
8. Feller, W.: An Introduction to Probability Theory and Its Applications, vol. 1. Wiley, New York (1968)
9. Deering, S.: Host extensions for IP multicasting. The Internet Engineering Task Force (IETF) (1989)
10. Turletti, T., Huitema, C.: Videoconferencing on the internet. IEEE/ACM Trans. Networking **4**(3), 340–351 (1996)
11. Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Comput. Netw. **38**(4), 393–422 (2002)
12. Schoch, E., Kargl, F., Weber, M., Leinmuller, T.: Communication patterns in VANETs. IEEE Commun. Mag. **46**(11), 119–125 (2008)
13. Verma, M., Huang, D.: SeGCom: secure group communication in VANETs. In: Proceedings of IEEE International Conference on Consumer Communication and Networking (CCNC), pp. 1–5. Las Vegas, NV, USA (2009)
14. Mittra, S.: Iolus: a framework for scalable secure multicasting. ACM SIGCOMM Comput. Commun. Rev. **27**(4), 277–288 (1997)
15. Rafaeli, S., Hutchison, D.: A survey of key management for secure group communication. ACM Comput. Surv. **35**, 309–329 (2003)
16. Wong, C.K., Gouda, M., Lam, S.: Secure group communications using key graphs. IEEE/ACM Trans. Networking **8**(1), 16–30 (2000)
17. Micciancio, D., Panjwani, S.: Optimal communication complexity of generic multicast key distribution. IEEE/ACM Trans. Networking **16**, 803–813 (2008)
18. Gerla, M., Kleinrock, L.: Vehicular networks and the future of the mobile internet. Comput. Netw. **55**(2), 457–469 (2011)
19. Berkovits, S.: How to broadcast a secret. In: Proceedings of the International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT), pp. 535–541. Springer, Brighton (1991)
20. Ballardie, A.: Scalable multicast key distribution. The Internet Engineering Task Force (IETF) (1996)
21. Lin, J., Huang, K., Lai, F., Lee, H.: Secure and efficient group key management with shared key derivation. Comput. Stand. Interfaces **31**(1), 192–208 (2009)
22. Lee, P., Lui, J., Yau, D.: Distributed collaborative key agreement and authentication protocols for dynamic peer groups. IEEE/ACM Trans. Networking **14**(2), 263–276 (2006)
23. Li, X., Yang, Y., Gouda, M., Lam, S.: Batch rekeying for secure group communications. In: ACM Proceedings of International Conference on World Wide Web (WWW), pp. 525–534. ACM, Hong Kong (2001)
24. Briscoe, B.: MARKS: zero side effect multicast key management using arbitrarily revealed key sequences. In: Rizzo, L., Fdida, S. (eds.) Networked Group Communication. Lecture Notes in Computer Science, vol. 1736, pp. 301–320. Springer, Berlin (1999)
25. Federal Information Processing Standards Publication: FIPS 180–3. Secure Hash Standard (SHS). http://csrc.nist.gov/groups/ST/toolkit/secure_hashing.html (2008)
26. Rivest, R.: RFC 1321, The MD5 message-digest algorithm. The Internet Engineering Task Force (IETF) (1992)

Index

M. Picone et al., *Advanced Technologies for Intelligent Transportation Systems*,
Intelligent Systems Reference Library 139, DOI 10.1007/978-3-319-10668-7