Driss Benhaddou · Ala Al-Fuqaha
Editors

Wireless Sensor and Mobile Ad-Hoc Networks

Vehicular and Space Applications

Springer

Wireless Sensor and Mobile Ad-Hoc Networks

Driss Benhaddou • Ala Al-Fuqaha
Editors

Wireless Sensor and Mobile Ad-Hoc Networks

Vehicular and Space Applications

Springer

Editors
Driss Benhaddou
Engineering Technology Department
University of Houston
College of Technology
Houston, TX, USA

Ala Al-Fuqaha
Western Michigan University
Kalamazoo, MI, USA

ISBN 978-1-4939-2467-7 ISBN 978-1-4939-2468-4 (eBook)
DOI 10.1007/978-1-4939-2468-4

Library of Congress Control Number: 2015932501

Springer New York Heidelberg Dordrecht London

Printed on acid-free paper

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

Preface

Wireless sensor network (WSN) is becoming ubiquitous in several applications including remote health care, fire tracking, integrated system health monitoring (ISHM), industrial automation, and space, to name a few. Their small size and pervasive computing characteristics enabled intelligence to be implemented deep in network-centered applications and systems. In particular, space applications are using WSNs to ensure the overall missions' success from asset management to crew heath monitoring. For instance, space crew health performance can be tracked down using telemedicine by taking advantage of sensor local processing and networking capabilities. The sensors can monitor the profile of key vital signs of astronauts and predict whether the astronaut is at the risk to be injured.

Implementation of WSN-based systems that provides end-to-end solutions requires innovative approaches at different levels of overall system implementation and integration. A WSN system involves components at different levels from sensor devices, sensor processing and network interfacing, energy-efficient and reliable protocols that support quality of service (QoS) requirement, middleware design that efficiently collects, archives the data, and makes it available for further processing and visualization, and applications that use the data to make decision pertaining to a myriad of solutions and applications. In space application, the system should be autonomous and scalable and support modularity and interoperability. Modularity is the ability to automatically plug a sensor in the network without the need to manually configure and set up the system, i.e., plug and play. Interoperability is very important, since a scalable system should not rely on one technology or one vendor. Sensor standards, such as IEEE 1451, are being developed to standardize such interfaces that will enable the implementation of smart sensor networks with plug and play capability. Scalability is another important characteristic, since we need to be able to deal with a large number of sensors (e.g., thousands or millions) and dynamically add or discard a sensor without the need for reconfiguring the whole system. Autonomy is very important characteristic of system deployed in space. It should be able to have self-configuration, self-healing, self-management, and self-update.

Concurrently, current network-centered applications are moving toward the Cyber-Physical Systems (CPS) that deeply integrate sensing, control, and networking in physical systems. CPS are expected to be autonomous and bring about innovations in different physical systems such as vehicular networks, building management, and smart grid, to name a few.

The Internet of Vehicles (IoV) integrates sensors and microcontrollers in the vehicles with fixed roadside infrastructure to form an intelligent *vehicle grid*. The vehicular cloud provides the communication protocols, computational infrastructure, services, and applications for the efficiency of the vehicle grid. The vehicular cloud resides on top of the vehicle grid and is the backbone for its operations. The vehicle grid is essentially a Vehicular Ad-hoc Network (VANET) of On-Board Units (OBUs) in vehicles and Roadside Units (RSUs) in the fixed road infrastructure. Vehicle OBUs comprise localization systems (e.g., Global Positioning System, Inertial Measurement Unit, etc.), processing units, sensors, and radio transceivers, mounted in and around the vehicle. RSUs are sensors and microcontrollers installed alongside and in the road, for example, cameras in traffic lights and road signs and pressure sensors and traffic light actuators on the road. The IEEE standards and protocols for Wireless Access in Vehicular Environments (WAVE) define the inter-networking for vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications. Commercialization of IoV is dependent upon the safety and security of the applications and services offered to users through the vehicle grid.

This book presents a series of challenges and work related to the application of WSN in space and vehicular applications. The book is divided into three parts: Part I focuses on the fundamentals of WSNs and includes Chaps. 1–4. Part II addresses applications of WSNs in Space and includes Chaps. 5–7. Part III gathers work of WSNs in vehicular applications and includes Chaps. 8–10.

Chapter 1, entitled "Introduction to Wireless Sensor Networks," coauthored by Zill-E-Huma Kamal and Mohammad Ali Salahuddin clarifies the key concepts and components of WSN architecture and implementations. It delineates the inherent characteristics of the WSNs and their smart sensor nodes. It also discusses the data delivery models and traffic patterns that instigate the design and development of novel network architecture protocols for WSN and distinguish them from its peers in other infrastructureless computing paradigms.

Chapter 2, entitled "Introduction to Mobile Ad-Hoc and Vehicular Networks," coauthored by Moussa Ayyash, Y. Alsbou, and Mohamed Anan presents Mobile Ad-hoc Networks (MANET) and their application in VANETs. It introduces their challenges and presents various protocols and applications tailored for vehicular networks.

Chapter 3, entitled "Routing in WSNs for Space Application," coauthored by Mohamed Riduan Abid and Driss Benhaddou describes routing protocols in WSN and the challenges associated with them. It classifies different protocols and presents in detail the energy efficiency components of these protocols.

Chapter 4, entitled "Middleware Architecture in WSN," coauthored by Mehdia Ajana El Khaddar Ajana, Hamid Harroud, Mohammed Boulmalf, and Mohammed Elkoutbi addresses the middleware design and related issues in WSN. Middleware

plays a key role in data collection, processing, and archiving of data for further processing and analysis. It also examines various approaches of middleware design, compares, and suggests different types of applications where each approach can be used. Finally, it proposes an enhanced middleware framework: FlexRFID for the integration of RFID and WSN.

Chapter 5, entitled "Space Applications of Low-Power Active Wireless Sensor Networks and Passive RFID Tags," coauthored by Richard J. Barton, Raymond S. Wagner, and Patrick W. Fink details the work developed in NASA JSC and includes low-power active WSN technology, standards, and space applications; rapid prototyping and testing of low-power active WSN devices and systems at NASA JSC; comparative performance evaluation of ZigBee and ISA100.11a in space habitat environment analogs; and passive RFID inventory tracking and sensing technologies, standards, and space applications.

Chapter 6, entitled "Predictive Data Reduction in Wireless Sensor Network Using Selective Filtering for Engine Monitoring," coauthored by David James McCorrie, Elena Gaura, Keith Burnham, Nigel Poole, and Roger Hazelden examines the application predictive data reduction in WSNs. Since transmissions consume a large portion of a node's energy budget, reducing the data transmitted in the network has an advantage of increasing the overall network lifetime. A new method for selective filtering of sensed data based on state identification is devised, using a skewed double exponentially weighted moving average filter for accurate state predictions.

Chapter 7, entitled "Space Crew Health Monitoring," authored by Azhar Rafiq focuses on crew health monitoring system in space application. With NASA's Vision for Space Exploration that will last a lengthy missions beyond the limits of evacuation or earth-based medical response, there is a need for capabilities that are autonomous medical practice. This chapter will present a system within a system where noninvasive physiological sensors are integrated into the larger computer system of NASA's MIII suit combining communications, avionics, and informatics. Physiological monitoring provides an infrastructure for health management between crewmember performing EVA and the spacecraft.

Chapter 8, entitled "AGORA: A Versatile Framework for the Development of Intelligent Transportation System Applications," coauthored by Mohammad Ali Salahuddin and Ala Al-Fuqaha deals with the AGORA platform for VANET applications. It is a novel framework for Intelligent Transportation Systems that interconnects wireless devices with pedestrians and vehicles through its infrastructure and its cloud services. The chapter presents a detailed discussion of the hardware and software components of AGORA architecture, followed by a thorough discussion of the suite of accompanying ITS applications and its development environment.

Chapter 9, entitled "Model, Analysis, and Improvements for Inter-Vehicle Communication Using One-Hop Periodic Broadcasting Based on the 802.11p Protocol," coauthored by Tseesuren Batsuuri, Reinder J. Bril, and Johan J. Lukkien presents models, analysis, and improvements in support of inter-vehicle communication

using IEEE 802.11p. The chapter also describes and evaluates one-hop Periodic Broadcast Communication (oPBC) technology for safety applications.

Chapter 10, entitled "A Survey of Security and Privacy in Connected Vehicles," coauthored by Lotfi Ben Othmane, Harold Weffers, Mohd Murtadha Mohamad, and Marko Wolf overviews security issues in WSN and VANET applications. The work provides a taxonomy for security and privacy aspects of connected vehicle. The aspects include security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. It also uses taxonomy to classify the main threats to connected vehicles, and existing solutions that address the threats.

We would like to acknowledge the contribution of collaborators, colleagues, and students who have directly and indirectly impacted the development of the material in this book.

Houston, TX, USA — Driss Benhaddou
Kalamazoo, MI, USA — Ala Al-Fuqaha

Contents

Part I
Overview, Architecture and Enabling Technologies

Introduction to Wireless Sensor Networks

Zill-E-Huma Kamal and Mohammad Ali Salahuddin

Abstract Wireless Sensor Networks (WSNs) enjoy great benefits due to their low-cost, small-scale factor, smart sensor nodes. Not only can they be employed in cumbersome and dangerous areas of interest, for monitoring or controlling the region, but they can also be deployed to automate mundane tasks. Early sensory units were expensive and lacked the computational and communicational capabilities of current smart sensor nodes, which can now sense, process, store, and forward data, all being powered by a battery.

Myriad applications exist that leverage WSNs as low-cost solutions for observing the habitat and environment, from military and civilian surveillance and target detection and tracking applications, to precision farming and agriculture, patient monitoring in health care, residential applications like energy management, for safety and efficiency in vehicular networks to outer space explorations.

The diversity of the applications of WSNs imposes varying design, implementation, and performance requirements on the WSNs. Therefore, for a thorough understanding of the different design and implementation techniques, we must understand the inherent characteristics of WSNs and their smart sensor nodes. This intrinsic nature of the application-specific WSNs makes classification and taxonomy delineation difficult and cumbersome.

In this chapter, we will delineate the inherent characteristics of the WSNs and their smart sensor nodes. Then, we will discuss the data delivery models and traffic patterns that instigate the design and development of novel network architecture protocols for WSN and distinguish them from its peers in other infrastructure-less computing paradigms.

We compare WSN with its peers, with respect to the problem space of WSN applications, followed by a brief overview of the challenges in programming WSN motes. Then, we present an overview of TinyOS, an operating system for WSN motes, and conclude with an overview of the challenges and limitations of WSNs.

Z.-E-H. Kamal (✉)
Montreal, Quebec, Canada
e-mail: huma.kamal@gmail.com

M.A. Salahuddin
Université du Québec à Montréal, Montreal, Quebec, Canada
e-mail: mohammad.salahuddin@ieee.org

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_1

1 Introduction to Wireless Sensor Networks

Originally, sensors were electromechanical detectors for measuring physical quantities. Their first use can be traced back to 1933, in the first room thermostats [1]. Early microelectromechanical systems (MEMS) consisted of a multi-chip, where a sensor and its electronics and mechanics were housed on separate chips and packages. This resulted in larger size, more cost, and lower yield of the sensor [2]. Recent advances in MEMS and integrated circuits (IC) have enabled the development of small-scale sensors and the integration of its actuators and electronics into one cost-effective high-performance chip. Over the past decade, these sensors have evolved into smart sensors, which now include an on-board processor, memory, and transceiver, all in a small-scale factor, powered by a battery source. These smart sensors constitute a node in the Wireless Sensor Network (WSN).

The benefit of the small-scale node is twofold, first, the low production cost and second, the easy and low installation cost. Currently, the price of a WSN node ranges in hundreds to thousands of dollars; however, it is envisioned to reduce to a couple of dollars with advances in technology and mass production [3]. The small size and low cost of the node allows an ease in the installation process, where nodes can be randomly air-dropped or precisely placed, based on the application.

These low-cost nodes with processing, storage, and sensing capabilities, coupled with the attractive infrastructure-less networking capabilities of the transceiver, market the WSN as a powerful, low-cost solution for numerous problems in diverse areas of research. They offer collaboration in a distributed environment for sensing and processing information. Vital information is routed through a multi-hop ad hoc network to a WSN sink, a collector of data or to a base station, which acts as a gateway to a fixed infrastructure. Variations of WSNs include nodes with actuators that comprise a Wireless Sensor and Actuator Network (WSAN) [4, 5], mobile nodes in WSNs [3], or nodes capable of handling multimedia content like audio and video streaming or still images in a Wireless Multimedia Sensor Network (WMSN) [6].

Myriad applications exist that leverage WSNs as low-cost solutions for observing the environment, for example, in military and civilian surveillance and target detection and tracking applications (e.g., [7–10]), for monitoring and controlling industrial processes [11], for precision farming and agriculture [12], environment and habitat monitoring [13, 14], patient monitoring in health care [15, 16], residential applications like energy management [17], in vehicular networks for safety [18] and efficiency [19] and even for outer space explorations [20].

In the following sections, we will discuss the various characteristics of WSN nodes and the intrinsic nature of WSN, followed by a discussion of the traffic patterns and network architecture protocols for WSN and their taxonomy. We will proceed with an overview of TinyOS, an operating system for WSN, and delineate a sample program to illustrate the methodology of programming in WSN. We will conclude with a summary of the challenges and limitations of WSN.

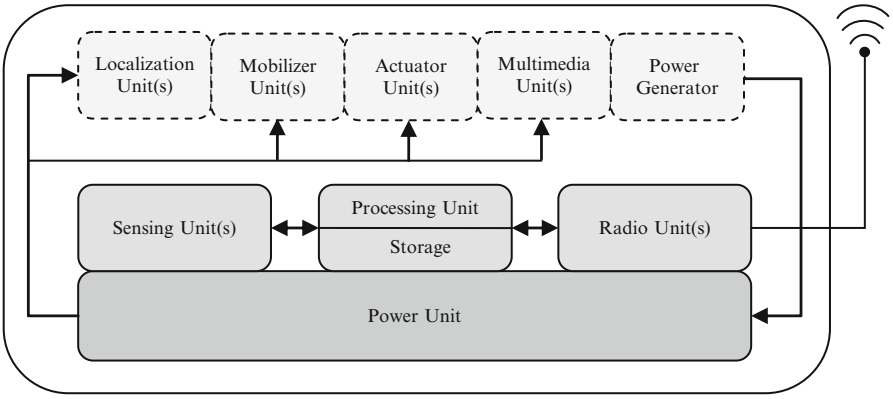

Fig. 1 A typical WSN mote with its fundamental units

1.1 WSN Nodes and Their Characteristics

Technological advances in MEMS and IC instigate the widespread availability of low-cost, small-scale sensors, which evolved into smart, battery powered sensors, with processing and communicating capabilities. These smart sensors constitute the WSN node, referred to as motes, hereafter, in honor of the first WSN nodes, which were University of Berkley's Rene and Mica Motes [21]. Figure 1 illustrates a typical WSN mote with its fundamental units. Minimally, a mote consists of a battery-operated unit with single or multiple sensors, a processing unit with storage and a transceiver. Typically, expansion slots exist or can be attached to expand the mote to include other application-specific units, such as global positioning system (GPS) for localization, or power harvesting units from solar or wind energy, or complementary metal–oxide semiconductor (CMOS) chips for multimedia capabilities.

These motes are placed on programming boards, interfaced with a computer, so that the WSN application and its operating system can be flushed into the memory of the mote. At this time, the motes can also be programmed with a specific identification number and/or group identification number. Various motes can also be programmed without a physical connection to the computer, known as over-the-air (OTA) programming.

WSN motes can vary greatly, with respect to the size, their cost, processing power, communication range, protocols, and operating systems. WSN motes can be as large as a shoebox, e.g. Sensoria Wireless Integrated Network Sensors (WINS) Next Generation (NG) 2.0 [22], or as miniature as a coin, like Moteiv Corporation's Tmote Mini [23], but typical WSN mote dimensions are in the order of a couple of centimeters [24].

WSN motes are typically equipped with multiple sensors for sensitivity to various environmental factors, e.g. mechanical, thermal, biological, chemical, optical,

and magnetic. Often motes have expansion slots that enable them to be equipped with mechanical actuators, wheels for mobility or CMOS chips or microphone for multimedia capabilities. The processors used in these motes can range from ultra-low-power 8 bit processors to more powerful 32 bit processors, similarly, memory space can vary from a couple of kilobytes to the order of megabytes [25].

Motes are equipped with short-range radio frequency (RF) transceivers to enable WSN applications [25] and to ease the query and retrieval of data from the WSN [26]. The use of a short-range radio directly influences the antenna size, since short-range radio waves have higher frequencies and shorter wavelengths they can be received and transmitted by small compact antennas. Through the high frequencies of 800–1,000 MHz, IEEE 802.15.4 or the 2.4 GHz Bluetooth, the low-power radios offer varying bandwidths [25]. The low-transmission range of the motes instigates a multi-hop WSN. Though current WSN motes are equipped with RF radios, communication via infrared, ultrasound, and inductive fields [27] has also been explored [25].

WSN motes can also be equipped with power generation units that harvest power ambient energy sources such as solar [28], mechanical, and thermal [29]. Table 1 presents a comparison of some WSN motes used in academia and industry w.r.t. these characteristics [25, 24, 30].

We will now consider the characteristics of WSNs with respect to the various application and environmental aspects.

1.2 Characteristics of WSNs

WSNs are deployed in a region of interest over a period of time. Since the motes have a short-range radio and a small coverage area, WSNs typically contain a large number of motes. These motes form multi-hop networks and collaborate with each other to maintain connectivity and coverage. Apart from the traditional concerns on collaboration and communication in an ad hoc environment, WSN are also plagued with power and energy management concerns, due to the battery-operated motes. In this section, we will delineate the various design criteria of application-specific WSNs.

Region of Interest The region of interest in WSN applications can be divided into those that are dangerous or isolated versus those that are mundane and cumbersome. In both scenarios, there is little or no infrastructure [3, 26]. For example, volcano monitoring [13], a dangerous task for humans, can be accomplished using low-cost motes that are deployed in the region. These motes are not only expendable but also provide critical data analysis and lifesaving information. However, a WSN used in life rhythm analysis for the elderly [31] can automate the mundane process of gathering vital signs and provides continuous statistics, rather than at fixed intervals.

Modes of Deployment There are two distinct mote deployment strategies, random and precise. In random deployment, motes are randomly distributed like nodes

Table 1 Comparison of some WSN motes

Platform	CPU	Clock (MHz)	RAM/Flash/EEPROM	Radio transceiver	BW (kbps)	Freq. (MHz)	OS
AWAIRS 1	Intel StrongARM SA1100	59–206	1M/4M	Conexant RDSSS9M	100	900	MicroC/OS
μAMPS	Intel StrongARM SA1100	59–206	1M/4M	National LMX3162	1,000	2,400	μOS
Dot	Atmel Atmega 163	8	1K/16K/32K	RFM TR1000	10	916.5	TinyOS
BT Node	Atmel Atmega 128L	8	4K/128K/4K	ZV4002 BT/CC1000	1,000	2,400	TinyOS
Smart-its	PIC 18F252	8	3K/48K/64K	Radiometrix	64	433	Smart-its
Mica2	Atmel Atmega 128L	8	4K/128K/512K	Chipcon CC1000	38.4	900	TinyOS
Mica2Dot	Atmel Atmega 128L	4	4K/128K/512K	Chipcon CC1000	38.4	900	TinyOS
iBadge	Atmel Atmega 128L	8	4K/128K	Ericsson ROK101007 BT	1,000	2,400	Palos
CENS Medusa MK2	Atmel Atmega 128L/Atmel AT91FR4081	4/40	4K/32K/136K/1M	RFM TR1000	10	916	Palos
iMote	Zeevo ZV4002 (ARM)	12–48	64K/512K	Zeevo BT	720	2,400	TinyOS
U3	PIC 18F452	0.031–8	1K/32K/256	CDC-TR-02B	100	315	Pavenet
RFRAIN	Chipcon CC1010 (8051)	3–24	2K/32K	Chipcon CC1010	76.8	0.3–1,000	RFRAIN Libraries
Nymph	Atmel Atmega 128L	4	4K/128K/512K	Chipcon CC1000	38.4	900	Mantis
Telos	TI MSP430F149	8	2K/60K/512K	Chipcon CC2420	250	2,400	TinyOS
MicaZ	Atmel Atmega 128L	8	4K/128K	Chipcon CC2420	250	2,400	TinyOS

Table 1 (continued)

Platform	CPU	Clock (MHz)	RAM/Flash/EEPROM	Radio transceiver	BW (kbps)	Freq. (MHz)	OS
Particle2/29	PIC 18F6720	20	4K/128K/512K	RFM TR1001	125	868.35	Smart-its
eyesIFXv2	TI MSP430F1611	8	10K/48K	Infineon TDA 5250	64	868	TinyOS
iMote2	Intel PXA 271	13–104	256K/32M	Chipcon CC2420	250	2,400	TinyOS
TelosB/Tmote Sky	TI MSP430F1611	8	10K/48K/1M	Chipcon CC2420	250	2,400	TinyOS
Ember RF Module	Atmel Atmega 128L	8	4K/128K	Ember 250	250	2,400	EmberNet
XYZ Sensor node	OKI ML67Q500x (ARM/THUMB)	1.8–57.6	4K/256K/512K	Chipcon CC2420	250	2,400	SOS
Ant	TI MSP430F1232	8	256/8K	Nordic nRF24AP1	1,000	2,400	Ant
ProSpeckz II	Cypress CY8C2764	12	256/16K	Chipcon CC2420	250	2,400	Speckle net
Fleck	Atmel Atmega 128L	8	4K/128K/512K	Nordic nRF903	76.8	902–928	TinyOS
Sun Spot	Atmel AT91FR40162S	75	256K/2M	Chipcon CC2420	250	2,400	Squawk VM (Java)
ECO	nRF24E1 (8051)	16	4K/512/32K	Nordic nRF24E1	1,000	2,400	–
SHIMMER	TI MSP430F1611	4/8	10K/2G	WML-C46A BT/CC2420	250	2,400	TinyOS
IRIS	Atmel Atmega 1281	8	8K/640K/4K	Atmel ATRF230	250	2,400	TinyOS
WaspMote	Atmel Atmega 1281	8	8K/128K/4K	eUnistone 31308/2	38.4	2,400	–
				Zig Bee		2,400/868/900	
				WiFi—802.11 b/g		2,400	
				GSM—SIM900 (SIMCom)		850/900/1800/1,900	
SquidBee	ATmega8/ATmega168	16	1K/8 or 16K/512K	MaxStream	250	2,400	–

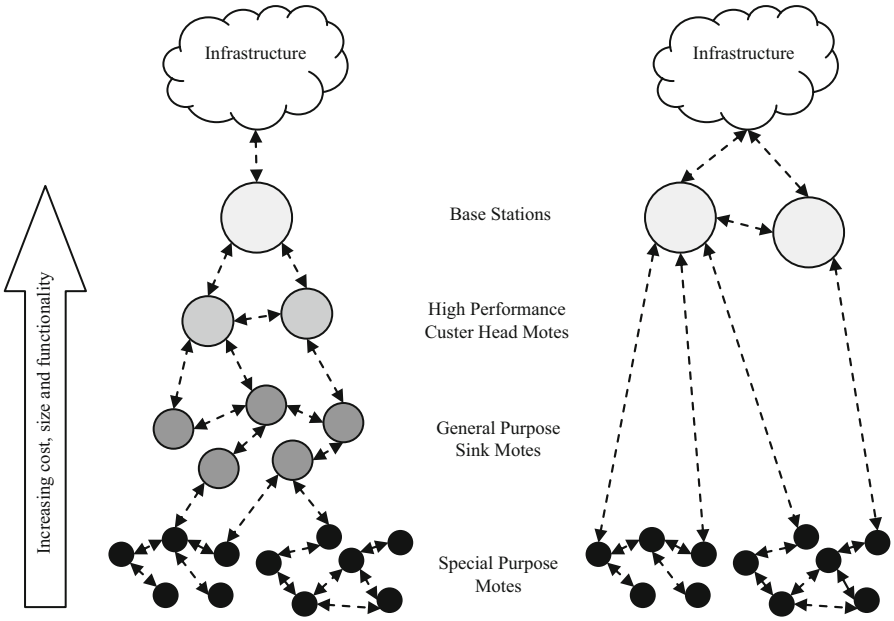

Fig. 2 WSN architecture, hierarchical (*left*) and flat (*right*)

in wireless ad hoc networks. This deployment could be rocket ejected or aircraft sown [32], e.g. in the aerial spread of motes for volcano monitoring [13]. Precise deployment usually consists of manual or pre-planned placement of motes, e.g. habitat monitoring on Skomer Island [33]. The deployment strategies used affect the structure of WSN and can have an impact on coverage of the region and cost of deployment [26].

Organization and Architecture WSN can be organized into two typical structures, flat or hierarchical. Figure 2 illustrates the flat and hierarchical WSN. In a flat network, all motes in the network have the same role and importance, whereas, in a hierarchical organization, motes are clustered or organized into groups with different motes playing different roles, such as general purpose sensing motes or data aggregators or forwarders. As we climb up the hierarchy, the mote's functionality increases with respect to its cost, size, processing power, storage size, etc.

For best results, WSN deployment should consist of a hierarchical organization of heterogeneous motes, integrating a larger number of low-power general purpose motes with smaller number of specialized or high-performance motes [24]. Furthermore, these motes can be equipped with actuators [4] or consist of mobile motes [3].

In WSN architectures, single or multiple base stations preside over the entire WSN. These base stations act as a gateway between the WSN and other fixed infrastructure, e.g. Internet. The base stations are typically, high-performance, human

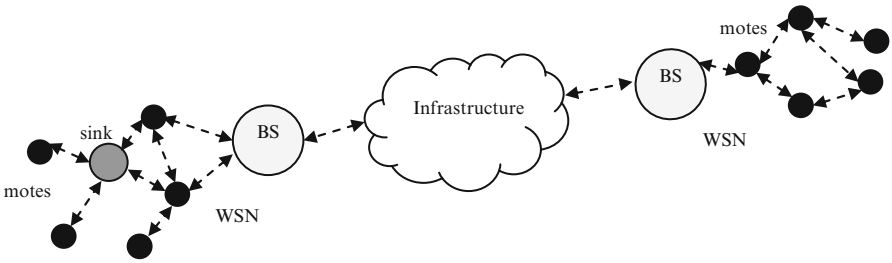

Fig. 3 Global WSN infrastructure

operated devices with rechargeable power source, e.g. laptops interfaced with a mote. Lower down the hierarchy would be specialized motes with slightly better configurations than the low-cost, low-power general purpose motes at the bottom of the hierarchy or pyramid. The special purpose motes offer data aggregation, or fusion, and more complex computational and communicational power, making them ideal sink nodes in the WSN. The general purpose motes at the bottom of the hierarchy typically act as data collectors and forwarders, as illustrated in Fig. 2.

Multiple WSN can be interconnected to form a global WSN, referred to as global sensing infrastructure [34], as illustrated in Fig. 3.

Lifetime of the WSN WSN are deployed with the intention of providing long-term data collection at previously unimaginable scales and resolutions [35]. Typical WSN habitat monitoring applications benefit from long-term data that help decipher data trends and are necessary to detect significant change in habitat [36]. Thus, the lifetime of a WSN is a fundamental characteristic. It is bounded by the finite power source of the battery-operated motes and the nature of the deployed region, and infeasible or cumbersome task of replacing and disposing these batteries. This necessitates power management in hardware and software components of the motes [37, 3] and in WSN protocols [38]. Reinforcement strategies are also being explored in WSN through power harvesting techniques, from solar [28], mechanical, and thermal [29] sources.

Ad Hoc Communication The lack of infrastructure in WSN is an intrinsic property of other communication systems like Bluetooth and mobile ad hoc networks (MANETs) [3]. Bluetooth employs discovery protocols and MANETs use robust and dynamic configuration protocols to establish and maintain the network infrastructure. Though these systems are the closest peers of WSN, there are fundamental differences that distinguish WSN from its peers [3]. First, WSN has larger number of motes with smaller transmission power and radio range, whereas Bluetooth and MANETs have smaller number of nodes with larger transmission power and longer radio ranges. Second, WSN and MANETs suffer from dynamic topology changes due to mote/node mobility and drop-out, but rate of mobility and drop-out is slower in WSN. Lastly, WSN motes have a finite, non-rechargeable power source, whereas Bluetooth and MANET nodes can be recharged. The critical power management in WSN imposes challenging requirements in WSN protocols and inhibits the direct

use of existing Bluetooth and MANET protocols in WSN [3]. Therefore, WSN needs power-aware protocols for a large-scale, autonomous, and heterogeneous network of low-processing power and low-transmission rate motes. These protocols are needed for establishing and maintaining a secure network infrastructure through self-discovery, configuration, and healing, routing and inherent fault-tolerance.

Self-configuration and Organization Various types of self-discovery algorithms exist for use in WSN, some are adaptations from Bluetooth discovery mechanism and others developed specially for WSN. Algorithms delineated in [39–41] and their like consist of a suite of protocols responsible for discovering motes and organizing the wireless infrastructure, despite topology changes and mote failures.

Connectivity and Coverage Establishing and maintaining connectivity and coverage is an essential step in the deployment of a WSN. It is important to determine the initial number of motes required to maintain connectivity and coverage in the region of interest, in face of mote failure and drop-out. It is generally assumed that mote communication and sensing range are uniform and unidirectional, usually depicted as circular regions (cf. Sect. 2.1.1, Fig. 4).

Furthermore, it is imperative for the configuration algorithms to maintain connectivity in the face of dynamic topology changes. Berman et al. [42] propose dominating k-connectivity, such that in the face of up to k mote failures, connectivity to a WSN sink is still maintained. Abbasi et al. [43] present a recovery technique for preserving connectivity in the face of mote failure. Others ([44–47], etc.) have also studied the matter of connectivity and coverage in WSN in great detail.

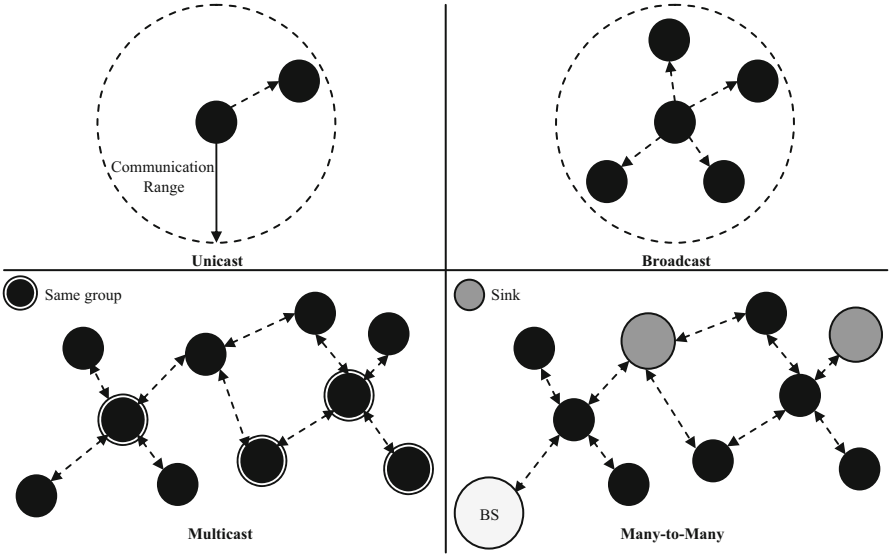

Fig. 4 WSN infrastructure-based data delivery models, unicast, broadcast, multicast, and many-to-many

2 Communication Patterns and Protocols in WSN

After the WSN configuration and organization algorithms have set up the WSN infrastructure for routing and communication. WSN motes can collaboratively perform the application-specific distributed tasks across the WSN. Over the lifetime of the WSN, human operated base stations, which are essentially gateways to fixed infrastructure, e.g. Internet backbone, are used to query and retrieve data from the WSN.

2.1 Communication Patterns

There are two different approaches in sampling and retrieving data from the WSN, namely, the push and pull approach. In a push approach, WSN motes are programmed to autonomously sample the environment at fixed intervals and *push* their data into the network. They push data towards data collectors like the base station or sink. On the other hand, in the pull approach, motes wait for an explicit command from the base station or sink to start sampling [48]. Both approaches have their own advantages and drawbacks. Obviously, push approach suffers from continuous sampling, bottleneck at sink or base station but enjoys low latency query response [48], whereas in the pull approach, there is an increase in the number of messages required for querying and there is a longer delay in query response, but conserves power by explicit sampling rather than continuous sampling [48].

WSN applications typically utilized the push approach to conserve energy, when radio communication was more expensive than sensing on a mote [48]; however, advances in radio technologies have significant improvements in the power consumption, resulting in ultra-low-power radios, e.g. ZigBee radio [48]. Bose and Helal [48] propose a hybrid push–pull approach to sampling and retrieving data from a WSN to leverage the benefits of both techniques. The push, pull, or hybrid approach yields three distinct network traffic patterns in WSN. The push approach causes a many-to-one communication pattern, where motes are pushing data towards the sink or base station. In the pull approach, before motes-to-sink communication occurs, the sink or base station sends the explicit command to start sensing, which is one-to-many communication. When motes are collaboratively communicating to self-configure, localize, or for data fusion at multiple sinks, many-to-many communication occurs [49].

Apart from these traffic patterns in WSN, Tilak et al. [50] decompose communication paradigms in WSN into two categories, that is, application-based and infrastructure-based communication. Application-based communication refers to getting the sensed data from nodes to application user, whereas infrastructure-based protocols are those that are used to provide the underlying primitives to achieve the application-based communication.

We will discuss data delivery models from the application and infrastructure-based communication perspectives, followed by a discussion of the network architecture in WSN.

2.1.1 Data Delivery Models

From the aspect of infrastructure-based communication protocols the flow of data packets is based on four distinct data delivery models: unicast, broadcast, multicast [50], or many-to-many. Unicast is one-to-one communication between two motes, versus, the one-to-many (i.e., all motes in the communication range of a mote) communication of broadcast. The motes within the communication range of a mote are often referred to as one-hop neighbor or simply neighbors. In multi-hop communication, data is routed between motes, using unicast communication.

Multicast pertains to communication of motes in a group. Motes are categorized into groups by the application and only process those packets that contain a group identification number that matches the motes group identification number. These WSN groups can form a connected or disconnected sub-network. In a disconnected group, multi-hop communication would be used to enable multicast data delivery models. If there is a geographic overlay on the multicast WSN and packets are transmitted to motes based on their specific geographical location [51], then this specialized form of multicast data delivery is known as Geocasting. Multicast data delivery model can also be specialized to behave in a many-to-one data delivery manner.

Finally, many-to-many data delivery models result in the presence of multiple sinks or gateways accessing the WSN for data and information [49]. These data delivery models in WSN are illustrated in Fig. 4.

From the perspective of the application-based communication protocols, the data delivery models consist of continuous, event-driven, query-driven, or a hybrid approach of these [50, 52], as illustrated in Fig. 5. In continuous, also known as periodic, data delivery models, data is transmitted from the motes to the sink or gateways at periodic intervals. In event-driven models, the motes are sensitive to one or more physical factors of the environment, when the sensed value (sensor readings) meets or exceeds a predetermined threshold, an event is triggered. The triggered motes propagate their sensor readings back to the sink(s) or gateways. In query-driven data delivery models, user initiates a query and network motes which meet the query criteria transmit their sensor readings. In the hybrid approach, one or more of the data delivery models are used together. For example, in a volcano monitoring application, the motes would be primarily event-driven. However, occasionally the user may want to poll or query the WSN for current seismic or temperature readings.

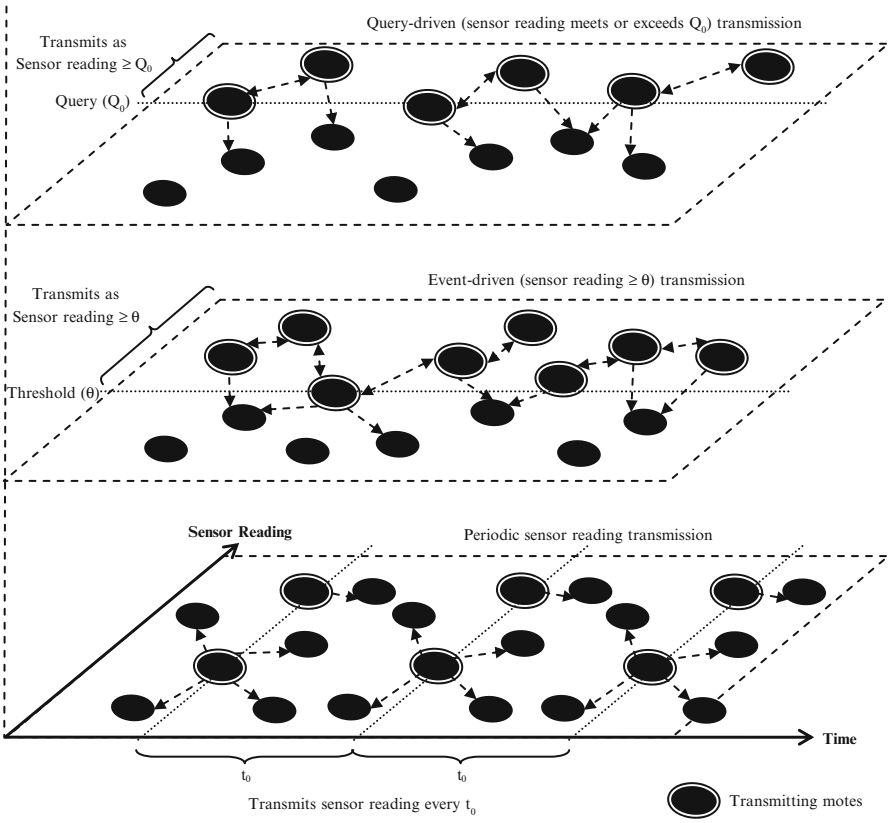

Fig. 5 WSN application-based data delivery models in WSN, periodic, event-driven, and query-driven

2.2 Network Architecture of WSNs

The inherent characteristics of the motes in a WSN, such as limited processing, memory and communication capabilities, impose design and performance criteria on the routing and communication protocols in WSN. Communication protocols are used to dictate, manage, and control all aspects of communication, from the lowest layer of accessing the physical medium to higher layers responsible for end-to-end transmission of packets and routing of data, to the top most application layer for data and packet formation. Figure 6 depicts an overview of the network architecture in WSN.

This is essentially different from typical network models due to the need for power management across all layers of the model. Furthermore, due to the constrained nature of motes and the application-specific nature of WSN, there is no clear demarcation in the network architecture layers, as in traditional network

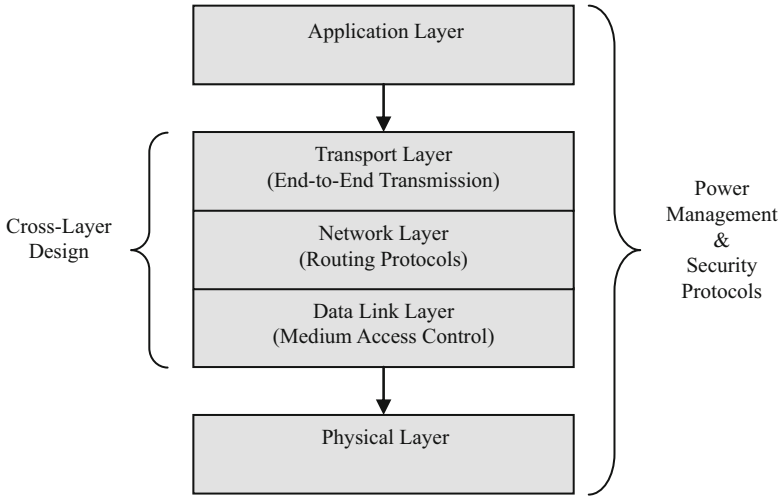

Fig. 6 Network architecture in WSN

architecture. Often, fundamental WSN communication protocols for querying, routing, and data delivery are integrated under data dissemination and aggregation techniques in WSN [53]. These limitations instigate the use of a cross-layer approach in developing network protocols for WSNs.

In the following sections, we will discuss the various concepts of the transport, network, and data link layers with respect to WSNs.

2.2.1 Transport Layer Protocols

Transport layer protocols are responsible for end-to-end transmission of data packets. These protocols include provisions for reliable or unreliable data delivery and for centralized or distributed congestion control. It is imperative that these protocols be designed carefully, since they can quickly overwhelm the constrained WSN. As WSN applications mature, it is important to design and implement efficient protocols for the network architecture to ensure reliability in a data-driven network. Primitives are built into transport layer protocols for reliable data delivery that prompt retransmission of packet(s) in the event of packet drop. This can be achieved by using packet sequence numbers, a series of acknowledgements to confirm delivery of packets, or time-out or round-trip intervals to access packet loss. These primitives allow senders and receivers to account for packets received and packets missed. Such primitives are crucial in WSN applications, which are essentially data-driven networks.

Furthermore, transport layer protocols are also responsible for congestion controlled. This is achieved by installing primitives that control transmission rate of motes, to ensure that the rate at which data packets are being transmitted does not

Table 2 Classification of some transport layer protocols for WSN

WSN transport layer protocols	Reliable	Unreliable	Congestion control		
			Distributed	Centralized	None
Flush	×		×		
STCP	×		×		
Hop	×		×		
RCRT	×			×	
Wisden	×				×
Tenet	×				×
RMST	×				×
WRCP		×	×		
IFRC		×	×		
Fusion		×	×		
CODA		×	×		
QCRA		×		×	
ESRT		×		×	
Surge (TinyOS)		×			×
CTP		×			×
RBC		×			×
CentRoute		×			×
Koala		×			×
XLP	×		×		
CRRT	×			×	
CTCP	×		×		
ERTP	×				×
GARUDA	×				×
DTSN	×				×
PHTCCP		×	×		

overwhelm the underlying network or create a bottleneck at the base station. Table 2 classifies some of the transport layer protocols developed for WSN with respect to these primitives [54–56].

2.2.2 Routing in Network Layer

As in Wireless Computing, routing protocols can be decomposed into reactive, proactive, cooperative [57], or hybrid protocols. Proactive routing protocols are also known as table-driven protocols since each node maintains a route table for all destinations at all times. Reactive protocols are on-demand routing protocols, which only discover routes to a destination when it is required, outdated, or invalid. The obvious trade-offs include data delivery latency, routing protocol traffic overhead, and storage requirements. Data delivery latency is low in networks using proactive routing protocols, since path information is readily available. However, there is

overhead incurred by the motes in the WSN, to discover and maintain paths to all destinations, even those that may never be used in the WSN. Routing protocol traffic overhead is minimized in reactive networks, when there is light network traffic and little changes to the network topology. Moreover, storage requirements of proactive routing protocols increase proportionally to the size of the network. In cooperative routing protocols, data is sent to a central entity that can further process the data and disseminate it accordingly [57].

Routing protocols can also be distinguished based on the communication entities involved in the routing protocol. In this perspective, there is node-centric and data-centric routing. In node-centric routing, data is routed between nodes, based on the node addresses. However, in data-centric routing, attribute based addressing is used, where nodes query for an attribute and only those nodes respond that can satisfy the query. For example, a mote can query for temperature greater than 72 °C, and those motes that have temperature readings more than 72 °C will respond.

Routing protocols are also designed for specific WSN topologies, classified as flat and hierarchical topologies, as illustrated in Fig. 2. On top of these WSN topologies, routing can also be geographical in nature, where packets are routed to motes closest to destination.

WSN routing protocols can also be based on different application defined criteria or operation-based attributes [57]. These operation-based attributes can include multipath, negotiation-based, query-based, QoS parameters (reliability, data integrity, energy efficiency), coherent [57] based routing techniques to increase reliability, security, etc., attributes.

Figure 7 depicts the various characteristics of WSN routing protocols, which can be integrated to design an efficient application-specific routing protocol for a WSN.

Multipath features in routing protocols can have a twofold benefit for WSN. Multipath routing can reduce frequency of updating routes, balance traffic load, and increase rate of data delivery [58]. This consequentially increases the lifetime and reliability of WSN. Earlier researchers proposed to use suboptimal routing paths with low energy consumption [59] or select a routing path based on the residual energy of the motes on the path [60]. The authors in [61] study the trade-off between traffic overhead and reliability using multipath routing in WSN. More recently, [58] uses multipath routing to increase lifetime of WSN and reduce collisions for transmission. Furthermore, researchers are utilizing ant colonization optimization, a swarm intelligence optimization technique, due to their proven success [62] in routing protocols for WSN. In [63], ant colonization is used to discover optimal routes to increase network lifetime and reliability, whereas [62] uses multipath routing to reduce congestion at the mote and link level. Authors in [64] survey swarm intelligence based routing protocols in WSN.

Routing protocols that are built on the pull approach, where queries can be propagated, are known as query-based routing protocols. Routing algorithms such as directed diffusion and rumor routing [57] are specifically implemented for query-driven data delivery models, where motes that meet the query requirements set up interest gradients or paths along which the data is routed to the sink.

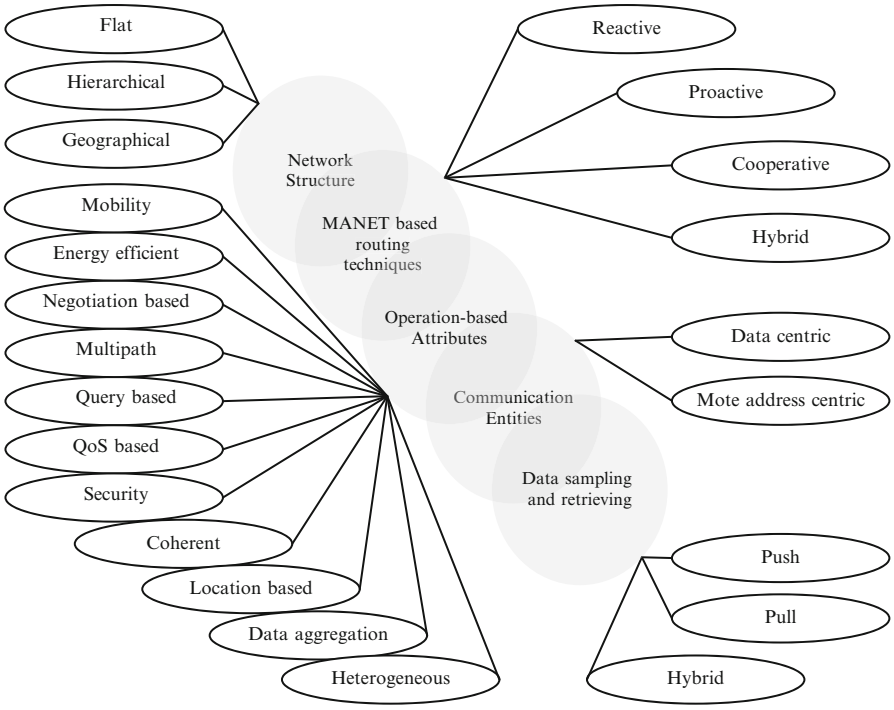

Fig. 7 WSN routing protocol characteristics

Negotiation-based routing algorithms suppress redundant data and use negotiation messages to ensure non-redundant data before transmission, e.g. Sensor Protocols for Information via Negotiation (SPIN) [65]. QoS-based routing ([66–69], etc.) ensures that routing in WSN meets application defined QoS metrics, of delay, energy, bandwidth, reliability, fault-tolerance, data integrity, etc. Data aggregation is also an important aspect of WSN and their applications, and is often accounted for when designing and developing routing protocols, to ensure efficient performance.

Not only is it important to secure the data and the motes in the WSN, but also the network traffic patterns. Since malicious motes can redirect traffic, inject false data or drop packets and cause havoc in the WSN, typical uses of asymmetric key cryptography or complicated symmetric key cryptography are infeasible for resource constraint WSN [70]. While there are techniques for implementing security on the data link layer in WSN, researchers are designing novel techniques for securing routing protocols [71–73], to increase reliability of WSN.

Table 3 classifies some WSN routing protocols based on the routing protocol characteristics illustrated in Fig. 7 [52, 57, 74–76, 71, 77–79].

Table 3 Classification of some WSN routing protocols

WSN routing protocol	Network structure	MANET technique	Operation-based attribute	Communication entities	Data sampling and retrieving
SPIN	Flat	Proactive	Data aggregation, negotiation-based, query-based, multipath	Data-centric	Pull
Directed diffusion	Flat	Reactive	Data aggregation, negotiation-based, query-based, multipath, localization, coherent	Data-centric	Pull
Rumor routing	Flat	Reactive	Query-based	Data-centric	Hybrid
TEEN and APTEEN	Hierarchical	Reactive	Data aggregation, localization, energy efficient	Data-centric	Push
LEACH	Hierarchical	Proactive	Data aggregation, localization	Mote address-centric	Push
ACQUIRE	Flat	Reactive	Data aggregation, query-based	Data-centric	–
COUGAR	Flat	Reactive	Data aggregation, query-based	Data-centric	Hybrid
PEGASIS	Hierarchical	Proactive	Data aggregation, localization, energy efficient	Mote address-centric	Push
GAF	Geographical	Proactive	Energy efficient	Mote address-centric	–
SEAD	Geographical	Proactive	Mobility, energy efficient	Mote address-centric	–
SPEED (SNFG)	Geographical	Proactive	QoS, energy efficient	Mote address-centric	–
CHR	Geographical	Proactive	Data aggregation, heterogeneous	Mote address-centric	–
APR	Geographical	Proactive	Security	Mote address-centric	–

2.2.3 Medium Access Control (MAC) Protocols at Data Link Layer

The MAC protocols at the data link layer manage how motes access the shared wireless medium and the backoff approach they employ in the event of a collision. In typical Wireless Networks, the radio channel is decomposed into multiple channels using techniques like time division multiple access (TDMA), frequency division multiple access (FDMA), or code division multiple access (CDMA) to allow collision-free transmission of multiple message, simultaneously. Traditional WSN motes used a simple single channel radio to conserve energy, which is required to maintain the complex multiple channel radios [80]. However, recent advances in research have given rise to multi-channel radios for WSN motes.

WSN MAC protocols can be broadly classified into schedule based or contention-based medium access protocols. Schedule or reservation based MAC protocols devise an assignment that motes can follow for accessing the physical medium to avoid collision. Contention-based MAC protocols do not require an assignment, they define a backoff algorithm for motes to follow, when there is a collision in accessing the physical medium. Schedule based protocols can be further decomposed into synchronous, asynchronous, and hybrid protocols.

Since the MAC protocol directly controls the radio on a mote, an important technique used by motes is called duty cycling, where active motes periodically turn their radio off and go to sleep [81]. Synchronous schedule based MAC protocols require time synchronization and topology information to ensure that neighboring motes are active at the same time to communicate. However, asynchronous protocols do not require such synchronization or topology information and instigate communication between motes in different active cycles.

Contention-based MAC protocols eliminate the overhead of synchronization or network topology from schedule based MAC protocols. Motes can transmit packets immediately and retransmit in the event of a collision. A refined approach is to sense the physical medium for transmission, and when the medium is idle then access the medium and transmit the packet. This is the general concept in carrier sense multiple access (CSMA), which is one of the canonical contention-based MAC protocols.

Ye et al. [82] have identified four significant aspects of wasteful energy consumption, namely, collision, overhearing, control packet overhead, and idle listening at the data link layer. When two or more motes transmit at the same time, their respective packets are corrupted and require retransmissions increasing energy consumption and latency. Energy consumed in overhearing is energy spent listening for packets destined for other motes. Sending and receiving control packets without useful data is also wasteful with respect to energy consumed. Listening to the channel even when there are no radio transmissions is considered idle listening, which is an energy consumption overhead [81].

Contention-based protocols include techniques like CSMA and offer multiple benefits like low implementation complexity, flexibility in face of mobility, and varying traffic patterns [80]. However, schedule based MAC protocols like TDMA have intrinsic energy conserving features since there are no collisions, overhearing,

Table 4 Classification of some WSN MAC protocols

WSN MAC protocol	Organization	Channel
SMACS	Frames	FDMA
T-MAC	Slots	Single
TRAMA	Frames	Single
BuzzBuzz	Random	Single
DW-MAC	Slots	Single
X-MAC	Random	Single
B-MAC	Random	Single
WiseMAC	Random	Single
PW-MAC	Hybrid	Single
S-MAC	Slots	Single
PicoRadio	Random	CDMA

and idle listening. Due to the energy conservation necessary in resource constraint WSN, it is important for MAC protocols for WSN to incorporate primitives for conserving energy by reducing collisions, overhearing, reduced control packet overhead, and idle listening, without the need for expensive time synchronization.

These MAC protocols require different degrees of collaboration and cooperation amongst the motes to achieve medium control access. Motes in the network could work completely independently in a random manner like those in contention-based MAC protocols, to slightly organized slotted MAC protocols and then to the very organized synchronous schedule based protocols like TDMA [80].

Various MAC protocols have been designed and implemented including synchronous duty cycling MAC protocols such as S-MAC [82], T-MAC, TRAMA, SCP, and DW-MAC. and asynchronous duty cycling MAC protocols such as X-MAC [83], B-MAC, WiseMAC, PW-MAC [81], and ASCEMAC [84]. An exhaustive MAC protocols survey is presented in [85]. MAC protocols are also being designed to include important features like security [86], power management [87], and QoS [88]. Table 4 delineates some of these protocols and their classification [80, 89].

3 WSN Applications and Problem Space

Numerous intrinsic WSN characteristics distinguish it from its peers in other infrastructure-less communication systems [3] and distributed computing environments [90], where their nodes are more powerful (w.r.t. processor and radio) and reliable than WSN motes. Figure 8 [90] illustrates the difference in the problem space of WSN with other computing paradigms, with respect to functionality of the network nodes, size of the network, and spatial-temporal awareness of the nodes in the network.

WSN are distinguishable from real-time embedded systems primarily due to the lack of spatial awareness in embedded system nodes. Real-time embedded

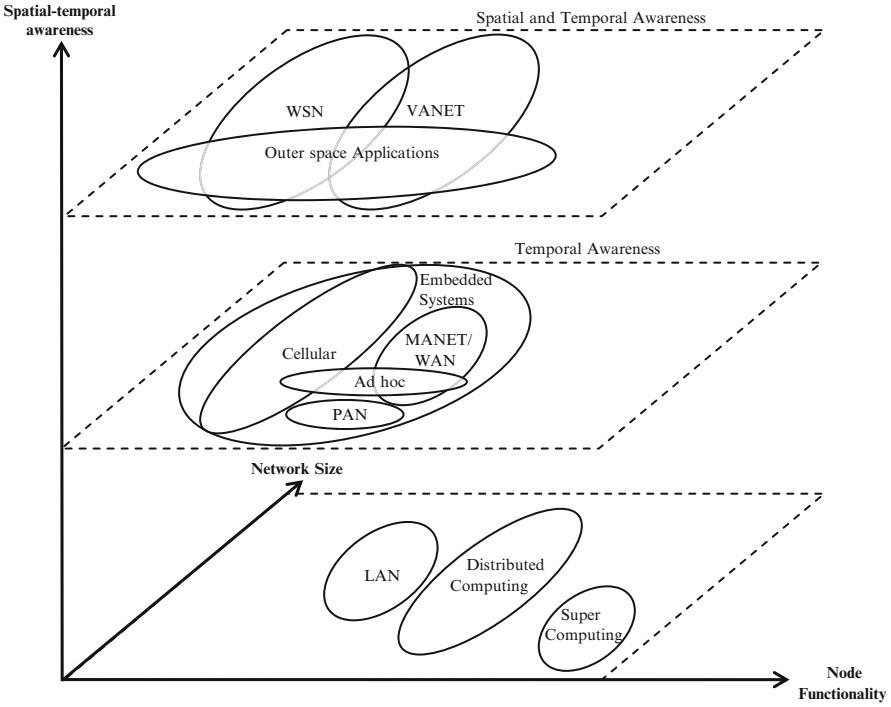

Fig. 8 Comparison of problem space of WSN with other computing paradigms

systems can include various other computing paradigms, especially, infrastructure-less mobile environments like Bluetooth personal area networks (PAN), MANET, cellular networks, and ad hoc networks.

WSN are significantly different from traditional distributed and supercomputing paradigms. Distributed computing includes paradigms such as peer-to-peer (P2P) networks, grid and high-performance computing. Apart from the lack of spatial and temporal awareness of the network nodes, another fundamental difference lies in the functionality of the network nodes. Typically, distributed and supercomputing paradigms consist of high-performance powerful nodes, as illustrated in Fig. 8.

WSN applications are sensitive to spatial and temporal parameters and can be classified into spatially and temporally related paradigms as follows. Firstly, in the spatial domain, WSN applications can be broadly classified into local and global WSN applications based on the size and coverage of the region of interest of the application. Similarly, WSN applications can also be temporally categorized into continuous, event-driven, query-driven, or hybrid applications (cf. Sect. 2.1.1, Fig. 5). This imposes a constraint on when the motes transmit data, either motes transmit data at periodic intervals, or transmit only the data that exceeds application defined triggers or threshold levels, or only those motes transmit data that meets

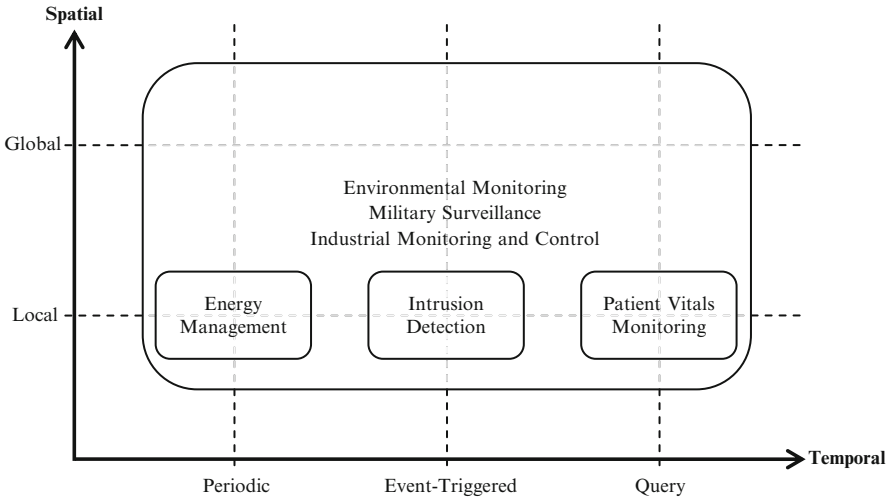

Fig. 9 Classification of WSN applications

a query criterion or a hybrid approach of these techniques. Figure 9 [49] broadly classifies some WSN applications into these domains.

WSN application operations cannot be discretely categorized into the different temporal domains, since these networks are fundamentally data-driven networks. This implies events could be triggered in an otherwise periodic sampling WSN application like environmental monitoring. Furthermore, periodic sampling WSN applications like patient vitals health monitoring could quickly become query-based, if the WSN for patient vitals monitoring is polled for specific vital reading. Therefore, WSN applications are usually designed to operate in multiple temporal domains. Figure 9 illustrates these characteristics of WSN applications and delineates some WSN application examples that accurately fit the domains.

Figure 10 [49] delineates the various design characteristics that form the intrinsic nature and behavior of WSN applications and their motes. This directly influences the design for the components of the network architecture of the WSN. The figure also summarizes the various WSN characteristics we have discussed in this chapter.

4 Programming WSNs

WSNs can only be fully utilized, if WSN programmers have adequate software platforms [49] for efficient and reliable programming of large number of motes. However, due to the lack of programming, debugging, and development environment platforms, WSN programmers are forced to implement application-logic intertwined with low-level WSN implementation issues [49, 90]. This makes the

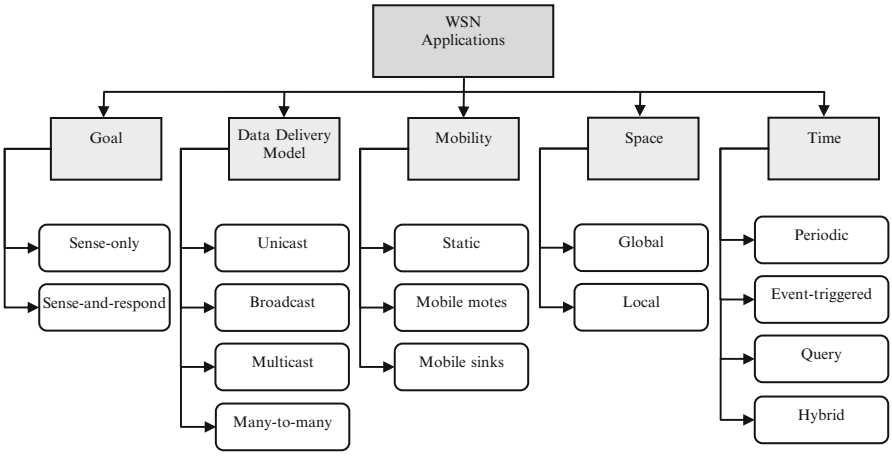

Fig. 10 WSN application design criteria

application code complex, harder to debug and non-scalable for complex application development due to the interdependencies between performance and function [90]. Furthermore, this sort of programming falls out of the expertise of WSN application domain experts [49]. Mottola and Picco [49] discuss various other design criteria and components of WSN programming languages and their affect on the design and development of WSN applications.

Myriad WSN motes exist that differ in functionality as delineated in Table 1. Manufacturers often ship an operating system with the motes to ease in the handling of hardware and software components. Most WSN operating systems (OS) are either event-driven or multithreaded [91]. Multithreaded OS are similar to traditional OS, therefore easier for programmers to manage, however, infeasible for resource constrained WSN. Therefore, lightweight multithreaded OS have been designed and implemented for WSN motes. Event-driven OS do not tax the resources of the WSN motes but pose a learning curve for programmers of WSN applications. Farooq and Kunz [91] survey and classify operating systems such as TinyOS, Contiki, and MANTIS, in WSN.

In the following section, we will discuss TinyOS, an operating system for WSN, and consider a component-based approach to programming applications for WSNs using TinyOS.

4.1 TinyOS: An Operating System for WSN Motes

Operating systems (OS) are programs that coordinate all other programs, such as process scheduling and memory access. in a computer. The typical size of operating systems for embedded devices ranges in the order of megabytes; however, TinyOS

is a much smaller operating system for WSN motes, approximately 400 bytes [92]. It is implemented in nesC [93], a variant of C. TinyOS is different from traditional OS, since it provides a framework and a set of component for programming motes to *build* application-specific operating system for a WSN application. Therefore, there is an operating system for each application. Typical TinyOS applications can range from approximately 15 Kbytes to more complex applications in the range of 64 Kbytes, of these the core operating system is about 400 bytes [92]. The networking architecture is encapsulated in Active Message interfaces, which are also implemented in nesC and offer fundamental communication primitives to TinyOS applications [49, 92]. Surge is the transport layer protocol implemented in TinyOS, which is unreliable and offers no primitives for congestion control [54]. Berkley MAC (B-MAC) is the asynchronous schedule based MAC protocol implemented in TinyOS [94], with interfaces available to change or adapt the MAC protocol.

TinyOS is a component-based programming model, where components are software abstractions of hardware components [92]. Therefore, a TinyOS application, which is really an application-specific operating system, is built on top of these reusable components [92], wiring them together to provide or use services. Components use interfaces to encapsulate a set of services [92]. For example, turning an LED on is a service that can be modeled by using the LED interface. Components use commands and events for inter-component communication. Commands to a component are requests to start a service, whereas events are signaled to mark the completion of the service at the component [92].

Programming in nesC consists of a configuration and module file. The configuration file connects components together, whereas the module file contains the implementation, by *providing* or *using* interfaces. Figures 11 and 12 illustrate a "hello world" TinyOS application, called Blink [95], where the red LEDS on a mote are toggled every 1,000 ms.

The `Blink.nc` configuration file shows how the components are wired together. A TinyOS application starts by executing the `init` command of the `StdControl` interface of the `Main` component. `Blink` configuration file wires the `Main` component's `StdControl` to `SingleTimer` and `Blink` component `StdControl`. This binds the implementation of `Main.StdControl` to `SingleTimer.StdControl` and `BlinkM.StdControl`. The `BlinkM.StdControl` interface is implemented in the module file, `BlinkM.nc`. Similarly, the `Timer` and `Leds` interfaces used in `Blink` actually invoke the

```
configuration Blink {}

implementation {
  components Main, BlinkM, SingleTimer, LedsC;
  Main.StdControl -> SingleTimer.StdControl;
  Main.StdControl -> BlinkM.StdControl;
  BlinkM.Timer -> SingleTimer.Timer;
  BlinkM.Leds -> LedsC;
}
```

Fig. 11 Blink.nc configuration file

```
module BlinkM {
  provides {
    interface StdControl;
  }
  uses {
    interface Timer;
    interface Leds;
  }
}
implementation {

  command result_t StdControl.init() {
    call Leds.init();
    return SUCCESS;
  }

  command result_t StdControl.start() {
    return call Timer.start(TIMER_REPEAT, 1000);
  }

  command result_t StdControl.stop() {
    return call Timer.stop();
  }

  event result_t Timer.fired()
  {
    call Leds.redToggle();
    return SUCCESS;
  }
}
```

Fig. 12 BlinkM.nc module file

`Timer` interface provided by component `SingleTimer`, and `Leds` interface of component `LedsC`, respectively. Note, `SingleTimer` and `Leds` provide software abstraction of the hardware components for the clock and LEDS.

The configuration file delineates how control of execution is passed amongst components. The module file (`BlinkM.nc`) provides the application-specific implementation by providing and using interfaces. `BlinkM.StdControl.init()` initializes the application and `BlinkM.StdControl.start()` starts the component. When `Blink` starts, it sets the timer to repeat every 1,000 ms. In the event-driven operating system, since, there is no other code to execute, `Blink` waits until event `Timer.fired()` is triggered. When the event is triggered, `Blink` toggles the red `LEDS`.

The application is compiled for the target platform, any TinyOS compatible device, e.g. micaz motes, and flushed on the mote hardware using a programming board, which interfaces a mote with the development environment on a computer or laptop.

5 Summary

WSNs typically consist of large number of heterogeneous motes organized in a hierarchical manner, to monitor and/or control a region of interest, based on ambient environment factors. Generally, the number of motes deployed in the region increases and the resources decrease as you move down the hierarchy. The mobile, infrastructure-less WSN are able to truly achieve untethered communications in dangerous or mundane regions for autonomous data collection, analysis, and response via actuators. However, the greatest resource limitation of WSN is the power source, which must be optimally used or alternatively harnessed to ensure network lifetime.

There is a lack of standards for traditional network concepts, like architecture, topology, routing, and security, since WSN are application-specific in nature. Due to the different types of motes and WSNs, there are interoperability issues amongst the heterogeneous hardware units [96]. Further limitations of WSN are attributed to the lack of software for programming and debugging motes and WSN [49, 90].

Energy conservation and power management in the radio and network communications are the underlying design criteria for protocols and primitives in WSN. However, Bose and Helal [48] show that advances in technology have allowed low-power radios, like the Atmel Zlink RCB [48] for effective use in WSN and now attention needs to be on the sensor sampling technique, which consumes significantly more energy than network and application costs. Anastasi et al. [94] present a survey of energy conservation techniques in WSN.

Furthermore, apart from energy conservation, security challenges in WSN are a make it or break it criteria in the widespread use of WSN in all envisioned domains. Boyle and Newe [97] compare some security protocols in use today in WSN, like SPIN [98], consisting of SNEP (Secure Network Encryption Protocol) and μTESLA (micro version of Timed Efficient Stream Loss-tolerant Authentication), LEAP [99], and TinySec [100], which aim at building security features into network protocols.

As recent routing protocols have gained from swarm intelligence, Kulkarni et al. [101] discuss the use of other computational intelligence paradigms like neural networks, fuzzy logic, evolutionary algorithms, reinforcement learning, and artificial immune system in designing and implementing effective and efficient protocols and primitives for WSN operations like routing, deployment, localization, security, scheduling, data aggregation, and QoS management.

To summarize, in this chapter we considered the characteristics and features of WSNs and their motes. Furthermore, we reviewed the fundamental concepts in WSNs pertaining to architecture, topology, data delivery models, transport, routing, and data link layer protocols. We also presented a classification of some of the protocols for transport, routing, and data link layers. We concluded with a discussion of TinyOS, an operating system for WSN motes, and exemplified programming WSN motes in nesC.

References

1. History of Innovation (1995–2012) [Online]. Available: http://www.ti.com/corp/docs/company/history/sensortimelinelowbandwidth.shtml. Accessed 6 Jan 2012
2. Weinberg H (2012) How they work: MEMS (micro electro-mechanical systems) technology. Sensorland [Online]. Available: http://www.sensorland.com/HowPage023.html. Accessed 6 Jan 2012
3. Akyildiz IF, Su W, Sankarasubramainiam Y, Cayirci E (2002) A survey on sensor networks. IEEE Commun Mag 40(8):102–114
4. Akyildiz IF, Kasimoglu IH (2004) Wireless sensor and actor networks: research challenges. Ad Hoc Netw 2(4):351–367
5. Xia F (2008) QoS challenges and opportunities in wireless sensor/actuator networks. Sensors 8(2):1099–1110
6. Akyildiz IF, Melodia T, Chowdury K (2007) A survey on wireless multimedia sensor networks. Comput Netw 51(4):921–960
7. Lamont L, Toulgoat M, Deziel M, Patterson G (2011) Tiered wireless sensor network architecture for military surveillance applications. In: International conference on sensor technologies and applications (SENSORCOMM), Nice, Saint Laurent du Var
8. Antepli MA, Gurbuz SZ, Uysal-Biyikoglu E (2010) Ferromagnetic target detection and localization with a wireless sensor network. In: Military communications conference (MILCOM), San Jose
9. Teng J, Snoussi H, Richard C (2010) Decentralized variational filtering for target tracking in binary sensor networks. IEEE Trans Mob Comput 9(10):1465–1477
10. Medagliani P, Ferrari G, Gay V, Leguay J (2013) Cross-layer design and analysis of WSN-based mobile target detection systems. Elsevier Ad Hoc Netw 11(2):712–732
11. Krishnamurty L, Adler R, Buonadonna P, Chhabra J, Flanigan M, Kushalnagar N, Nachman L, Yarvis M (2005) Design and deployment of industrial sensor networks: experiences from a semiconductor plant and the North Sea. In: 3rd international conference on embedded networked sensor systems (SenSys 05), San Diego
12. Sikka P, Corke P, Valencia P, Crossman C, Swain D, Bishop-Hurley G (2006) Wireless adhoc sensor and actuator networks on the farm. In: International conference on information processing in sensor networks, Nashville
13. Song W-Z, Huang R, Xu M, Ma A, Shirazi B, LaHusen R (2009) Air-dropped sensor network for real-time high-fidelity volcano monitoring. In: International conference on mobile system, applications and services (MobiSys 09), Krakow
14. Ceriotti M, Mottola L, Picco GP, Murphy AL, Corra M, Pozzi M, Zonta D, Zanon P (2009) Monitoring heritage buildings with wireless sensor networks: the Torre Aquila deployment. In: International conference on information processing in sensor networks (IPSN), San Francisco
15. Ko J, Lim JH, Muvaloiu-E R, Terzis A, Masson GM, Gao T, Destler W, Selavo L, Dutton RP (2010) MEDiSN: medical emergency detection in sensor networks. ACM Trans Embed Comput Syst (TECS) 10(1):1–29
16. Chen Y, Shen W, Huo H, Xu Y (2010) A smart gateway for health care system using wireless sensor network. In: International conference on sensor technologies and applications (SENSORCOMM 10), Venice
17. Ota N, Ahrens S, Redfern A, Wright P, Yang X (2006) An application-driven architecture for residential energy management with wireless sensor networks. In: IEEE international conference on mobile adhoc and sensor systems, Vancouver
18. Song H, Zhu S, Cao G (2008) SVATS: a sensor-network-based vehicle anti-theft system. In: IEEE conference on computer communications, Phoenix
19. Tubaishat M, Zhuang P, Qi Q, Shang Y (2009) Wireless sensor networks in intelligent transportation systems. Wirel Commun Mob Comput 9(3):287–302, Special Issue: Distributed Systems of Sensors and Actuators

20. Vladimirova T, Bridges CP, Paul JR, Malik SA, Sweeting MN (2010) Space-based wireless sensor networks: design issues. In: IEEE aerospace conference, Big Sky
21. Culler D, Hill J, Horton M, Pister K, Szewczyk R, Woo A (2002) MICA: the commercialization of microsensor motes. Sensors Magazine, 1 April 2002 [Online]. Available: http://www.sensorsmag.com/networking-communications/mica-the-commercialization-microsensor-motes-1070. Accessed 1 Feb 2012
22. Merrill W, Kaiser W (2004) Wireless integrated network sensors (WINS) next generation. Airforce Research Laboratory, Rome
23. Wireless Applications – Moteiv launches wireless mote with interface to mobile devices. Sensors, 11 May 2007 [Online]. Available: http://www.sensorsmag.com/wireless-applications/news/moteiv-launches-wireless-mote-with-interface-mobile-devices-2523. Accessed 7 Jan 2012
24. Hill J, Horton M, Kling R, Krishnamurthy L (2004) The platforms enabling wireless sensor networks. Commun ACM 47(6):41–46
25. Healy M, Newe T, Lewis E (2008) Wireless sensor node hardware: a review. In: IEEE sensors, Lecce
26. Yick J, Mukherjee B, Ghosal D (2008) Wireless sensor network survey. Comput Netw (Elsevier) 52(12):2292–2330
27. Sun Z, Akyildiz IF (2010) Magnetic induction communications for wireless underground sensor networks. IEEE Trans Antennas Propag 58(7):2426–2435
28. Bhuvaneswari P, Balakumar R, Vaidehi V, Balamuralidhar P (2009) Solar energy harvesting for wireless sensor networks. In: International conference on computational intelligence, communication systems and networks (CICSYN), Indore
29. Seah WK, Eu ZA, Tan H-P (2009) Wireless sensor networks powered by ambient energy harvesting (WSN-HEAP) – survey and challenges. In: International conference on wireless communication, vehicular technology, information theory and aerospace & electronic systems technology (Wireless VITAE), Aalborg
30. Bokareva T. Mini hardware survey [Online]. Available: http://www.cse.unsw.edu.au/~sensar/hardware/hardware_survey.html. Accessed 7 Jan 2012
31. Uchiyama T, Uehara Y, Mori M, Saito H, Tobe Y (2006) A system for analyzing life rhythm using wireless sensors on mules. In: International conference on mobile data management (MDM), Nara
32. Romer K, Mattern F (2004) The design space of wireless sensor networks. IEEE Wirel Commun 11(6):54–61
33. Naumowicz T, Freeman R, Kirk H, Dean B, Calsyn M, Liers A, Braendle A, Guilford T, Schiller J (2010) Wireless sensor network for habitat monitoring on Skomer island. In: IEEE conference on local computer networks (LCN), Denver
34. Fok C-L, Roman G-C, Lu C (2007) Towards a flexible global sensing infrastructure. ACM SIGBED Rev 4(3):1–6, Special Issue on the Workshop on Wireless Sensor Network Architecture
35. Polastre J, Szewczyk R, Mainwaring A, Culler D, Anderson J (2004) Analysis of wireless sensor networks for habitat monitoring. In: Wireless sensor networks. Kluwer Academic, Norwell, pp 399–423
36. Schoonmaker P, Luscombe W (2005) Habitat monitoring: an approach for reporting status and trends for state comprehensive wildlife conservation strategies. 22 March 2005 [Online]. Available: http://www.defenders.org/resources/publications/programs_and_policy/biodiversity_partners/habitat_monitoring.pdf. Accessed 8 Jan 2012
37. Agarwal R, Martinez-Catala R, Harte S, Segard C, O'Flynn B (2008) Modeling power in multi-functionality sensor network applications. In: International conference on sensor technologies and applications (SENSORCOMM), Cap Esterel
38. Dargie W (2012) Dynamic power management in wireless sensor networks: state-of-the-art. IEEE J Sensors 12(5):1518–1528
39. Sohrabi K, Gao J, Ailawadhi V, Pottie GJ (2000) Protocols for self-organization of a wireless sensor network. IEEE Pers Commun 7(5):16–27

40. Saglam O, Dalkili ME (2009) A self organizing multihop clustering protocol for wireless sensor networks. In: International conference on mobile ad-hoc and sensor networks (MSN), Fujian
41. Kacimi R, Dhaou R, Beylot A (2008) Energy-aware self-organization algorithms for wireless sensor networks. In: IEEE global telecommunications conference (GLOBECOM), New Orleans
42. Berman K, Annexstein F, Ranganathan A (2006) Dominating connectivity and reliability of heterogeneous sensor networks. In: IEEE international conference on pervasive computing and communications workshop (PerCom), Pisa
43. Abbasi A, Younis M, Baroudi U (2010) Restoring connectivity in wireless sensor-actor networks with minimal topology changes. In: International conference on communications (ICC), Cape Town
44. Ghosh A, Das SK (2008) Coverage and connectivity issues in wireless sensor networks: a survey. Pervasive Mob Comput 4(3):303–334
45. Wang P, Sun Z, Vuran M, Al-Rodhaan MA, Al-Dhelaan A, Akyildiz IF (2011) On network connectivity of wireless sensor networks for sandstorm monitoring. Comput Netw 55(5):1150–1157
46. Lu M, Wu J, Cardei M, Li M (2009) Energy-efficient connected coverage of discrete targets in wireless sensor networks. Int J Ad Hoc Ubiquitous Comput 4(3–4):137–147
47. Wang X, Xing G, Zhang Y, Lu C, Pless R, Gill C (2003) Integrated coverage and connectivity configuration in wireless sensor networks. In: International conference on embedded networked sensor systems (SenSys), Los Angeles
48. Bose R, Helal A (2010) Sensor-aware adaptive push–pull query processing in wireless sensor networks. In: International conference on intelligent environments (IE), Kuala Lumpur
49. Mottola L, Picco GP (2011) Programming wireless sensor networks: fundamental concepts and state of the art. ACM Comput Surv (CSUR) 43(3):1–51
50. Tilak S, Abu-Ghazaleh NB, Heinzelman W (2002) A taxonomy of wireless micro-sensor network models. ACM SIGMOBILE Mob Comput Commun Rev 6(2):28–36
51. Villalba L, Orozco A, Cabrera A, Abbas C (2009) Routing protocols in wireless sensor networks. Sensors 9(11):8399–8421
52. Akkaya K, Younis M (2005) A survey on routing protocols for wireless sensor networks. Elsevier Ad Hoc Netw 3(3):325–349
53. Niculescu D (2005) Communication paradigms for sensor networks. Commun Mag 43(3):116–122
54. Paek J, Govindan R (2007) RCRT: rate-controlled reliable transport for wireless sensor networks. In: Proceedings of the 5th international conference on embedded networked sensor systems (SenSys'07), Sydney
55. Rathnayaka AJD, Potdar VM (2013) Wireless sensor network transport protocol: a critical review. J Netw Comput Appl 36(1):134–146
56. Vuran M, Akyildiz I (2010) XLP: a cross-layer protocol for efficient communication in wireless sensor networks. IEEE Trans Mob Comput 9(11):1578–1591
57. Al-Karaki JN, Kamal AE (2004) Routing techniques in wireless sensor networks: a survey. IEEE Wirel Commun 11(6):6–28
58. Wang Z, Bulut E, Szymanski BK (2009) Energy efficient collision aware multipath routing for wireless sensor networks. In: IEEE international conference on communications, Dresden
59. Shah RC, Rabaey JM (2002) Energy aware routing for low energy ad hoc sensor networks. In: IEEE wireless communications and networking conference, Orlando
60. Chang JH, Tassiulas L (2004) Maximum lifetime routing in wireless sensor networks. IEEE/ACM Trans Netw 12(4):609–619
61. Dulman S, Nieberg T, Wu J, Havinga P (2003) Trade-off between traffic overhead and reliability in multipath routing for wireless sensor networks. In: IEEE wireless communications and networking, New Orleans
62. Saleem K, Fisal N, Hafizah S, Kamilah S, Rashid R (2009) A self-optimized multipath routing protocol for wireless sensor networks. Int J Recent Trends Eng (IJRTE) 2(1):93–97

63. Okdem S, Karaboga D (2009) Routing in wireless sensor networks using an ant colony optimization (ACO) router chip. Sensors 9(2):909–921
64. Saleem M, Di Caro GA, Farooq M (2011) Swarm intelligence based routing protocol for wireless sensor networks: survey and future directions. Inform Sci 181(20):4597–4624
65. Kulik J, Heinzelman W, Balakrishnan H (2002) Negotiation-based protocols for disseminating information in wireless sensor networks. Wirel Netw 8(2/3):169–185
66. Huang X, Fang Y (2008) Multiconstrained QoS multipath routing in wireless sensor networks. Wirel Netw 14(4):465–478
67. Kusy B, Lee HJ, Wicke M, Milosavljevic N, Guibas L (2009) Predictive QoS routing to mobile sinks in wireless sensor networks. In: International conference on information processing in sensor networks (IPSN), San Francisco
68. Ben-Othman J, Yahya B (2010) Energy efficient and QoS based routing protocol for wireless sensor networks. Elsevier J Parallel Distrib Comput 70(8):849–857
69. Chen Y, Nasser N, El Salti T, Zhang H (2010) A multipath QoS routing protocol in wireless sensor networks. Int J Sensor Netw 7(4):207–216
70. Zhou Y, Fang Y, Zhang Y (2008) Securing wireless sensor networks: a survey. IEEE Commun Surv Tutorials 10(3):6–28
71. Sheu J-P, Jiang J-R, Tu C (2008) Anonymous path routing in wireless sensor networks. In: IEEE international conference on communications (ICC), Beijing
72. Zhang Z, Jiang C, Deng J (2010) A secure anonymous path routing protocol for wireless sensor networks. In: IEEE international conference on wireless communications, networking and information security (WCNIS), Beijing
73. Zahariadis T, Leligou H, Voliotis S, Maniatis S, Trakadas P, Karkazis P (2009) Energy-aware secure routing for large wireless sensor networks. World Sci Eng Acad Soc Trans Commun 8(9):981–991
74. Singh S, Singh M, Singh D (2010) Routing protocols in wireless sensor networks—a survey. Int J Comput Sci Eng Surv (IJCSES) 1(2):63–83
75. Koliousis A, Sventek J (2007) Proactive vs reactive routing for wireless sensor networks. Department of Computing Science, University of Glasgow, Glasgow
76. Du X, Lin F (2005) Improving routing in sensor networks with heterogeneous sensor nodes. In: Vehicular technology conference, Stockholm
77. Kim H, Abdelzaher T, Kwon W (2003) Minimum-energy asynchronous dissemination to mobile sinks in wireless sensor networks. In: 1st international conference on embedded networked sensor systems (SenSys'03), Los Angeles
78. Yao Y, Gehrke J (2002) The cougar approach to in-network query processing in sensor networks. ACM SIGMOD Rec 31(3):9–18
79. Lindsey S, Raghavendra C (2002) PEGASIS: power-efficient gathering in sensor information systems. In: IEEE aerospace conference, Big Sky
80. Langendoen K, Halkes G (2005) Energy efficient medium access control. In: Embedded systems handbook, CRC, pp 34.1–34.29
81. Tang L, Yanjun S, Gurewitz O, Johnson DB (2011) PW-MAC: an energy-efficient predictive-wakeup MAC protocol for wireless sensor networks. In: IEEE INFOCOM, Shanghai
82. Ye W, Heidemann J, Estrin D (2002) An energy-efficient MAC protocol for wireless sensor networks. In: INFOCOM 2002, 21st annual joint conference of IEEE computer and communications societies, New York
83. Buettner M, Yee GV, Anderson E, Han R (2006) X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In: Proceedings of 4th international conference on embedded networked sensor systems (SenSys), Boulder
84. Ren Q, Liang Q (2005) An energy-efficient MAC protocol for wireless sensor networks. In: IEEE Globecom, St. Louis
85. Bachir A, Dohler M, Watteyne T, Leung K (2010) MAC essentials for wireless sensor networks. IEEE Commun Surv Tutorials 12(2):222–248
86. Rao P, Rani R, Sankar S, Pradeep K, Kumar N, Kumar V (2010) Secure MAC for wireless sensor networks through RBFNN. Int J Eng Sci Technol 2(8):3510–3514

87. Hu Q, Tang Z (2011) ATPM: an energy efficient MAC protocol with adaptive transmit power scheme for wireless sensor networks. J Multimedia 6(2):122–128
88. Yigitel M, Incel O, Ersoy C (2011) QoS-aware MAC protocols for wireless sensor networks: a survey. Elsevier Comput Netw 8(1):1982–2004
89. Langendoen K (2011) The MAC alphabet soup served in wireless sensor networks – taxonomy. Delft University of Technology, 16 March 2011 [Online]. Available: http://www.st.ewi.tudelft.nl/~koen/MACsoup/taxonomy.php. Accessed 12 July 2012
90. Sugihara R, Gupta RK (2008) Programming models for sensor networks: a survey. ACM Trans Sensor Netw (TOSN) 4(2):1–29
91. Farooq M, Kunz T (2011) Operating systems for wireless sensor networks: a survey. Sensors 11(6):5900–5930
92. Levis P, Madden S, Polastre J, Szewczyk R, Whitehouse K, Woo A, Gay D, Hill J, Welsh M, Brewer E, Culler D (2005) TinyOS: an operating system for sensor networks. In: Ambient intelligence. Springer, New York, pp 115–148
93. Gay D, Levis P, von Behren R, Welsh M, Brewer E, Culler D (2003) The nesC language: a holistic approach to networked embedded systems. In: Proceedings of programming language design and implementation (PLDI), San Diego
94. Anastasi G, Conti M, Di Francesco M, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Elsevier Ad Hoc Netw 7(3):537–568
95. TinyOS Tutorial Lesson 1: getting started with TinyOS and nesC. TinyOS, 9 September 2003 [Online]. Available: http://www.tinyos.net/tinyos-1.x/doc/tutorial/lesson1.html. Accessed 8 Jan 2012
96. Thusu R (2010) Wireless sensor use is expanding in industrial applications. Sensors, 1 June 2010 [Online]. Available: http://www.sensorsmag.com/networking-communications/wireless-sensor/wireless-sensor-use-is-expanding-industrial-applications-7212. Accessed 8 Jan 2012
97. Boyle D, Newe T (2007) Security protocols for use with wireless sensor networks: a survey of security architectures. In: International conference on wireless and mobile communications (ICWMC), Guadeloupe, French Caribbean
98. Perrig A, Szewczyk R, Tygar JD, Wen V, Culler D (2002) SPINS: security protocols for sensor networks. Wirel Netw 8(5):521–534
99. Zhu S, Setia S, Jajodia S (2003) LEAP: efficient security mechanisms for large-scale distributed sensor networks. In: ACM conference on computer and communications security, Washington, DC
100. Karlof C, Sastry N, Wagner D (2004) TinySec: a link layer security architecture for wireless sensor networks. In: International conference on embedded networked sensor systems, Baltimore
101. Kulkarni RV, Forster A, Venayagamoorthy GK (2011) Computational intelligence in wireless sensor networks: a survey. IEEE Commun Surv Tutorials 13(1):68–96

Introduction to Mobile Ad-Hoc and Vehicular Networks

Moussa Ayyash, Y. Alsbou, and Mohamed Anan

Abstract Mobile ad–hoc and vehicular networks are excellent examples of the proliferation of wireless data communication technologies. A mobile ad-hoc networks (MANET) consists of wireless devices which can dynamically be setup and can operate without any centralized control. One promising application of MANETs is the vehicular ad-hoc networks (VANETs). A VANET is a combination of an architectural network and an ad-hoc network. VANETs are distributed self-organizing networks of mobile vehicles. This chapter introduces both MANETs and VANETs, their characteristics and challenges, and presents the various protocols and applications optimized for them.

1 Introduction

The field of wireless communication is witnessing an unprecedented growth in the scale and diversity, and becoming more popular than ever before. This is due to the proliferation of wireless data communication technologies and low cost mobile devices.

There are two approaches for enabling wireless communication between hosts. The first approach is based on an established cellular network infrastructure. The second approach is to organize an arbitrary and temporary ad-hoc network among all mobile nodes interested in communicating with each other. The major challenge of the former approach is that such networks are limited to places where there is an existing cellular network infrastructure. On the other hand, ad-hoc networks have smaller topology where individual nodes have limited bandwidth, transmission ranges, and physical security. Also, ad-hoc topology dynamically changes all the

M. Ayyash (✉)
Chicago State University, Chicago, IL, USA
e-mail: msma@ieee.org

Y. Alsbou
Mutah University, Mutah, Jordan

M. Anan
Alfaisal University, Riyadh, Saudi Arabia

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_2

time where nodes can appear and disappear at different points within the network. However, since ad-hoc networks do not rely on pre-established infrastructures, they can be deployed rapidly. Ad-hoc networks provide users with on-demand setup and unconstrained connectivity, which is useful in widely varying environments such as military and emergency situations at places with damaged or no pre-existing communication infrastructure.

The continuous progress in wireless communications has led to new research trends, which enable wireless networks to be highly deployed in daily life. Mobile Ad-hoc Networks (MANETs) is the most promising field of ad-hoc networks. Recently, the widespread adoption of several wireless standards has guided into an impressive increase in the wireless data networks. Due to this, academia and commercial sectors are looking for more applicable solutions for these wireless technologies. One promising application of MANETs is the Vehicular Ad-hoc Networks (VANETs) where it can effectively be considered a subset of MANETs. Despite a large set of potential applications in MANETs environments, VANETs are amongst the very first deployed large-scale instance of MANETs. This chapter introduces both approaches, their challenges, and presents the various protocols and applications optimized for them.

2 Mobile Ad-Hoc Networks

A MANET is a self-organized network. It consists of wireless nodes/devices, which can dynamically be set up and can operate without any centralized control or pre-existing infrastructure. The nodes in MANET use the wireless medium to communicate with other nodes if they are in their radio range. In a MANET, every node can act as a data terminal and/or a computing device. Examples of MANET devices include laptops, wearable computers, PDAs, and mobile phones. These kinds on networks have outstanding attributes, which characterize and affect their operations. These features are detailed below [1]:

1. *Dynamic network topology*: MANETs have a dynamic network topology due to node mobility. Therefore, the topology may change continuously which in turn affects the connectivity among wireless devices. As a result of this, MANETs should adapt rapidly to network changes and conditions in order to sustain mobility, traffic, resources, and propagation conditions.
2. *Self-organized network nature*: In MANETs, each node is an independent and self-governing terminal. A node may act as a host, a router, or both. In addition, the nodes are responsible for dynamically exploring and finding out other nodes to communicate with or handle some network configurations like addressing, and position location issues.
3. *Multi-hop routing*: In MANETs, nodes communicate with each other through single or multi-hop routing protocols in order to deliver data packets from a source to destination. A node must know some information about its neighbors for building its routing table, while the network topology can change quite often

in a MANET. Thus, routes may change while in use and become no longer valid in a very short time [2].

4. *Limited-resources terminals*: Generally, MANET terminals are mobile nodes with limited resources like processing capabilities, small memory sizes, and low power storage. Such resources must be optimized in order for these devices to efficiently perform computing and communicating functions.
5. *Distributed function*: Due to the infrastructureless nature of MANETs, the control management of the network is distributed among the mobile nodes. Therefore, nodes involved in a MANET should collaborate amongst each other to implement routing and distributed functions. These distributed functions include distributed coordination function (DCF), point coordination function (PCF), and a modified PCF to operate over infrastructureless networks by combining the operation of both DCF and PCF [3].
6. *Fluctuating link-capacity*: Mobility and resources scarcity usually causes high error rates in MANETs. Moreover, their radio transmission is vulnerable to noise, fading, multiple access and interference conditions. These conditions in addition to the high bit-error rate make wireless connections and links to be unstable.
7. *Security*: The security concerns in MANET are different than those that exist in conventional networks because wireless links are more susceptible to attacks [4]. MANETs also lack central administration and prior organization. Security solutions in MANETs must be established without reference to any centralized solutions. These solutions should be executed by the cooperation of all existing nodes in the network. That is due to the fact that using an inefficient authentication protocol in MANETs makes it easier for hackers eavesdrop and gain access to confidential information, which leads to a vulnerability increase. In addition, security solutions need to consider malicious attacks not only from outside but also from within the network.
8. *Quality of service (QoS)*: Real-time multimedia applications have different QoS requirements. In MANETs, the ability to provide QoS guarantees is critical because wireless links have variable capacity, high loss rates, and high latency. In addition, the weakness in QoS provisions is due to dynamic nature of topologies, which will result in frequent links breakages. Furthermore, the service quality of the network varies with time depending on the resource availability in the wireless medium and in the nodes [5].
9. *Interoperation*: The self-organization property of MANETs is a challenge when two independently formed networks come physically close to each other. For these two networks to join each other is not trivial. This is because the networks may be using different synchronization, physical layer, media access control (MAC) layer, routing, or security protocols.

Despite the aforementioned constraints, MANETs offer numerous benefits. MANETs are useful in situations where a fixed infrastructure is not possible, too expensive or unreliable. In addition, due to their self-organizing and self-administering capabilities, MANETs can rapidly be deployed with minimum cost, time, and planning. This allows users to access and exchange information regardless of their geographic position or proximity to infrastructure.

2.1 MANETs Applications

During the past few years, a noticeable growth in the era of mobile computing has been witnessed. This rapid expansion, which includes devices, applications, and protocols is solely focused on cellular or wireless local area networks (WLANs), not taking into account the great potential offered by MANETs [6]. Moreover, MANETs are not limited to operate as a stand-alone network but can be attached to cellular networks or to the Internet. As a result, MANETs are expected to become an important part of the 4G architecture. Such architecture aims to provide pervasive computer environments that support users in accomplishing information and communicating anytime, anywhere and from any device. This opens the means for various new and exciting applications of MANETs. The set of applications for MANETs is diverse. These applications appear, but are not limited to areas such as [7–10]:

1. *Emergency and crisis management applications*: MANETs are very applicable in disarray or inoperative areas like fire, flood, or earthquake where restoring communications quickly is essential for emergency or rescue operations. MANETs' infrastructures could be built very rapidly compared to long time and costly efforts required for constructing wired communications.
2. *Local level applications*: MANETs can connect an immediate and temporary multimedia network using notebook computers to exchange information among users at a conference or a hall. Another useful application on the local level is in-home networks where wireless devices can communicate directly to share information. Similarly, in some civilian environments like trains, stadium, aircraft, MANET communications will have many applications.
3. *Commercial applications*: As an example, in business scenarios, the necessity for joint computing and information exchanges might be more important outside office environments than inside a building. Therefore, sometimes people do need to have outside meetings to cooperate and exchange information on a given project. Other commercial scenarios include ship-to-ship ad-hoc mobile communication, law enforcement, etc.
4. *Military battlefield applications*: MANETs were mostly created for military purposes and applications [11]. In these situations, the importance of ad-hoc networking stems from maintaining an information exchange network among the soldiers, vehicles, and military information headquarters.

2.2 MANETs Protocols

In ad-hoc network, nodes can move in an uncontrolled manner. This dynamic network with rapid topological changes cause frequent route failure. The conventional routing protocols such as link state and distance vector are not suitable for such

network. They may work within an ad-hoc network with low mobility where the topology is not changing very often. Therefore, different types of routing protocols need to be designed for MANETs.

MANETs independent research groups have produced many different ad-hoc routing protocols. MANETs routing protocols can be classified into either proactive, reactive, or hybrid protocols. In proactive protocols, each node stores routes to all destination nodes in a routing table. This helps in forwarding packets without delay but these types of protocols require storage and continuous updates to these tables. Examples of proactive protocols include: global state routing (GSR), hierarchical state routing (HSR), and destination-sequenced distance vector routing (DSDV) [2].

In contrast, reactive protocols require a node to perform a route search for every new destination. This approach eliminates the overhead of storage, maintenance, communication, and continuous route updates. However, the delay of forwarding packets increases due to the time needed for finding routes to the destination. Examples of reactive protocols include: Ad-hoc on-demand distance vector routing (AODV), dynamic source routing (DSR), location aided routing (LAR), and temporally ordered routing algorithm (TORA) [2].

There exist a number of hybrid MANETs protocols that are globally reactive and locally proactive. In these protocols, a route within a zone is immediately available but a reactive approach is needed for routes outside a zone. The zone routing protocol (ZRP) suite is an example of such category. The intrazone routing protocol (IARP), interzone routing protocol (IERP), broadcast routing protocol (BRP), and neighborhood discovery protocol (NDP) are examples of hybrid protocols under the ZPR protocol suite.

2.3 Technological Challenges in MANETs

MANETs impose many operation and functional challenges. In addition, they also suffer from technical challenges on all layers of the *open system interconnection* (OSI) network protocol stack [6]. In this regard and due to mobility, the physical layer must react and deal with fast and vast changes in link characteristics; the MAC layer must use fairness mechanisms to allow fair channel access, and minimize packet collisions due to hidden and exposed terminals. At the network layer, nodes need to cooperate to solve routing issues and paths selecting from source to destination. The transport layer must be modified in order to be able to deal with packet losses and delay characteristics that are much higher in MANETs than wired networks. Applications and session layers should be able to handle possible disconnections and reconnections. Furthermore, all network protocol developments need to integrate smoothly with traditional networks and take into account possible security problems.

3 Vehicular Ad-hoc Networks

One promising application of MANETs is the VANETs where it can effectively be considered as a subset of MANETs. A VANET is neither an architectural network nor an ad-hoc network but a combination of both. Ad-hoc networks have been studied for some time but VANETs will form the biggest ad-hoc networks ever implemented; therefore, issues of stability, reliability, and scalability are very important to be considered.

Generally, VANETs are distributed self-organizing networks of mobile vehicles. These vehicles operate as mobile nodes in a wireless network. In an ad-hoc network, nodes communicate directly with each other. However, VANETs are different from other ad-hoc networks due to: high node mobility, the variable node density, and the unpredictable and harsh communications environment [12]. Once multi-hop communication is employed, a VANET enables a vehicle to communicate with other out of sight vehicles or even out of radio transmission range. For the communications in a VANET to be realized, vehicles and roads are equipped with wireless communication devices which include on-board units (OBUs) for vehicles and stationary nodes called roadside units (RSUs) deployed along the roads [13]. RSUs are devices distributed along the road to perform local tasks related to the traffic. RSUs may be traditional devices like traffic lights or dedicated devices such as multi-hop relays to improve the connectivity between vehicles. In addition, these devices can be structured as: a stand-alone, connected with each other to form a network or connected to the Internet and linked to central servers [14, 15]. Both OBUs and RSUs devices have wireless/wired communications capabilities. The role of the OBU is to collect data from other units in the vehicle and communicate with other OBUs in other vehicles and with RSUs via one or more radio links. Based on this, there are two communication scenarios in VANETs: vehicle-to-vehicle (V2V) (i.e., between OBUs) and vehicle-to-road station (V2R) (i.e., between OBUs and RSUs) [12, 13]. The main components of a VANET and how they perform communications are indicated in Fig. 1.

VANETs use various advanced wireless technologies such as dedicated short-range communications (DSRC), which is a WiFi technology appropriate for VANETs operations [13]. DSRC uses wireless short-to-medium range communication in a vehicular environment and has the potential for enhancing transportation safety and improving highway efficiency [16]. In addition, the DSRC is employed in VANETs due to its fast response and high data transfer rates in rapidly changing communication environments.

Nowadays, VANETs are considered an important part of large intelligent transportation systems (ITS) to improve the performance in order to provide road safety, comfortable driving, and distribution of updated information about the roads [17]. Moreover, other goals of the ITS are to reduce traffic congestion, waiting times, and fuel consumptions.

A complete ITS with the aid of VANETs helps drivers to acquire real-time information about road situations and give them the opportunity to react on time.

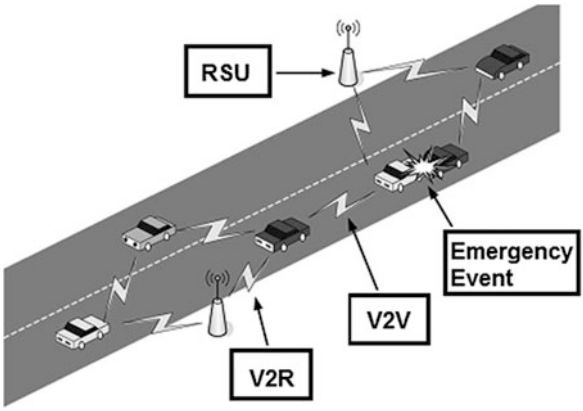

Fig. 1 VANET main components

For example, vehicles send warning messages about an accident to allow drivers to take proper decisions prior to entering the crash zone [18, 19]. Also, current transportation conditions are communicated to facilitate driving by taking new routes in case of congestion [20].

3.1 VANETs Applications

Future vehicle manufacturers consider a number of potential applications based on VANETs technology. A VANET communication platform allows an enormous variety of applications aimed at administration, companies, drivers, and people in the vehicle. Although the spectrum of these applications is very wide, the main applications of VANET can be categorized as [13, 21, Rawashdeh and Mahmud, 1]:

1. *Safety-related applications*: Safety-related applications are amongst the most important VANETs applications. This is due to its relevance to life-critical situations where the existence of such service may prevent injuries and life-endangering accidents. These include pre-collision warning, electronic road signs, traffic light violation warning, online vehicle diagnosis, and road condition detection. These types of applications usually employ short-range communication to perform real-time detection and provide warnings to drivers like: municipal traffic management, traffic congestion detection, route planning, highway tolling, and public transportation management. The following are some examples of VANETs safety applications:
   - *Cooperative collision avoidance (CCA)*: The main goal of this application is to prevent collisions by helping vehicle drivers in detecting possible obstacles on the road. This service will be triggered automatically whenever there is a possibility of collisions. For example, in case of detecting a possible collision situation or accident, a warning or notification messages are sent

to the vehicles approaching the accident area. Also, these messages may be forwarded by these vehicles to other vehicles. Therefore, the drivers can take appropriate actions or the vehicle itself can stop or decrease the speed automatically. Another example is when CCA messages are sent to vehicles approaching the accident area to avoid crashes during lane change. One of the proposed techniques to disseminate CCA messages on a highway was presented in [22]. Another type of CCA is the emergency warning messages (EWM). Some of the proposed techniques that related to EWM are presented in [9, 10, 23].

- *Cooperative driver assistance system (CDAS)*: This service depends on the exchange of information among vehicles in order to broaden the range of perception. By transmitting this information, drivers are informed about hazards, obstacles or traffic flow ahead, resulting in more efficient and safe driving. In this way, information could be transferred to each vehicle in the network to notify about a traffic jam that has started at a certain point of the road.
- *eCall*: eCall is a project of the European Commission intended to bring rapid assistance to drivers involved in a collision anywhere in the European Union. For example, in case of crash, an eCall-equipped vehicle automatically calls the nearest emergency center even if no passenger is able to speak. This call includes the location of the crash site, therefore, shortly emergency services know that there has been an accident, and its exact location. eCall cuts emergency services' response time where it goes down to 50 % in the countryside and 60 % in built-up areas.
- *Cooperative intersection collision avoidance (CICA)*: CICA is employed to avoid collisions at road intersections. This service is achieved by utilizing the installed RSUs at road intersections. An RSU periodically dispenses messages about the intersection status to the approaching vehicles. These messages may include information about: (1) Traffic signal state and time remaining until the traffic switches to a new state, (2) State of the vehicles (location, speed, etc.) approaching the intersection that are within a relevant distance/time from the intersection, (3) Intersection environmental conditions (weather, visibility, road surface at the intersection, etc.) [24].

2. *Traffic managements and control*: This application is to help traffic flow in reducing traffic congestion, fuel consumption, and travel time. In contrast to the safety applications, this application is less strict on real-time constraints, where if the messages are delayed, there is no real threat to life. The messages based on this service are used to describe the condition of roads and traffic in certain areas like highways or city centers [25].
3. *Applications for administration*: This will present a safe and fast means of information support from vehicles without stopping them or interrupting their movement. This is done by storing the necessary information about the vehicle and automatically transmits the information once an authorized device requests it. In addition, vehicle identification service will help police in different ways such

as checking if a vehicle and its driver have the necessary documentation, or if a violation is detected; a report would be automatically processed.
4. *Infotainment applications*: Examples of these applications are: video and music sharing, location-based restaurant or store reviews, carpooling, and social networking. These applications have become attractive add-ons in the vehicle market. The networking of infotainment systems will surely be a trend in the near future.
5. *Commercial and comfort applications*: This type of applications is to provide and improve comfort and traffic efficiency to the passengers. This may include current traffic, weather information, and interactive communication. Other services include the reception of "wireless advertising" commercial vehicles, and roadside infrastructure about their businesses or gas stations. An important aspect of these applications is that they should not consume the bandwidth and interfere with safety applications. The main concern should always be given to the safety data where using separate physical channels is a feasible solution. Examples of these applications include: **Electronic toll collection, Entertainment applications, and Internet access** [26].

3.2 Requirements of VANET Applications

In order for the VANET applications to be practical, they should have several basic requirements. These requirements may be summarized as follows:

1. *Scalability*: scalability is one of the vital factors that VANET should consider. This is due to the large number of vehicles that the vehicular networks may incorporate. Therefore, the deployment of hybrid architecture, for example in-network aggregation techniques and P2P technologies, may provide a scalable information exchange.
2. *Availability*: availability of VANETs plays an important role in real-time interaction between vehicular networks and the physical world. In addition, availability has a major impact on the safety and efficiency of road systems to withstand unexpected system failures.
3. *Context-awareness*: this implies that VANETs protocols should be flexible to real-time transportation system changes, including vehicle density and movement, traffic flow, and road topology changes.
4. *Security and privacy*: due to the cyber physical nature of VANET, security and privacy are critical. Without considering security, on-board computer systems will make confidential driver and vehicle information vulnerable to others.

3.3 VANETs Characteristics and Challenges

As mentioned earlier, VANETs represent a specific aspect of MANETs. Nevertheless, the characteristics of a VANET are unique compared to other MANETs [27]. VANETs are distributed, and self-organizing wireless network constructed by mobile vehicles; and therefore characterized by very high node mobility and limited degrees of freedom in the mobility patterns [28]. This means that VANETs are generally characterized by a fast changing topology, continuous network fragmentation, small network diameter, and most importantly unique security challenges. However, some research results on MANETs are not applicable to VANETs because of the unique characteristics of VANET which are different than the ones in MANETs. The following are the main characteristics and constraints of VANETs compared to MANETs [1]:

1. *Processing and energy*: VANETs, unlike MANETS, have no limitations with respect to energy. In MANETs, energy limitation is one of the main challenges. In addition, VANETs have no constraints with respect to processing capability, which in turn allows for supporting several communications interfaces.
2. *Environment and mobility*: In contrast to MANETs' limitations to open spaces or indoors, VANETs allow vehicle to move across road infrastructures, on highways, or within a metropolitan area. However, both VANETs and MANETs suffer radio obstacles because of buildings, bridges, and multipath and fading effects, which significantly influence the mobility model and radio transmission quality.
3. *Type of information and diffusion*: Because VANETs main application is road safety, the nature of communications will be from a source to multiple receivers. Therefore, the information diffusion depends on vehicle location. In such positions, communications are mainly unidirectional.
4. *Network topology, size, and connectivity*: As mentioned previously, VANETs, unlike MANETs, are characterized by very high mobility due to vehicle speed. Hence, a vehicle can rapidly enter or depart the network in a very short period of time, which allow for rapid and continuous topology changes [29]. This will result in difficulties in vehicle link connectivity to the network where many paths may be disconnected before they can be utilized, which is a key challenge in VANETs [27]. Moreover, VANETs size plays an important role in making devising suitable protocols and architectures more difficult.
5. *Variable network density*: The VANET density depends on vehicular number in specific area, which is highly changeable. In traffic jam situations, like city centers, the network can be categorized as very dense network whereas in suburban traffics it could be a light network.
6. *Security*: One of the most important requirements for VANETs is security. The information sent from vehicles about their situation must reveal their actual state. That is because when a vehicle sends false data to other vehicles, drivers may take wrong decisions, which may affect their safety. Therefore, the deployment of a security solution must cope with these conditions and constraints.

7. *Routing*: Since VANETs are a special type of MANETs, the most commonly used MANET routing protocols have been tested and evaluated for use in a VANET environment like destination-sequenced distance vector (DSDV), DSR, and ad-hoc on-demand distance vector (AODV). However, the existing routing algorithms used in MANETs are much less suitable in a VANET environment. This is due to the fact that VANETs issues and challenges are different than these of MANETs. Examples of such challenges include: network topology, mobility patterns, demographics, density of vehicles at different times of the day, rapid changes in vehicles arriving and leaving the VANET, and the fact that the width of the road is often smaller than the transmission range. All aforementioned challenges make the use of these conventional ad-hoc routing protocols inadequate. Therefore, routing in VANETs has been studied and investigated widely in the past few years to find scalable routing protocols that deal with the frequent path disconnections caused by vehicle mobility [30]. In addition, researchers proposed efficient approaches that can improve throughput and enhance packet delivery ratio in VANETs [31].
8. *Quality of service (QoS)*: Generally, QoS in wired networked can often be achieved using many techniques like resource reservation protocols and adequate infrastructure. However, this is not guaranteed in dynamic, ad-hoc environments, such as VANETs. This is because VANETs do not have reliable infrastructure and poses a rapidly changing topology. Based on this, several research papers have been published to provide some possible mechanisms to guarantee QoS in VANETs [22, 32].
9. *Broadcasting*: Broadcasting is one of the vital challenges in VANETs due to the considerable number of messages transmitted [33]. This is because the wireless communication technologies employed in VANETs are not capable of handling broadcast transmissions as a result of frequent message collisions leading to frequent retransmissions by vehicles. Moreover, this will reduce the messages delivery rate and increase message delivery time, which will result in unreliable network especially in critical situations and safety messages. Hence, providing reliable broadcast messages with minimal overheads for VANETs is very challenging [34–36].

3.4 Standards for Wireless Access in VANETs

There are several standards for wireless access in VANETs. These standards include protocols related to transponder equipment and other communication protocols related to security, routing, addressing services, and interoperability protocols [33]:

1. *Dedicated short-range communication (DSRC)*: DSRC is a standard used for short-to-medium range communications service, which aims to bring vehicular networks to North America. DSRC was developed to support V2V and V2R communications. DSRC provides high data transfers and low communication

latency in small communication zones. In 1999, the United States federal communications commission (FCC) allocated 75 MHz of spectrum at 5.9 MHz to be used by DSRC. DSRC was approved to be used based on the IEEE 802.11a physical layer and 802.11 MAC layer [37]. DSRC is a free but licensed spectrum, which means that it is more restricted in terms of its usage [38]. The spectrum of DSRC is categorized into seven channels. Every channel is of 10 MHz wide. These channels are used as follows: one channel is allocated for safety communications to broadcast safety data to alter about dangerous driving conditions. Two other channels are used for future use. All the remaining channels are service channels (SCH) and can be used for either safety or non-safety applications [33].

2. *Wireless access in vehicular environments (WAVE) (IEEE 802.11p)*: Wireless connections among VANETs elements can be achieved using the existing 802.11a devices of 54 Mbps data rates [39]. Due to VANETs special characteristics like varying driving speeds, dynamic topologies, and traffic patterns, IEEE 802.11 MAC operations endure additional overheads when used in vehicular environments. This will affect the data exchange speed especially in safety applications. To deal with these difficulties in the MAC operations, the DSRC was migrated to the IEEE 802.11 standard group which renamed the DSRC to IEEE 802.11p WAVE [40]. By merging DSRC into IEEE 802.11 family, WAVE will be adopted to be the most commonly used standard for VANETs [40]. WAVE uses orthogonal frequency division multiplexing (OFDM) to split the signal into several narrowband channels to provide a data payload communication capability of 3, 4.5, 6, 9, 12, 18, 24, and 27 Mbps in 10 MHz channels.

References

1. Ríos XC, Pérez DP (2011) Performance evaluation of realistic scenarios for vehicular ad hoc networks with Citymob and Nctuns simulator. Enginyeria de Telecomunicació. http://upcommons.upc.edu/pfc/bitstream/2099.1/12310/1/PFC_-_Performance_Evaluacion_of_Manhattan_Downtown_Scenarios_for_VANETs_with_CityMob_and_NCTUns.pdf
2. Preetida VJ, Sugata S, Bai R, Singhal M (2006) DOA: DSR over AODV routing for mobile ad hoc networks. IEEE Trans Mob Comput 5(10):1403–1416
3. Crespo C, Alonso-Zarate J, Alonso L, Verikoukis CV (2009) Distributed point coordination function for wireless ad hoc networks. In: Proceedings of the 69th IEEE vehicular technology conference, VTC Spring 2009, 26–29 April 2009, Hilton Diagonal Mar, Barcelona, Spain
4. Mamatha T (2012) Network security for MANETS. Int J Soft Comput Eng 2(2):65–68. ISSN: 2231-2307
5. Houda L (2008) Wireless ad hoc and sensor networks. Wiley-ISTE, USA. ISBN: 1848210035
6. Hoebeke J, Moerman I, Dhoedt B, Demeester P (2004) An overview of mobile ad hoc networks: applications and challenges. J Commun Netw 3:60–66
7. Goyal P, Parmar V, Rishi R (2011) MANET: vulnerabilities, challenges, attacks, application. Int J Comput Eng Manage 11, pp 32–37, ISSN (Online): 2230-7893
8. Frodigh M, Johansson P, Larsson P (2000) Wireless ad hoc networking: the art of networking without a network. Ericsson Rev 4:248–263

9. Yang H, Yun H, Ye F (2004a) Security in mobile ad-hoc networks: challenges and solutions. IEEE Wirel Commun 11(1):38–47
10. Yang X, Liu J, Zhao F, Vaidya NH (2004b) A vehicle-to-vehicle communication protocol for cooperative collision warning. In: Proceedings of the first annual international conference on mobile and ubiquitous systems: networking and services, Boston, USA
11. Rajabhushanam C, Kathirvel A (2011) Survey of wireless MANET application in battlefield operations. Int J Adv Comput Sci Appl 2(1):50–58
12. Booysen M, Zeadally S, Rooyen G (2012) A performance comparison of media access control protocols for vehicular ad-hoc networks (VANETs). IET Netw 1(1):10–19
13. Rawashdeh ZY, Mahmud SM (2011) Communications in vehicular networks. In: Mobile ad-hoc networks: applications. InTech in Rijeka, Croatia
14. Casteigts A, Nayak A, Stojmenovic I (2009) Communication protocols for vehicular ad hoc networks. Wirel Commun Mob Comput. Published online in Wiley InterScience
15. Kosch T, Kulp I, Bechler M, Strassberger M, Weyl BR (2009) Communication architecture for cooperative systems in Europe—[automotive networking series]. IEEE Commun Mag 47(5):116–125
16. Bai F, Daniel D, Krishnan H (2010) Toward understanding characteristics of dedicated short range communications (DSRC) from a perspective of vehicular network engineers. In: Proceedings of the annual international conference on mobile computing and networking, MOBICOM 2010, pp 329–340
17. Tripp C, Mateos M, Soto P, Mezher A, Aguilar-Igartua M (2012) Smart city for VANETs using warning messages, traffic statistics and intelligent traffic lights. In: Intelligent vehicles symposium 2012, pp 902–907
18. ElBatt T, Goel SK, Holland G, Krishnan H, Parikh J (2006) Cooperative collision warning using dedicated short range wireless communications. In: Proceedings of ACM VANET 2006, pp 1–9
19. Xu Q, Mark T, Ko J, Sengupta R (2007) Medium access control protocol design for vehicle–vehicle safety messages. IEEE Trans Veh Technol 56(2):499–518
20. Nadeem T, Dashtinezhad S, Liao C, Iftode L (2004) Traffic view: traffic data dissemination using car-to-car communication. ACM Mob Comput Commun Rev (MC2R) 8(3):6–19
21. Raya M, Hubaux J (2007) Securing vehicular ad hoc networks. J Comput Secur 15(1):39–68
22. Biswas S, Tachikou R, Dion F (2006) Vehicle-to-vehicle wireless communication protocols for enhancing highway traffic safety. IEEE Commun Mag 44(1):74–82
23. Maihöfer C, Cseh C, Franz W, Eberhardt R (2003) Performance evaluation of stored geocast. In: Proceedings of the IEEE 58th vehicular technology conference, Orlando, FL, USA
24. Benmimoun A, Chen J, Neunzig D, Suzuki T, Kato Y (2005) Communication based intersection assistance. In: Proceedings of the IEEE intelligent vehicle symposium, Las Vegas, NV, USA
25. Kihl M, Sichitiu M, Joshi HP (2008) Design and evaluation of two geocast protocols for vehicular ad-hoc networks. J Internet Eng 2(1)
26. Wang SY (2007) The potential of using inter-vehicle communication to extend the coverage area of roadside wireless access points on the highway. In: Proceedings of the IEEE international conference on communications, Glasgow, UK
27. Blum JJ, Eskandarian A, Hoffman LJ (2004) Challenges of intervehicle ad hoc networks. IEEE Trans Intell Transp Syst 5(4):347–351
28. Prasanth K, Duraiswamy K, Jayasudha K, Chandrasekar C (2010) Packet transmission analysis in vehicular ad hoc networks using revival mobility model. Int J Adv Netw Appl 1(1):252–257
29. Wang SY (2004) Predicting the lifetime of repairable unicast routing paths in vehicle-formed mobile ad hoc networks on highways. In: IEEE PMRC 2004
30. Chung S-E, Yoo J, Kim C-K (2009) A cognitive MAC for VANET based on the WAVE systems. In: Proceedings of 11th international conference on advanced communication technology (ICACT 2009), vol 1, pp 41–46

31. Jinyuan S, Chi Z, Yuguang F (2007) An ID-based framework achieving privacy and non-repudiation. In: Proceedings of IEEE vehicular ad hoc networks, military communications conference (MILCOM 2007), pp 1–7
32. Zhu J, Roy S (2003) MAC for dedicated short range communications in intelligent transport systems. IEEE Commun Mag 41(12):60–67
33. Zeadally S, Hunt R, Hassan A (2010) Vehicular ad hoc networks (VANETS): status, results, and challenges. Telecommun Syst 3(1):1–25
34. Ciccarese G, De Blasi M, Marra P, Palazzo C, Patrono L (2009) On the use of control packets for intelligent flooding in VANETs. In: Proceedings of IEEE wireless communications and networking conference (WCNC 2009), pp 1–6
35. Amoroso A, Roccetti M, Nanni M, Prati L (2009) VANETS without limitations: an optimal distributed algorithm for multi-hop communications. In: Proceedings of 6th IEEE consumer communications and networking conference 2009, CCNC 2009, Las Vegas
36. Yang L, Guo J, Wu Y (2009) Piggyback cooperative repetition for reliable broadcasting of safety messages in VANETs. In: Proceedings of 6th IEEE consumer communications and networking conference 2009, CCNC 2009, Las Vegas
37. Standard specification for telecommunications and information exchange between roadside and vehicle systems—5 GHz band dedicated short range communications (DSRC) medium access control (MAC) and physical layer (PHY) specifications. ASTM E2213-03, September 2003.
38. Notice of proposed rulemaking and order FCC 02-302. Federal Communications Commission, November 2002
39. IEEE Standard 802.11 (2007) IEEE Std. 802.11-2007, Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications
40. IEEE P802.11p/D3.0, draft amendment for wireless access in vehicular environments (WAVE), July 2007

Routing in WSNs for Space Application

Mohamed Riduan Abid and Driss Benhaddou

Abstract Wireless Sensors Networks are indispensable for applications whereby data needs to be sensed and transmitted from unfriendly terrains e.g., planets' surfaces.

The inherent ad-hoc nature of WSNs, whereby no topology can be defined in advance, imposes challenges to *Routing*. Sensor nodes have to detect each other and build an ad-hoc topology with specific gateways (e.g., sinks) that forward data to the control plane.

In this chapter, we present a deployment topology for WSNs in space applications, and delineate the challenges towards an optimal routing protocol that accounts basically for energy efficiency, self-organization, self-adaptation, and the use of multiple gateways. The latter disseminate data towards the control plane. In this context different WSNs routing protocols are surveyed and relevant trade-offs are presented and discussed.

1 Introduction

The Wireless Sensor Networks (WSNs) technology [1] has been around for long. In academia, it was first tackled, in the early 1990s, within the context of the military-borrowed MANETs (Mobile Ad-hoc Networks) [2–4] technology. Since then, the technology witnessed good interest from both academia and industry. With the recent advance in MEMS (Micro-Electro-Mechanical Systems), which substantially reduced the sensors cost and size, WSNs is strongly back and providing a further enlarged set of applications, e.g., distributed control (home automation, energy efficiency, smart grids), weather and ecosystem monitoring (temperature, humidity, light, sound, movement), military, positioning and tracking, security, health care, disaster management, and space applications.

M.R. Abid (✉)
School of Science and Engineering, Alakhawayn University in Ifrane, Hassan II Ave, Mail Box 1005, Ifrane 53000, Morocco
e-mail: R.Abid@aui.ma

D. Benhaddou
University of Houston, College of Technology, Houston, TX, USA

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_3

Indeed, the reduced cost of sensors (which implies a reduced WSN deployment cost) encouraged the adoption of WSNs for a plenty of new emerging applications, e.g., Smart Grids [5, 6], IoT (Internet of Things) [7], and space application. These applications stipulate a large-scale deployment of the technology. Unlike small-scale deployments, where the sensed data is usually forwarded towards a sole sink node, Large-Scale WSNs (LS-WSNs) support multiple sinks, and the sensed data is routed via the "closest" path (to the sink) that exhibits less wireless hops and less energy consumption.

Indeed, in small-scale WSNs, the sensors do not have the choice to adhere to multiple sinks. Instead, they forward data to the sole sink (that serves as a gateway to the control place) and form a tree-based topology with fat-links at the edged closer to the sink: a fact that complicates the load-balancing task. In LS-WSNs, e.g., space applications, sensors can adhere to different sinks, and thus favoring the load-balancing task by distributing the traffic among the alternative routes (to the sink) in the network, and this would optimize energy consumption since the frames will not witness stringent congestion. This latter, when present, forces the node to retransmit the frames (because of the high loss rate), and this increases energy consumption, which is the most important resource in WSNs.

In large-scale WSNs, the ability to adhere to different sinks permits the sensors to build clusters in the form of sink-rooted trees. In such a topology, the routing protocol should account for two basic metrics: energy consumption and load-balancing. In the case where sinks have a wireless connection to the control place (e.g., in space applications), a wireless mesh network is formed to connect the sinks, and a different routing strategy needs to be adopted.

In this chapter, we present the subtleties of WSNs from a routing point of view, and we propose an appropriate architecture that further eases the WSNs deployment, especially for space applications.

In Sect. 2, we present the topology of WSNs for space applications. Sect. 3 surveys alternative routing protocols, highlights the relevant challenges, and discuss the challenges in the light of energy efficiency, self-organization, and self-adaptation. Finally, we conclude in Sect. 4.

2 WSNs: Topology

WSNs can be deployed in two basic scenarios: 1. Small scale, e.g., spanning a building, and 2. Large scale, e.g., spanning large field (of tens of km^2) on planets. In the first scenario, see Fig. 1, there is usually a single sink that serves as a gateway to the control plane. The sink can be connected to the control plane via the Internet or any other kind of private network (e.g., a Satellite network). In such a scenario, the sensors are constrained to forward the data to the single sink, in contrast to the large-scale deployment where multiple sinks are available. Thus, the routing protocol is straightforward, and can be either a tree-based routing one where the sensors connect to the sink via a single hop connection or a mesh-based one where the sensors connect to the sink via a multi-hop connection, and whereby sensors act as routers on behalf of each other.

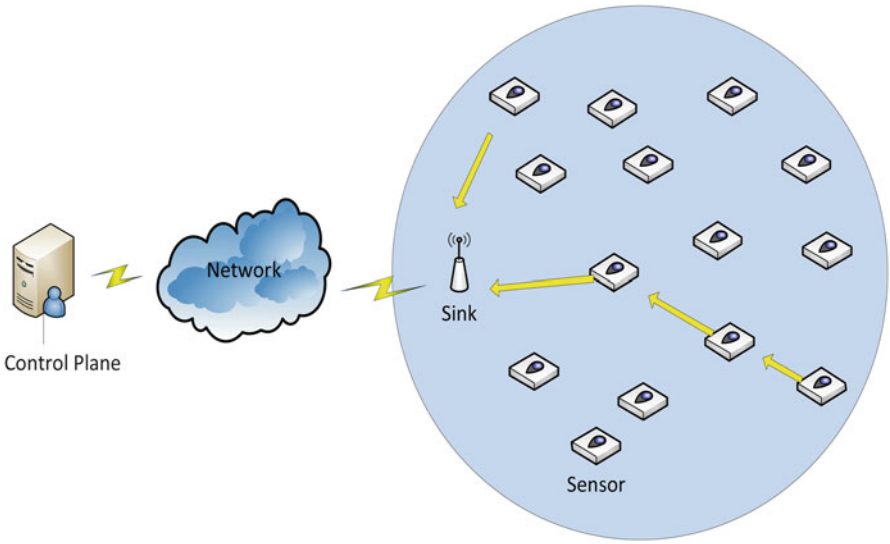

Fig. 1 Small-scale WSNs with a single sink node

On the other hand, LS-WSNs support multiple sinks in order to cover the relatively large area where the sensors are deployed. In such a case, every sensor selects the best sink to adhere to while accounting for the two basic metrics of the number of hops and the load-balancing, see Fig. 2. In this figure, sensor node #1, which was adhering to sink-1 in Fig. 1 via the path 1–2–3, is now adhering to sink 2, via the path 1–4, and this is because the latter path might exhibit better metrics than the former one.

The presence of multiple sinks affects the topology of the deployed WSNs since the connections from the sensors to the sinks become dynamic one and adapt to the link quality as well as to the load-balancing. Furthermore, the sinks, when they are wirelessly connected to the control plane, which is the common case (e.g., in space applications where all communication is wireless), form a wireless mesh topology as well, see Fig. 3. In such a case, the routing protocol has to be adapted to communicate the sensed data between the sinks. The routing protocol might be different from the one adopted in routing between sensors, as the sinks might use different wireless technologies, e.g., Wi-Max, Wi-fi, a fact which alleviates the energy consumption constraint.

3 Routing

Multi-hop routing is a key issue in Wireless Sensors Networks. Besides being multi-hop wireless networks, WSNs are exhibiting the stringent and inherent constraint of Power limitation.

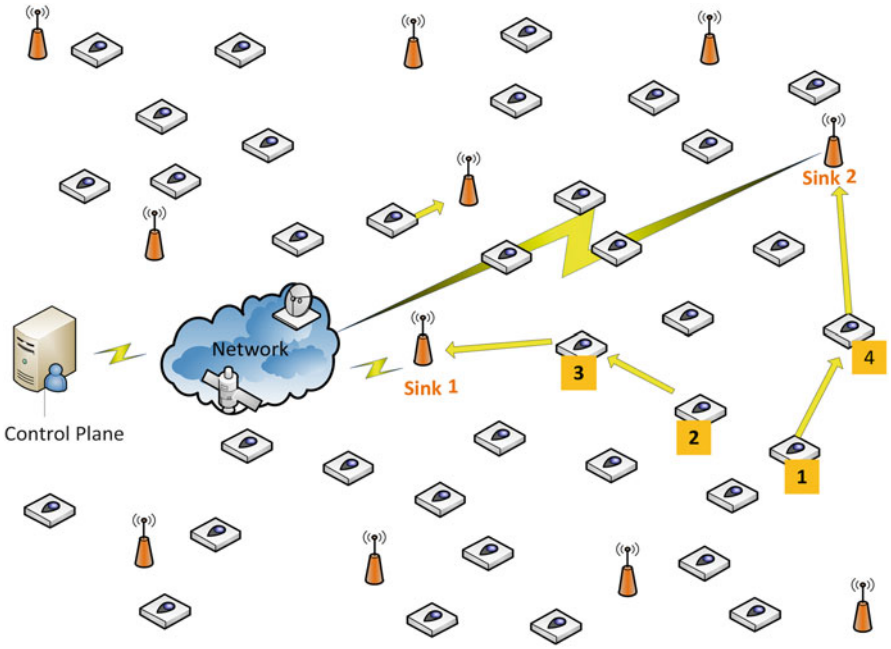

Fig. 2 Large-scale WSNs with a multiple sink nodes

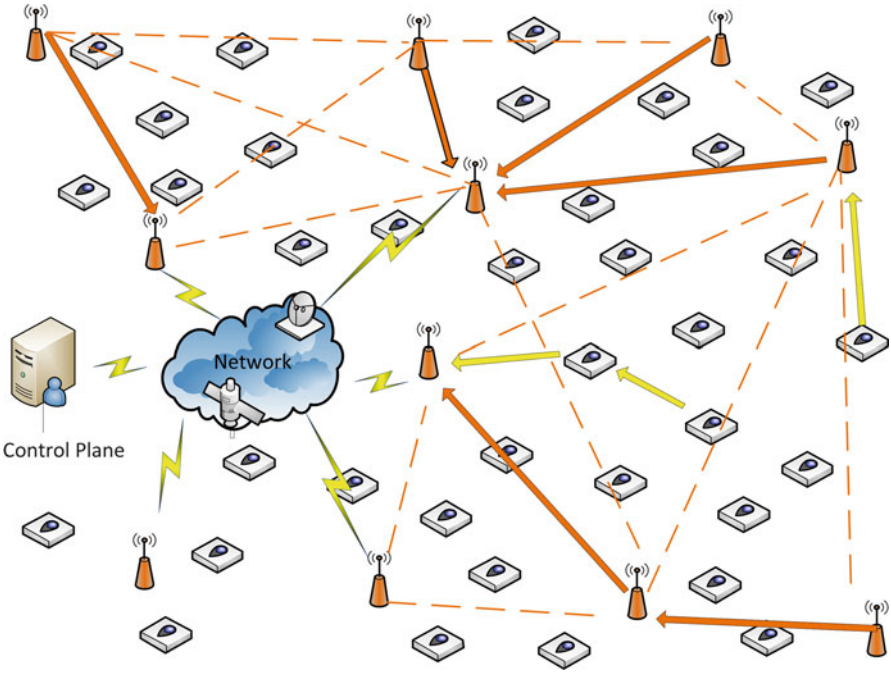

Fig. 3 Mesh routing among the sinks

Ideally, the routing is optimal only and only if the "Routing Tables Entries" are regularly updated as a reaction to the changes in the network topology and links quality. This is done via an extensive exchange of packets/frames between the routers. In WSNs, the topology and the link quality is continuously changing because of inherent unstable quality of the air-interface, and because of the mobility of sensors in some scenarios, e.g., Vehicular systems. Since, WSNs cannot cope with such an extensive exchange of packets due to the limitation in power (most of the sensors are usually locally powered, e.g., via batteries), the routing protocol is becoming a challenge indeed.

In space applications, WSNs will be mostly deployed in large areas spanning tens of km^2 at the surface of planets (e.g., moon, mars) where the sensors will communicate real-time data about physical measures like temperature, movement, gravity, electromagnetic fields, soil, and humidity. The sensed data need to be reliably communicated to the control planes where scientists are setting, and ready to analyze and mine the data. In such a scenario, the deployed WSNs will exhibit an ad-hoc mesh topology (aka. unstructured topology) since the number of deployed sensors can be huge (e.g., thousands), and thus they will randomly be scattered in the area.

In such Wireless Mesh Networks (WMN), there are three types of nodes:

1. Sensors: These are the devices that sense the data
2. Ordinary sinks: These are the devices that collect the data from the sensors in its vicinity
3. Gateway sinks: These are the devices that aggregate data from the ordinary sinks, and forward it to the control plane via a private network, e.g., Satellite network.

Besides collecting data from the sensors, the ordinary sinks are routing data on behalf of each other towards the gateway sinks, see Fig. 4.

For space applications, the data flow will have two patterns: 1. From Sensors to Ordinary sinks (SO plane) and 2. From Ordinary sensors to Gateway sensors (OG Plane). See Fig. 4: The links in "yellow" are in the SO plane, and the links in "pink" are in the GO plane.

The routing in both planes is a wireless multi-hop one, and the routing protocol has to choose the path that exhibits better reliability and less energy consumption. To this latter end, every routing protocol has a *routing metric* that evaluates the *cost* of the transmission on a specific path. The cost of a route is usually computed as the sum of the cost in the single hops along the path. Next section surveys most of the adopted routing metrics in wireless multi-hop routing.

3.1 Wireless Multi-hop Routing Protocols

Multi-hop wireless routing protocols are classified into two main categories: 1. Reactive routing and 2. Proactive routing.

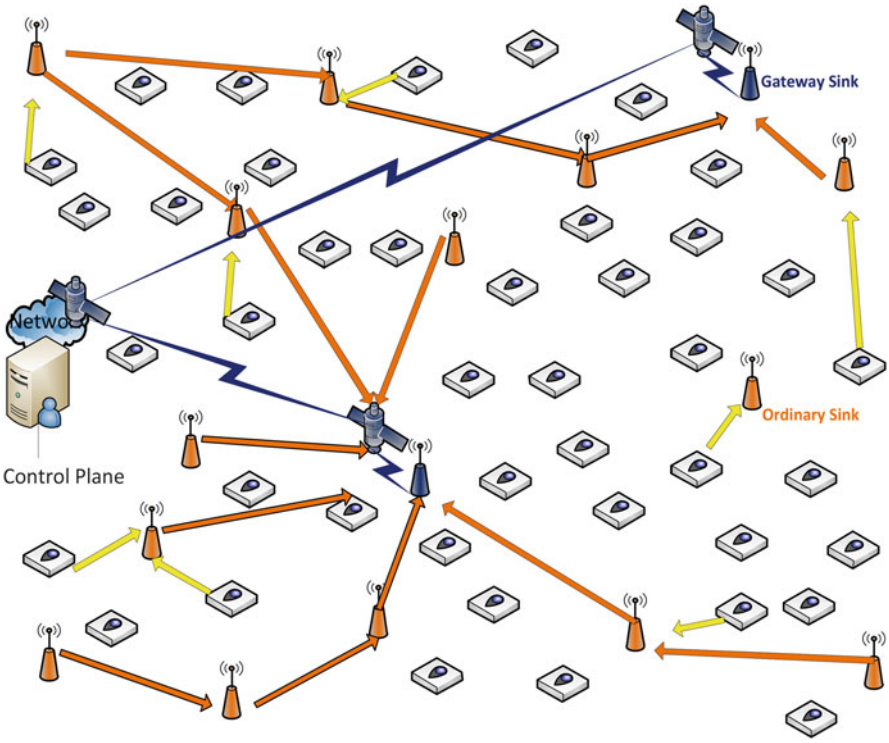

Fig. 4 Data flow planes

Reactive Routing

Reactive routing protocols initiate route discovery requests only when needed, such as in case of a route failure or a route time-expiration. The most well-known one in literature is the Adhoc Ondemand Distance Vector (AODV) [8] (The actual routing protocol in the Zigbee standard is an adaptation of AODV.)

In AODV, when a node needs a path to a certain destination, it broadcasts a PREQ (Path Request) message that contains the address of the destination. Every PREQ message bears a unique sequence number that distinguishes it from other PREQ messages. When an intermediate node receives a PREQ message, it first checks the freshness of the received message by comparing its sequence number with the last locally stored sequence number. A PREQ message is processed only when it has a greater sequence number or when it has the same sequence number but exhibiting a better route.

After receiving a fresher PREQ message, intermediate nodes update their reverse path towards the source, update the routing metric by including the weight of the last hop, and then re-broadcast the updated PREQ message. Subsequent MPs (Mesh Point) proceed the same way until the PREQ reaches the destination.

When the destination node receives the PREQ message, it checks its freshness in the same way as other intermediate nodes, then it updates its reverse path towards

the source. The destination then formulates an RREP (Route Reply) message which is afterwards uni-casted towards the source. Intermediate MPs receiving the RREP message update their forward path towards the destination, update the routing metric, and then forward the updated RREP towards the source.

In case of node failure, the intermediate MPs detecting the failure send an RERR (Route Error) message towards the source to inform it about the breakage of the link. The source can then re-initiate a new route discovery using a new PREQ message.

Proactive Routing

In the proactive mode, the sink periodically broadcasts PREQ messages bearing unique sequence numbers. The protocol is very similar to reactive routing, presented in the last section, in terms of how intermediate nodes react and process PREQ messages. The only differences are:

- In reactive routing, PREQ messages are sent in a reactive way (i.e., when needed), whereas in proactive mode, PREQ messages are sent proactively (i.e., in advance) and on a periodic basis (e.g., every 1 s).
- In reactive routing, PREQ can be sent by both types of nodes (i.e., sensor and sinks), whereas in proactive mode, PREQ messages are sent only by sinks. These latter represent the trees roots. The intermediate sensors that serve as routers on behalf of each other do only forward PREQ messages.

By having every sink periodically broadcasting PREQ messages, tree-like topologies are built with sinks as roots and sensors as tree nodes. By comparing the routing metrics of the different PREQ messages, sensors select the best sink and thus adhere to the tree whose root is the selected sink.

The proactive protocol is ideal for the scenario where WSNs exhibit a tree-like topology where all the traffic is directed towards/from the sinks that connect the WSNs to the Control Plane. Next section highlights the most known routing algorithms in WSNs.

3.2 WSN Routing Protocols

WSNs are wireless multi-hop networks. However the routing protocols should account for the stringent constraint of power consumption, thus the existing multi-hop routing protocols need to be adapted to cope with the constraint.

Generally the WSNs routing protocols can be classified into two main categories: 1. Hierarchical routing and 2. Location-based routing.

Hierarchical Routing

In hierarchical routing, the sensors are hierarchized to form clusters, and each cluster elects a node to be the head. The cluster head is responsible for aggregating, processing, and forwarding the data, while other sensors are only responsible for sensing the data. The main point here is that the elected head should have enough

(high) energy levels. The head role can be circulated among all sensors in order to balance the energy levels at the different sensors.

The most well-known routing protocol in this category is the LEACH (Low Energy Adaptive Clustering Hierarchy) protocol [8]:

- LEACH uses a randomized rotation for the selection of the Cluster Head (CH). The sensors use a probabilistic approach to decide on when to nominate themselves as CHs, and the decision is made by each node independently of other nodes, and this to minimize the overhead of communication and coordination between the different nodes, a fact that would result in energy loss due to the exchange of further packets between the nodes. The CH determination function depends on the ratio of CHs in the network, the frequency of successful election for the nominated nodes, and the time it was last elected.

Other variations of LEACH have been proposed, e.g., TL-LEACH [9] and MELEACH-L [10]:

- TL-LEACH (Two-level Hierarch LEACH) proposes two levels of hierarchies in contrast to LEACH which uses just one. These levels are *primary* and *secondary*. The sensors only communicate with the secondary heads. These latter communicate only with the primary heads. The main goal behind this two-level hierarchy is to minimize the number of nodes that need to communicate with the control plane as only the primary heads are allowed to.
- MELEACH-L is a further adaptation of More Energy-efficient LEACH (MELEACH [11]) to cope with the specificities of large-scale WSNs by controlling the size of each cluster and separating the cluster heads from the backbone nodes. MELEACH-L solves the problems of the channel assignment among neighbor clusters and the cooperation among cluster heads during data collection by having node switch its channel among a set of 20 channels ranging from Channel 0 to Channel 19. Channel 0 is used as the common broadcast channel on which the different sensors and heads can coordinate and synchronize.

There are other protocols different from the LEACH-flavored ones, e.g., TEEN [12], APTEEN [13], EECS [14, 15], HEED [16], and PEGASIS [17]. Next, we briefly outline the main characteristics of some of these protocols.

- Power-efficient Gathering in Sensor Information Systems (PEGASIS) [17] is chain-based protocol developed to enhance LEACH protocol and improve the lifetime of WSN. The main goal of PEGASIS is to extend the network lifetime. To do so, each node needs to communicate only with its closest neighbor (avoiding the use of long range transmissions) and they take turns to communicate with the base station. PEGASIS assumes that each node knows the location's information of other nodes at the beginning. In PEGASIS, nodes use the received signal strength to measure the distance to all neighboring nodes and then they adjust their signal strength to reach only the closest node. The sensor nodes take turns in being the sinks. It uses local coordination among the nodes on choosing the closest nodes and changing the sink node to increase the lifetime of all nodes.

Simulation results showed that PEGASIS is able to increase the lifetime of the network twice as much the lifetime of the network under LEACH protocol. Its main drawback here is that it assumes that each node can reach the base station directly which is not practical in many cases. In addition, PEGASIS introduces excessive delay for distant nodes in the chain.

- TEEN (Threshold-sensitive Energy-Efficient sensor Network protocol) and APTEEN (Adaptive Periodic Threshold-sensitive Energy-efficient sensor Network protocol) [12, 13] are two hierarchical protocols developed for time sensitive applications. Sensor Nodes limit the transmission frequency but are actively probing the medium and keep track of threshold information sent by the cluster head. Nodes send sensed data only when the threshold is within the time delay requirement. TEEN introduces two thresholds to the LEACH mechanism; a Hard Threshold (H_T) that defines the acceptable range for a sensed value and a Soft Threshold (S_T) to measure the change on the sensed value. A node will report data only when the absolute sensed value exceeds the hard threshold bounds or when the change in the sensed data is bigger than S_T. The main drawback of TEEN is that in the occurrence of a failure to deliver the thresholds to the nodes in the field, the nodes will never sense and will never communicate which means the user does not receive any data from the network. TEEN is highly suitable for time critical sensing applications and since message transmission consumes more energy than sensing operations, the energy consumption will be less than that in proactive networks.

Maximum Energy Cluster Head (MECH) [18]: Is a proposed routing protocol that presents some enhancements over LEACH protocol; first, MECH constructs clusters based on radio range with a defined maximum number of members on each cluster in order to avoid having widely spread or excessively populated clusters. This creates a cluster topology more equally distributed. In addition, using a hierarchical tree routing mechanism, MECH reduces the distance of transmission from the cluster heads to the base station.

Hierarchical Power-Aware Routing (HPAR) [19]: This protocol divides the network into groups of sensors. Groups are formed based on their geographic proximity and clustered together as a zone and each zone is treated as an entity. Messages are routed along the path which has the maximum remaining power over the entire minimum remaining, called the max–min path. HPAR's goal is to minimize the total power consumption and maximize the minimal residual power of the network. The protocol uses Dijkstra's algorithm to find the path with the least power consumption and then finds a path that maximizes the minimal residual power in the network. The routing algorithm in this protocol selects the route with the best combination of both parameters.

Reliable Energy-Aware Routing Protocol (REAR) [20]: REAR establishes routing paths considering the residual energy capacity of each sensor node. By supporting multi-path routing, REAR can change from current optimum path to an alternative second routing path when sensor node's energy capacity on the current routing path is degraded under a given threshold value. Route discovery in REAR works as follow:

A source node broadcasts a Multi-path Route Request message (MREQ) to find a routing path for a destination node, after receiving the MREQ message, nodes continuously forward it after waiting for a forwarding delay according to their energy level. (Nodes with low energy levels wait longer to retransmit the MREQ packets.) Due to this approach, source nodes will find first about those routes with high energy values and they will establish routing paths using these high energy nodes. In addition, REAR allows each sensor node to confirm success of data transmission by providing a DATA-ACK-oriented packet transmission.

Location-Based Routing
In location-based protocols nodes are addressed by their location where distance to next neighboring nodes can be estimated by signal strength or by GPS receivers. In most cases location information is needed in order to calculate the distance between two particular nodes so that energy consumption can be estimated. Since, there is no addressing scheme for sensor networks like IP-addresses and they are spatially deployed on a region, location information can be utilized in routing data in an energy-efficient way.

- Geographical and Energy-Aware Routing (GEAR)
  GEAR [21] uses the geographic information to disseminate queries to approximate regions. It uses energy-aware and geographically informed neighbor selection to route a packet toward the destination region. It follows the query-response model. This routing protocol assumes that each node knows its location, energy level, and its neighbors' locations and energy levels. Since GEAR is a location-based routing, each sensor node will require localization hardware, such as GPS (Global Positioning System).

  In GEAR, each node keeps an estimated cost and a learning cost of reaching the destination through its neighbors. The estimated cost is a combination of residual energy and distance to destination. The learned cost is a refinement of the estimated cost that accounts for routing around holes in the networks. A hole occurs when a node does not have any closer neighbor to the target region than itself. If there are no holes, the estimated cost is equal to the learned cost. The learned cost is propagated one hop back every time a packet reaches the destination so that route setup for next packet will be adjusted.

  There are two phases in the algorithm:

  – Nodes forward packets towards the target region; upon receiving a packet, a node checks its neighbors to see if there is one neighbor that is closer to the target region than itself. If there is more than one, the nearest neighbor to the target region is chosen as the next hop.
  – Once the packet reaches the targeted region, nodes forward the packets within the region by either recursive geographic forwarding or restricted flooding.

- GEDIR (The Geographic Distance Routing) [22] and its variants [23, 24] of routing protocols use location, distance, progress, and directions methods in their routing algorithms. GEDIR is a greedy method that always selects the node closest to destination. DIR technique selects the best neighbor that is within the

minimum angular distance (i.e., direction) to the destination. The speed of data is adapted based on next hop selection.

Cross-Layer Routing Approach

Cross-layer design is another paradigm for optimum energy-efficient strategy in WSN. The central idea of cross-layer design is to optimize the control and exchange of information of the routing layer using two or more other layer's related information in the OSI model to achieve routing performance improvements. This will exploit interactions among various protocol layers [25]. For example, knowledge about channel conditions using PHY and MAC layers, for instance, will enable routing to make optimized decision as shown in Fig. 5.

Cross-Layer Routing Protocol for Wireless Sensor Networks (XLRP) [26] proposed a protocol that supports different transmitting power levels based on the volume of the data being transmitted as an energy-efficient mechanism. It uses information from the application and the physical layers in determining an efficient network path.

XLRP protocol different from conventional cross-layer design in four aspects:

- It relies on the information from the application and physical layer.
- It supports transmission at varying power levels, XLRP switch the transmission power level based on the volume of the data being transmitted.
- It saves energy by switching OFF unintended receivers based on the power of the received radio signal.
- It shifts from beacon based approach to piggy-packing approach to reduce the control overhead of the protocol.

Integrated Routing and Interest dissemination System (IRIS) is a cross-layer network protocol for WSN [27]. It is an integrated data gathering and interest dissemination system. It is an efficient convergecasting protocols, i.e., actively sends data to the sink, and supports sleep scheduling and interest dissemination. This protocol was implemented in TMoteSky test-beds and showed a positive experimental behavior [28].

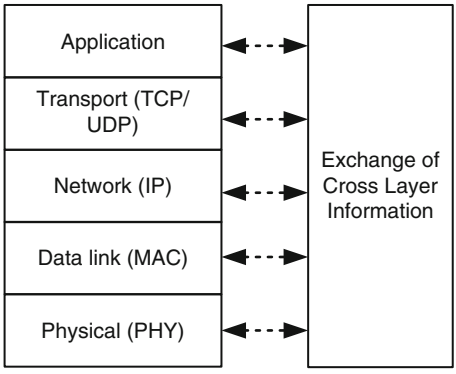

Fig. 5 Cross-layer information exchange

Self-Organizing WSNs

Self-Organizing Network Services with Evolutionary Adaptation [29]: Proposed a framework for developing adaptive and scalable network services using a Self-Organizing and Evolutionary-based approach. The authors presented an approach where Network Services are viewed as groups of autonomous agents that interact with each other in the network environment. These agents are capable of simple behavior (replication, migration and death) and they use an evolutionary adaptation mechanism based on Genetic Algorithms (GAs) to evolve their behaviors and improve their fitness values (e.g., response time to a service request). This approach was tested using simulation and the results demonstrated the ability of autonomous agents to adapt to the network environment. This protocol may be suitable for disseminating network services in dynamic and large-scale networks where a large number of data and services need to be replicated, moved, and deleted in a decentralized manner.

Hierarchical On-Demand Routing for Self-Organized Networks [30]: Presents a Hierarchical On-Demand Routing Protocol enhanced with the characteristics of the Self-Organized Systems. This protocol divides the network into areas, according to the tightness of the cooperative work between nodes. All nodes in a given area share the same wireless channel which means that transmission occurring in one area does not interfere with other areas' transmission. The network is divided into two area types: common areas and a backbone area. A node in each common area is designed as the gateway node, and it acts as a bridge between common and backbone areas. This gateway node has two channels with which it can communicate with nodes in both common and backbone areas. The authors proposed two different routing protocols: an intra-area routing protocol and an inter-area routing protocol. The intra-area routing protocol allows nodes to discover and maintain a route to each other node in the same area, including the area's gateway node. This protocol includes hop-by-hop acknowledgements, handles retransmissions, route advertisement, and gateway advertisements.

The inter-area routing protocol provides a mechanism to make every gateway knows the node information in its own area and to exchange this information with other areas' gateways by using advertisement packets. The approach was compared with Ideal Distance-Vector Routing Protocol (IDVR), Ideal Link-State Routing Protocol (ILSR), and Flooding Protocol (FR), where the HOR approach performed the worst for Packet delivery ratio and end-to-end delay, but proved that the protocol introduced less overhead in the network than IDVR, ILSR, and FR.

A Self-Organizing Network based on Naturally Occurring Structures [31]: The authors presented a real-life application for Self-Organizing Wireless Networks using Adaptive Social Hierarchy. In their research they show how the biological principle known as the social dominance hierarchy can be applied to Wireless Networks to form hierarchies based on each individual energy level. In social dominance hierarchy individuals in a group have a hierarchy level based on measurable characteristics such as their size or weight (as it occurs naturally in many species). The goal of this protocol is to extend the network's lifetime by routing traffic more actively through those nodes with a large amount of energy. The simple

rule in this protocols states that a node should deliver its data packets to any higher-rated node that it encounters. At the same time, nodes on the network adjust and update their level in the hierarchy by periodically informing their current energy level. This protocol uses a very common structure in Animal Kingdom applied to Communication Networks, resulting in a system that scales well to large numbers of nodes and uses no centralized control.

Energy-Aware Routing Algorithms

Routing protocols described in the previous sections were categorized based on network structures and protocols used. Regardless of the structure or the protocol, the routing engine in a node will need to run an algorithm that will populate the forwarding table. These algorithms need to select the next node that would minimize the energy consumed by that node as well as the subsequent nodes in the network. The following section discusses two techniques based on determining the next hop using fittest note selection and fuzzy logic based algorithms.

A. Fittest node selection

The salient parameters that define the fitness of a node are: battery level, hierarchy level, and link distance. The fitness score is calculated by each node in a localized manner, for example: Node A is would like to determine the next hop, nodes B, C, and D are within A's range. Node A calculates a fitness score for each one of its neighbors (B, C, and D), and based on the score, node A selects the next hop router. Note that the fitness scores for B, C, and D are calculated by node A and this Fitness score might differ from another node's point of view. These parameters are designed during network design and each is given a value that may change.

Nodes follow a set of simple rules that defined their behavior:

1. Calculate the fitness score of the neighboring nodes (see equation below) and determine the fittest nodes.
2. Randomly select one of the "**fittest**" node.
3. Update the hierarchy level based on the selected node level plus 1.
4. If router node notifies a change in its "fitness score" or if an acknowledgement was not received, execute rule number 1.

Each node executes these rules considering only the neighbor nodes, regardless of the overall behavior of the system. Nodes can publish their information within data packets or control packets. This information is sent in the form of an array of bits (encoded in two bytes), to reduce the size of the packets being sent. For instance, when Node A hears a packet sent by node B, it opens the packet and reads a 8-bit field that contains the hexadecimal value 9D (8 bits), and it converts this number into its binary representation as seen in Fig. 6.

The representation of this array of bits will be as follows (from right to left):

1. **Mode**: The mode indicates if a sensor can or can't be a router; a node will change its mode to 0 when its battery level is "down," it can still sense and send information but it can't route.

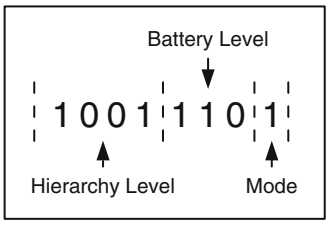

Fig. 6 Binary representations of a node's characteristics

2. **Battery level**: It indicates the amount of energy left in the node's batteries. If the battery level is high, the probability of being selected as router will be higher.
3. **Hierarchy level**: Is the position in the hierarchical tree of the node, where level 0 is sink node and level 15 means unconnected (a node that has not found a router yet). A node with a low number in the hierarchy level is more attractive to other nodes, because it is closer to the base station.

In addition, both data and broadcast packets contain two fields with node's x and y coordinates in the field, this information is used by nodes to calculate its distance from the sender of the packet.

B. Computation of the fitness score

Using battery level, distance, hierarchy, and mode nodes calculate the fitness scope using the following function:

$$\text{Fitness Score (FS)} = \left[(B \times \text{Bw}) + \left(\frac{\text{Dw}}{D} \right) + \frac{\text{Hw}}{H} \right] \times \text{MODE}$$

where B is battery level and Bw is weight of the battery level parameter, D is distance and Dw is weight of the distance parameter, H is hierarchy level and Hw is weight of the hierarchy level parameter.

The selection of the weight will depend on the design of the network and the expected parameters that will influence the Fitness Score (FS). Let us consider the "ideal router node" as example to understand the fitness score estimation and the role of parameter weights. For a node that is computing FS, the ideal router node will have full battery level, the lowest hierarchy level and will be at the closest distance possible (anything below 50 m, which is a number decided by the designer) from the computing node.

Applying the parameter values for the ideal router node, we get

$$\text{Fitness Score (FS)} = [(7 * \text{Bw}) + (\text{Dw}/50) + (\text{Hw}/1)] \times \text{MODE}$$

Now we can define the values for weights based on the required influence of each parameter on the overall fitness score. In our network, since the main concern is energy conservation, we consider the following parameter influences on the final FS: 60 % for battery level, 20 % each for distance and the hierarchy level.

Table 1 Fitness scores for node A's neighboring nodes

Node	Battery level	Distance	Hierarchy level	Mode	Fitness score
B	6	60	9	1	66
C	6	90	9	1	61
D	4	65	8	1	57

Assuming that the perfect fitness score will be equal to 100 points, we have to find the values for Bw, Dw, and Hw that will achieve the weight distribution, i.e. if the FS of an ideal router is equated to

$$100 = [(7 * \mathrm{Bw}) + (\mathrm{Dw}/50) + (\mathrm{Hw}/1) * 1]$$

(7*Bw) should comprise 60, and the other two components, (Dw/50) and (Hw/1), should account for 20 each satisfying the equation and the decision criteria. Thus, the values Bw = 7.14, Dw = 1,000, and Hw = 20 will satisfy the equation, but to ease the calculation process, Bw is round up at 8, then the following equation resulted:

$$96 = [(7 * 8) + (1,000/50) + (20/1) * 1]$$

The algorithm was simulated using OMNET++ and the results were published in [33]. The originally selected values (Bw = 8, Dw = 1,000, and Hw = 20) showed an average remaining battery level of 250 J and a standard deviation of 298. The weights in the formula give flexibility to the routing protocol. For example, if there is an application where nodes are mobile, mobility could be included among the decision parameters, the weight for mobility can be increased, this will increase the "influence" of mobility parameter in the fitness score and those nodes with the lowest mobility will become the most attractive carriers. Or if in a given application, the nodes do not depend on a battery, the weight of the battery level can be reduced to lower its influence on the overall fitness score calculations. This allows us to use the same natural route selection method for different scenarios and applications with careful choice of parameters and the corresponding weight distribution. Once the fitness score is calculated nodes, use the Roulette wheel method to select the next hop node.

– The Roulette wheel method

The Roulette wheel method is a biased random selection procedure based on the fitness score of each node. For example, given that node A has already calculated the fitness scores for nodes B, C, and D with the following results (see Table 1).

Then, an imaginary roulette wheel is constructed with a segment for each individual in the population. Figure 7 shows that the size of the segment is based on the fitness score for each node.

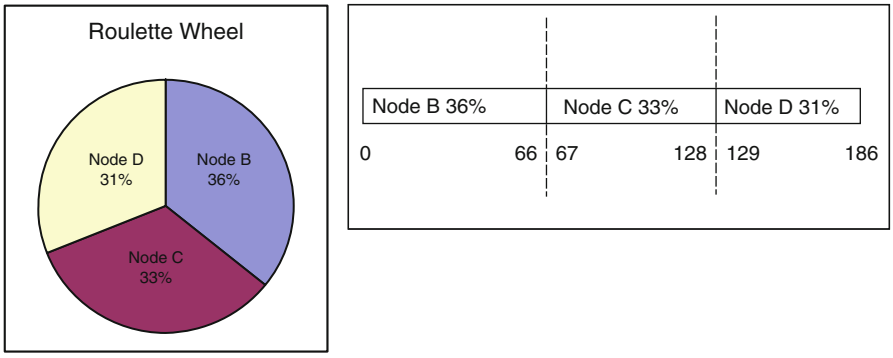

Fig. 7 Roulette wheel with two different views

It can be seen in Fig. 7 that node A can select one of the nodes by generating a random number between 0 and 186. With this approach the node with the highest fitness score has a higher probability of being selected. Using random selection instead of just selecting the one with the highest fitness score diminishes the likelihood of a node being selected as a router node for all the nodes within the same range and this would highly increase its battery consumption. The fitness score calculation and the random selection process enable to network to self-adapt. The following section describes the self-adaption process.

C. Self-adaptation

The fitness score provides an indication about the "weak" and the "strong" nodes. The parameters are not static and change constantly through time. In addition, each node has the capability of modifying its own parameters in a selfish manner, trying to save energy or fighting to survive.

Three parameters should be considered as self-modifiable:

1. The power used in transmission: If a node A has selected a router node, it now can adapt its transmission range to only use the necessary power to reach its parent node.
2. The hierarchy level: When node A selects its router node it will now have a hierarchy level equal to its router node plus one.
3. The Mode: This is a value that should only be modified when a node is running out of battery and its depletion is close. A node with its mode value set as zero will not be selected as parent node by other nodes.

Nodes Operation

All nodes have different IDs but are programmed with the same code. The base station node has a node ID of 1 to differentiate its behavior from the rest of the nodes in the network. When a node is turned on it verifies its node ID to see if it is a base station as it shown in Fig. 8.

If a node checks its own ID and it is equal to 1, the node sends a broadcast packet with its information. This packet triggers the beginning of the whole operation in the network. Then it has to wait in listening mode for data packets. Because the base station node usually has an unlimited source of power, energy should not be a concern and receiving packets should be its main task. On the other hand, non-base station node will start listening as shown in Fig. 9.

The node should be in promiscuous mode until it receives a packet. When a packet is received, the node reads the *status* field of the received packet; this field contains the parameters of the host that sent the packet. Using a simple modulo-2 operation, the node determines the Mode of the sending node which enables a node to determine whether the sender could be a router node. Therefore, if the result of this operation is 0, the packet should be discarded and the node should remain in promiscuous listening until another packet arrives (see Fig. 10).

After checking that the sender is available to route packets, those fields in the packet related to the sender's position (*x* and *y* fields) are then read. To continue, the node has to calculate its distance to the sender by using the Pythagorean Theorem with its own position and the information gathered from the received packet. The rest of the information needed to calculate a node's fitness score is contained in the *status* field of the received packet, and the number in this field has to be converted to its binary representation (described above). Having the information about the sender's *battery level*, *hierarchy level,* and *distance*, the node can use the fitness score equation to calculate the sender's fitness score (see Fig. 11).

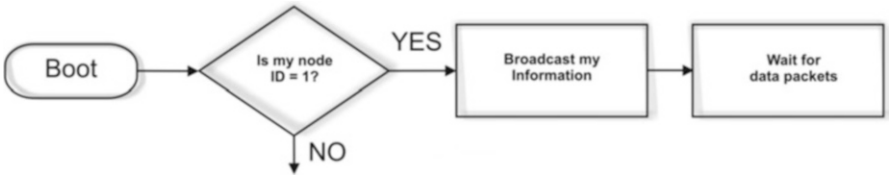

Fig. 8 Flow diagram for base station node's booting

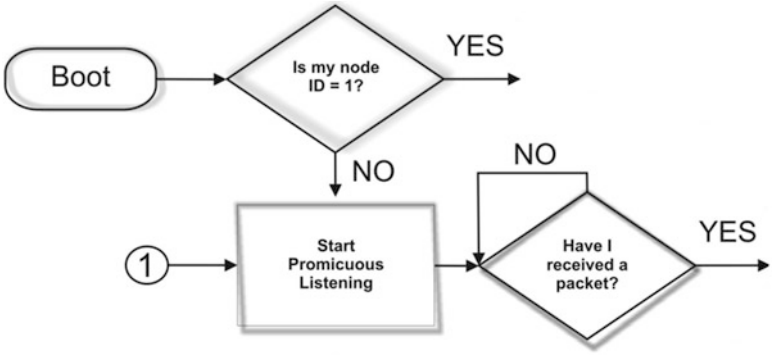

Fig. 9 Nodes operation after booting

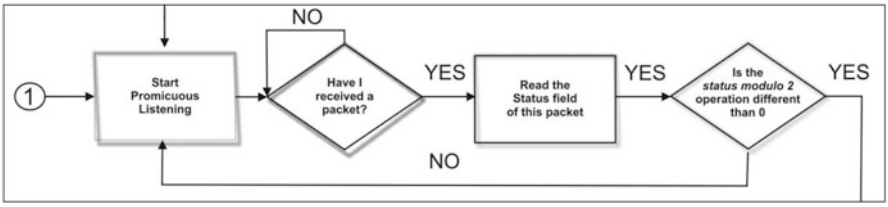

Fig. 10 Status field and the modulo 2 operation flowchart

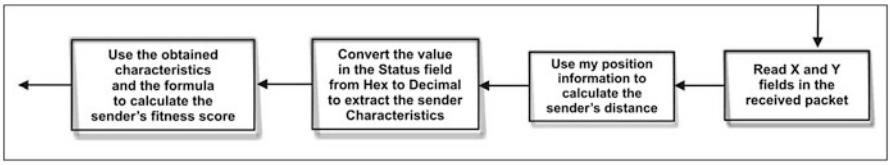

Fig. 11 Values and distance flowchart

After obtaining the characteristics of the sender, the node has to determine the relationship with this neighbor. If the packet was originated by its router node, the packet will be discarded and the node will return to steady state operation. But if the packet was not originated by its current router node and if its hierarchy level is not max (15), the sender's fitness score will be compared with the current selected router and the packet will only be used if the sender's fitness score is bigger than the current router node, if the fitness score its lower or equal, the packet will be dropped. This procedure is described in Fig. 12.

If the sender can be used as a possible router, the sender information will be added to a linked list that contains the neighbors that could be selected as a router node. This linked list contains the following fields: neighbor node ID, Fitness score, distance, fitness score. Once the information has been included in the linked list, the node has to generate a random number between 0 and the last node's fitness score using the roulette method described above (Fig. 13).

Now that a router node has been selected, the node adjusts its hierarchy level and reports the level to its neighbors, as described in Fig. 14. The node will enter to a steady state operation to perform its regular functions of sensing, computing, and communicating the information to the sink.

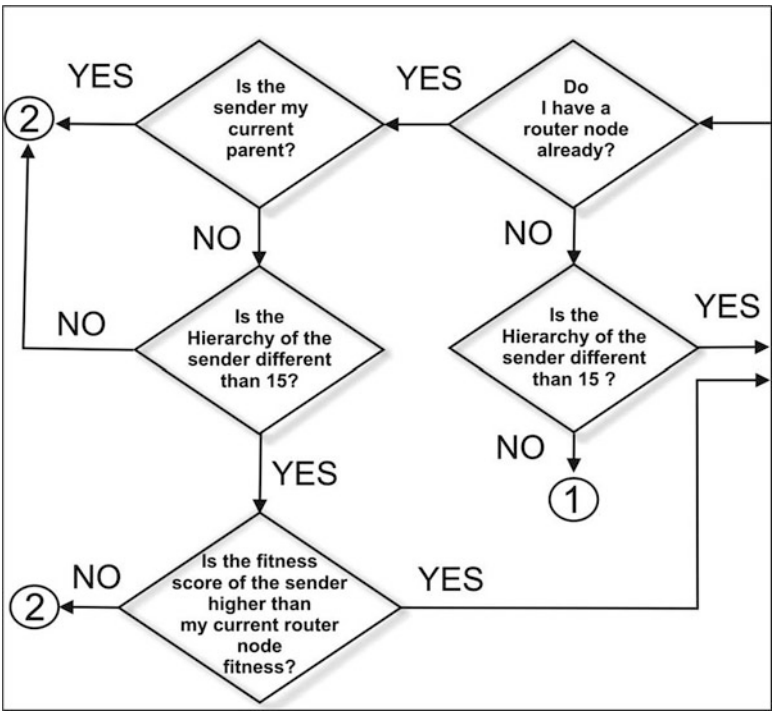

Fig. 12 Connections, current parent and hierarchy testing

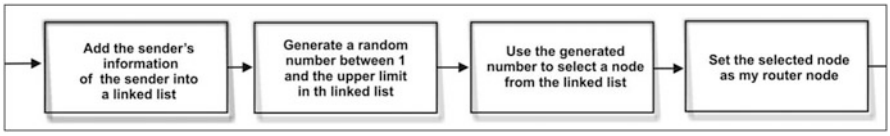

Fig. 13 Linked list and node selection

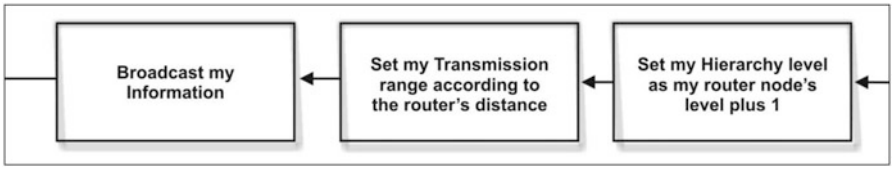

Fig. 14 Setting hierarchy, transmission level, and broadcasting

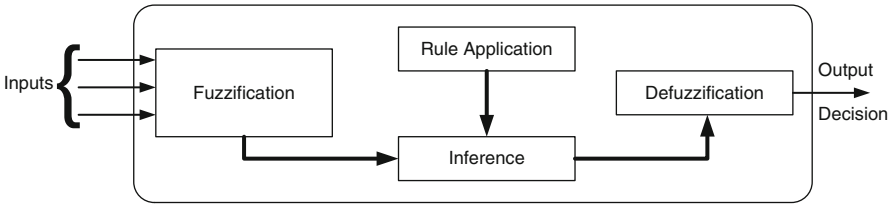

Fig. 15 Fuzzy logic controller

D. Fuzzy logic

Another way to enable the routing engine to select the next hop is through fuzzy Logic [34]. Fuzzy logic is a decision-making tool that is used to help in making with multiple input parameters and makes an efficient trade-off between significance and precision. In this algorithm, instead of using the fitness equation, the system uses a fuzzy logic to make a decision. The Fuzzy Logic Algorithm has a powerful capability of handling uncertainty and ambiguity. The four basic elements of any fuzzy logic controller are shown in Fig. 15, they are the fuzzifier, inference engine, Fuzzy Rule Base (FRB), and defuzzifier, where the process is performed in these four steps. Further description could be found in reference [34].

4 Conclusion

This chapter presented WSNs routing protocols and proposed an appropriate architecture that further eases the WSNs deployment, especially for space applications. Different routing protocols have been reviewed and presented. Routing protocols play a key role in energy efficiency and different techniques that enable energy efficiency have been presented. The chapter introduced an interesting concept of multigateway architecture where a WSN has multiple paths to different gateways to deliver the data. A routing protocol based on natural selection and self-organizing and self-adapting network has been presented in detail. This protocol will enable nodes to select the closest gateway regardless of the number of gateways in the network.

References

1. Akyildiz I, Su W, Sankarasubramaniam Y, Cayirci E (2002) Wireless sensor networks: a survey. Comput Netw 38(4):393–422
2. IETF, Mobile Ad-hoc Networks (MANET) Working Group. http://datatracker.ietf.org/wg/manet/charter/. Accessed Jan 2015

3. Conti M, Giordano S (2007) Multihop ad hoc networking: the theory. IEEE Commun Mag 45(4):78–86
4. Conti M, Giordano S (2007) Multihop ad hoc networking: the reality. IEEE Commun Mag 45(4):88–95
5. Satyajayant M, Guoliang X, Dejun Y (2012) Smart grid – the new and improved power grid: a survey. IEEE Commun Surv Tutorials 14(4):Fourth Quarter
6. Gungor VC, Sahin D, Koçak T, Ergüt S, Buccella C, Cecati C, Hancke GP (2011) Smart Grid Technologies: communication technologies and standards. IEEE Trans Ind Inf 7(4):529–539
7. Atzori L, Iera A, Morabito G (2010) The internet of things: a survey. Comput Netw 54(15):2787–2805
8. Perkins CE, Royer EM (1999) Ad-hoc on-demand distance vector routing. In: WMCSA
9. Heinzelman W, Chandrakasan A, Balakrishnan H (2000) Energy-efficient communication protocol for wireless microsensor networks. In: Proceedings of the 33rd Hawaii international conference on system sciences (HICSS'00), January 2000
10. Loscri V, Morabito G, Marano S (2005) A two levels hierarchy for low energy adaptive clustering hierarchy (TL-LEACH). In: Proc. IEEE 62nd vehicular technology conference
11. Chen J, Shen H (2008) MELEACH-L: more energy-efficient LEACH for large-scale WSNs. In: Proceedings of 2008 4th international conference on wireless communications on networking and mobile computing, Dalian, October 2008, pp 1–4
12. Chen J, Shen H (2007) MELEACH an energy-efficient routing protocol for WSNs. Chin J Sens Actuators 20:2089–2094
13. Manjeshwar A, Agarwal DP (2001) TEEN: a routing protocol for enhanced efficiency in wireless sensor networks. In: First international workshop on parallel and distributed computing issues in wireless networks and mobile computing
14. Manjeshwar A, Agarwal DP (2002) APTEEN: a hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks. In: Proceedings of the 16th international parallel and distributed processing symposium, pp 195–202
15. Ye M, Li C, Chen G, Wu J (2005) EECS: an energy efficient clustering scheme in wireless sensor networks. In: Proceedings of the 24th IEEE international performance, computing, and communications conference (IPCCC), Phoenix, AZ, USA, 7–9 April 2005, pp 535–540
16. Ye M, Li C, Chen G, Wu J (2006) An energy efficient clustering scheme in wireless sensor networks. Ad Hoc Sens Wireless Netw 3:99–119
17. Younis O, Fahmy S (2004) HEED: a hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks. IEEE Trans Mobile Comput 3:366–379
18. Lindsey S, Raghavendra C (2002) PEGASIS: power-efficient gathering in sensor information systems. IEEE Aerospace Conf Proc 3(9–16):1125–1130
19. Chang R, Kuo C (2006) An energy efficient routing mechanism for wireless sensor networks (MECH). In: Proceedings of the 20th international conference on advanced information networking and applications, vol 2. April 2006
20. Li Q, Aslam J, Rus D (2001) Hierarchical power-aware routing in sensor networks. In: Proceedings of the DIMACS workshop on pervasive networking, May 2001
21. Shin K, Song J, Kim J, Yu M, Mah P (2007) Reliable aware routing protocol for wireless sensor networks (REAR). In: 9th international conference on advanced communication technology, February 2007
22. Yu Y, Estrin D, Govindan R (2001) Geographical and energy-aware routing: a recursive data dissemination protocol for wireless sensor networks. In: UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, May 2001
23. Watanabe M, Higaki H (2007) No-beacon GEDIR: location-based ad-hoc routing with less communication overhead. In: Fourth international conference on information technology (ITNG'07), 2–4 April 2007, pp 48–55
24. Takehira T, Higaki H (2012) IRDT-GEDIR: shorter delay wireless multihop routing in sensor networks. In: 4th international congress on ultra-modern telecommunications and control systems and workshops (ICUMT), 3–5 October 2012, pp 857–863

25. Takehira T, Higaki H (2012) IRDT-GEDIR: next-hop selection in intermittent wireless multihop sensor networks. In: 9th international conference on ubiquitous intelligence & computing and 9th international conference on autonomic & trusted computing (UIC/ATC), 4–7 September 2012, pp 894–899
26. Fu B, Xiao Y, Deng H, Zeng H (2014) A survey of cross-layer designs in wireless networks. IEEE Commun Surv Tutorials 16(1):110–126
27. Gunasekaran R, Hairong Q (2008) XLRP: cross layer routing protocol for wireless sensor networks. In: Proceedings of the wireless communications and networking conference (WCNC), March 31–April 3, 2008, pp 2135–2140
28. Camillò A, Nati M, Petrioli C, Rossi M, Zorzi M (2013) IRIS: integrated data gathering and interest dissemination system for wireless sensor networks. Ad Hoc Netw 11(2):654–671
29. Camillo A, Petrioli C (2012) Hands on IRIS: lessons learned from implementing a cross layer protocol stack for WSNs. In: Proceedings of IEEE global communications conference (GLOBECOM)
30. Nakano T, Suda T (2005) Self-organizing network services with evolutionary adaptation. IEEE Trans Neural Netw 16(5):1269–1278
31. Chun Y, LeiMinga X, MeiLin S (1999) Hierarchical on-demand routing for self-organized networks. In: Fifth Asia-pacific conference on communications and fourth optoelectronics and communications conference, October 1999
32. Markham A, Wilkinson A (2006) The adaptive social hierarchy – a self organizing network based on naturally occurring structures. In: 1st bio-inspired models of network, information and computer systems, December 2006
33. Bernal Velazquez CF, Benhaddou D, Balakrishnan M, Anan M (2011) Energy efficient data dissemination in wireless sensor networks based on natural selection. In: 7th international in wireless communications and mobile computing conference (IWCMC), 4–8 July 2011, pp 577–582
34. Jaradat T, Benhaddou D, Balakrishnan M, Al-fuqaha A (2013) Energy efficient cross-layer routing protocol in wireless sensor networks based on fuzzy logic. In: 9th international wireless communications and mobile computing conference (IWCMC), 1–5 July 2013, pp 177–182
35. Benhaddou D, Balakrishnan M, Yuan X, Chen J, Rungta M, Barton R, Yang H (2009) "Wireless Sensor Networks for Space Applications: Network Architecture and Protocol Enhancements," in Sensors and Transducers journal Vol.7, Special Issue "MEMS: From Micro Devices to Wireless Systems", pp. 203–212

Middleware Architecture in WSN

Mehdia El Khaddar Ajana, Hamid Harroud, Mohammed Boulmalf, and Mohammed Elkoutbi

Abstract Sensors integrated into the environment, machinery, and structures, and coupled with the efficient delivery of sensed information could provide tremendous benefits in a wide range of applications such as improved manufacturing productivity, enhanced homeland security, fewer catastrophic failures, and improved emergency response. The design and development of these applications should address the challenges dictated by Wireless Sensor Network (WSN) characteristics on the one hand and the targeted applications on the other hand. One of the novel emerging approaches used to address these challenges is the design of middleware for WSN. Middleware refers to distributed software that can bridge the gap and remove impediments between the heterogeneous hardware platform and the backend applications requirements. In recent years, research has been carried out on WSN middleware from different aspects and for different purposes. WSN can be used with other identification technologies such as Radio Frequency Identification (RFID). In an integration system of RFID and WSN, RFID is used to identify objects while WSN can provide context environment information about these objects. This integration increases system intelligence in pervasive computing. This chapter provides a comprehensive review of the existing middleware for WSN, seeking for a better understanding of the current issues and future directions in this field. It also examines the various approaches of middleware design, compares and suggests different types of applications where each approach can be used. Finally, it proposes an enhanced middleware framework; FlexRFID for the integration of RFID and WSN.

M.E. Ajana (✉) • M. Elkoutbi
SI2M Lab, ENSIAS, Rabat, Morocco
e-mail: mehdia.ajana@gmail.com

H. Harroud
WML Lab, Alakhawayn University in Ifrane, Ifrane, Morocco

M. Boulmalf
International University of Rabat (UIR), Rabat, Morocco

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_4

1 Introduction

Wireless Sensor Networks (WSNs) have special features and diverse applications. In order to satisfy the requirements of WSN, there is a need to provide adaptation functions in order to fill the gap between applications and network protocols. The adaptation functions should take into consideration the constrained resources of WSN while providing quality of service to applications. One approach to satisfy this adaptation is WSN middleware. A WSN is a wireless and resource constrained network in terms of bandwidth, computation, communication capabilities, and energy, which needs to support diverse applications simultaneously [1]. Among the applications supported by WSN we find: infrastructure monitoring, environment monitoring, object tracking, habitat monitoring, battlefield monitoring, and healthcare monitoring. WSN topology is also variable due to node mobility, radio range, routing possibilities, switching between sleep and active states, and depletion of energy. Developing applications for WSN is a tedious job as the developers need to meet the above numerous constraints. Designing a middleware is a novel approach which provides a generic interface that can be used by different applications to meet WSN constraints and get the required QoS parameters from the network [1].

Middleware refers usually to software that sits on top of the operating systems and network protocols and below the application level. It hides details of low-level hardware, facilitates application development and management, and satisfies application requirements [2]. There are some other issues which could be addressed by designing a middleware: resource management at the middleware level is more easy and flexible compared to the OS level and application level, adding security features is also more appropriate at the middleware level supporting multiple applications, integration of WSN with other technologies and networks is possible with a middleware, and middleware can provide runtime environment for supporting and coordinating multiple applications [2].

WSNs have special requirements compared to traditional networks and distributed systems, and therefore require different designs of middleware solutions. In this chapter we examine the different middleware challenges, design principles, approaches and architecture for WSN, present a brief survey and comparison of the existing middleware for WSN, list the WSN application needs, investigate the integration of Radio Frequency Identification (RFID) and WSNs, and suggest a middleware solution for this integration called FlexRFID.

2 Background and Concepts

Sensors are small embedded sensing platforms with computing and communication capabilities, which combine low cost, flexible and fast deployment, resilient self-management and embedded intelligence for cooperatively delivered, value added services [3]. Such nodes are called sensor nodes and are a type of transducer that converts some physical phenomenon such as heat, light, and sound into electrical

signals. Each sensor node is capable of only a limited amount of processing. But when coordinated with the information from a large number of other nodes, they have the ability to measure a given physical environment in great detail [3].

Thus, a sensor network can be described as a collection of sensor nodes which co-ordinate to perform some specific action. Unlike traditional networks, sensor networks depend on dense deployment and coordination to carry out their tasks [3]. Sensors have seen early adoption in a number of application spaces. They can be used to instrument and monitor environments; track assets through time and space with respect to some workflow or process; detect changes in the environment defined to be of significance that humans are unable or are put at risk to perceive; control a system with respect to the environment within a defined range of changes; and adapt services to improve their utility [3].

Two adjacent technologies to WSN should be noted. First, RFID, which pioneered the "Internet of Things" concept, is a distinct technology whose purpose is to identify and track real-world objects. Second, *Information Management* is a major technology field that provides the means of making sense of the vast amounts of information which WSNs will inevitably produce. By integrating some data processing into the sensor nodes themselves, a first level of information management can be pushed into the network [3].

Sensors are networked like RFID tags; however, there exists a difference between them. RFID tags generate a fixed electrical signal, whereas sensor data may change because it is the output from a transducer that converts varying physical phenomena into varying signals. Also, the data processing is likely to be more complex for sensors than tags [3]. RFID is an automatic identification technology with ability to wireless communication (read and write data without direct contact) and without the necessity for line-of-sight. RFID is a method relying on storing and remotely retrieving data using devices called RFID tags or transponders. RFID data can read through the human body, clothing, and non-metallic materials. RFID is in use all around us. If you have ever chipped your pet with an ID tag, or paid for gas using Speed Pass, you've used RFID. In addition, RFID is increasingly used with biometric technologies for security [3].

Middleware refers to a layer that sits between the applications and hardware devices. It helps abstract the heterogeneity of the underlying computing environment and offers the capabilities to reuse software components on demand. It can offer a number of services that satisfy the non-functional requirements of an application such as scalability, reliability, dissemination, service discovery, efficiency, security, and privacy [4]. There exist many WSN middleware solutions that use different approaches in order to add the needed values and services required by the applications as described in Sect. 6.

3 Middleware Challenges for WSN

WSNs have unique characteristics that make them advantageous over traditional networks. They are easily deployable at a large scale, inexpensive, low power and

self-organizing. WSN were originally motivated by military applications and are emerging for a spectrum of new applications in recent years [5].

The middleware refers broadly to software that sits between the operating system and the enterprise information systems. The design and development of WSN middleware is not trivial and is associated with many challenges. It needs to deal with many challenges related to WSN characteristics on the one hand and the applications on the other hand [6].

In this section, we try to articulate the challenges associated with middleware for WSN.

3.1 Hardware Resources

A sensor node should accomplish three basic operations without resources exhaustion: sensing, data processing, and communication. The WSN middleware should be energy aware, providing mechanisms that enable lower power communication for an efficient use of the processor and memory. For example, the middleware can turn off most of the device's components depending on the application's needs, in order to save energy. WSN consists of tiny sensors with very small memory, computation power, and battery power. Middleware for sensor networks have to be lightweight to work under limited resource availability [5].

3.2 Changes in Network Topology and Size

Different factors such as device failure, malfunctioning, mobility, moving obstacles and interferences lead to frequent changes in network topology. The network should be scalable enough to allow the addition of nodes anywhere anytime without affecting network performance, depending on application needs. WSN middleware should therefore support mechanisms for self-configuration, self-maintenance of sensor nodes, and fault tolerance. WSN middleware must be scalable and adapt to network size extension in terms of number of nodes, number of users, etc. in order to operate over a long period of time [5].

3.3 Heterogeneity

WSN middleware should establish system mechanisms in order to interface with various types of hardware and networks, and also to meet the major challenge of bridging the gap between hardware technology's raw potential and the broad needed applications [5].

3.4 Data Fusion

WSN middleware should provide mechanisms to merge and synthesize raw data collected by various sensor nodes to a processed and high level format that is easily understandable by business applications. Communicating this processed data to backend applications is another major challenge for WSN middleware design [7].

3.5 Network Organization

WSN middleware should support the dynamicity of sensor networks and adapt to their changing environment. An efficient design of routing protocols is needed so that the sensor networks could support long running applications. Also, knowledge of the network is essential for a network to operate properly that is why an Ad hoc network resources discovery should be provided. For example, a sensor node needs to know its own location in the network in addition to the whole network topology. WSN middleware must also consider the limited resources of sensor nodes; energy, bandwidth, and processing power that are dynamically changing [5].

3.6 Application Knowledge

WSN middleware should include mechanisms for injecting application knowledge and requirements in the infrastructure of WSN. This will allow an adaptation of the network monitoring process to the application communication requirements. For example, the WSN middleware can get Quality of Service (QoS) requirements from the application and adapt the network configuration to provide the required QoS [5].

3.7 Quality of Service

WSN middleware should address the various QoS features; response time, availability, bandwidth allocation, etc. in order to ensure a reliable service to the applications. It should be designed based on trade-offs among performance metrics such as network capacity or throughput, data delivery delay, and energy consumption. QoS in WSN can be viewed from two perspectives: application specific that are related to the application such as sensor node measurement and coverage, and network specific which refers to how the supporting communication network can meet application needs while efficiently using network resources such as bandwidth and power consumption. QoS mechanisms for WSN are different from those used in traditional wired networks because of constraints such as dynamic topology and

resource limitations. So, WSN middleware should adapt itself when the required QoS and the state of the application change, and provide new mechanisms to maintain QoS over an extended period [4].

3.8 Security

Sensor networks are often deployed for sensitive applications like patient monitoring, military, and environment systems. Data collected and shared by these sensors should be secure, authentic, and tamper free [7]. In order to ensure this security, encryption algorithms are required to hide critical data, WSN monitoring services are required to protect from external attacks, and strong authentication and access control policies need to be deployed to ensure safe WSN functionality for the users [4].

However, most standard security mechanisms that consume a lot of resources cannot be implemented in sensor nodes due to their limited resource availability and low computational power. WSN middleware should therefore provide a security service for the collection and distribution of the sensors data depending on application domains and requirements, while maintaining acceptable power consumption and desirable network performance when necessary [7].

3.9 Integration with Other Systems

Most sensor network applications are using sensor or RFID networks along with their supporting software components that need to be integrated with enterprise or web systems. Collected data is useless unless it is given to its intended target and integrated to achieve the ultimate goal [5].

Middleware for WSN should provide generic and real-time services that allow the sensor and RFID networks to adapt to the changes and provide consistent data to various types of WSN applications that deal with real-time phenomena. This requires providing a set of interfaces that facilitate the integration and validate the exchanged information between the WSN and the external system [4].

4 WSN Middleware Design Principles and Approaches

There are basically four middleware functions for WSN as listed in [1]. The middleware should provide standardized system services in order to deploy diverse backend applications easily. It should also provide a steady environment that helps supporting multiple applications and creating new ones. WSN middleware should take care of the efficient utilization of system resources through the provision

of algorithms that dynamically manage the variable network resources of WSN. Finally, the middleware should be able to optimize the required network resources and satisfy the most of QoS dimensions. This can be achieved through negotiation between applications and low-level network protocols. In order to perform this, the WSN middleware needs to abstract application specific features as well as network protocols, and construct an effective mapping between them based on the current network status and the required QoS. This mapping can be seen as middleware services that are invoked by applications to manage network resources and provide the required QoS [1].

4.1 WSN Middleware Design Principles

As stated in [2] the key design principles for WSN middleware can be described as follows. The WSN middleware should support *data centric* communication since in WSN, the application is not interested in the sensor nodes themselves but rather in the data they sense. WSN middleware should consider *energy efficiency* and *resource management*; it should deal with restrained resources in terms of energy, memory, and size, in order to provide the required services while increasing the sensor nodes' life. The WSN middleware should allow for *scalability*; meaning that it should provide acceptable performance levels even if the network grows in the future. It should also support *in-network processing* by involving the sensor nodes in decision taking about how to operate a network based on the transmitted data. Examples of in-network processing are but not limited to data aggregation, data dissemination, encoding, and compression. WSN applications have more QoS requirements than traditional networks applications, because the sensor nodes work collaboratively, process each other's data, and need to be aware of the data that they are forwarding. Therefore, WSN middleware should support in addition to *network level QoS* metrics such as throughput and delay, *application level QoS* metrics like accuracy, deployment, reliability, and accuracy. WSN middleware should also consider the dynamic WSN network organization in order to adapt to the frequent changes in WSN networks topology. WSN middleware should be designed taking into consideration the *heterogeneity of hardware technology* and allow for reconfiguration, execution, and communication with these devices, while providing mechanisms to inject application knowledge into WSN's infrastructure [2].

4.2 WSN Middleware Approaches

There are different middleware approaches for WSN. Some of these approaches are the virtual machine, database, application driven, message-oriented, and modular programming middleware. The *Virtual Machine Approach* is flexible and contains virtual machines (VMs), interpreters, and mobile agents. Developers can write

applications in separate small modules that the system distributes through the network using specially made algorithms. These algorithms consider the overall energy consumption and resource usage limitations. The modules are then interpreted by the VM. The problem of the Virtual Machine approach is the overhead introduced by the instructions. In the *Database Approach*, the whole sensor network is treated as a distributed virtual database system. It provides an easy-to-use interface for the user to issue SQL like queries to extract target data from the sensor network. This approach is good for regular queries, but does not support rendering data in real-time; it only provides approximate results. The *Application Driven Approach* allows designing a middleware that takes into consideration application requirements to fine-tune the network. For example, an application can dictate network operations management and QoS requirements. This approach has great performance advantages, but the tight coupling with applications may lead to a specialized middleware instead of a general purpose one. The *Message-oriented middleware Approach* is based on the Publish-Subscribe mechanism to facilitate message exchange between the sensor nodes and the sink nodes. This approach is adequate for WSN where most applications are based on events such as WSNs, and its strength lies in the support for asynchronous communication that helps in saving energy in the WSN. In *Modular Programing Approach*, applications are developed as modular programs to facilitate transmission through the network by consuming less energy than a whole application. However the code is incompatible for limited resources devices since the code instruction does not allow hardware heterogeneity [2, 4].

Hereafter, we give a comparison for the different middleware approaches discussed above based on the application type. Table 1 presents this comparison with a suggestion of the type of applications where each middleware approach could be used. Ease of use refers to what degree does the middleware relieve the user from handling the heterogeneity of the network, and openness refers to how easily we can modify the system once the functional requirements change [2, 5]. All the other comparison parameters are discussed in Sect. 3 above.

5 WSN Middleware Architecture

The middleware generally gathers information from both the application and network protocols, determines how to support the connected applications, and at the same time adjusts network protocol parameters. Sometimes the middleware goes under the network protocols layer and interfaces with the operating system directly. WSN middleware needs to dynamically adjust network protocol parameters and configure the sensor nodes based on application requirements in terms of performance improvement, QoS, and energy conservation. The WSN middleware can abstract the common properties of applications and offer general purpose services that can be used by a wide range of applications. In general, a middleware solution may consist of three components: *resource management*, which is a functional

Table 1 Comparison of middleware approaches for WSN [2, 4]

Main features	Openness	Scalability	Heterogeneity	Mobility	Ease of use	Power awareness	Application type
Approaches							
Virtual machine	Full	Full	Partial	Full	Little or none	Yes	Dynamic applications
Database	Little or none	Little or none	Little or none	Little or none	Full	Partial	Event-driven applications
Modular programing	Full	Full	Little or none	Full	Full	Yes	Dynamic applications
Application driven	Full	Full	Little or none	Little or none	Full	Yes	Real-time applications
Message oriented	Full	Full	Partial	Partial	Full	Yes	Event-driven applications

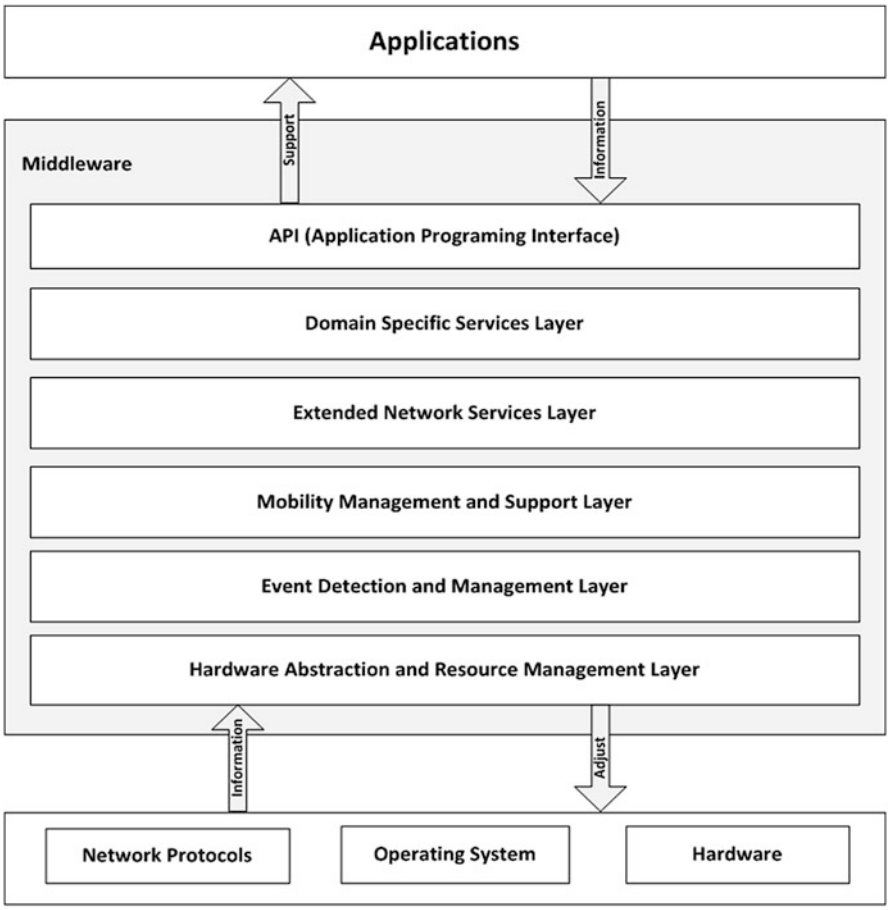

Fig. 1 General WSN middleware architecture

element that monitors the network status and gets application requirements, *event detection and management* that is used to detect and manage events, and *API* (Application Programming Interface) which is invoked by the applications in order to use services of the middleware and achieve the required performance and QoS parameters [1, 8, 9]. Figure 1 shows a general architecture of a WSN middleware.

WSN middleware should provide data management functions since it is dealing with a data-centric technology: WSN. These data management functions can include the following:

Data dissemination: in WSN the data sensed by the sensor nodes need to be transmitted to a special node or a sink for more analysis, control, and management. Data dissemination protocols which are related to routing protocols are required to provide an effective data transmission. The major difference between a data dissemination protocol and a routing protocol is that the former is general and

designed to find a path between source and destination, and the latter should guarantee successful transmission from nodes to sink. Data dissemination protocols consist of at least the following two phases: the initial phase of triggering data transmission that is initiated by the sink, and the data transmission phase when sensor nodes report data to the sink [1].

Data compression: many characteristics of WSN make it possible to implement effective data compression techniques. First, neighboring sensor nodes tend to collect correlated data especially when their deployment is dense in the network. Second, this correlation may become more apparent on the path from the sensor nodes to the sink due to the treelike logical topology of most WSNs. Third, the occurrence of an event in WSN may be assimilated as a random process whose information content can be extracted easily. Fourth, the application semantics in WSN may enable data aggregation and data fusion. Fifth, the data reading and reporting in WSN can be reduced, thanks to the tolerance of applications for possible errors in data. As stated in [1], compression includes the following techniques: information theoretic-based techniques such as Distributed Source Coding Using Syndromes (DISCUS), data aggregation-based compression schemes such as tiny aggregation service for TAG, and sampling of a random process.

Data storage: data storage sensor nodes store data related to the sensed events for future use. When considering data storage, several questions need to be answered concerning the type of data that need to be stored, where this data should be stored, and for how long. This will help defining the data storage requirements of WSN. There exist two types of data in WSN: the raw data collected directly by the sensor nodes and the results from the processed data collected initially. Examples of data storage techniques that have been proposed in the literature include: External Storage (ES), Local Storage (LS), Data Centric Storage (DCS), Provenance Aware Data Storage (PADS), and Multi-Resolution Storage (MRS) [1].

6 Existing Middleware for WSN

Some WSN middleware solutions have already been proposed. In this section, we list an example of middleware solution for each of the WSN middleware approaches listed above, and compare these solutions to explore the pros and cons.

6.1 Impala

Impala is a lightweight middleware that uses *Modular Programing Approach* and basically provides efficient mechanisms to support dynamic applications. It has automatic behavior which increases its fault tolerance and network self-organization, and is an event-driven middleware that achieves effective application adaptation. Impala is intended to act as a resource manager, operating system, and

event filter on top of which specific applications can be installed and run. Therefore it is separated into two main layers. The upper layer contains the applications and protocols. The lower layer has three main agents: an application adapter, an application updater, and an event filter. The application adapter handles application adaptation to various runtime conditions on the basis of different scenarios such as performance, energy efficiency, robustness, and other attributes. The application updater receives software updates and installs them effectively on the sensor node by taking into account trade-offs such as limited size, constrained bandwidth, high node mobility, wide range of updates, code memory management, and propagation protocol. The event filter dispatches events to the application adapter and updater and initiates processing chains such as the timer, packets sent, and device events such as device failure. Applications, the application adapter, and the application updater are all programmed into a set of event handlers which are invoked by the event filter when events are received. Impala middleware is energy-efficient, its software is easily updatable, but it does not support QoS. Also, its code instructions nature does not allow hardware heterogeneity and makes it unsuitable for devices with limited resources [1, 2].

6.2 Mate

Mate is among WSN middleware that uses the *Virtual Machine Approach* as an abstraction layer to implement its operation and tackle the different WSN challenges. Mate is a byte code interpreter that runs on top of the TinyOS [10] operating system designed specifically for sensor networks. Mate code is broken into capsules of 24 instructions, each of which is a single byte long, and can fit into a single TinyOS packet. This makes the programs quickly and easily injectable into sensor network.

Mate has concise programs which are resilient to failure, which makes the network dynamic, flexible, and easily reconfigurable. It also provides efficient network and sensor access. However, in terms of energy, Mate is only energy-efficient for short running or sleepy applications. For long running applications, it incurs high CPU overhead. Also in its current state, Mate is only byte codes; a higher-level language and programming models for application development are needed [2, 5].

6.3 Middleware Linking Applications and Networks

Middleware Linking Applications and Networks (MiLAN) is a WSN middleware providing QoS support to applications and using the *Application Driven Approach*. It was originally designed for medical advising and monitoring. MiLAN allows the application to know how it performs using the collected data from heterogeneous

sensors and how to combine sensors in order to satisfy its QoS requirements, using the graph based approach. MiLAN is tightly coupled with the network stack, thus it lacks support for OS and hardware heterogeneity and does not address mobility. It also requires that the applications have prior knowledge of the type of sensors used [2].

Two classes of applications are defined in MiLAN: data driven applications that aim to collect and analyze data, and state based applications that can change their requirements depending on the data received. In the latter case the middleware is needed to enable the applications to affect actively both the network and the sensors themselves. For its operation, MiLAN receives the following information: variables of interest to the application, the QoS required for each variable and the level of QoS that data from each sensor or set of sensors can provide for each variable. MiLAN takes a description of applications requirements and checks the network conditions to fulfill the performance required. It emphasizes on extending the runtime of the application rather than efficient utilization of sensor power. In order to account for sensor power optimization, MiLAN suggests modifications in routing protocols for energy conservation according to the application [1].

6.4 Sensor Information and Networking Architecture

SINA stands for Sensor Information and Networking Architecture. It is a WSN middleware based on the *Database Approach* which allows applications to issue queries into the network, collect results from the network, and monitor changes within this network. SINA middleware is a database like system that considers a sensor network as a collection of datasheets, where we find a collection of attributes of each sensor node in each datasheet. Therefore, the sensor network can be queried using SQL-like language. SINA has mainly three components: hierarchical clustering, attribute-based naming, and location awareness. *Hierarchical clustering* consists of grouping of nodes based on their closeness or their energy levels into clusters. *Attribute-based naming* is suitable for data-centric networks like WSN and used instead of id-based naming. Sensor nodes should be *location aware*, and that is possible using GPS [2].

6.5 MIRES

MIRES is a middleware using the *Message-Oriented Approach*, used for traditional fixed distributed systems. It has numerous advantages over the traditional request-reply model, and provides an asynchronous communication model adapted for WSN applications. It also implements a Publish-Subscribe communication infrastructure using a component-based programming model. Mires has basically three main components: a Publish-Subscribe core component which manages the communication

between middleware services, and the topics list in order to give the right topic to the related application. It has also a routing component, and some additional services such as aggregation. MIRES middleware does not support QoS and security [2].

6.6 BioNet Middleware for NASA's Exploration Mission

BioNet is a different middleware infrastructure that is used by **NASA**'s engineers whom wish to communicate, control, command, and/or monitor a facility, crew member, or some physical phenomena. The *BioNet software framework* was developed by researchers at the University of Colorado in conjunction with NASA, which funded research to develop a software framework to standardize software development among its contractors, grantees, and own internal software developers. With this standard software framework, NASA can direct a multi-institution software development to use BioNet middleware to develop software to collect data from a variety of digital devices, networks, sensors, and other automated systems [11].

BioNet integrates different data from heterogeneous sensors into a unified command and control system. The BioNet software framework provides data interoperability by normalizing the data from the different data-producing sensors and control devices. The software framework can store, transport, and display voice, video and data in a single unified data management system. BioNet has an open architecture, is a standards-based solution, and was designed for multi-developer use [12]. BioNet software framework aims to facilitate software development by allowing developers and programmers to devote their time to meeting project requirements rather than dealing with standard low-level details of providing a working system, and therefore reducing overall development time [12].

The BioNet middleware software is composed of two entities: (1) The BioNet software framework and (2) The BioNet developers' kit (or "DevKit"). The "Dev Kit" provides a *C source code* module that uses standard C language libraries that serve as a detailed example for any developer. In addition to C language bindings, Perl and Python bindings are also available. Software developers utilize the "DevKit" to interface to system hardware and to compose BioNet application clients to ingest BioNet data. A BioNet network is made up of a set of peers. *Peers* can join and leave the BioNet network at any time. There are two kinds of peers: "Hardware Abstractors" (HABs) and "Clients." The HABs interface with specific sensors and effectors and export (publish) these devices and their data to the BioNet network. HABs are *publishers* of sensor data and consumers of actuator settings. Clients get data from HABs and process them, and provide settings for actuators to HABs. Clients are *subscribers* of sensor data and providers of actuator settings. Clients choose what data they want to subscribe to. Together the HABs and clients perform information exchange utilizing efficient *publication/subscription* (pub/sub) communications [11].

BioNet can be used for a variety of applications including *environmental monitoring*: temp, pressure, humidity, atmospheric contaminants, etc., *structural monitoring*: leak detection, impact detection, seismic, stress and strain, *physiological monitoring*: health assessment, exercise cardio, BP/ECG, and Voice, Video, and Science Data [12]. BioNet is utilized for the integration of various *inventory management* hardware including handheld and portal-based radio frequency identification, RFID, systems, "smart" shelves, and the "smart" receptacles for RFID asset tracking. BioNet was also ported to the *Apple iPod/iPhone* for the NASA LHWT project, thereby providing a mobile handheld command and control device [11].

BioNet software framework has many benefits and features for space exploration including: Data interoperability, Multi-vendor solution set, Independent development, Mobility, Standards-based interoperable network communications, Scalability, Network robustness and reliability, Platform independence, Ease of application composition, Eases burden of flight software certification, Unified enterprise data management system, Automated data replication and back-up, Consistent operational interface, Integrated network security, Delay tolerant communications, Engineering data analysis, Easy support of add-on or retro-fit activities, and Endpoint re-programmability [11, 12].

6.7 Others

It is good to mention that there has been other research effort in middleware design for WSN, and most of the middleware are following one of the approaches discussed above and even use similar mechanisms like the solutions presented before. One example is Agilla [13] that is based on Mate and extends it by providing injection of mobile code into the sensor network to deploy user applications. This has been proved to be more suitable than the flooding mechanisms used by Mate. Garnet [14] is a middleware framework which provides abstraction for managing data streams within the sensor network. It offers a collection of system services such as receivers, filtering and dispatching services, and resource manager. Another example is Automatic Service Composition (AutoSec) [15] which is an Application Driven middleware providing support for dynamic service brokering for a better use of resources within a distributed system. AutoSec has an architecture that goes deep in the network and performs resource management and organization within a sensor network by providing the required QoS for applications on per-sensor basis. This is done by choosing a combination of information collection and resource provisioning policies from a given set based on user needs and system needs. Other examples of WSN middleware solutions include but are not limited to the following: IrisNet (Internet-Scale Resource-Intensive Sensor Networks Services) [16], AMF (Adaptive Middleware Framework) [17], DSWare (Data Service Middleware) [18], CLMF (Cluster-Based Lightweight Middleware Framework) [19], MSM (Middleware Service for Monitoring) [20], DDS (Device Database System) [21], MidFusion

(Middleware for Information Fusion) [22], TinyLIME (an extension of Lime for the sensor network environment) [23], TinyDB (a Declarative Database for Sensor Networks) [24], Cougar [25], MagnetOS [26], SensorWare [27], etc.

6.8 Comparison of WSN Middleware

We present hereafter in Table 2 a comparison of WSN middleware solutions. As we can see from Table 2, QoS and security are still the most ignored features in the existing middleware solutions. Security in WSN middleware should be taken seriously into consideration in the design of future middleware solutions, because future deployments of sensor networks will be mostly done for sensitive applications [7, 9].

7 WSN Application Needs

System requirements in WSN applications could change significantly due to the wide variety of possible applications of WSN. For example, energy efficiency, low data rate, and one-way communication requirements are typically dominant in environmental monitoring applications. However, in a typical industrial application the requirements are but not limited to reliability, security, high data rate, and two-way communication. Although requirements are strongly application dependent, energy efficiency is still the most important issue in the design of WSNs; high energy efficiency means long network lifetime and limited network maintenance and deployment costs [28].

WSN applications are deployed for very specific purposes and over a broad spectrum. The selection of WSN middleware and its corresponding services is greatly affected by the target of WSN applications. There exist various categories of WSN applications that need different services from the middleware. The possible categories of WSN applications are but not limited to the categories described hereafter [4].

7.1 Scale

WSN can be either used in body sensor networks or in a small size deployment with few sensor nodes or in a large size with many sensor nodes distributed over a large area. The WSN middleware is needed in the latter case in order to provide services that deal with efficiency, reliability, and manageability of the network [4].

Table 2 Comparison of WSN middleware [7, 9]

Features	Programming abstraction features			Middleware service features	
Proposal	Interface type	Abstraction level	Programming paradigm	Functional	Un-functional
Impala	None	None	Mobile agent	Code management Resource management Dynamic topology	SecurityScalability Adaptability
Mate	Imperative (motle, byte code)	Local	Virtual machine	Code management Resource management Resource management	Security Adaptability
TinyDB	Declarative (ACQP language)	Global	Database Graphic UI	Data management Resource management Dynamic topology Code management	None
Agilla	Imperative (compile-like)	Local	Mobile agent	Code management Data management Resource management Dynamic topology	Adaptability Scalability
TinyLime	None	Simple local	None	Code management Data management Resource management Dynamic topology	Security
SensorWare	Declarative	Local	Database	Code management	None
MagnetOS	Imperative (java language)	Global	Virtual machine	Code management Resource management	None
MiLAN	None	None	None	Resource managementResource discovery	None
SINA	Declarative (cells data format SQTL)	Global	Database	Data management	None

Table 2 (continued)

Features	Programming abstraction features			Middleware service features	
Cougar	Declarative (XML data format, SQL-like language)	Global	Graphic UI	Data management	None
FACTS	Declarative (facts, rules, functions, XML like)	Global	Rule based	Code management Data management	None
Proposal	Interface type	Abstraction level	Programming paradigm	Functional	Un-functional
AutoSec	None	None	None	Service discovery Resource management	None
DSWare	Declarative	Global	None	Data management Resource management Storage supporting	Real-time Reliability
Mire	Imperative	Local	Graphic UI	Data management	None
Enviro Track	Imperative object-based (context object and tracking objects)	Global	None	Data management Resource discovery Resource management	None
BioNet	Standardized application interface Consistent operational interface	Global: abstracting networks and hardware for the application developer	Database	Data interoperability Engineering data Analysis Endpoint re-programmability	Mobility Scalability Network robustness and reliability Integrated network Security Delay tolerant communications

7.2 Accessibility

WSN can be used in accessible areas such as patients monitoring as in inaccessible areas such as monitoring underwater pipelines. The middleware is highly needed for the inaccessible deployments of WSN in order to provide self-organized, energy-efficient, and configurable services. The middleware services will ease the replacement of nodes when they are damaged or run out of power. The deployment service that enables the efficient installation of new and updated software is also needed in WSN middleware, for the case of inaccessible areas [4].

7.3 Criticality

WSN can be used for uncritical applications such as environmental sensing as for highly critical applications requiring high security and QoS for sensed data such as battlefield surveillance. The WSN middleware should provide security and QoS services for the critical applications type [4].

7.4 Complexity

WSN environments can be divided into two types: simple and complex. Simple WSN environments have structured distributed homogeneous sensor nodes such as buildings' monitoring. However, complex WSN environments have high dynamic unstructured distributed heterogeneous nodes. The WSN middleware should include services that provide high abstractions to hide the complexity and heterogeneity of the underlying environment in order to deal with the latter case. These services should be transparent to the backend applications and interoperable with various types of devices and sensors. Also, as long as complexity increases, there will be a need for efficient resource discovery and self-organization services in order to meet the applications requirements and extend the life of the network [4].

7.5 Traffic Granularity

Some types of WSN only report information about exceptions such as a sudden increase in a forest temperature that may indicate a fire nearby. Other types of sensors send continuous streams of information and lead to a generation of high communication load on the network such as cameras and audio recorders. In the latter case, there is a high need for middleware services that can keep optimal performance in the network and efficiently manage high traffic [4].

7.6 WSN: Standalone Versus Integrated Applications

In contradiction with a standalone deployment, WSN may need to be integrated with other systems such as healthcare systems, SCM applications, and other enterprise systems. In this case, the middleware should include a service that enables an easy integration with other systems [4].

8 Integration of RFID with WSN

8.1 Mixed Sensor Networks

The *smart ubiquitous environment* is the vision of an environment where computing devices, integrated into everyday objects and activities, are enabled to interact and cooperate with each other and with the environment by the exchange of data and information "sensed" about the environment, while reacting autonomously to the "real/physical world" events and influencing it by running processes that trigger actions and create services with or without direct human intervention. In this environment, the computing devices are so imbedded, fitting, natural, completely permeate the life of the user, and become a helpful but invisible force, assisting the user in meeting his or her needs without getting in the way [9].

The goal of researchers is to create a system that is pervasively and unobtrusively embedded in the environment, completely connected, intuitive, effortlessly portable, and constantly available. This is supported by the new technological movements that are already well underway such as nanotechnology and wireless computing. In order to achieve this goal, such systems need to exhibit a kind of awareness about theirs and others' situation and to interact in a way that is sensitive, adaptive, responsive to their users capabilities, needs, habits, and emotions [9].

Today's pervasive computing widely uses RFID and WSN. In WSNs, hundreds to thousands sensor nodes sense the physical environment and send the sensed data to the sink by multi-hops. Examples of WSN applications include: military applications and environment monitoring. In RFID, a unique ID called Electronic Product Code (EPC) is assigned to an RFID tag which identifies a real-world object. RFID applications cover many areas such as healthcare, supply chain management, industry automation, and library management. [9].

The integration of RFID and WSN will bring many advantages in the future of ubiquitous computing. This integration can both provide real-world tracking of objects and the context information to the objects, which can increase considerably the automation of an information system. The existing EPCglobal architecture for RFID can be used to share information from the sensor nodes, due to the lack of standards in WSNs. An extension of RFID EPCglobal architecture; ESN (EPC Sensor Network) has been proposed in [9] for capturing, filtering, and sharing both RFID and sensor data. A middleware solution using the ESN architecture collects

RFID and sensor data from distributed sensors and readers, and processes this large volume of data in real-time. After aggregating, filtering, and grouping, the processed data are sent to the upper layers until they get to the backend interested applications. We present hereafter our middleware solution for the integration of both RFID and WSNs; FlexRFID.

8.2 *Multi-layer Middleware Based on RFID and Sensor Information*

The FlexRFID middleware [29] is integrated into a three-tier architecture consisting of hardware layer, middleware layer, and backend applications layer. The hardware layer referred to as the *Diverse Types of Sensors and Devices* layer comprises RFID readers, sensors, and other industrial automation devices. Such approach allows incredible flexibility in the selection of devices, lets companies build their enterprise solutions without handling low-level programming, and allows creating an intelligent sensor network, where RFID readers are choreographed with other devices such as sensors. There are diverse makes and models of devices, which require a middleware layer that monitors, manages, coordinates, and obtains data from the different devices. In FlexRFID, these functions are taken care of before processing the raw data and applying business logic to them. Our approach is to use a Device Abstraction Layer (DAL) that abstracts the interaction with the physical network of devices. The FlexRFID middleware incorporates three other layers which are: Business Event and Data Processing Layer (BEDPL), Business Rules Layer (BRL), and Application Abstraction Layer (AAL).

The *DAL* is responsible for interaction with various devices and data sources independent of their characteristics. The DAL has three other sub-layers; the *Data Source Abstraction Module (DSAM)*, the *Device Abstraction Module (DAM)*, and the *Device Management and Monitoring Module (DMMM)* [29].

The *BEDPL* acts as a mediator between the DAL and the AAL. The services accepted by the BEDPL are first authorized by the BRL and then allowed to issue commands to the DAL in order to get the raw data and process them accordingly. Similarly the raw data are carried from the DAL, processed, and passed on to the AAL by this layer. Services provided by the BEDPL include the following: data dissemination, data aggregation, data transformation, data filtering, duplicate removal, data replacement, data writing, security, and privacy [29].

The BRL is a policy-based management engine that defines the rules that grant or deny access to resources and services of the FlexRFID middleware, and enforces different types of policies for filtering, aggregation, duplicate removal, privacy, and different other services. This is achieved by determining the policies to apply when an application requests the use of a service in the BEDPL. The BRL consists of the following components: the *Middleware Policy Editor (MPE)*, the *Middleware Policy Repository Database (MPRD)*, the *Middleware Policy Enforcement Point (MPEP)*,

and the *Middleware Policy Decision Point (MPDP)*. Policies are operating rules used to maintain order, security, consistency, or other ways of successfully achieving a task. Examples of policies that should be available in the BRL are: Access policy, data replacement policy, quality of service policy, and privacy policy [29].

The *AAL* provides various applications with an interface to the hardware devices, through which the applications request the set of services provided by the FlexRFID middleware with hidden complexity. This layer provides a high level of software abstraction that allows communication among the enterprise applications and the FlexRFID middleware [29].

Figure 2 shows a detailed architecture of the FlexRFID middleware adapted to RFID and WSN data and connected to diverse backend applications. The FlexRFID middleware in this case of integration is structured in six layers of components as shown in Fig. 2.

The *Hardware Abstraction Module* is placed directly over the hardware (sensor nodes and RFID) Operating System. Its primary task is to provide transparency and platform independence by hiding individual OS peculiarities and machine level complexities.

The *Protocol Management Module* is oriented to address the dynamic and mobile nature of WSNs as opposed to infrastructure-based wired systems. In the latter systems routing and packet transmission are taken care by dedicated routers, whereas in WSNs and wireless ad hoc networks each node acts as a router. This module shields the applications from heterogeneous key distribution, authentication, routing and communication protocols. This provides unified communication over distinct communication standards like Zigbee, 6LoWPAN, or other IEEE 802.15.4-based protocols, and provides network support for packets being transmitted between network domains that use different routing protocols.

The *Common Services Module* defines higher-level in-network services such as data aggregation, localization and filtering to achieve efficiency, optimization, and quality of data. Sensor nodes may report duplicate information, out-of-range, and erroneous data. Hence sensor nodes may implement data-centric forwarding techniques to reduce unnecessary data transmission. Consequently this module provides services for data management such as data filtering, data replacement, duplicate removal, privacy, security, data dissemination, data writing, data aggregation, and data transformation, and other services which are code management, integration management, resource management, and resource discovery management.

The *Domain-Specific Services Module* is commissioned to address the requirements of specific domains such as healthcare, aerospace, and environment monitoring. For example in healthcare, it is desired to have a set of domain-specific middleware services for applications such as patient monitoring systems, life support systems, and cardiac systems. Therefore, this layer provides applications with domain-specific abstractions resulting in easily accessible remote sensor services [30]. Presents a scenario for the integration of FlexRFID in healthcare and the different trends and challenges related to this integration.

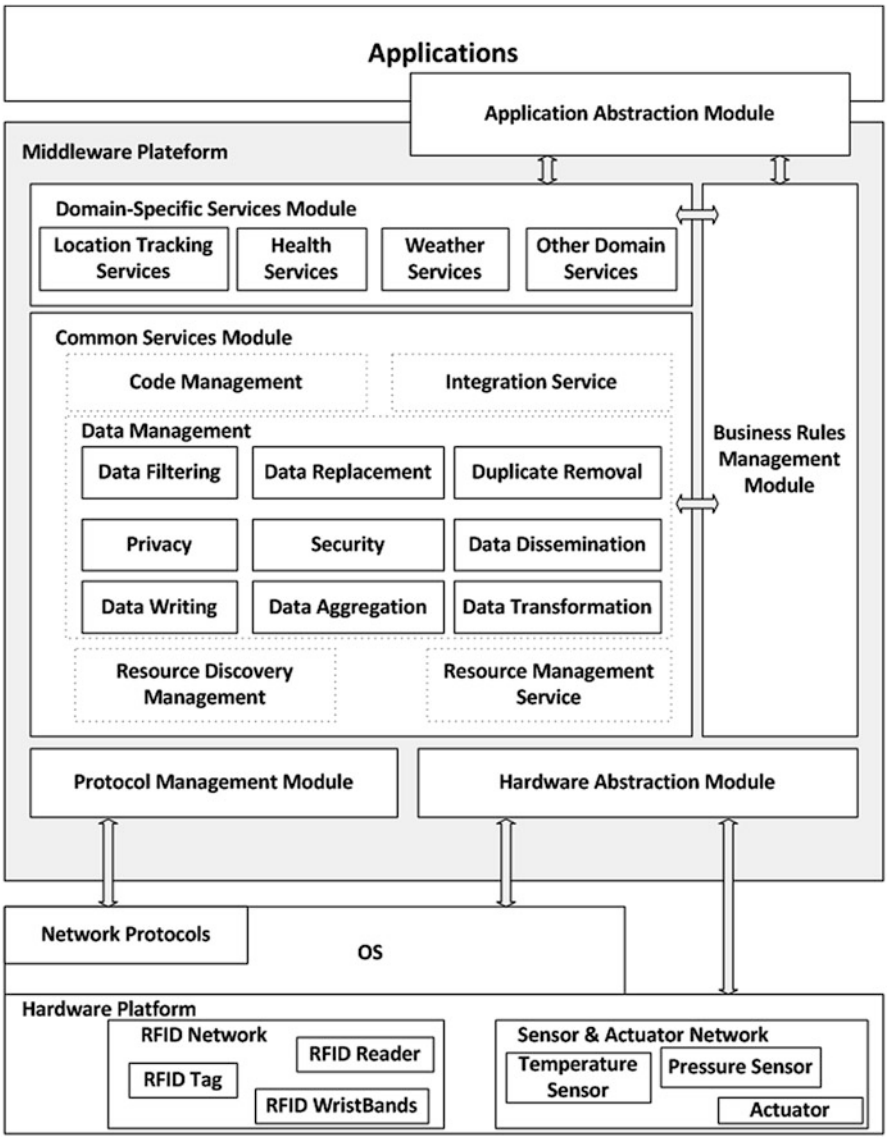

Fig. 2 FlexRFID middleware for the integration of RFID and WSN

The *Business Rules Management Module* as described above holds the applications business rules and is based on application-defined policies that get applied to the services before delivering data to the backend applications.

The *Application Abstraction Module* referred to as AAL above provides abstraction between the middleware and all the interested enterprise applications.

9 Conclusion

WSNs, a new and rapidly evolving field, is expected to change completely our lives in the near future. In this chapter, we have presented a review of the state of art middleware solutions for WSN. We went through the design principles, the different middleware approaches, and several existing middleware architectures for WSN, and then compared the different approaches and solutions by including suggestions where each approach could be used. We observed that while scalability and power saving issues can be compensated, it comes at a trade-off with QoS. The main objective of this chapter is to provide a comprehensive understanding of the current issues in WSN middleware design for better future academic research and industrial practice. We can conclude from this chapter that there is a lot of research yet to be carried out before a perfect WSN middleware can be built and tested.

There are more very important WSN middleware architectures than those covered in this chapter [1]. Future sensor nodes are expected to be resource-rich and require the development of new applications requiring high data rates, complex processing, and strict QoS requirements. Middleware for supporting such applications should be flexible and can be adapted to changes in the environment and devices. It should also comprise more functions such as fusion, migration of fusion points, effective adaptation between sensor nodes and applications, and the ability to change the device behavior more dynamically [31].

In the last section of this chapter we presented in detail the adaptation of our middleware solution called FlexRFID for the integration of RFID and WSN and the various applications for which it can be used.

References

1. Sohraby K, Minoli D, Znati T (2007) Middleware for wireless sensor networks. In: Wireless sensor networks technology, protocols, and applications. Wiley. Online version available at: http://www.knovel.com/web/portal/browse/display?_EXT_KNOVEL_DISPLAY_bookid=4513&VerticalID=0
2. Radhika J, Malarvizhi S (2012) Middleware approaches for wireless sensor networks: an overview. Int J Comput Sci Issues 9(6):224
3. Baugé T (2009) Wireless sensor networks. Thales Research and Technology. Available online at: www.thalesgroup.com
4. Mohamed N, Al-Jaroodi J (2011) A survey on service-oriented middleware for wireless sensor networks. J Service Oriented Comput Appl Arch 5(2):71–85. doi:10.1007/s11761-011-0083-x
5. Hadim S, Mohamed N (2006) Middleware for wireless sensor networks: a survey. Paper presented at the 1st IEEE international conference on communication system software and middleware (COMSWARE 2006), New Delhi, 8–12 January 2006
6. Omer KR, Kasten O, Mattern F (2002) Middleware challenges for wireless sensor networks. ACM SIGMOBILE Mobile Comput Commun Rev 6(4):59–61. doi:10.1145/643550.643556
7. Molla M, Ahamed SI (2006) A survey of middleware for sensor network and challenges. Paper presented at the 2006 IEEE international conference on parallel processing workshops (ICPPW), Columbus, 14–18 August 2006

8. Afzal SR, Huygens C, Joosen W (2009) Extending middleware frameworks for wireless sensor networks. Paper presented at the IEEE international conference on ultra-modern telecommunications (ICUMT 2009), St. Petersburg, Russia, 12–14 October 2009
9. Wang W, Sung J, Kim D (2008) Complex event processing in EPC sensor network middleware for both RFID and WSN. Paper presented on the 11th IEEE symposium on object oriented real-time distributed computing (ISORC), Orlando, 5–7 May 2008
10. TinyOS documentation wiki (2013) Getting started with TinyOS. Available online at: www.tinyos.net
11. Gifford K, Williams S, Maiorani L et al (2010) BioNet middleware and software framework in support of space operations. Paper presented on the SPACEOPS 2010 conference, hosted by NASA, Huntsville, 25–30 April 2010
12. Gifford K, Foore L (2007) BioNet command, control, and communications infrastructure for NASA's exploration mission. bioserve.colorado.edu/bionet. Available online at: https://caneus.org/fbw/downloads/2007/Gifford-FooreBriefing.pdf
13. Fok CL, Roman GC, Lu C (2009) Agilla: a mobile agent middleware for self-adaptive wireless sensor networks. ACM Trans Autonomous Adapt Syst 4(3), Article No 16. doi:10.1145/1552297.1552299
14. St Ville L, Dickman P (2003) Garnet: a middleware architecture for distributing data streams originating in wireless sensor networks. In: 23rd international conference on distributed computing systems workshops, Providence, 19–23 May 2003
15. Han Q, Venkatasubramanian N (2001) AutoSeC: an integrated middleware framework for dynamic service brokering. IEEE Distrib Syst Online 2(7)
16. Gibbons PB, Karp B, Ke Y et al (2003) IrisNet: an architecture for a worldwide sensor web. IEEE Pervas Comput 2(4):22–33
17. Yu X, Niyogi K, Mehrotra S et al (2003) Adaptive middleware for distributed sensor environments. IEEE Distrib Syst Online 4(5). doi:10.1109/MDSO.2003.1
18. Li S, Son SH, Stankovic JA (2003) Event detection services using data service middleware in distributed sensor networks. Paper presented at the 2nd IEEE international conference on information processing in sensor networks (IPSN), Palo Alto, 22–23 April 2003
19. Yu Y, Krishnamachari B, Prasanna VK (2004) Issues in designing middleware for wireless sensor networks. IEEE Netw 18(1):15–21
20. Ahamed SI, Vyas A, Zulkernine M (2004) Towards developing sensor networks monitoring as a middleware service. Paper presented at the international conference on parallel processing workshops (ICPPW), Montreal, Quebec, Canada, 15–18 August 2004
21. Bonnet P, Gehrke J, Seshadri P (2000) Querying the physical world. IEEE Pers Commun 7(5):10–15
22. Alex H, Kumar M, Shirazi B (2008) MidFusion: an adaptive middleware for information fusion in sensor network applications. Inform Fusion 9(3):332–343. doi:10.1016/j.inffus.2005.05.007
23. Curino C, Giani M, Giorgetta M et al (2005) TinyLIME: bridging mobile and sensor networks through middleware. Paper presented at the 3rd international conference on pervasive computing and communications (PerCom), Kauai Island, 8–12 March 2005
24. Madden SR, Franklin MJ, Hellerstein JM et al (2005) TinyDB: an acquisitional query processing system for sensor networks. ACM Trans Database Syst (TODS) – Special Issue: SIGMOD/PODS 30(1):122–173
25. Rahman MA (2009) Middleware for wireless sensor networks: challenges and approaches. Paper presented at the current internet trends seminar on internetworking, Helsinki University of Technology, Espoo, Spring 2009
26. Wang MM, Cao JN, Li J et al (2008) Middleware for wireless sensor networks: a survey. J Comput Sci Technol 23(3):305–326
27. Boulis A, Han CC, Shea R (2007) SensorWare: programming sensor networks beyond code update and querying. Pervasive Mobile Comput Arch 3(4):386–412
28. Buratti C, Conti A, Dardari D et al (2009) An overview on wireless sensor networks technology and evolution. Sensors 2009(9):6869–6896. doi:10.3390/s90906869

29. Ajana ME, Harroud H, Boulmalf M et al (2009) FlexRFID: a flexible middleware for RFID applications development. Paper presented at the 6th IEEE international conference on wireless and optical networks communications (WOCN), Cairo, Egypt, 28–30 April 2009
30. Ajana ME, Harroud H, Boulmalf M et al (2012) Emerging wireless technologies in E-Health trends, challenges, and framework design issues. Paper presented at the 3rd IEEE international conference on multimedia computing and systems (ICMCS), Tangier, Morocco, 10–12 May 2012
31. Lopez TS, Kim D (2007) A context middleware based on sensor and RFID information. Paper presented at the 5th IEEE international conference on pervasive computing and communications workshops (PerComW), White Plains, 19–23 March 2007

Part II
Space Applications

Space Applications of Low-Power Active Wireless Sensor Networks and Passive RFID Tags

Richard J. Barton, Raymond S. Wagner, and Patrick W. Fink

Abstract The purpose of this chapter is to demonstrate the increasing importance of Wireless Sensor Networks (WSNs) in all aspects of NASA space exploration activities, review several areas of recent development, and examine in some detail two WSN technology areas being explored at NASA Johnson Space Center (JSC). The two technology areas of emphasis are low-power active WSN protocols and architectures and passive wireless Radio-Frequency Identification (RFID) inventory tracking systems. The methodologies and motivations for developing each of these WSN technologies for space application are explored in the chapter, and the current state of the art in each area as well as the anticipated future developments are reviewed. In addition, devices, systems, as well as operational concepts currently under development at JSC are described, and the results of a recent performance evaluation study of alternative low-power active WSN technologies undertaken at JSC are presented and discussed.

The specific topics covered in the chapter include:

- Introduction to low-power active WSN technology, standards, and space applications,
- Rapid prototyping and testing of low-power active WSN devices and systems at NASA JSC,
- Comparative performance evaluation of ZigBee and ISA100.11a in space habitat environment analogs,
- Introduction to passive RFID inventory tracking and sensing technologies, standards, and space applications,
- Development of passive RFID inventory tracking systems at NASA JSC.

1 Introduction

Wireless Sensor Networks (WSNs) have the capacity to revolutionize data gathering in both spaceflight and terrestrial applications. A WSN consists of small, low-

R.J. Barton (✉) • R.S. Wagner • P.W. Fink
NASA Johnson Space Center, Houston, TX, USA
e-mail: richard.j.barton@nasa.gov

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_5

power nodes that can sample local sensors, optionally perform local data analysis and signal processing, and wirelessly transmit raw or processed data to a central collection point. WSNs provide a substantial advantage over traditional, wired instrumentation since they do not require wiring trunks to connect sensors to a central hub. This allows for easy sensor installation in hard to reach locations, easy expansion of the number of sensors or sensing modalities, and reduction in both system cost and weight.

While this technology offers unprecedented flexibility and adaptability, implementing it in practice is not without its difficulties, particularly with respect to achieving reliability that is on par with wired sensor approaches. Examples of these difficulties include complex and dynamic scattering environments, contention for spectrum between multiple wireless systems, radio frequency interference from other sources, and battery lifetime. Multiple government agencies and industrial partners have recognized the need for standardization of WSNs, in part to defray costs that could be prohibitive for a single entity but also to bring together the best practices and technologies for achieving common needs. Thus, solutions to WSNs for critical government and industrial applications have emerged in the form of standards-based WSNs.

This chapter focuses on space-agency and space-exploration applicable standards for wireless sensing with the goal of providing interoperable, reliable, and scalable networked wireless sensing and asset management solutions. Throughout the chapter, the emphasis is on applications of low-power active WSN standards and passive Radio-Frequency Identification (RFID) standards, and we largely restrict attention to the subset of standards that have received the most attention (primarily due to maturity and commercial/industrial adoption) within the international space agencies and international bodies such as the Consultative Committee for Space Data Systems (CCSDS). In fact, much of the discussion in this chapter is derived from different CCSDS documents that the authors helped to draft [1, 2].

The chapter is organized as follows. An introduction to low-power active WSN technology for space applications is given in Sect. 2. A discussion of rapid prototyping and testing of low-power active WSN devices and systems at Johnson Space Center (JSC) is discussed in Sect. 3, and in Sect. 4 a comparative performance evaluation of ZigBee and ISA100.11a in space habitat environment analogs is presented. In Sect. 5, the emphasis switches to RFID, beginning with a brief history of asset management in human spaceflight applications and the introduction of RFID technology addressing this application. The development of passive RFID inventory tracking systems at NASA JSC and the concept of Autonomous Logistics Management (ALM) are discussed in Sect. 6.

2 Introduction to Low-Power Active WSN Technology for Space Applications

This discussion has appeared in an earlier publication of these results [19].

Low-power active WSNs are extremely useful for monitoring and controlling the behavior of spacecraft and launch systems during testing phases on the ground or during nominal operations in orbit. Successful monitoring and control in these situations is the key to ensuring the correct functioning of various onboard systems and structures and the long-term reliability of the spacecraft. These data are also highly significant when compiling lessons learned that will be applied to building better space systems and increasing the reliability of future space components.

Hundreds and often thousands of data measurement locations are required to adequately monitor spacecraft behavior, steadily increasing the mass (acquisition systems, cables, and harnesses) and the project costs and time (installation and verification of each new sensor). The use of low-power active wireless technologies is foreseen to reduce the integration effort, cost, and time typically required to instrument a high number of physical measurement points on a space structure. Technicians should need less time to integrate and verify their installations, while the risk of mechanically damaging interfaces during the process should be reduced. Large structures should see health monitoring equipment mass reduced, while last-minute changes in the instrumentation (e.g., addition/removal of sensing nodes at measurement points) should be easier to accept at project level. One of the by-products of using wireless technologies in space systems is the extra flexibility introduced when implementing wireless fault-tolerance and redundancy schemes.

An important consideration in our approach to the application of low-power active WSNs is the desire to provide a scalable wireless infrastructure for low-rate data transport that will: (1) augment the *overall networking infrastructure* in a spacecraft rather than to provide dedicated data transport to particular end-to-end application-specific subsystems, (2) support traffic generated by diverse sensor types, multiple application-specific devices, and devices supplied by multiple different vendors, and (3) facilitate operation of multiple wireless networks in the same bandwidth with minimal interference. This implies in particular that we focus on providing wireless data transport across the lower levels of the Open System Interconnect (OSI) network model and not on achieving higher-level application-specific behavior. That is, the chosen technology should ensure interoperability of many different low-data-rate wireless devices on a common network at the physical (PHY) layer and data link (DL) layer, which includes the Medium Access Control (MAC) sublayer, so that data packets generated by new devices entering the network will be transported by the existing network devices without regard to the sensor or application that generated the data in the packet payload.

Towards that end, we consider standards that are derived from the progenitor of almost all low-power active WSN standards, *IEEE 802.15.4*. In the nomenclature of the OSI model, IEEE 802.15.4 provides a MAC and PHY for low-data-rate wireless personal area networks. A number of PHYs with various operating frequencies,

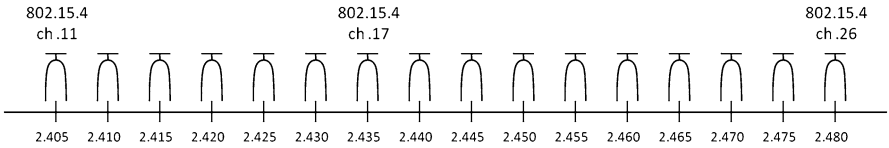

Fig. 1 802.15.4 channel assignment

modulation techniques, and achievable data rates are available in 802.15.4, but the most popular by far has been the 2.4 GHz Direct Sequence Spread Spectrum (DSSS) PHY outlined in the original version of the standard and carried through to the 2011 revision [3].

There are 16 channels available under the 2.4 GHz PHY, numbered sequentially from 11 to 26. These are illustrated in Fig. 1 above. Center frequencies start at 2.405 GHz and are spaced 5 MHz apart. The spectral mask of each channel nominally extends ± 3.5 MHz from the center frequency (the -30 dB absolute limit), but effective bandwidth is typically considered ± 1 MHz from the center of the channel.

In addition to the PHY options, 802.15.4 mandates a Carrier Sensing Medium Access with Collision Avoidance (CSMA-CA) MAC layer. This mechanism is intended to facilitate non-interference with other 2.4 GHz systems, such as 802.11 a/b/g/n Wi-Fi. When the MAC layer has an outgoing packet to send, it will first perform a clear channel assessment, listening on its assigned channel for any signal energy. Should it detect no other traffic, it will pass the packet down to the PHY layer for transmission; otherwise, it will back off for a randomly selected period before trying again.

The 802.15.4 specification is adequate only for establishing direct communication links between wireless sensor nodes. Thus, a WSN using 802.15.4 as its communication stack will (essentially) be constrained to the so-called *star topology*, with each node reporting directly to an aggregation point such as an Ethernet-connected gateway. This, of course, limits the potential geographic scope of the network to the radius of an individual 802.15.4 radio, which can be fairly small when relying on batteries alone and practicing sound power economy. This has led to several extensions of 802.15.4 that provide greater functionality.

2.1 ZigBee

The *ZigBee* standard [4] overcomes these limitations by specifying the remainder of the layers required to build a complete, multi-hop OSI protocol stack on top of 802.15.4. ZigBee begins by adopting the 2.4 GHz DSSS PHY and CSMA-CA MAC of 802.15.4 directly, picking a single channel from the options in Fig. 1 and operating with a listen-before-talk mode. It then specifies a network layer (NWK) for multi-hop mesh networking. Ad-hoc On Demand Distance Vector (AODV)

routing is chosen as the means of route discovery and maintenance in the mesh network (though source routing, specifying each next hop directly in the packet header, is also allowed). Finally, an application layer (APP) with a number of optional application profiles is defined. These profiles provide standard interfaces for application domains such as home automation, health care, and smart energy systems.

ZigBee defines three classes of devices: ZigBee coordinators (ZCs), ZigBee routers (ZRs), and ZigBee end devices (ZEDs). ZEDs are the most simple and are only capable of generating their own traffic—they cannot participate in the multi-hop routing of other nodes' packets. This role falls to the ZRs, which can both generate packets of their own and relay the packets of other ZigBee nodes. Finally, a single ZC is responsible for network formation, and only one can be active at a time in a given ZigBee network. It is also typical for the ZC to act as the gateway to a higher-throughput backbone network and, therefore, the destination of ZigBee network packets. The latter role, however, is a matter of convenience and not addressed in the ZigBee specification.

Due to their responsibilities at the NWK layer, ZCs and ZRs must remain active at all times and cannot put their radio stack processors into low-power sleep states. ZEDs can (and should) cycle into low-power states whenever possible to maximize the lives of their power supplies. Typically, ZEDs are considered to be battery powered, while ZRs and ZCs are assumed to be mains powered. This is not mandated in the standard, but it helps to maximize the overall network lifetime.

2.2 ISA100.11a

Like ZigBee, the International Society of Automation's ISA100.11a [5] is based on IEEE 802.15.4, adopting the 2.4 GHz DSSS PHY. However, rather than adopting the 802.15.4 MAC in its entirety, ISA100.11a mandates an alternate medium access approach. The carrier-sensing mechanism is still employed to check for interference, but rather than restricting itself to operation on a single channel, ISA100.11a makes use of all 16 channels shown in Fig. 1. To do so, it coordinates frequency hopping among sender/receiver node pairs, requiring central management from a Network Manager entity. Time is slotted into (typically 10 ms) intervals, and nodes must be synchronized relative to each other so that slot boundaries are aligned. When one node (e.g., a sensor node) wishes to communicate with another (e.g., a network gateway) at an application level, it must first request communication resources from the Network Manager. This initiates building a route (using graph routing) and configuring transmit and receive slots for each pair of nodes on the path so that they can listen for and talk to each other on specific channels at specific times. A node on the route with a packet to send cycles through the available channels at designated transmit slots, and the associated receiving node cycles through those same channels waiting for a transmitted packet. When the transmitting node receives a MAC-level acknowledgement it declares the transmit attempt successful. If the transmitting

node is unsuccessful on a number of channels, it may select an alternate next-hop neighbor, according to its routing graph. Particularly troublesome channels can be blacklisted by the Network Manager over time so that they are skipped by nodes during frequency hopping; these channels may then be whitelisted at a later time should they start to indicate better performance. Taken together, the features of this alternative MAC build time, frequency, and spatial diversity into ISA100.11a. It is notable that, as a consequence of the inherent time synchronization, all routers are capable of sleeping their radio processors to conserve power.

ISA100.11a provides an object-oriented APP layer suited to industrial process control. But, just as the use of ZigBee application profiles is optional, so is the use of the ISA100.11a APP layer. A tunneling service is provided to allow arbitrary protocols to use the underlying networking functions. It should be noted here that the WirelessHART industrial WSN protocol [6] provides a very similar service to ISA100.11a. In fact, the two approaches share a similar design lineage, as they are both based on Dust Networks' Time Synchronized Mesh Protocol [7]. While the two are not strictly compatible, they share many of the same features and advantages. WirelessHART, however, does not allow its application layer to be bypassed and is targeted strictly at providing a wireless interface for Highway Addressable Remote Transducer (HART) control equipment. For this reason, we have chosen to focus on ISA100.11a, which is far more open. Finally, the IEEE 802.15.4e-2012 amendment adds a scheduled MAC option to 802.15.4 which draws inspiration from both ISA100.11a and WirelessHART. As this specification is organized under the banner of IEEE, it may become the preferred form of frequency-hopping WSN MAC. However, at the time of publication of this chapter, there are no readily available commercial instantiations of 802.15.4e, and therefore we must only introduce it as a point of interest.

2.3 Tradeoffs

ZigBee and ISA100.11a in many ways provide very different solutions to the WSN problem. ZigBee is targeted primarily at the home and office automation market. Network setup must be (essentially) instant and effortless, and radio processors must be affordable. ISA100.11a is targeted at the industrial processing monitoring and control market, where loss of data (even for non-critical operations) can be costly for operators. Network behavior must be predictable, reliable, and tolerant of RF interference and harsh environmental conditions.

As a consequence, the two protocols are each strong in areas where the other is weak. ZigBee radios are inexpensive, costing on the order of US$10 in small quantities; ISA100.11a radios are an order of magnitude more expensive. ZEDs can find and join a nearby ZigBee network within seconds; in ISA100.11a, the Network Manager must run a complex bandwidth allocation algorithm as nodes join the network and request communication contracts—a process that can take several minutes. However, when fully configured, the ISA100.11a network is designed

to be very robust with respect to performance in the presence of significant RF interference, a claim that ZigBee cannot make. ISA100.11a also provides reliably low latency on packet delivery from source to destination; ZigBee suffers from variable (and sometimes extremely high) latency. ISA100.11a's scheduled channel access technique allows all nodes, including routers, to sleep. ZigBee routers must remain awake at all times. Not surprisingly, greater robustness and determinism comes at a price: ISA100.11a nodes generally achieve a lower maximum data rate than ZigBee nodes due to the inefficiencies of the slotted channel access scheme.

2.4 Coexistence with 802.11

Recall from Fig. 1 the channel assignments of 802.15.4: center frequencies for 16 channels are spaced 5 MHz apart, starting at 2.405 GHz. The spectral occupancy of each channel is $f_c \pm 1$ MHz, where f_c represents the center frequency. 802.15.4 channels at 2.4 GHz are numbered starting at 11 (an artifact of the 10 channels at 800 and 900 MHz available under other 802.15.4 PHYs). Unfortunately, the Wi-Fi standards IEEE 802.11b,g,n occupy the same frequency band as 802.15.4, and as a result, one of the primary sources of interference for WSNs using the 802.15.4 PHY will be 802.11 radios.

For 802.11b,g and the single-channel (20 MHz) mode of 802.11n, center frequencies start at 2.412 GHz and enumerate 13 channels, each spaced 5 MHz apart. Channel widths are $f_c \pm 11$ MHz. In North America, operation is permitted only in channels 1–11; elsewhere in the world, channels 12–13 may also be used [8]. To minimize mutual interference of Wi-Fi networks, operation on either of channels 1, 6, or 11 is typically recommended in North America, as these are orthogonal with guard bands in between (see Fig. 2). IEEE 802.11n networks can also operate in channel-bonding (40 MHz) mode with channels twice as wide ($f_c \pm 21$ MHz). Throughput is higher in 40 MHz mode, but two 802.11n 40 MHz networks cannot co-exist without mutual interference due to the greater channel usage (Fig. 3).

When 802.11 is configured with 20 MHz channels using the suggested North American channel assignments (1, 6, or 11), Wi-Fi-free bandwidth can be guaranteed for 802.15.4 in channels 15, 20, 25, and 26. The former two lie in the guard bands between channel pairs 1/6 and 6/11, and the latter two lie in the empty bandwidth above channel 11. However, when Wi-Fi channel assignment is not

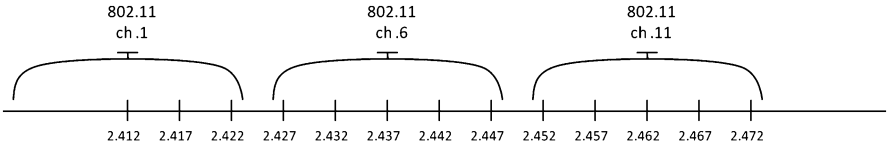

Fig. 2 802.11b,g and 802.11n (20 MHz) channel assignment

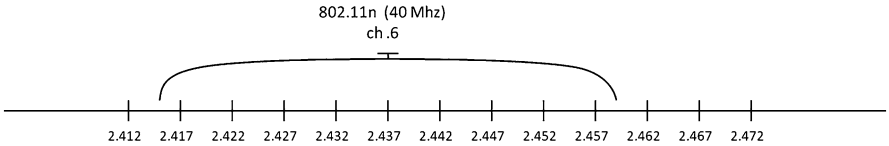

Fig. 3 802.11n (40 MHz) channel assignment

tightly coordinated, and 40 MHz channels or upper Wi-Fi channels allowed outside of North America are in use, there may be significantly less (possibly none) guaranteed bandwidth for 802.15.4-based WSNs.

3 Rapid Prototyping and Testing of Low-Power Active WSN Devices and Systems at JSC

This discussion has appeared in an earlier publication of these results [18].

Available commercial off-the-shelf (COTS) components implement a variety of standard wireless communication protocols with a range of capabilities. Some are suited for higher data rates, while others are optimized for lower power. Some utilize single-hop (point-to-point) protocols, while others implement multi-hop, mesh network transport. Finally, some offer best-effort service with a lightweight protocol, while others are optimized for robustness in the face of wireless interference.

To guide down-selection of these technologies and their individual vendor implementations, we must understand how they will perform in expected spaceflight environments, which can be very different from terrestrial application environments. Spacecraft tend to be composed of a number of small (or very small) resonant cavities, introducing multi-path radio frequency (RF) interference dissimilar to that in most terrestrial applications and giving rise to unique fading and self-interference problems. Such tight physical constraints also result in unique inter-system interference problems, which can be expected with greater frequency as wireless solutions proliferate on orbit. Both of these interference behaviors must be characterized prior to widespread deployment of wireless systems in space. Additionally, as power to un-tethered wireless devices is likely to be limited in spaceflight applications, operational power requirements of wireless systems and protocols must be well understood to guarantee their continued availability over a vehicle's lifetime.

Characterizing these facets of wireless technologies in ground tests requires access to a number of representative vehicle environments and test facilities. Test protocols must be determined, hardware must be built and software written to guarantee fair comparisons between alternatives, and results must be archived and disseminated among mission managers interested in applying the technologies. The primary goal of guiding proper down-selection of wireless standards and vendor implementations with respect to mission requirements motivates carefully

structured experiments that vary only the choice of radio (protocol and implementation) under consideration and keep all other aspects of the experimental setup constant. To aid in this process, we have developed a suite of modular rapid-prototyping hardware, collectively referred to as the Modular Integrated Stackable Layers (MISL) system to serve as a reference platform for developing and conducting test procedures. In this section we discuss the MISL hardware and software architecture as well as some examples of test methodology and test environments.

3.1 MISL Hardware Architecture

To test a given radio, at a minimum we must (1) provide it with power; (2) interface it with an antenna; (3) command it to execute a series of data transmissions and keep track of any data it receives. Therefore, the choice of hardware components supporting the radio will have an impact on the overall test results and must be uniform across experiments. Standardizing power sources and antenna selections is not particularly difficult. More consideration must be given, however, to the mechanism by which we choose to drive the experiment from software.

Many modern wireless protocols, especially those capable of forming networks, require their "radio" to do more than simply modulate and transmit waveforms. These protocols often implement many of the layers of the OSI model for communications systems [9]. This may include the APP layer at which the test routines will reside. Thus, for many choices of protocol/implementation, the radio on its own, were it programmable at the APP layer, would suffice to form a complete test unit when paired with a power source and antenna. Unfortunately, this will not always be the case, as many vendors do not open the source code for their communication stack implementations and, hence, do not allow additional code development on their radio processors.

Thus, we have adopted a model whereby the APP layer resides on an external application processor interfaced with a radio processor that implements all remaining OSI layers as required by the protocol under investigation. This allows standardization of all test elements (APP processor, power source, antenna) across experiments with the exception of the particular radio being tested. It also has important consequences for the design of the test software suite.

MISL is designed for easy interchange of components to allow for quick prototyping of integrated systems. Modularity drives the form factor, which is not optimized for size or weight. MISL factors its basic test article into three modular components: a power supply board, a processor board, and a communications board. An interface is provided for adding arbitrary sensor modules to enable rapid prototyping of embedded, wireless sensing applications, if desired. Figure 4 illustrates one example of a MISL test article, which in this case comprises four interconnected modules (including a sensor package).

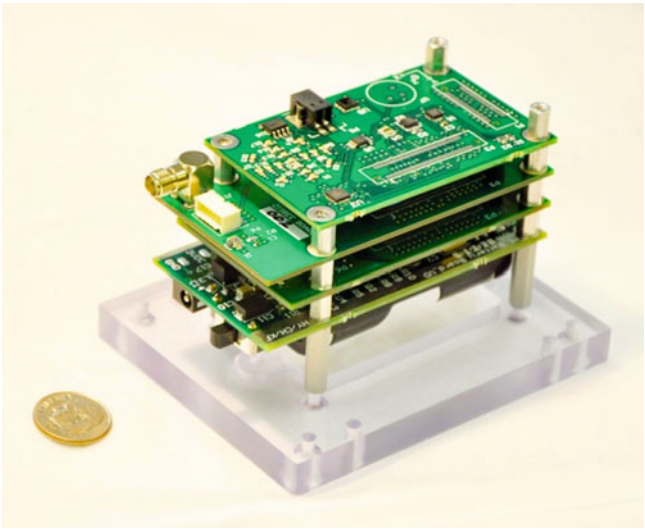

Fig. 4 Modular Integrated Stackable Layers (MISL) unit, containing power, processor, wireless communication, and sensor modules (photo courtesy: Mary Lynne Barends)

3.2 MIS Software Architecture

To evaluate test articles in relevant environments, we must be able to generate specific patterns of wireless traffic from each individual unit. This involves writing test applications running on the unit's application processor that interact with the driver software for the particular radio to which the processor is interfaced. Ideally, comparing two different radios under the same application should involve changing only (1) the physical radio module interfaced with the test article, and (2) the driver compiled into the test application for that specific radio. This enables running the same test application on top of each radio without modification, ensuring a fair comparison between competing protocols.

The driver in question resides on the application processor and translates application-level communication requests into the application programming interface (API) calls peculiar to the vendor-specific implementation of the communication protocol running on that radio. Differing standards—and even deferent vendor's implementations of the same standard—may have dissimilar APIs. Thus, drivers must be structured such that a uniform presentation is made to the application layer, regardless of the particular protocol/implementation running under the driver.

This motivates building drivers such that each supports a certain basic set of calls, for example:

configure(): implement any hardware and network configuration steps necessary to begin sending and receiving packets.

send(message, address): given a message and the address of a receiving radio, send the message byte string as a packet payload to the receiver.
receive(message): provide a callback function to be executed on the payload of a received message at the APP layer.
service(): service outstanding tasks on the radio driver (allows for interleaving radio driver functions with other APP tasks in a single-threaded processor).

Multiple protocol- and vendor-specific details may be hidden behind the implementations of these API calls. For example, with protocols such as IEEE 802.11 and the IEEE 802.15.4-based ZigBee, configuration of the network is quite straightforward. Provided a transmitting radio has the proper security credentials and knows the network address of a receiver, it is allowed to immediately begin sending and receiving messages once authenticated. For a protocol such as ISA100.11a, however, a radio must request bandwidth from a network manager entity to use in sending messages once it has been authenticated in the network. Until the manager allocates the resources and notifies the radio, the associated application cannot begin sending its data.

Similarly, a gateway interface to which network data may be transmitted for transport outside of the wireless network must be somewhat standardized. The gateway itself may have an arbitrary configuration in hardware. In many cases (ISA100.11a, 802.11 a/b/g/n) the gateway may be a vendor-supplied part. In others (ZigBee), the gateway may be either vendor-supplied or built from components similar to those used for the MISL test article. Regardless, the gateway interface must be capable of receiving, logging, and displaying the contents of packets sent to the gateway by the test article radios. It must also be able to send packets to network radios either individually or as a group.

3.3 Test Methodology

WSN tests must evaluate the performance of candidate radios according to a number of metrics. Some possible examples of useful test procedures include the following.

Application-Level Throughput Test From each node, send a known sequence of APP messages to the network gateway at a pre-defined rate. At the gateway interface, collect and analyze these messages to determine the percentage of successful and unsuccessful message deliveries.

Application-Level Bandwidth Test From a given node (or multiple nodes), attempt to send a known sequence of packets as quickly as possible. At the gateway interface, collect and analyze the messages to determine the time elapsed between successful reception of the first and last messages.

RF Interference Immunity Test Repeat the application-level throughput test, but simultaneously introduce a controllable source of RF interference. For basic in-band jamming, an RF signal generator may be used. For more sophisticated tests

of system coexistence, other RF systems may be used as interferers. Wireless interference from these other RF systems may be generated randomly RF systems may be generated randomly (e.g., according to or approximating real-world usage patterns) or deterministically. In the latter case, the receiver of an interfering system transmitter/receiver pair may be analyzed to additionally determine the reciprocal impact of the system under test on the jamming system.

Node-Level and Network-Level Power Consumption Test For any of the tests described above, record the amount of power consumed by each node during an experiment. In the case of multi-hop networks of radios, this data can be aggregated to determine network-level power draw for a given application data flow.

3.4 Test Environments

The Wireless Habitat Test Bed (WHAT) at JSC provides a unique platform for characterizing the behaviors of wireless systems in an analog crewed environment (Fig. 5). The cylindrical WHAT is approximately the size of an International Space Station (ISS) module or future habitat at 10 ft in diameter and 20 ft in length. It has recently been used to characterize the relative performance of the ISA100.11a and ZigBee Pro WSN protocols in the presence of 802.11g Wi-Fi interference [18].

Fig. 5 NASA-JSC Wireless Habitat Test Bed

Numerous equipment racks and storage bins outfit the WHAT along its walls, running down its length and extending from its ceiling down to a metallic deck. The deck is mounted approximately 2 ft above the WHAT's lowest point, and the space underneath may be outfitted with more equipment or used as storage space.

The WHAT provides a highly RF-reflective environment that can be used to study the performance of candidate RF systems in future habitation-class vehicles. And since the metallic WHAT shell provides good insulation from the external environment, it provides an RF-quiet space to which we can begin to selectively add interferers to study the ability of multiple candidate RF systems to co-exist in a crewed environment.

An RF anechoic chamber is also located in the same NASA-JSC facility as the WHAT, giving further options for control of RF interference levels. Using the anechoic chamber, we can obtain ground-truth data on radio performance in the absence of the multi-path self-interference induced by the WHAT environment.

4 Comparative Performance Evaluation of ZigBee and ISA100.11a in Space Habitat Environment Analogs

In this section, we summarize the results of a rigorous laboratory analysis of the performance of ZigBee Pro and ISA100.11a. Additional details and discussion of the results of this study can be found in [18].

As robustness to RF interference emerges as the principal claimed differentiator between the more complex ISA100.11a approach and the simpler ZigBee approach, we study message delivery rates in a representative crewed environment in the presence of varying levels of interference. To achieve parity between the two approaches, we attempt to hold constant as many variables as possible across our experiments except for those being tested. Since ZigBee and ISA100.11a adopt very different multi-hop routing protocols (which ultimately merit their own, separate comparison), we restrict ourselves to a single-hop (star) topology with each node transmitting data directly to a central hub. This effectively reduces our study to a comparison of the MACs used by ZigBee (CSMA-CA, single channel) and ISA100.11a (scheduled, channel-hopping). The study was performed in the JSC WHAT and an 802.11g-enabled client radio carrying traffic generated by the Iperf network traffic tool provided varying levels of RF interference. Further details on the experimental methodology are discussed below.

Our results show that, in general, the more lightweight ZigBee protocol performs quite well at low to moderate interference levels, but its performance degrades as interference increases. Conversely, the more complex and costly ISA100.11a protocol continues to perform well as single-channel Wi-Fi interference levels increase.

4.1 Experimental Methodology

Experiments were conducted using MISL and MISL-compatible hardware. As discussed above, this architecture comprises a generic application processor that communicates with a radio processor. The application processor implements the APP layer in the OSI model, formatting packet payloads to pass to the radio processor, which is responsible for all the lower layers in the networking stack.

4.2 Hardware

For these experiments, the Texas Instruments (TI) MSP430-F5438 microcontroller was selected as the application processor to interface with the ZigBee and ISA100.11a radio processors. By writing the driver code on the MSP430 to expose similar APP interfaces for both the ZigBee and ISA100.11a radios, we were effectively able to run the same application-level code on top of each WSN protocol.

The Texas Instruments CC2530 system-on-a-chip 802.15.4 radio was chosen as the ZigBee radio processor. The CC2530 was configured as a ZigBee Network Processor (ZNP) using Texas' Instruments Z-Stack (v. 2.5.0). Z-stack is a ZigBee Alliance-certified implementation of the ZigBee Pro protocol (ZigBee-2007 stack profile 2) with golden unit status. The Nivis VersaNode 210 (VN210), which is the first commercially available radio implementing ISA00.11a, was chosen as the ISA100.11a radio processor. For our experiments, the VN210 was loaded with the Nivis ISA100.11a Full API firmware (v. 4.3.9). The power levels on all radios were normalized to about $+3$ dBm.

The nodes in each of the two networks must report to a gateway. For the ZigBee implementation, we used another CC2530 designated as the Coordinator of the network. For the ISA100.11a implementation, a Nivis VersaRouter (VR) 900 was used. The VR900 implements the Network Manager and Backbone Router roles in the ISA100.11a network.

4.3 Software

Embedded software drivers written to allow the MSP430 to communicate over an SPI interface to both of the TI ZNP and Nivis VN210. The radio modules each exposes a unique API, peculiar to the details of the underlying WSN protocol behavior. The software drivers attempt to wrap a generic API around these so that the application code resident on the MSP430 has a similar interface regardless of which protocol it is using to send packets. This interface includes three high-level functions: (1) join a network, (2) send packets to a gateway, (3) receive packets from a gateway.

Once a node joined a network, it waited for a command to configure itself and begin sending packets. Upon receipt of this command, a timer was established with a requested periodicity, and each time the timer fired, the node formatted and sent a packet to its radio for transmission over the WSN. Packets were 76 B[1] in length, with the first 2 B encoding a sequential counter and the remaining bytes set to zero. This process repeated up to a requested maximum count, at which point the node disabled its timer and began idling, waiting for a command to re-configure and begin a new experiment.

4.4 Wi-Fi Interference

We chose 802.11g as our representative Wi-Fi protocol for its ubiquity as well as its lower spectral efficiency compared to 802.11n. That is, a lower-throughput Wi-Fi flow would typically be needed to present detrimental interference over 802.11g than over 802.11n. Results should extend to 802.11n flows at higher-throughput levels, and this is a subject of ongoing investigation.

We used a D-Link Xtreme-N Gaming Router configured to operate in 802.11 g-only mode to serve as the access point for Wi-Fi network. This router interfaced through a wired Ethernet connection to a laboratory workstation and through a wireless connection to a laboratory laptop. The laptop used a Linksys Dual-Band Wireless-N USB adaptor as its 802.11g interface. The Iperf network testing tool was used to generate UDP traffic from the laptop to the workstation at requested throughput rates. As only one leg of this UDP flow is wireless (laptop to router), the achieved UDP flow rate represents all the Wi-Fi traffic present during the experiment.

4.5 Experimental Setup

Five WSN nodes and one gateway were used in each experiment along with the Iperf client laptop (sender of Wi-Fi traffic) and the 802.11g router (recipient of Wi-Fi traffic). The WSN gateway and the Wi-Fi router were generally co-located within the WHAT, as were node 2 and the Iperf client. Nodes 1 and 2 were placed in separate equipment bays along the port side of the WHAT, and nodes 4 and 5 were placed in separate equipment bays along the starboard side. Node 3 was perched on a beam running along the top of the forward end of the WHAT. ZigBee nodes were configured as ZEDs and ISA100.11a nodes were configured as non-routing devices. Each node thus reported directly to its gateway, forming a star networking topology.

[1]The maximum length of a packet payload in the Nivis ISA100.11a implementation.

Another advantage of using the WHAT for performance testing is that the effects of unknown, external RF systems can be isolated. A portable spectrum analyzer was used at the outset of experimental runs to ensure that the background RF environment was quiet. A noise floor below −70 dBm was required before experiments could begin.

4.6 Performance Metrics

We were interested in understanding how the performance of ZigBee and ISA100.11a was affected as the level of Wi-Fi traffic increased in a highly reflective, enclosed environment. As a performance metric, we focused on application-level message delivery rates. This allowed us to measure how successful a remote sensing application is likely to be when sending a packet back to its gateway. As mentioned above, each packet from a node contained a single element in a sequence of 16-bit counters. We computed the packet delivery rate by analyzing packets received at the WSN gateway following each experiment to determine the percentage of packets received successfully relative to those transmitted. Using packet length and the frequency of packet transmission, message delivery rate can be related to goodput (application-level throughput) through simple arithmetic.

We considered two configurations of the interferer for the ZigBee case as shown in Fig. 6: direct 802.11g interference and sideband 802.11g interference. ZigBee was set to 802.15.4 channel 17 in both cases. For direct interference, Wi-Fi traffic was sent on channel 6, placing the ZigBee center frequency (2.435 GHz) as close as possible to the Wi-Fi center frequency (2.437 GHz). For sideband interference, Wi-Fi traffic was sent on channel 4 (2.427 GHz). This placed the operational ZigBee channel in the region of the 802.11 spectral mask where signal energy is degraded relative to the energy at the center frequency.

In the ISA100.11a case, all 16 of the 802.15.4 channels were in use and potentially subject to 802.11 interference (Fig. 7). We considered the case where 802.11g is set to channel 6. This interferes with the use of 802.15.4 channels 17 and 18 near the center frequency and with the channels 16 and 19 in the sidebands.

We utilized WSN transmission rates of 1 packet every 1, 5, and 10 s, representing a high-to-moderate duty cycle on a low-data-rate network. We utilized 802.11 flows

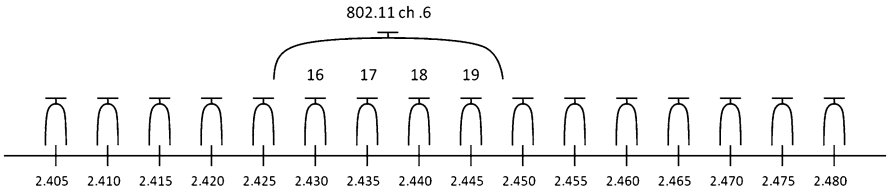

Fig. 6 ISA100.11a Wi-Fi interference channel mask

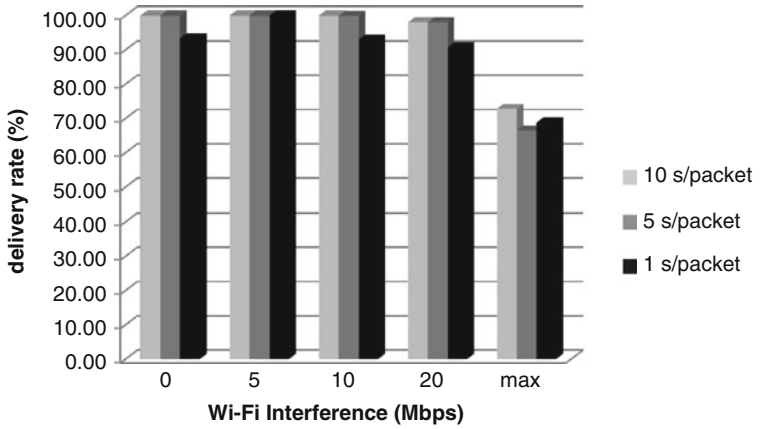

Fig. 7 ZigBee message delivery rate—direct interference

of 0 Mbps (no Wi-Fi interference), 5, 10, and 20 Mbps. We also utilized maximum single-flow throughput, which is around 25–30 Mbps in practice with the Wi-Fi equipment used in our experiments and which varies slightly with the amount of WSN traffic present.

In all cases, nodes transmitted packets at the desired rate for 1 h. This duration was chosen because it provided the ISA100.11a network manager with sufficient time to optimize its channel-hopping algorithm in response to detected interference. Experiments were repeated three times for each rate combination, and average message delivery rates were computed as a percentage of successful packet deliveries.

Finally, it is of note that we disabled end-to-end packet acknowledgements in the ZigBee network, since ISA100.11a does not support such a service. Once a packet reached its destination, no packet was returned to the original sender to indicate delivery success. Thus, the amount of traffic generated by the two networks is directly comparable.

4.7 Experimental Performance Results

Results from the ZigBee direct interference case are plotted in Fig. 7. It can be seen that for the lower WSN transmission rates of one packet every 5 and 10 s, ZigBee exhibited average message delivery rates near 100 % as Wi-Fi interference starts at 0 Mbps and increases to 10 Mbps. ZigBee achieved even a 98 % delivery rate for these lower WSN throughput levels as interference increased to 20 Mbps. When the Wi-Fi interference was increased to its maximum possible level, however, performance drops off. ZigBee delivery rates averaged approximately 66 % for the 5 s ZigBee transmission periodicity and approximately 73 % for the 10 s periodicity. Thus, in these cases ZigBee performance degraded with both increasing Wi-Fi and ZigBee transmission rates, as one would expect.

A more interesting trend emerges with the ZigBee transmission rate of 1 packet every second. Examination of the 0 Mbps case in which no Wi-Fi interference was present immediately gives one pause: ZigBee achieved approximately a 93 % delivery rate on average, where one would expect 100 %. Indeed, perfect delivery was achieved in the 5 Mbps interference case, and results degraded to near 90 % for 10 and 20 Mbps before finally falling to 69 % for the maximum interference case. While the latter results follow our intuition, the former is quite perplexing.

Scrutiny of the 15 individual node delivery rates across the three 1 s/packet experiments for 0 Mbps interference illuminates the problem: 14 of the averages were 100 %, while node 5 in the third test reported approximately a 1 % delivery rate. This result is due to a troublesome behavior of the ZigBee hardware: occasionally, a node declared itself an orphan and failed to re-associate itself with its coordinator, either permanently or intermittently. In such a case, excessive ZigBee traffic generated by the orphaned-and-searching node can be observed on a spectrum analyzer outside the operational ZigBee channel (17), facilitating identification of the anomaly. Using this technique, we identify three such instances in the ZigBee direct interference experiments, giving an orphaning rate of 1.3 %. In Fig. 8, the ZigBee direct interference delivery rates are displayed after being recomputed to remove these outlying cases. We see that in this case, the results are far more intuitive: delivery rates of approximately 100 % were achieved across all ZigBee transmission rates as Wi-Fi interference increased to 10 Mbps, with a degradation at 20 Mbps interference to 98 % for the 5 and 10 s/packet cases and 91 % for the 1 s/packet case. At maximum Wi-Fi interference, transmission rates decreased to approximately 73 % for the 1 and 20 s/packet cases and 66 % for the 5 s/packet case. It should be noted that achievable Wi-Fi rates varied across experiments in the maximum interference case, so average delivery rates may not be exactly comparable between the three ZigBee transmission rates.

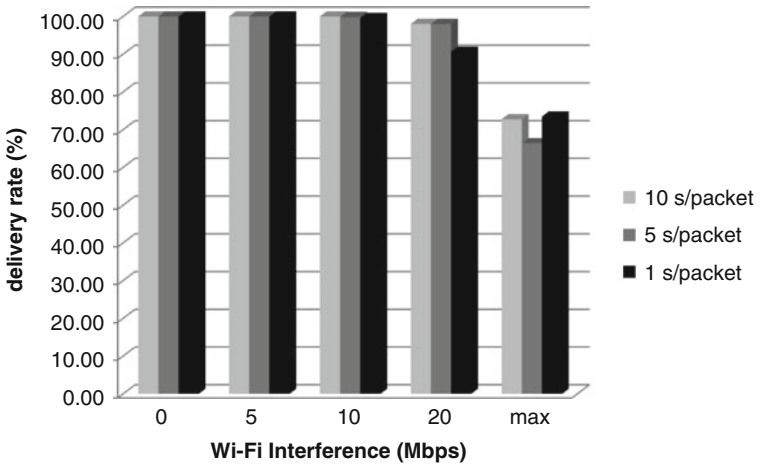

Fig. 8 ZigBee message delivery rate—direct interference (outliers removed)

Results from the indirect ZigBee interference experiments are plotted in Fig. 9 (note that the 0 Mbps interference results are simply repeated from the direct interference case). Again, delivery rates held at approximately 100 % for moderate levels of Wi-Fi interference, declined to the 95–98 % range as interference increased to 20 Mbps and then to the 69–78 % range as interference increased to the maximum achievable level. As before, three experiments were identified as each containing a single instance of an orphaned node, and the results with these outliers removed are shown in Fig. 10. Performance in both the direct and indirect interference cases is roughly comparable, both considering and removing the orphaned node cases. Thus,

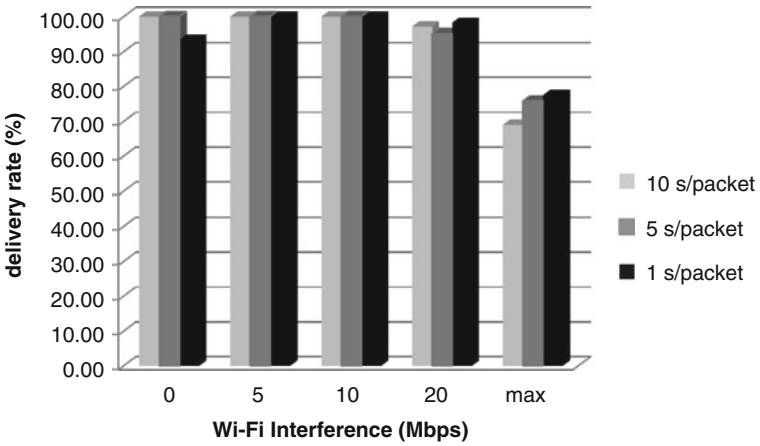

Fig. 9 ZigBee message delivery rate—indirect interference

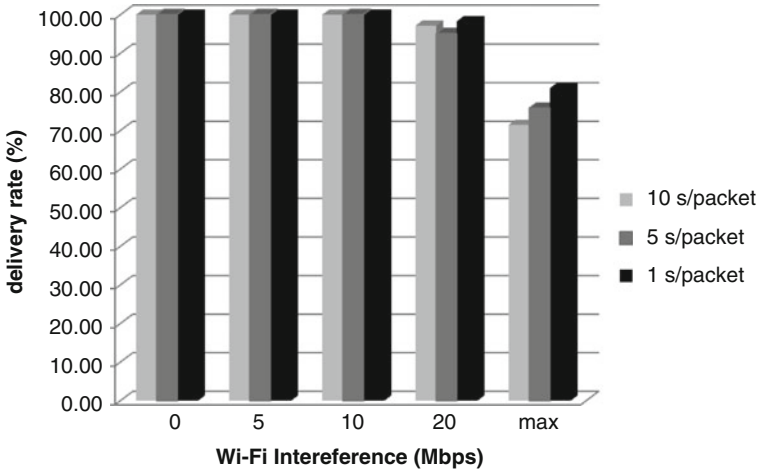

Fig. 10 ZigBee message delivery rate—indirect interference (outliers removed)

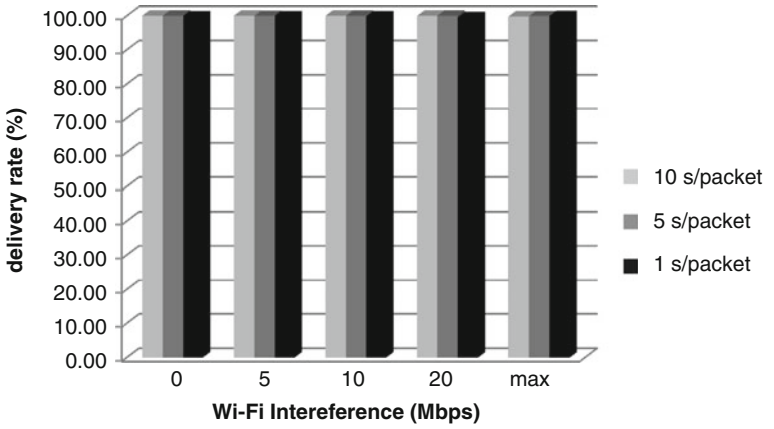

Fig. 11 ISA100.11a message delivery rate

it appears that in the experimental scenario, location of the ZigBee center frequency relative to the Wi-Fi center frequency has little impact on the overall delivery rates achieved by ZigBee.

Finally, in Fig. 11, we show the performance of ISA100.11a in the presence of the same Wi-Fi interference scenarios. ISA100.11a maintains the good track record of ZigBee under the low-to-moderate interference cases, but it really shines in the high and maximum interference cases, continuing to return a delivery rate in excess of 99 %. This indicates that, as a single channel becomes arbitrarily bad, ISA100.11a effectively exploits its frequency-hopping algorithm to find the free channels it needs to send its packets.

4.8 Summary of Performance Evaluation Results

Based on the results of this experimental performance evaluation, we can conclude that both ZigBee and ISA100.11a have their place in aerospace applications. Where moderate to high latency and packet loss can be tolerated, or where the spectrum can be carefully managed to guarantee non-interference for the operational lifetime of the WSN, ZigBee Pro should work well. It has the advantages of low unit cost, fast network formation, and the highest throughput achievable using an 802.15.4 PHY/MAC radio as the basis for a larger WSN stack.

ZigBee Pro can, however, be susceptible to high levels of 802.11g interference, resulting in delivery success rates of roughly 65–75 % for the use cases presented here. While 20 Mbps and higher of Wi-Fi interference may seem excessive, consider that a single MPEG-2 encoded HDTV-quality video stream requires 19.4 Mbps

of throughput [10]. Streaming video is quickly becoming a driving application for wireless communications, and it is reasonable to assume that the appetite for it will grow even as networking technologies advance. Thus, even 802.11n, with its greater spectral efficiency, is likely to be pushed toward the saturation point by mission requirements that require increasing levels of HD video coverage for situational awareness and public outreach.

Perhaps more troublesome is the tendency of the ZigBee Pro nodes used in these experiments to occasionally orphan themselves. It is possible that this behavior could be an artifact of the particular TI ZigBee Pro stack implementation we use, but it is notable that the TI stack is a ZigBee Alliance certified Golden Unit, and the behavior persisted across multiple versions of the stack during our development process. While the approximately 1 % orphaning rate presented here is not overly alarming, we observed that the likelihood of a node becoming an orphan seems correlated with the node's transmitter output power. Indeed, we collected a second set of direct and indirect interference ZigBee data using an output power of −3 dBm instead of +3 dBm, and we observed that the likelihood of a node becoming an orphan was closer to 15 %—a substantial increase. This greater tendency to orphan contributed in large part to the delivery success rates of 30–60 % that we observed in the higher interference cases. More alarmingly, these reduced average rates were often due to individual nodes delivering approximately 0 % of their packets over the course of an experiment. This is clearly unacceptable in applications where an individual node may be producing arbitrarily crucial data at any given time. Further work is required to explore this behavior across output power ranges for both WSN standards investigated here.

ISA100.11a, with its emphasis on high reliability, is likely the better choice when the interference environment is uncertain and not easily managed. Satellites and interplanetary probes, for example, may have the luxury of implementing the strict spectrum management and environmental determinism necessary to guarantee a highly reliable ZigBee deployment. In an evolving environment such as the ISS, however, equipment configuration changes over time, and crew members are constantly in motion. Both affect the quality of the RF channel and are not easily characterized. Moreover, as the capabilities of a vehicle such as the ISS evolve over a 20–30-year operational lifetime, new RF systems are likely to be added as technology advances. Although rigid spectrum management has served space agencies well for more than 50 years of spaceflight, the increasing proliferation of wireless devices on orbit will only serve to make this task more and more intractable. As COTS radios become more commonly used as a cost-saving measure, shared spectrum is likely to become the reality on orbit that it already is terrestrially. Use of precious bandwidth by one system cannot preclude its re-use by an unforeseen peer system years down the road.

5 History of Asset Management Techniques in Space Applications

This section provides an overview of asset management challenges in the context of long duration human space exploration missions, and the early efforts to evaluate the effectiveness of RFID technology in addressing those challenges. It sets the stage for Sect. 6, which describes the concept of ALM for human spaceflight systems.

5.1 Asset Management in Human Spaceflight

Inventory management is a critical function in many aspects of space operations, both in flight and ground segments. On the ground, thousands of controlled components and assemblies are stored in bond rooms across multiple centers and space agencies. These inventories are tightly controlled, typically using manual processes such as paper tags on individual items or small collections of identical items, such as small bags with screws or other fasteners. Bag inventory is often tracked by inking out the previous count and replacing with a revised count. In some instances, the process is aided with optical barcode technology.

Other ground operations also require complex inventories, including tracking all laboratory and office equipment with significant value. For example, at Johnson Space Center, a database containing approximately 36,000 items is maintained. Inventory audits of such equipment are currently very labor intensive and involve periodic room-by-room examinations and scanning of optical barcodes for each tagged item. Many inventory items require careful monitoring to assure, for example, that expiration dates are not exceeded. Replacement of consumables can also be highly critical; monitoring delivery and restocking of compressed gases and chemicals requires careful attention to assure, for example, that identical or compatible replacements are made.

Inventory management for flight applications entails an even greater degree of control, as improperly substituted items and early depletion of certain items can be catastrophic. Most short duration missions do not involve restocking, so resupply logistics are non-existent, but initial stocking and tracking of inventories is nonetheless quite important. For most long duration missions, resupply efforts are inherently complex, expensive, and infrequent. To date, the most extensive space-based inventory management operation has been the ISS. More detail on ISS inventory management, as well as a brief history of inventory management in human spaceflight, is provided below.

In early human spaceflight, such as the Apollo missions, inventories were kept on paper with diagrams showing inventory stowage locations. Even on NASA's Space Shuttle Orbiter, the crew was given hardcopy descriptions of item locations, without serial or model numbers. Figure 12 below shows an example of an Orbiter stowage

STS-109 MIDDECK STOWAGE
FORWARD LOCKERS

Food, Menu
FRED

Clothing, CDR
Clothing, CDR

Bags
- Helmet Stowage (2)
- Inflight Stowage, Restraint (10)
- Jettison Stowage (10)

Bungee, Adjustable (7)
Canister, WCS (Coffee Can)
Covers
- HUD (4)
- Parachute (7)

Hoses
- Personal Hygiene
- WCS Canister

(Cont)
Kits
- Comm
  - Cables
    - Comm, 4 ft
    - Comm, 14 ft
  - Mic, Handheld (3)
  - VHLS (2)
- Saliva
  - Mirror (2)

O2 Bleed Orifice
Pip Pin (12)
Pip Pin, Escape Pole (Spare)
Switch Guard, Computer
Tape
- Gray, 1 in.
- Gray, 2 in.

Ziplock, 8 in (20)
Ziplock, 12 in (8)

Air Bottles
Breaker Bar, 3/8 in.
Breakout Box
Filter, Waste Water Dump
Kit, RMS D&C
Turnbuckles

FDF I Bag, WVS

Food, Menu
Food, Menu

Food, Menu
Food, Menu

Clothing, PLT
Clothing, PLT

Fig. 12 Example (STS-109) of Space Shuttle Orbiter stowage list

location diagram. The Orbiter crew did have access to similar inventory information through an onboard laptop database, but additional assistance with item location was often required and entailed radio communication with Mission Control.

Although NASA's Skylab program provided valuable experiences in logistics management, the vast majority of knowledge pertaining to this topic in long duration space missions has been derived from experiences on the ISS. The inventory on ISS is vast, with about 20,000 items that are tracked with a software database application known as the Inventory Management System (IMS). In most cases, the inventory is densely packed, and packed in containers by people other than the end users. Within ISS, much of the inventory is mobile, both in the sense that intentional relocation is often required, and in the unintended sense when items that are insufficiently tethered relocate themselves. The item population is highly inhomogeneous and complex in nature, owing to three causes. First, the ISS serves both as living quarters and laboratory quarters, and the limited onboard volume often results in overlapping stowage space. Second, the types of experiments performed on ISS span human health and exploration, technology testing for enabling future exploration, research in basic life and physical sciences, and earth and space science. This research portfolio is fairly dynamic, and the supporting supplies and equipment tend to be quite diverse, irreplaceable, and expensive. Third, the crews are multi-national and are typically rotated on 6-month intervals. The criticality of logistics management is further underscored by the difficulty of resupply, the cost per unit mass to launch

Fig. 13 Cargo Transfer Bags (CTBs) on the International Space Station

cargo (about US$30 per gram in 2012), and the consequences of not being able to find the right items in a timely manner. Such consequences extend beyond monetary measures and might include loss of life, mission, or science.

Both flight and ground crews update the IMS database daily. A handheld optical barcode reader is used to update the onboard database, and the IMS application performs complex updates. The ground and flight segment databases are synchronized by uplinking and downlinking "delta files." The common transport apparatus for smaller items on the ISS is the Crew or Cargo Transfer Bag (CTB—see Fig. 13). The cargo ranges from crew clothing to office supplies, pantry (food) items, and personal effects. The CTBs are packed on the ground, and like items within a CTB are usually stored in Ziploc bags. For some cargos, items are tracked both at the Ziploc bag level and at the individual item level. For other cargo types, tracking resolution extends only to the Ziploc bag level. In addition, optical barcode tags are also affixed directly to the CTBs.

In the 2008 timeframe, approximately 500 CTBs were onboard the ISS at any given time. The CTBs are typically stacked several deep and are often restrained by crisscrossing elastic bungee lines, which are sometimes referred to as "bungee jails." Optical barcode-based inventory audits require approximately 20 min per day for each crewmember. The time required to inventory a single CTB is also about 20 min. The process requires removal of each Ziploc bag, removing each individual tagged item from a Ziploc bag, sealing the Ziploc bag, positioning and orienting the barcode to enable line-of-sight reading, and then re-bagging the items. The process is greatly complicated by the zero-g environment, which requires extra care to prevent items from floating out of reach.

Metallic drawers embedded in rack configurations constitute another common storage structure on the ISS. These drawers contain numerous items such as band-aids, hygienic wipes, tweezers, aspirin and other medications, test tubes, and laboratory supplies.

In addition to the tracking of smaller items packed in CTBs and metallic drawers, localization of larger pieces of equipment has, at times, also proven to be difficult. Such difficulties might arise, for example, when the sought item is stored behind other cargo or closeout panels. Although this situation does not occur often, crew time can be significantly impacted when it does. Moreover, inability to locate critical equipment in a timely manner can entail obvious safety implications.

5.2 Introduction of RFID for Asset Management in Human Spaceflight

In the early 2000s, NASA began investigating the use of RFID technology to assist with the onboard auditing of inventories and with searches for missing items [11]. In order to avoid the extra mass, volume, and logistics associated with battery-powered tags, NASA focused on two types of battery-free RFID tag technology. With both types, all of the necessary power required by the tag is derived through the electromagnetic radiation from an interrogator. One tag technology type utilizes surface acoustic wave (SAW) [12] piezoelectric crystals that convert incoming electromagnetic pulses, received from an antenna, into acoustical pulse energy that propagates along the surface of the crystal. A series of metallic reflectors are printed on the crystal's surface such that a small amount of the incident pulsed RF energy is reflected, from each reflector, back toward the antenna and then radiated back to the interrogator. The spacing between the reflectors on the crystal and the electromagnetic loading imparted by each reflector determine both the delay between pulses and the phase of successive pulses, hence providing the means to encode information on each tag. The second tag technology comprises an integrated circuit that is powered by rectifying the incident electromagnetic energy from the interrogator. Once powered, the integrated circuit impacts the tag antenna impedance such that power at the antenna port is received or reflected (backscattered) in an alternating fashion. The tag's unique identification determines the sequence in which power is received and reflected, and the interrogator subsequently decodes the signal from the tag.

In 2005, RFID studies were commissioned, including tests of the EPCglobal Class 1 Generation 1 standard, which is an example of an integrated circuit type RFID tag. The read accuracy of the standard was believed too low to warrant immediate pursuit in flight applications. During this timeframe, NASA funded the Haughton-Mars Project Expedition at Devon Island [13]. One of the four major objectives was to conduct field experiments with RFID tagging, reading, and automated database management to facilitate tracking of people, supplies, and

vehicles. An RFID gate experiment at a base tent demonstrated a factor of 2–3 in inventory management time, and the technology was used to track personnel movements. The work also led to follow-on concepts such as a self-aware "smart cabinets" and use of handheld readers for "fast checkout" of exploration vehicles.

Later tests in 2006 of SAW RFID [12] showed greater promise than the previous tests with EPCglobal Class 1 Generation 1. Two years later, the first spaceflight RFID tests were conducted as an International Space Station Detailed Test Objective. The test involved rotating a CTB in front of a fixed SAW RFID interrogator. In addition, the interrogator was used to locate a "hidden" piece of equipment. Even though the read accuracy was less than the target 95 %, the ease of audit, when compared with the optical barcode process, was found to be a significant improvement.

In parallel with the flight SAW experiments, in 2008 NASA conducted ground tests of the EPCglobal Class 1 Generation 2 (http://www.gs1.org/sites/default/files/docs/uhfc1g2/uhfc1g2_2_0_0_standard_20131101.pdf), or EPC-C1G2, standard for interrogation of CTB cargos. The second generation of this standard showed considerable improvement over the first and over SAW RFID for the interrogation of tags in the CTBs. An additional study commissioned for NASA's Orion [14] likewise found the Generation 2 implementation to be greatly superior to Generation 1 for interrogation of cargo within CTBs.

In 2011, NASA had to replace the obsolete onboard optical barcode scanners, and many newer barcode scanners also supported EPC-ClG2 interrogators. Within this same timeframe, NASA, the European Space Agency, and the Federal Space Agency (Russian), jointly accepted the EPC-C1G2/ISO 18000/6C standard as a recommended practice for space-based asset management (http://public.ccsds.org/publications/archive/881x0m1.pdf). This provided an opportunity to begin operational evaluation of RFID technology on the ISS. The process to achieve a minimum 95 % read accuracy of typical CTB contents comprises scanning the reader at a distance of about 2–4 in. from all six faces of the CTB for a total time of about 20 s. Although this represented a substantial savings compared to the 20 min barcode inventories, the process had to be further modified to prevent the incorrect association of tags read from CTBs residing in the background. The remedied process required relocation of CTBs to ISS regions with fewer background tags, increasing the per CTB audit time from 20 s to 2 min. Although other remedies were devised and tested, such as a mobile, trash can-sized, conductive textile structure for isolating individual CTBs, attention was already shifting to fully automate logistics management, for all future crewed spacecraft [15–17]. Figure 14 below shows a web application screen depicting multiple integrated RFID technologies applied to logistics management within a habitat test bed.

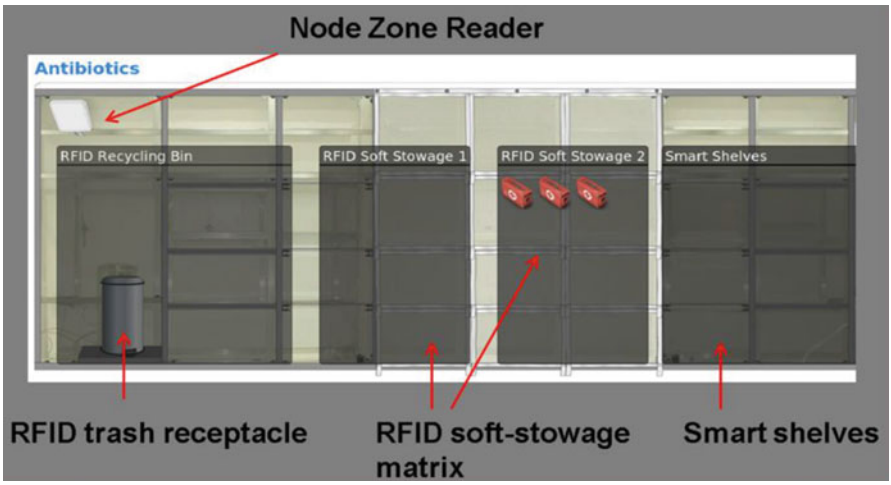

Fig. 14 Depiction of habitat test bed with multiple RFID technologies for asset management

6 Autonomous Logistics Management

6.1 Overview of ALM in Human Spaceflight

Although revised definitions, breadth of scope, and NASA technology roadmaps of ALM are in work at the time of this publication, the following definition captures the core meaning:

> ALM comprises the integrated tracking of location, availability, and status of all mission hardware and software to facilitate decision making with respect to consumables usage, spares availability, and the overall health and capability of human exploration mission vehicles and habitats and their subsystems.

6.2 ALM Technology Approach: Asset Tracking

In this section, we review ALM technology approaches currently at various levels of development for the purpose of achieving accurate inventory databases, including location as well as quantities. In order to facilitate general ALM solutions for human spacecraft, the applicable ALM technologies have been classified according to the following technologies: dense zone, sparse zone, and smart software applications. "Dense zone" is used to describe regions characterized by high densities of RFID-tagged items. Dense zone technologies typically involve readers connected to a feed or antenna within a drawer or enclosure that is electromagnetically shielded, in which case the enclosure essentially becomes an RF cavity. If properly shielded, good isolation can be achieved, thus reducing the likelihood of an external tag

Fig. 15 Prototype ISS RFID-enabled ISIS drawer

Fig. 16 ISS Human Research Facility pantry drawer interior with embedded reader in *upper left corner*

being erroneously associated with interior contents. Moreover, even relatively low cavity Q-factors permit RFID signal penetration into fairly dense populations of tags. Examples of an implementation of such a dense zone technology include the ISIS (International Subrack Interface Standard) RFID-enabled drawer shown in Fig. 15. This drawer incorporates an embedded RFID reader (EMBER) and an optional, externally mounted touchscreen, which permits on-demand interrogation. The interrogator can also be activated on a schedule or on events, such as the door opening or closing. Figure 16 shows another RFID-enabled enclosure, which is the ISS Human Research Facility 8PU pantry drawer. This RFID drawer was launched in 2013 and is expected to become operational in 2014.

Sparse zones are defined as spacecraft habitat regions exclusive of the dense zones. So, while these regions obviously include the open module spaces, they are also inclusive of the cracks, crevices, and air duct vents where items in zero-g sometimes accumulate. Technologies considered for sparse zones include zone readers as well as mobile readers, with many possible implementations of either group. For example, zone readers might refer to antennas with fixed coverage patterns that monitor tags in a specific zone. EPC-C1G2 readers commonly switch between two or more RF antenna ports, so that a set of antennas connected to a reader can cover multiple exclusive or overlapping regions. Zone readers also encompass arrays that can be electronically steered to monitor different regions or to track one or more items in motion. Mobile readers considered within spacecraft habitats include handheld readers, notebook- or computer-embedded readers, and wearable readers. Robotic-borne readers are also being evaluated. For example, the airborne free-flying robotic vehicle, SPHERES (Synchronized Position Hold Engage and Reorient Experimental Satellites (http://www.nasa.gov/mission_pages/station/research/experiments/311.html)), shown in Fig. 17, below, is being considered as a mobile RFID interrogator through joint work at NASA's Ames Research Center and Johnson Space Center.

Fig. 17 ISS payload SPHERES—a potential RFID reader platform

Fig. 18 REALM test bed: RFID Cargo Transfer Bags within a cell of a soft stowage matrix

To illustrate some of the challenges of RFID-based ALM, RFID-signal penetration experiments were conducted in the WHAT (Fig. 5), which is a ground analog of an aluminum horizontal cylinder habitat architecture measuring 6 m in length by 3 m in diameter. The objective of these experiments was to quantify the percentage of tags read from fixed antennas at the two entrances of the habitat, with the antennas switched sequentially, a transmit power of 1 W, and antenna gains of about 8 dBi. Two reader antennas at one entrance are shown in Fig. 18. The tags items are stowed in CTBs, and four CTBs are placed in a soft stowage matrix, as shown in the left and right sides of Fig. 18. Figure 19 shows the initial results from this test, with the left-side graph depicting the success of tags read from antenna 1 (Fig. 18), and the right-side graph depicting the success of tags read from antenna 2. In both graphs, each quadrant represents one of the spatially corresponding CTBs, and each sphere represents a group of tags within the CTB. Red-colored spheres denote that none of the tags in the group were read, green represents 100 % of the tags in the group were read, and other colors represent grades of success in between. It can be readily seen that a fixed reader transmitting 1 W of power is incapable of reading all tags within a dense cluster at about 3 m distance.

Introduction of a mobile interrogator, such as an RFID-enabled free-flyer, should increase the number of tags read within the soft stowage matrix, and such tests are in progress at the time of this writing. Nonetheless, it is unlikely that a

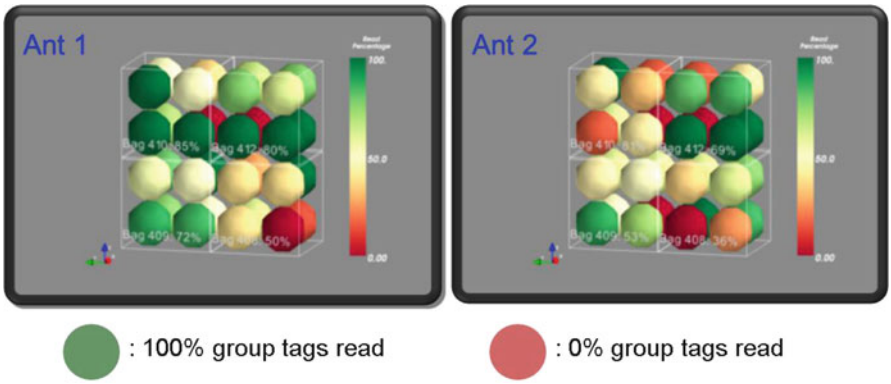

Fig. 19 Tag read success corresponding to the setup of Fig. 18

space-borne ALM system will rely entirely on RFID on-demand reads of all tagged items. Although technically such a system is possible, other considerations will probably render such a system improbable. For the ISS problems associated with retro-fitting an in-flight vehicle complicate a complete on-demand RFID system. For example, prime vehicle power is not readily available everywhere, an embedded or zone reader might be desired. For future habitats beyond low-Earth orbit, size, weight, and power constraints must be considered in the selection of ALM technologies. For example, an abundance of RF cable throughout the vehicle is not likely to be consistent with mass goals. Higher transmit power is likely to be incompatible with power budgets as well as safety and interference considerations. The mobile free-flyer will likely be employed to reduce the mass associated with the RFID solution, and the ability of the free-flyer to closely approach the RFID targets should reduce the required RF transmit power. Other technologies of an autonomous free-flyer will have to be matured to realize this concept.

So, given the complexity of the space habitat environment and the difficulties in reading dense populations of tags from even relatively short distances, for both the ISS and future space habitats, it is likely that the third technology area, smart software applications, will be leveraged to relax requirements associated with the dense and sparse zone technologies. For example, instead of maintaining that all tagged items be readable at any time, the smart application might infer the location of certain items. Of the three technology areas, this one is the least mature for spaceflight applications and related studies have just begun.

Finally, incorporating sensor telemetry into the capabilities offered by passive RFID tags is a natural progression, allowing RFID tags to function similarly to the active 802.15.4-based networks discussed earlier in this chapter.

References

1. Spacecraft Onboard Interface Systems–Low Data-Rate Wireless Communications for Spacecraft Monitoring and Control. Report Concerning Space Data System Standards, CCSDS 882.0-M-1. Magenta Book. Issue 1. May 2013
2. Wireless Network Communications Overview for Space Mission Operations. Report Concerning Space Data System Standards, CCSDS 880.0-G-1. Green Book. Issue 1. CCSDS, Washington, DC, December 2010
3. IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Std. 802.15.4™-2011. IEEE, New York, 2011
4. ZigBee Specification. ZigBee document 05347r17. ZigBee Alliance, San Ramon, 2007
5. Wireless Systems for Industrial Automation: Process Control and Related Applications. ISA-100.11a-2011. ISA, Durham, 2011
6. Industrial Communication Networks – Wireless Communication Network and Communication Profiles – WirelessHART™. IEC 62591 Ed. 1.0 IEC, 2010
7. Technical Overview of Time Synchronized Mesh Protocol (TSMP). White paper. Dust Networks
8. 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. IEEE Std. 802.11™-2012. IEEE, New York, 2012
9. Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model. International Standard, ISO/IEC 7498-1:1994, 2nd edn. ISO, Geneva, 1994
10. Thonet G, Allard-Jacquin P, Colle P (2008) ZigBee-Wifi Coexistence, Schneider Electric White Paper and Test Report
11. Shull S, Powers A, Schellhase A (2005) Use of radio frequency identification technology for International Space Station inventory tracking, Biennial Research and Technology Development Report, NASA JSC October 2005
12. Brown P et al (2007) Asset tracking on the International Space Station using global SAW Tag RFID technology. In: IEEE ultrasonics symposium
13. de Weck O, Simchi-Levi D (2006) Haughton-Mars Project Expedition 2005, Final Report, NASA/TP-2006-214196, January 2006. URL: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20060013129.pdf
14. Jones E et al (2008) RFID in space: exploring the feasibility and performance of Gen 2 tags as a means of tracking equipment, supplies, and consumable products in cargo transport bags onboard a space vehicle, NASA Final Report, October 2008
15. Fink PW, Barton RJ, Wagner RS, Ngo PH, Gifford KK (2009) Unified communications for space inventory management. In: AIAA space 2009 conference proceedings, September 14, 2009
16. Chu A (2010) Assessment of RFID read accuracy for ISS water kit. JSC-65920. NASA JSC, Houston, August 2
17. Chu A (2010) RFID portal test at the wireless habitat test bed. JSC-64867. NASA JSC, Houston, July 28, 2010
18. Wagner RS, Dufour JF, Braham SP, Barton RJ (2013) An Overview of the Smart Sensor Inter-Agency Reference Testbench (SSIART), IEEE Aerospace Conference, March, 2013
19. Wagner RS, Barton RJ (2012) Performance Comparison of Wireless Sensor Network Standard Protocols in an Aerospace Environment: ISA100.11a and ZigBee Pro, IEEE Aerospace Conference, March, 2012

Predictive Data Reduction in Wireless Sensor Networks Using Selective Filtering for Engine Monitoring

David James McCorrie, Elena Gaura, Keith Burnham, Nigel Poole, and Roger Hazelden

Abstract In a wireless sensor network, transmissions consume a large portion of a node's energy budget. Data reduction is generally acknowledged as an effective means to reduce the number of network transmissions, thereby increasing the overall network lifetime. This paper builds on the Spanish Inquisition Protocol (SIP), to further reduce transmissions in a single-hop wireless sensor system aimed at a gas turbine engine EGT monitoring application. A new method for Selective Filtering of sensed data based on state identification has been devised, using a skewed double exponentially weighted moving average filter for accurate state predictions. Low transmission rates are achieved even when significant temperature step changes occur. A simulator was implemented to generate flight temperature profiles similar to those encountered in real-life, which enabled tuning and evaluation of the algorithm. The results, summarised over 280 simulated flights of variable duration (from approximately 58 min to 14 h), show an average reduction in the number of transmissions by 95, 99.8, and 91 % in the take-off, cruise, and landing phases, respectively, compared to transmissions encountered by a sense-and-send system sampling at the same rate. The algorithm generates an average error of 0.11 $\pm$ 0.04 °C over a 927 °C range.

D.J. McCorrie (✉) • E. Gaura • N. Poole
Cogent Computing ARC, Coventry University, Priory Street, Coventry CV1 5FB, UK
e-mail: mccorrid@coventry.ac.uk; e.gaura@coventry.ac.uk; n.poole@coventry.ac.uk

K. Burnham
Control Theory and Applications Centre, Coventry University, Priory Street, Coventry CV1 5FB, UK
e-mail: k.burnham@coventry.ac.uk

R. Hazelden
TRW Conekt, Stratford Road, Solihull B90 4GW, UK
e-mail: Roger.Hazelden@trw.com

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_6

1 Introduction

Wireless sensing and data transmission technologies are now generating much interest for sensing and instrumentation applications in aircraft, both within academia and industry. The main applications are for monitoring the performance and structural integrity of rotating and fixed structural components, as well as sub-systems such as engines and landing gear. For aircraft the key advantages of wireless sensors over conventional wired devices include:

1. Enabling more sensors to be fitted (giving better control and monitoring of systems) without the weight penalty of large and bulky wiring harnesses
2. Easy installation (including retrofit) and removal of sensors
3. More flexible and convenient monitoring of rotating and moving structures

Recent technology improvements have enabled robust, reliable, long-life wireless systems to be deployed in other industries and these are now inspiring development for the aerospace sector.

For example, the automotive industry is now mass-producing wireless tyre pressure monitoring sensors. These fit inside the tyre and measure pressure (0–4.5bar) and temperature (-40 to $+125\,^{\circ}\mathrm{C}$) [17]. Typically these sensors are powered for a 10-year, 150,000-mile life by a non-replaceable battery, and withstand $>1{,}500\,\mathrm{g}$ centrifugal loading and 100 g shock. A proprietary protocol is used to transmit data from each wheel back to a central receiver inside the car. Other mass-produced wireless sensors include temperature and humidity sensors for home weather stations, wireless room thermostats for domestic heating systems, and various switching and sensing devices for commercial building management systems.

Power management is critical to successful deployment of a wireless sensing system. Autonomous wireless sensors will need to derive their power from batteries and/or energy harvesting from the environment (in the form of e.g., light, heat, or vibration). As batteries are not allowed in many parts of the aircraft, all the power for the wireless sensors will need to come from energy harvesting, and the technologies for this are still under development. The sending or receiving of data via the wireless link is usually the biggest consumer of power, so it is important to minimise the amount and frequency of data transmission. The amount of power required to activate and read the sensing element(s) must also not be underestimated, so the sensor sampling strategy must be carefully thought out. If data is to be relayed from one sensor node to another, care must be taken not to overload any individual node with messages which could exhaust its energy supply or lead to data corruption or interference.

In the UK alone several large projects are currently reporting positively on wireless sensor network (WSN) based developments for this sector. Such projects include the Wireless Intelligent Sensing Devices Project (WISD) [4] which demonstrated a WSN deployment on a helicopter rotor in-flight, and, Wireless Data Acquisition in Gas Turbine Engine Testing Project (WIDAGATE) [10] which demonstrated high yield acquisition of data from wireless sensors on a cold engine.

Another is the Wireless Technology for Novel Enhancement of Systems and Structures Serviceability (WiTNESSS) project [11] which looked more generically at multi-purpose WSN deployments on aircraft. The project successfully delivered a generic support architecture for large WSN deployments as well as functional demonstrators of the technology.

Following the successful outcome of these and other research projects, wireless measurement is now being considered as an attractive option particularly for aircraft engines. It could reduce the complexity, weight, and cost of engine monitoring as well as provide increased sensor density, higher data rates, and enhanced sensor deployment flexibility [18].

Whilst *in-flight* engine monitoring and control based on WSN is a long-term aim, in the short-to-medium term, the use of wireless instrumentation is envisaged mainly for engine test and development. A wired engine test harness, for example, consists of over 3,000 sensors (of which there are over 1,000 thermocouples) connected by 12 km of cable, making the test infrastructure costly and location dependent. There is thus a strong business case to offer more flexibility for the engine instrumentation process.

The research the authors have engaged with is towards *in-flight* engine monitoring. It aims to create highly efficient autonomous thermocouple (HEAT) system prototypes for EGT monitoring. A primary goal is to deliver robust, long lived wireless thermocouple systems which sample at rates of over 1 Hz. The drive towards long lived, low power wireless systems is essential to the domain, given that nodes need to run until the next service of the engine to avoid disruption of operation. Current battery technology operational temperature limits precludes the use of battery powered systems in this application. Heat and vibration energy harvesting has been suggested as a means to overcome both of these limitations [16]. Reducing power consumption would allow measurement systems to be powered by existing energy harvesting technology [1].

The HEAT hardware system developed is composed of multiple battery powered thermocouple sensor nodes, located around the circumference of an engine casing. Figure 1 shows a prototype sensor network deployed for testing on a cold Rolls-Royce Trent 900 gas turbine engine. Each sensor node can sample at 1–5 Hz from a Type-K thermocouple and reports the EGT data back to the sink node. Nodes use the CC2530 low powered ZigBee radio [15] and a MAX6675 cold junction compensation chip [9]. The nodes support Low Power Sleep.

When a HEAT sensor node's power consumption was analysed, it was found that a transmission of a single sample accounts for 75 % of the total consumption, whilst sleeping and sensing account for 24 % and 1 % of the consumption, respectively. Figure 2 presents a node's current draw over a 1 s cycle. Reducing the number of transmissions will thus provide considerable savings in power consumption and increase the network lifetime.

The contribution brought by this paper consists of a method for drastically reducing RF transmissions of EGT wireless sensing data. The work builds on previous research in the area of dual prediction schemes (DPSs) for wireless sensor systems. In particular, we propose an application bespoke state prediction and

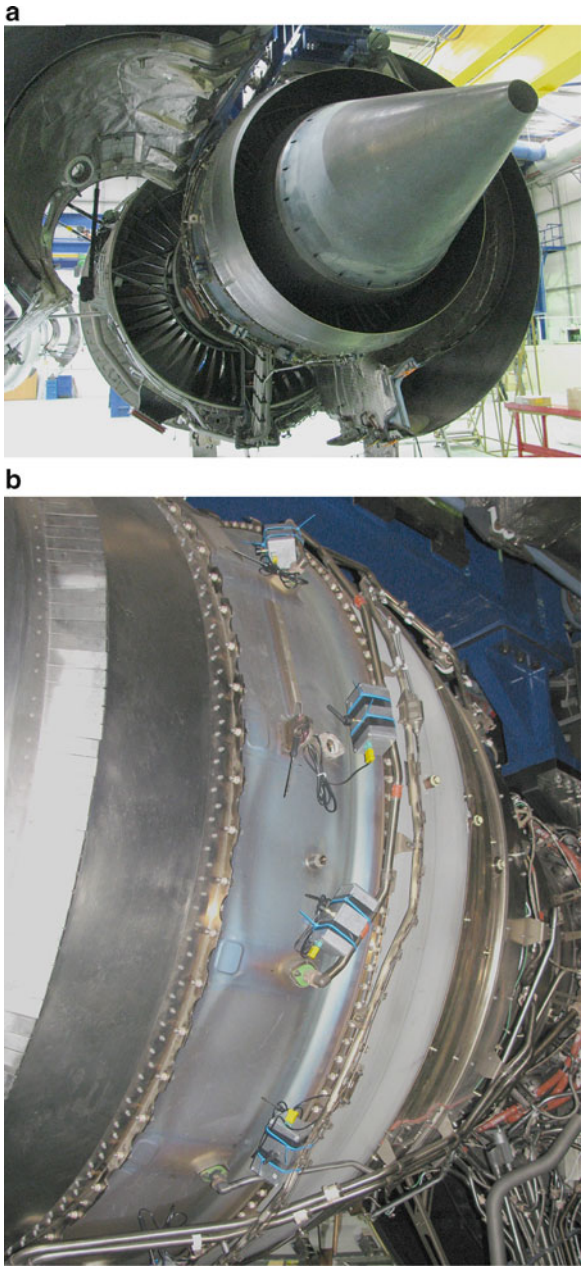

Fig. 1 HEAT nodes deployed on a Rolls-Royce Trent 900 gas turbine. (**a**) Engine rear view. (**b**) Nodes at the standard test point locations

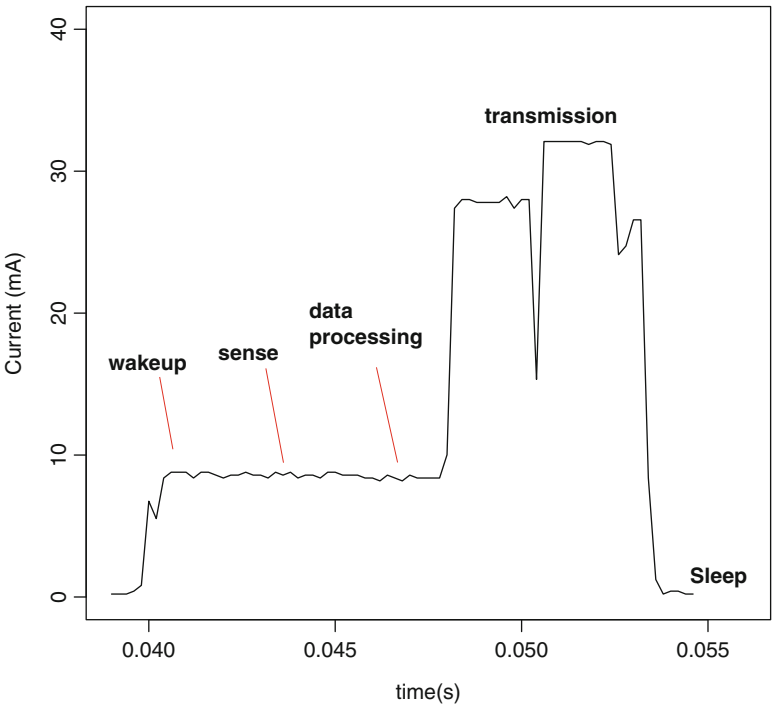

Fig. 2 A 1 s sensing, communication, and sleep cycle for a HEAT node

selective filtering method to be used in conjunction with the Spanish Inquisition Protocol (SIP) [5] (described in Sect. 4). The proposed method is generally suitable for applications where the data stream exhibits a combination of steady states and significant step changes. The performance of the algorithm developed is tightly correlated with the absolute values of the sensor readings. Thus in order to tune and evaluate its performance an EGT simulator was also developed.

The remainder of this paper is structured as follows: Sect. 2 briefly describes related work in the area of data reduction for wireless sensor systems. Section 3 describes the simulator used to evaluate the proposed algorithm. Section 4 considers the SIP—the fundamental data reduction method within the HEAT system. Section 5 presents the integration of the Selective Filtering (SF) algorithm with the SIP. Results are given in Sect. 6, and concluding remarks are presented in Sect. 7.

2 Related Work

Whilst it is important to reduce the number of transmissions in a wireless networked system, it is equally important to accurately capture the phenomena being

monitored. For the application at hand, transmission reductions through sampling rate reduction cannot be considered; HEAT nodes need to ensure sampling at least at 1 Hz. Data compression and reduction is however an alternative approach to long lived networks with specified requirements for data quality.

Many methods for data compression and reduction within a WSN have been proposed [14]. Dictionary based compression algorithms such as lossless entropy compression (LEC) [8] provide a byte reduction on a per packet basis, although would require sensor readings to be buffered in order to reduce the number of transmissions. Schoellhammer et al. [13] model the sensor readings using a linear model, by buffering the readings until the residual error of a linear fit exceeds a predefined error threshold. Due to the real-time requirement of the HEAT system, buffering approaches are not applicable.

The class of DPS algorithms solve the problem of having to buffer sensor readings. By using a model on the sensor node and the sink node, new readings can be predicted without having to transmit further data. When the error between the models exceeds a tolerable threshold, new model parameters are transmitted. Large reductions in the numbers of transmissions can be accomplished using the DPS approach, while keeping a real-time knowledge of the system's state [2]. A variety of implementations exist for the approach described above. Jain et al. [7] use a Dual Kalman filter (DKF) as the system model. Santini and Römer [12] use a least mean-square (LMS) filter. Le Borgne et al. [3] present a general method for adaptively selecting the model using a statistical procedure termed *racing*. Such a method allows the most optimal model, from a discrete set of models stored on the sensor node, to be learned over time. Although considerable transmission reductions are reported for a variety of case studies, none of these approaches respond well to step changes in the sensed data. (A summary of the results of these algorithms can be found in [3, 5].) These works have, however, inspired the authors towards the reported developments.

3 EGT Simulator

The EGT simulator attempts to produce flight-like data to be used in evaluating the proposed algorithms. The simulator consists of an EGT phenomenological model (PM) and a thermocouple sensor model (SM). For the purpose of this paper, the output from the PM, denoted s, is considered as the actual EGT. The output from the SM is taken to be the thermocouple readings, denoted y. The simulator architecture can be seen in Fig. 3 and a representative flight EGT profile is shown in Fig. 4.

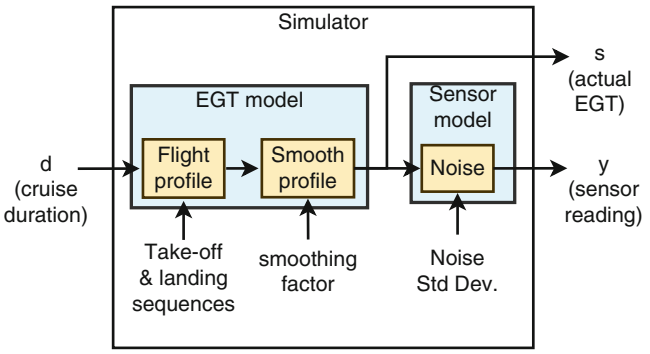

Fig. 3 Simulator architecture

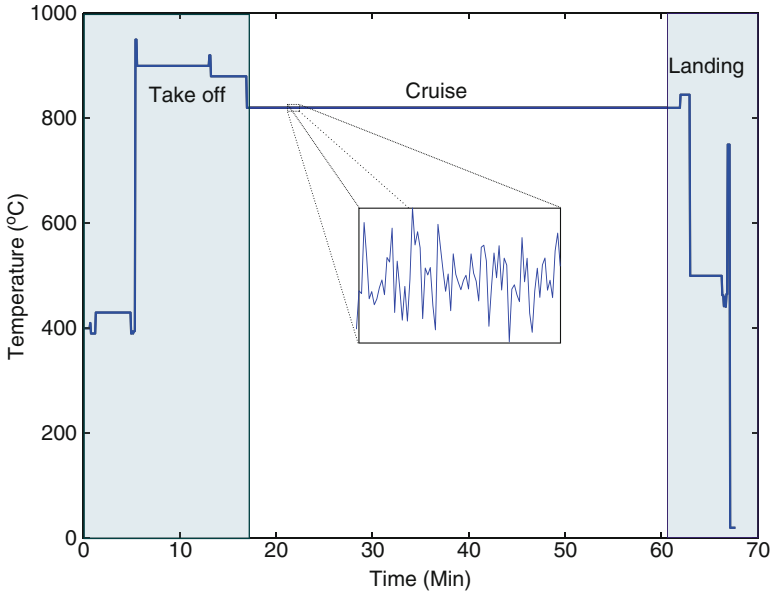

Fig. 4 Typical 70-min flight with three main phases: take-off, cruise, and landing. The *zoom box* shows the superimposed sensor noise

3.1 Flight Profile

Flight profiles are split into three main phases: take-off, cruise, and landing. At the start of the flight the EGT is relatively low. During take-off, the EGT rises sharply to around 1,000 °C where it remains fairly constant for the majority of the flight. During the final landing phase, the EGT decreases to ambient temperature, with some oscillation to replicate observed practice.

Table 1 Flight sequence take-off phase, which is identical in every simulated flight

Phase	EGT (°C)	Duration (s)
Start engine	400	40
	410	5
	390	30
Taxi	430	220
Manoeuvring	390	5
	394	7
	390	4
	394	8
	395	3
Take off	950	10
	900	430
	920	10
Pull back	880	224

Table 2 Flight sequence landing phase, which is identical in every simulated flight

Phase	EGT (°C)	Duration (s)
High idle	500	200
Landing	463	10
	442	5
	459	7
	441	4
	465	8
Reverse thrust	750	15
Engine off	20	30

For the purpose of this simulation study, the take-off and landing sequences are considered to be consistent. These two EGT sequences can be found in Tables 1 and 2 for take-off and landing, respectively.

The cruise sections of the flight are of variable duration, which is regarded as an input to the flight profile simulator, and denoted d. To simulate a typical cruise phase of the flight, a nominal EGT of 830 °C is chosen. Furthermore, to accommodate different altitudes and weather conditions, a uniform distribution of the EGT about the nominal value of $\pm$ 30 °C is randomly selected at the start of each cruise section.

Over the course of the cruise phase, there are additional randomly implemented transient adjustments to the flight pattern. These adjustments are made at random times during every 90–120 min, i.e. in a uniformly distributed random manner. These transients consist of an increase in EGT by 25 °C for 60 s. Following this transient a further cruise section randomly chosen from the uniform distribution and the cycle repeats until the cruise phase is complete.

3.2 Smoothing

Once the flight profile sequences have been generated, smoothing is applied to provide a gradual transition between flight phases, thus facilitating a realistic EGT of the engine between the various steady state sections. To realise this, an exponentially weighted moving average (EWMA) filter [6] with a value for the smoothing factor, denoted α, of 0.4 is applied to the generated time series.

3.3 Sensor Model

Meggitt (UK) Limited have provided thermocouple temperature samples from a Viper Engine test run (Langley, 2011, Thermocouple temperature data sets, private communication). From this it has been found that the standard deviation of the noise at cruise temperature is around 0.52 °C. Consequently, Gaussian noise, with a standard deviation of 0.52 °C, is added to the output of the PM to simulate realistic thermocouple data.

4 Spanish Inquisition Protocol

The Spanish Inquisition Protocol is a generic data reduction algorithm developed by Goldsmith and Brusey, designed to reduce the number of transmissions in a WSN [5]. The underlying principle is that transmissions should only be made when sensor readings are not as expected, i.e. when some pre-defined change threshold is violated. By using a model of the system, sensor readings can be reconstructed at the sink node within a defined error tolerance.

A model state vector, denoted X_t, is calculated on the sensor node and shared with the sink. This state vector is used on both the sensor and sink to predict the future system state at every time step. As the predicted state, denoted X'_t, diverges from the actual measured state, so the reconstruction error, defined as $\epsilon = |X'_t - X_t|$, increases. When this error exceeds a defined threshold, a new model state vector is calculated and shared, see Algorithm 1.

4.1 Piecewise Linear Model

The data reduction that SIP provides is dependent on model quality and the calculation of the predicted state, denoted X'_t. In this paper a piecewise linear model is used, as demonstrated in [5]. The model state vector is defined as

Algorithm 1 Pseudocode for node and sink in the Spanish Inquisition Protocol. (Reproduced from Goldsmith and Brusey [5] with permission)

Node:
$y \leftarrow$ querysensor()
$\mathbf{x}' \leftarrow$ estimate new state $(y, \mathbf{x}_{\text{old}}, t_{\text{old}})$
$\mathbf{x}_{\text{sink}} \leftarrow$ predict sink state $(\mathbf{x}_{\text{sink}}, t_{\text{sink}})$

if $|\mathbf{x}' - \mathbf{x}_{\text{sink}}| > \varepsilon$:
    transmit $(\mathbf{x}')$
    $\mathbf{x}_{\text{sink}} \leftarrow \mathbf{x}'$
    $t_{\text{sink}} \leftarrow t$
$\mathbf{x}_{\text{old}} \leftarrow \mathbf{x}'$
$t_{\text{old}} \leftarrow t$

Sink:
[On receipt of new state estimate($\mathbf{x}$)]
    $\mathbf{x}_{\text{sink}}(t) \leftarrow \mathbf{x}$; $t_{\text{last}} \leftarrow t$

[Estimate value for time(t)]
    if $t \geq t_{\text{last}}$
        predict from $\mathbf{x}_{\text{sink}}(t_{\text{last}})$
    else
        interpolate from neighbouring $\mathbf{x}_{\text{sink}}$

$X_t = (\overline{x}_t, \Delta\overline{x}_t)^T$, where $\overline{x}_t$ denotes the predicted value and $\Delta\overline{x}_t$ denotes the predicted rate of change. It is then possible to predict future states between transmissions using

$$X'_t = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} X_t.$$

5 Signal Estimation Using Selective Filtering

The more accurate the predicted rate of change, the longer the state prediction will remain within the allowable error range. Hence, the more predictable the signal the greater the reduction in transmissions. It is important, therefore, to remove as much noise from the signal as possible, which is done by filtering the data.

Selective Filtering (SF) is a rule based method of selecting between multiple filters, each optimised for a different part of the signal. This allows a greater reduction in the effect of noise on signal readings.

The signal is modelled as a sequence of states and transitions, which allows them to be filtered separately. State transitions are determined by a predefined set of rules specific to the application. Each state has its own filter, selected from a bank of filters and its own predictor, which estimates the rate of change.

The state machine used for this particular application is shown in Fig. 5. An example of the states during a typical flight can be seen in Fig. 6. The remainder of this section looks at the state identification rules and the predictors used.

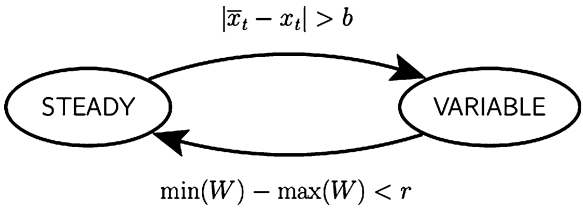

Fig. 5 Flight state machine

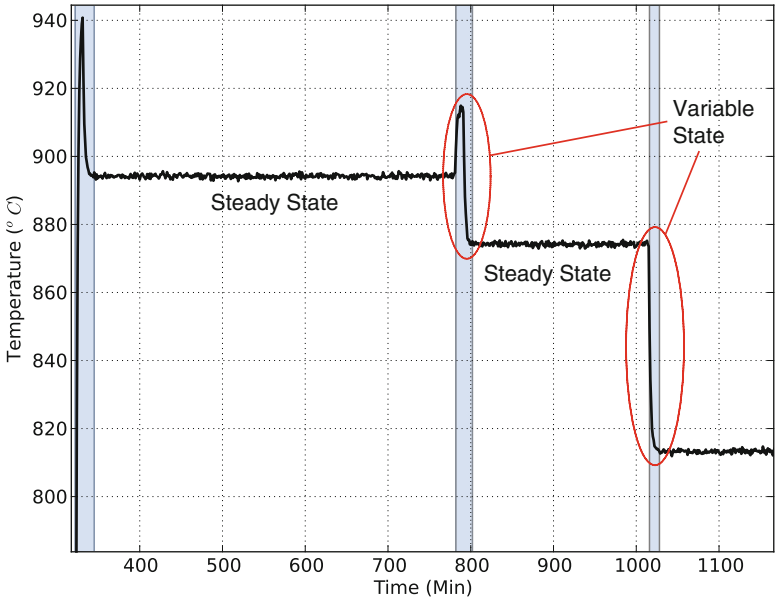

Fig. 6 System states within the flight profile

5.1 *Identifying a Steady State*

Steady states are defined as periods of low rate of change. A state change is identified when a sensor reading deviates from its expected value by more than a specified threshold. The initial steady state has been defined as the state when the aircraft starts with the engine in idle prior to taxiing. While in steady state, $\overline{x}_t$ is found by filtering the sensor sample x_t with an EWMA filter having an α value of 0.01.

As the EWMA filter in this state responds slowly to signal changes, $\overline{x}_t$ is taken as the expected steady state value. An envelope bounded by an upper and lower breakout threshold, denoted b, is formed as shown in Fig. 7. A state change is triggered when $|\overline{x}_t - x_t| > b$. An appropriate value for the breakout threshold b for this particular application was found to be 1.6 °C, as this is adequate to avoid triggering due to noise (standard deviation of the noise is 0.52 °C, see Sect. 3.3), yet small enough to catch all step changes.

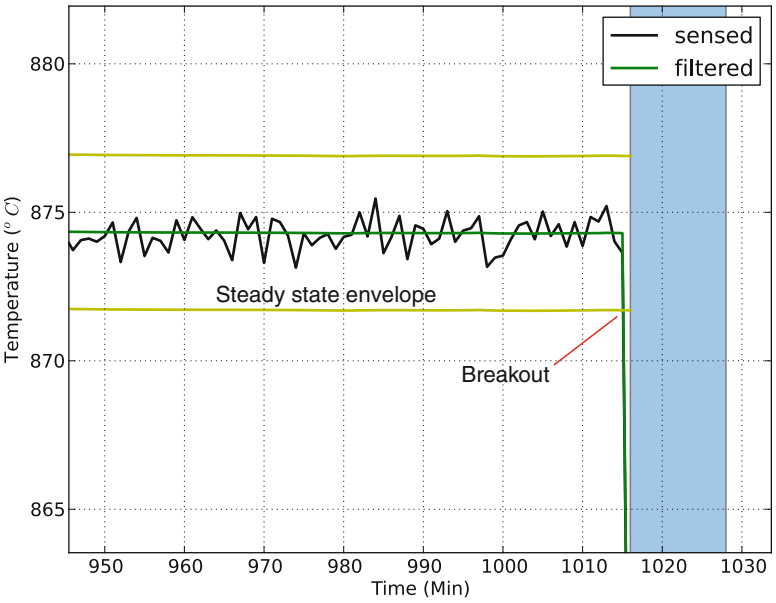

Fig. 7 Showing the breakout from the cruise envelope that signals state change

5.2 *Identifying a Variable State*

A variable state is defined as a period of large rates of change in the data. During this state a second filter can be used, or alternatively, samples can remain unfiltered.

A moving window, denoted W, of the n most recent samples is stored. A steady state resumes when the range of values in the window is less than a given threshold, r, such that $\max(W) - \min(W) < r$, see Fig. 8.

5.3 *State Prediction*

Accurate state prediction depends on the calculation of the two components in the SIP model state vector. The SF method improves the quality of the prediction, denoted $\overline{x}_t$, by reducing signal noise. This section considers the method for calculating the rate of change component of the state vector. When the rate of change is accurate, the model will take longer to diverge from the actual sensor readings, resulting in fewer transmissions. Recognising that the rates of change in the variable and steady states are different, improvement can be gained by reinitialising the model state vector on each state change.

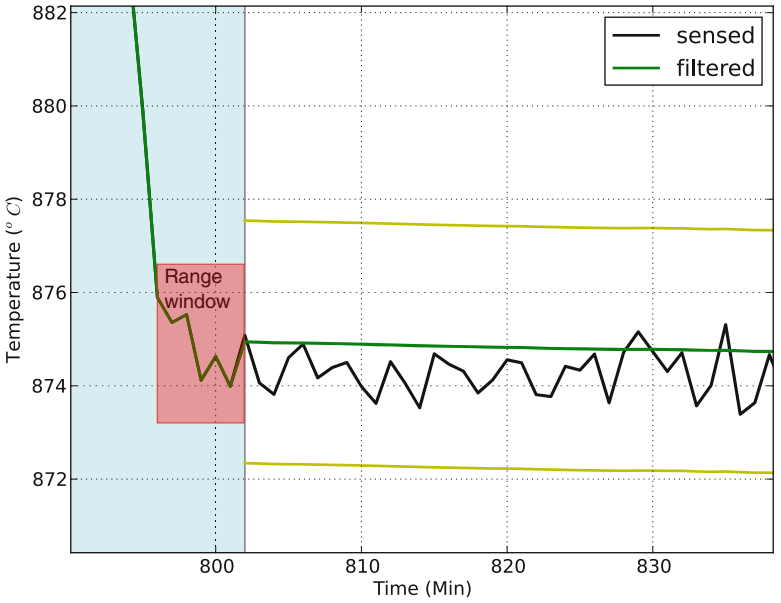

Fig. 8 Showing the resumption of stable state

In the original SIP $\Delta\overline{x}_t$ is calculated using the current and last value of the estimate, denoted $\overline{x}_l$, i.e.

$$\Delta\overline{x}_t = \frac{\overline{x}_t - \overline{x}_l}{t}.$$

5.3.1 Steady State

In a steady state, there is, by definition, little change in the temperature. The best estimate of the initial $\Delta\overline{x}_t$ was found to be 0.

5.3.2 Variable State

In order to gain a more accurate prediction for X_t, and to add some smoothing, a double EWMA (dEWMA) filter [6] was used in the variable state. When transitioning from a steady to a variable state, the initial expected value is defined as $\overline{x}_c = x_c$, where c denotes the time of the transition to the variable state. Here the expected rate of change is calculated as

$$\Delta\overline{x}_c = \frac{x_c - \overline{x}_l}{c - l},$$

where $\overline{x}_l$ is the last expected value from the preceding steady state at time l, and x_c is the sensor reading at the time of transitioning. Subsequent predicted values are calculated as

$$\overline{x}_t = \alpha x_t + (1-\alpha)\left(\overline{x}_{t-1} + \Delta\overline{x}_{t-1}\right),$$

where α denotes the value of the smoothing factor for the predicted signal value. Moreover, the rate of change is also filtered and updated at each time step using the calculation,

$$\Delta\overline{x}_t = \beta\left(\overline{x}_t - \overline{x}_{t-1}\right) + (1-\beta)\,\Delta\overline{x}_{t-1},$$

where β denotes the smoothing factor for the predicted rate of change.

5.4 Biased Rate of Change Utilising Prior Knowledge

While the rate of change calculated with the dEWMA filter provides more accurate predictions of the changes in the value of the signal than the original SIP method, they are generally over estimated. Since a variable state starts with a large change in temperature followed by a convergence to a steady state (near zero rate of change), it can be assumed that there is a high probability of future rate of change to be smaller than the rate of change at the previous time step.

Based on this assumption, it is postulated that $\Delta\overline{x}_t$ can be better predicted with

$$\Delta\overline{x}_t = \gamma\left(\beta\left(\overline{x}_t - \overline{x}_{t-1}\right) + (1-\beta)\,\Delta\overline{x}_{t-1}\right),$$

where γ determines the strength of the bias of the rate of change towards a steady state. Figure 9 shows the reconstructed signal before and after bias. Moreover, an appropriate value of β for this particular system was found to be unity, placing the greatest importance on the new rate of change estimate. The rate of change calculation can then be simplified to

$$\Delta\overline{x}_t = \gamma\left(\overline{x}_t - \overline{x}_{t-1}\right).$$

6 Results

This section presents the simulated results, using the simulator described in Sect. 3. A comparison is made between the performances of the Kalman filter, the EWMA filter, and the SF with the skewed dEWMA predictor, as described in Sect. 5.3, when

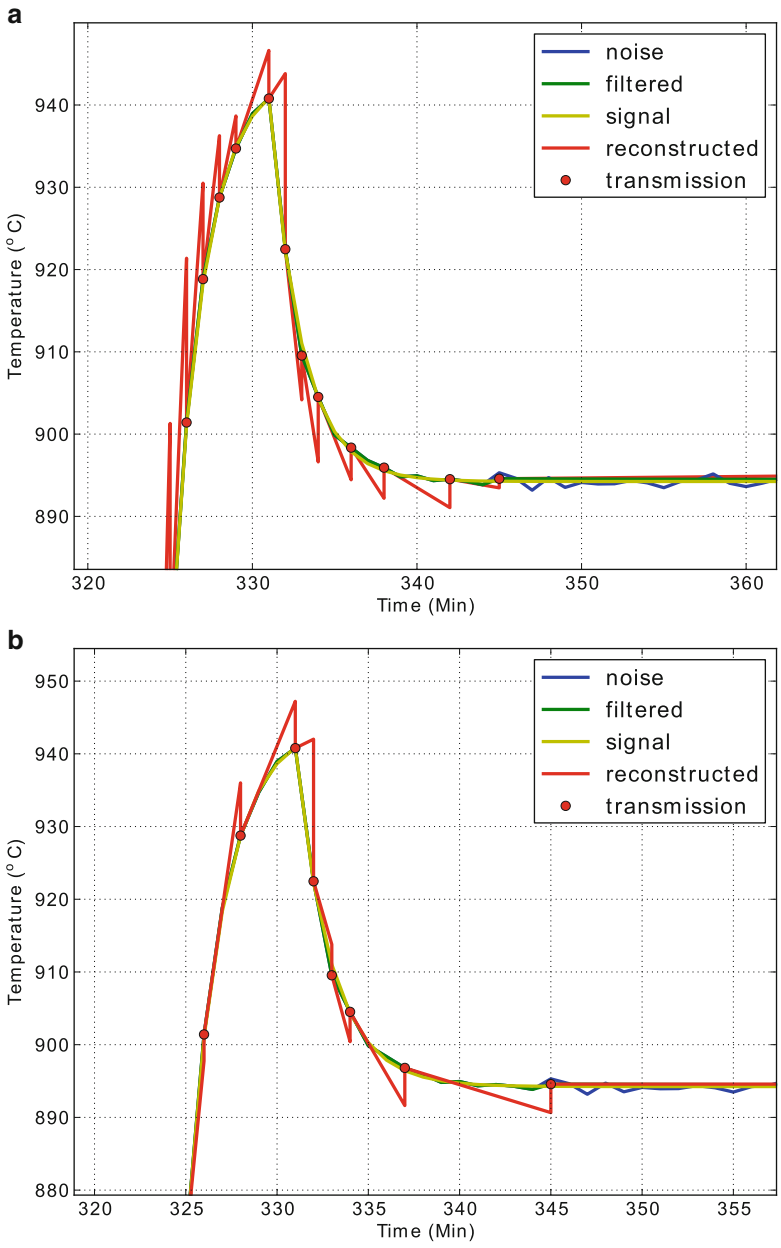

Fig. 9 More accurate rate of change estimation after bias. (**a**) Before bias. (**b**) After bias

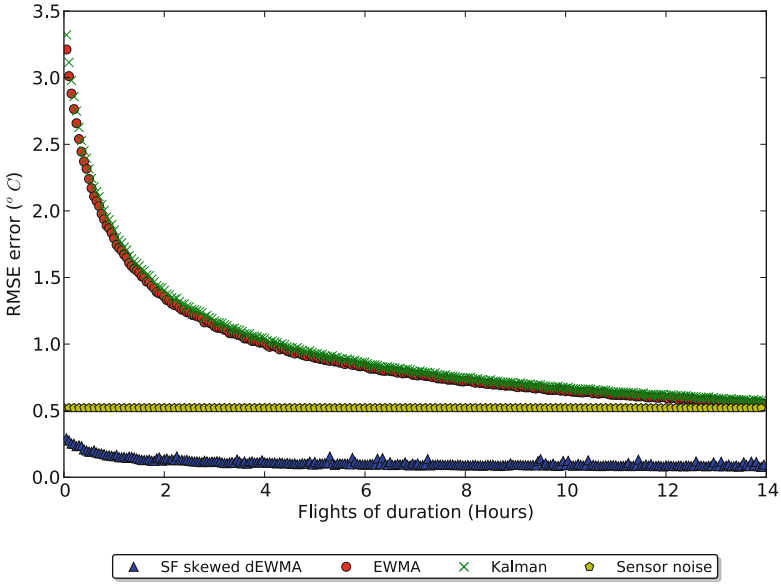

Fig. 10 RMSE error between the original signal and the filtered value

applied to the simulated EGT data. In particular it is of interest to assess their ability to filter the data, as well as the impact on the number of transmissions when they are used in conjunction with SIP.

The use of SF has been found to reduce the noise while preserving the underlying signal. This can be seen in the reduction of the root mean squared error (RMSE) between the filtered value and the noise free signal, while the EWMA and Kalman filters increased the RMSE. The latter is because the signal corruption outweighs the noise reduction benefits. In order to provide a fair comparison in further tests, the EWMA and Kalman filters were tuned so that they both have a maximum error from the original signal of 3.5 °C. The simulations were run again with the new filter parameters and resulting RMSE can be seen in Fig. 10.

When using SF, far greater reductions in transmissions can be gained with a lower SIP error threshold. The cruise duration d was fixed to 1 h and 499 flights simulated, while varying the SIP error threshold from 0 to 5 °C. Figure 11 shows the expected lifetime of a node, for each generated flight, for each algorithm. It is important to note that the expected lifetime shown is the relative effective lifetime—assuming the nodes are turned off between flights—this would be the expected number of days if the aircraft serviced flights of the given durations only.

Using an error threshold of 0.5 °C, 280 random flights are generated with cruise lengths from 0 to 13 h at 3-min intervals. Table 3 shows the percentage number of samples that would need to be transmitted per hour for each algorithm. Figure 12 shows the expected effective battery lifetime of a node running each algorithm.

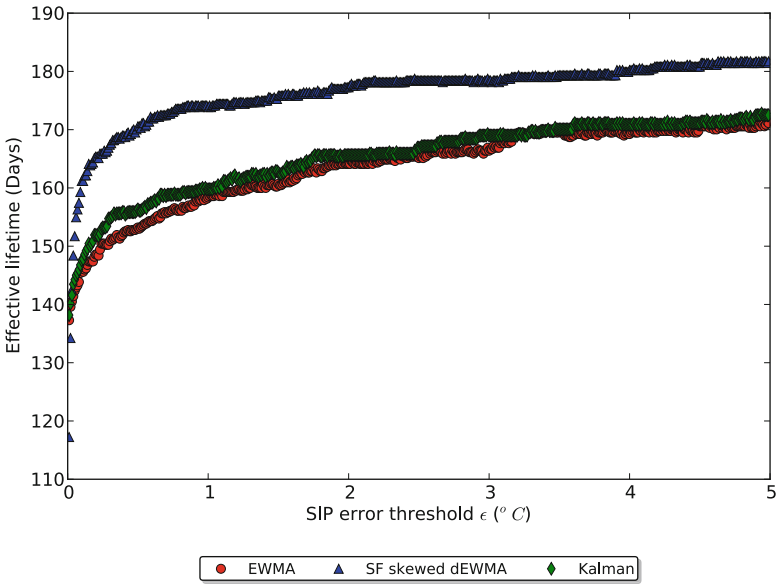

Fig. 11 Expected lifetime of a HEAT node as the error threshold is varied, when used continuously in flights lasting 1 h

Table 3 Mean percentage of samples transmitted per hour in each phase of a flight

Samples transmitted per hour (%)			
Filter	Take off	Cruise	Landing
EWMA	54.7	52.9	93.6
Kalman	50.7	48.7	86.9
SF	6.9	0.3	21.4

The total system error arising from the use of each algorithm can be compared in Fig. 13. When the EWMA filter and the Kalman filter are used, there is an RMSE of $1.29 \pm 0.05\,°C$ and $1.27 \pm 0.06\,°C$, respectively, over a range of $927\,°C$; the error observed being higher in a variable state than a steady state. However, the SIP and SF with a skewed dEWMA approach has an RMSE of $0.46 \pm 0.15\,°C$.

7 Conclusions

When we implemented the new SF algorithm in the SIP and evaluated it with the HEAT system, we found that more accurate signal estimates were made. This is because the filter lag at the step changes has been greatly reduced, resulting in reduced sensor error, as well as the overall system error. Additionally by utilising domain specific knowledge of the phenomena, we make the prior assumption that after a thermal shock—a dramatic step change—the system will seek thermal

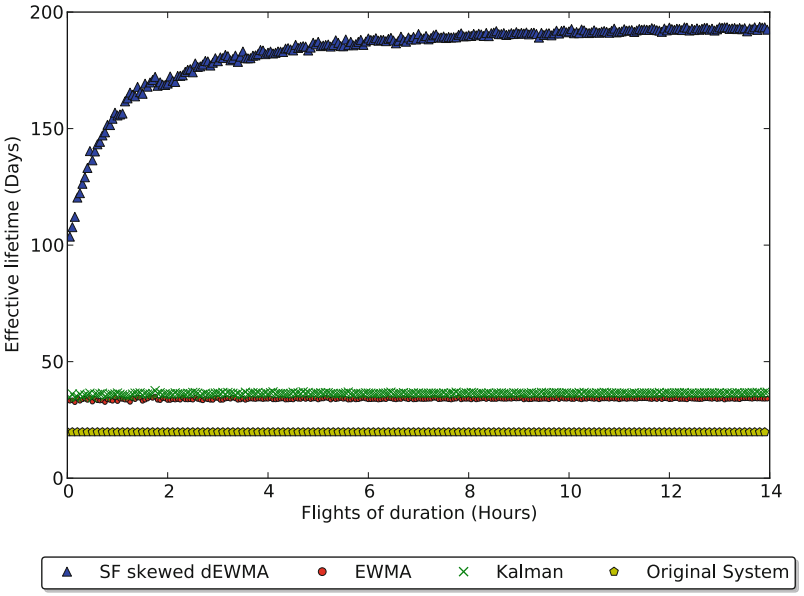

Fig. 12 Comparing the expected effective battery lifetime when using different data reduction methods

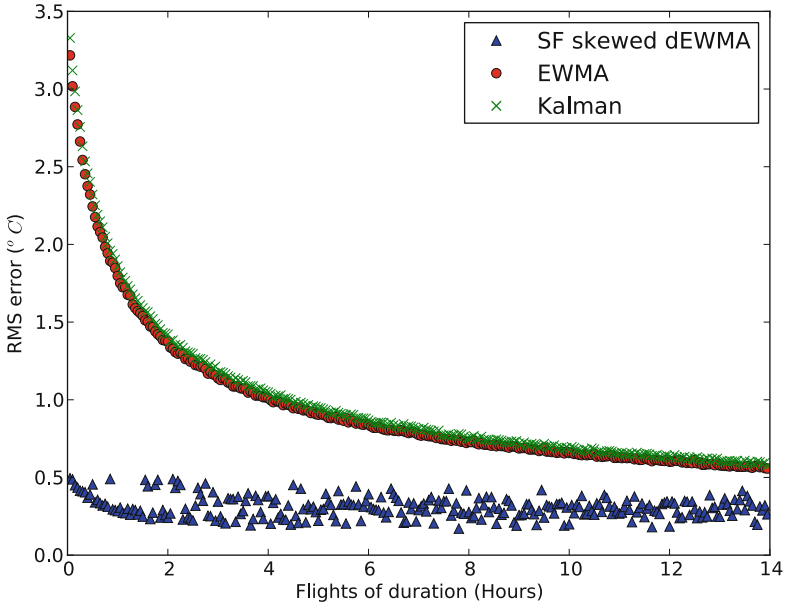

Fig. 13 Algorithm error, given a 0.5 °C threshold

equilibrium. Based on this assumption the skewed dEWMA results in more accurate predictions of the rate of chance in step changes. The overall results of this method predict considerably increased battery lifetime for a sensor node.

Preliminary work by the authors shows the method presented in this paper can be adapted to other steady state systems with significant step changes, and the savings achieved are similar to that of the simulated results here.

There are some assumptions that have been made during the evaluation of this work, which may affect performance in a real deployment. The simulation assumes normally distributed noise, with a fixed standard deviation. However, there are some real-world applications where this is not the case, although in general, the assumption of Gaussian noise is reasonable—and where not, appropriate filters may be substituted for the dEWMA. Other applications present situations where the system parameters change over time. For such systems, further improvements to this method may be achieved using adaptive filters and an adaptive breakout threshold in response to the varying system parameters.

A further assumption is that the temperature in the steady state is constant, i.e. no variation; in reality, one could expect there would be some variation. Further transmissions would be needed to capture those variations, and actual transmission reduction rates will depend on the amount of variation.

The encouraging results presented in this paper provide opportunity and impetus for further exploration in this area. For example, the algorithm presented uses a linear model and it is considered that further reductions in transmission are possible if a non-linear model were to be used. Important application information may also be gained from the identification of the system states, which may be used in future on-node algorithms.

Acknowledgements The authors acknowledge the support of Meggitt (UK) Limited, Basingstoke, UK; Rolls-Royce, Strategic Research Centre, Derby, UK; TRW Conekt, Solihull, UK and Engineering and Physical Sciences Research Council (EPSRC).

References

1. Adnan H (2011) Energy harvesting: state-of-the-art. Renew Energy 36(10):2641–2654
2. Anastasi G, Conti M, Francesco MD, Passarella A (2009) Energy conservation in wireless sensor networks: a survey. Ad Hoc Netw 7(3):537–568
3. Borgne Y-AL, Santini S, Bontempi G (2007) Adaptive model selection for time series prediction in wireless sensor networks. Signal Process 87(12):3010–3020
4. Burrow S, Lieven N, Ambrosio PE, Clare L, Lester G, Heggie W, Stark J, Hazelden R, Cooper P, Stinchcombe M (2008) WISD: wireless sensors and energy harvesting for rotary wing aircraft health and usage monitoring systems. In: Proceedings of nanoPower forum, Irvine
5. Goldsmith D, Brusey J (2010) The Spanish inquisition protocol—model based transmission reduction for wireless sensor networks. In: Sensors, 2010 IEEE, pp 2043–2048
6. Holt CC (2004) Forecasting seasonals and trends by exponentially weighted moving averages. Int J Forecast 20(1):5–10

7. Jain A, Chang EY, Wang YF (2004) Adaptive stream resource management using Kalman filters. In Proceedings of the 2004 ACM SIGMOD international conference on management of data. ACM, New York, pp 11–22
8. Marcelloni F, Vecchio M (2009) An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks. Comput J 52(8):969–987
9. Maxim (2002) MAX6675: cold-junction-compensated K-thermocouple- to-digital converter. Maxim, rev 1 edition
10. Mitchell J, Dai X, Sasloglou K, Atkinson R, Strong J, Panella I, Cai L, Mingding H, Wei A, Glover I et al (2011) Wireless communication networks for gas turbine engine testing. Int J Distributed Sensor Networks
11. Pinto J, Lewis GM, Lord JA, Lewis RA, Wright BH (2010) Wireless data transmission within an aircraft environment. In: Proceedings of the 4th European conference on antennas and propagation (EuCAP), 2010, pp 1–5
12. Santini S, Romer K (2006) An adaptive strategy for quality-based data reduction in wireless sensor networks. In: Proceedings of the 3rd international conference on networked sensing systems (INSS 2006), pp 29–36
13. Schoellhammer T, Greenstein B, Osterweil E, Wimbrow M, Estrin D (2004) Lightweight temporal compression of microclimate datasets. In; Conference on local computer networks, pp 516–524
14. Srisooksai T, Keamarungsi K, Lamsrichan P, Araki K (2012) Practical data compression in wireless sensor networks: a survey. J Netw Comput Appl 35(1):37–59
15. Texas Instruments (2011) SWRS081B: a true system-on-chip solution for 2.4-GHz IEEE 802.15.4 and ZigBee applications
16. Thompson HA (2009) Wireless sensor research at the Rolls-Royce control and systems university technology centre. In: Wireless communication, vehicular technology, information theory and aerospace electronic systems technology, pp 571–576
17. TRW Automotive (2011) Tire pressure monitoring systems. datasheet
18. Yedavalli R, Belapurkar R (2011) Application of wireless sensor networks to aircraft control and health management systems. J Control Theory Appl 9:28–33

Space Crew Health Monitoring

Azhar Rafiq

Abstract In NASA's strategy for continued expeditions human explorers are expected to undertake long duration missions which will increase the requirement for maintaining good health in instances of extreme conditions including higher radiation exposure and lower gravity environment. These circumstances pose greater challenges to the astronaut such as metabolic stress, decompressions sickness and radiation exposure. To support the health and performance of the astronaut there is an essential need to monitor their status in the space suit. In this paper a prototype low-power, wireless physiological monitoring system (VPack) is presented. Multiple non-invasive sensors are integrated into a wearable device that can be used to monitor the health and wellness of space suit occupant during extra vehicular activities (EVA). The integrated software and hardware interoperability capabilities are outlined here with inclusion of medical informatics and wireless network communication.

1 Introduction

The efforts in the space program have evolved to include recent efforts towards successfully resupplying the International Space Station (ISS). The cargo delivered allows for continued research studies but dictates duration of stays in reduced gravity environment that are months long. Consequences on the human physiology are greater with the absence of gravitational influence.

Future efforts are towards more challenging flight such as to identify, capture, and relocate an asteroid. Additionally, there is interest in longer duration flights to other planets such as MARS. With increased duration of exploration missions the probability of medical events will increase and capability for medical care will become more significant. A series of medical events have occurred from the early Mercury programs to today's ISS missions providing data on human ability to work in space [1]. Thus illnesses and injuries during space flight can be classified relative to degree of possible responses or intervention [2]. The three possible

A. Rafiq (✉)
Department of Surgery, Virginia Commonwealth University, Richmond, VA, USA
e-mail: arafiq@vcu.edu

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_7

categories of illnesses are (1) resolution with minimal or no intervention, (2) fatal incidence unless response applied, (3) fatal incidence regardless of intervention efforts. Medical incidence has also occurred during extra vehicular activity (EVA). For instance, during STS-100 a crewmember had a burning sensation in both eyes during EVA that was attributed to exposure of eyes to Anti-fog solution (soap). Additionally, crew members become susceptible to high nitrogen concentrations in their blood. Countermeasures for this complication require implementation of Exercise Prebreathe protocol as a method for more efficient denitrogenization of EV crew prior to EVA from ISS. To better manage complications during longer missions crew and system environmental monitoring is essential.

An integrated vehicle health management (IVHM) system has been developed and implemented to monitor environmental health of vehicles during orbit flight [3, 4]. Integrated monitoring includes hardware and software technologies embedded in the vehicle subsystems, maintenance operations, and launch and mission operations elements that provide both real-time and life-cycle vehicle health information. However, health monitoring of crew members, especially during EVA, has lagged. In this example of a system for Wireless Sensor Network (WSN) a non-invasive sensor mesh network is presented that allows for crew physiological monitoring within the EVA suit.

2 Background

A non-intrusive BioSensor solution with a network of sensors for use in the space suit during EVA called the VPack was designed for continuous physiological monitoring. The sensor data captured was integrated into NASA's communication, avionic and informatics (CAI) computational subsystem on the outside of the space suit. This effort was to clarify the requirements for future physiological monitoring within EVA suits in collaboration with NASA engineers and scientists leading the Research and Technology Studies (RATS) team at Johnson Space Center (JSC), Houston, TX.

The technical vision was to establish a highly flexible and scalable platform with miniature biomedical sensors, activity sensing, and providing data analytics.

Capabilities of the VPack system are as follows:

- Continuously monitor the health and well-being of astronauts in EVA traverse
- Utilize low power personal area network (PAN) wireless communications to interconnect sensor unit into miniature form factors appropriate to embed in the space suit
- Provide data analysis of physiological vital signs, activity level, and location to detect deviations from expectations and predictive determinations
- Be a consistent solution applicable across transitions through echelons of mission phases

- Protect astronaut privacy in accordance with the Health Insurance Portability and Accountability Act (HIPAA) of 1996 requirements and adherence to requirements of NASA mission and medical operations requirements
- Support the flight surgeons on mission support console, but increase their accuracy of monitoring tasks

The VPack system combined vital sign monitoring, automatic fall detection, and location determination into an unobtrusive headband device. This device was used to detect astronaut location for simulated activities of daily EVA, as well as to detect their physiological status with exertion. Vital sign data was collected on a scheduled basis configurable for each astronaut and was designed to display an alert to notify the subject and support staff on console if the data trends were beyond the accepted threshold of the astronaut.

The VPack system also contains backend informatics software that will be able to trend physiological data for each user. As the subject's mobility and task performance increased during mission EVA the normal parameters associated with the wearer were monitored continuously. If any decline in health or mobility becomes apparent or if an emergency occurs, an additional alert would display for flight surgeons on console as well as create a warning display to Heads up Display in the helmet to provide visual and audio notification. This data was also configured to relay automatically in a secure manner compliant with HIPAA regulations to a mobile device such as a cellular telephone or to a computer terminal at a remote station.

3 System Design and Implementation

This system design included a network of sensors measuring physiological data packets from non-invasive, FDA approved sensors. This architecture can be viewed as being organized in a progression of layers for data processing:

1. Sensor network—the series of sensors are non-invasively attached to the skin surface for data capture.
2. Sampling and conditioning electronics—the signals are filtered and converted to a digitized form.
3. Data pre-processing—this phase includes calibration and time stamping of the captured data.
4. Data archive and analysis—in this capacity customized software organizes the data and analyzes the series in comparison with acceptable ranges.
5. Communication stack—this capability provides data relay for real-time observation by multiple parties.

3.1 Sensor Network

In this study, an embedded multiple sensors system was integrated that keeps the weight and size of the system at a minimum in the space suit, while allowing the power consumption for extended EVA missions. By designing embedded sensor technology, each physiological sensor or status sensor for the astronaut could be an element of the integrated digital space suit environment.

The sensor network includes a plethysmograph sensor (AD Instruments, Colorado Springs, CO), galvanic skin conductance sensors (Stoelting, Wood Dale, IL), one skin surface temperature sensor (PASPORT, Roseville, CA), and one pulse oximeter sensor (Nonin, Plymouth, MN). The plethysmograph uses an infrared photoelectric sensor to detect levels of blood volume perfusion in tissue. The skin conductance is a function of sweat gland activity at the skin surface, which is controlled by the sympathetic nervous system. A low voltage is applied to the skin between two probes and the skin's current conduction is measured. The skin conductance is one of the fastest responding measures of stress response. The skin temperature sensor measures the subject's body surface temperature. The pulse oximeter consists of an infrared optic probe with signal transfer to a microprocessor, monitoring the functional saturation of arterial hemoglobin (SpO_2), as well as the heart rate (HR).

Placement of the sensor network was made in accordance with requirements suggested by the senior engineer of the EVA division at JSC. Also, placement of sensors was approached since blood volume shifts towards the head in a microgravity environment making the need for blood analysis essential. In this regard the sensors had to be placed peripheral to the suit operations and not interfere with the assigned tasks that the subject would perform in simulated exploration activities. In addition all electronics associated with the sensors systems had to be external to the suit. For easy placement the sensors were integrated into a flexible head band worn by the subject with pulse oximetry and plethysmograph located on the lateral aspect of the forehead. The skin temperature and galvanic skin resistance sensors were placed on the opposing temporal surface.

3.2 Sampling and Conditioning Electronics

A hardware package called the VPack included data acquisition and conditioning boards in order to capture the steady stream of data from the sensors (Fig. 1). Using a wired architecture, sensor data was routed into the VPack processor module at a rate of 100 samples per second. With the exception of the pulse oximeter, sensor data packets are in analog format. Analog signals were amplified by the embedded amplifier in the data acquisition board in order to obtain the desired spectrum of values. Additionally, band-pass filter circuits were used to remove interfering frequencies from the amplified signals. The filtered data packets were routed to a 12-Bit A/D converter where the analog signals were converted to digital format.

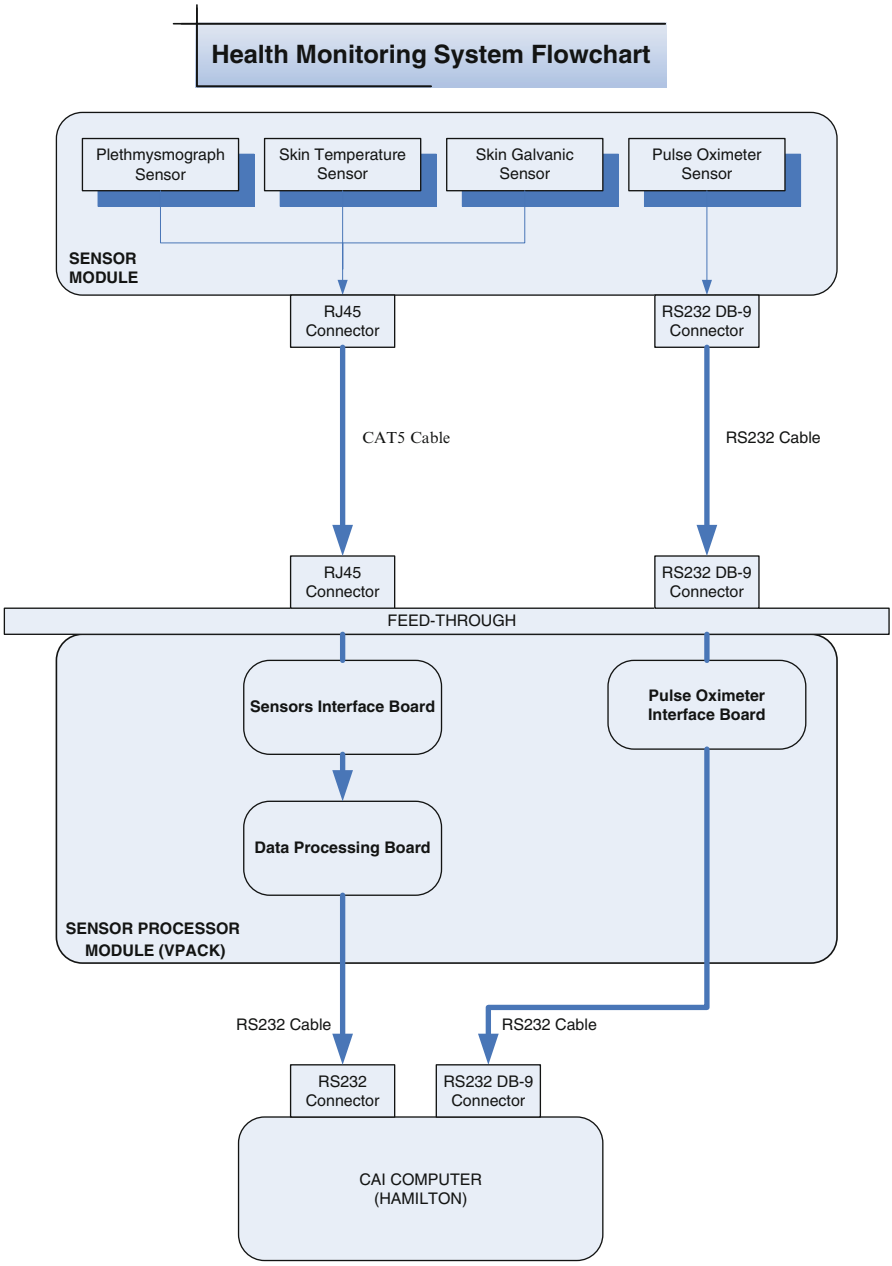

Fig. 1 Schematic of hardware integration in VPack biomedical sensor system

The digitized output of the sensor signals from the VPack was routed via an RJ45 connector to a suit specific computer system, the CAI Pack, with custom built software for data acquisition.

Data from the pulse oximeter sensor consists of 5 bytes that are captured 75 times per second. The data were digitized by an OEM III module to relay digitized signals out via an RS 232 connector to a serial port on the designated CAI Pack.

3.3 Data Pre-processing

The parallel data streams from all the sensors are routed from the VPack via cable into the designated communications, avionics, and informatics computer system (CAI Pack) housed exterior to the standard backpack of the space suit. The CAI Pack is a laptop PC designed to perform the following functions during EVA: data capture, data storage, onboard processing, display, and transmission of data. Series of other software packages were also operating in the CAI Pack supported by numerous teams supporting EVA sessions. These included gesture recognition, human factor coordination with task order lists and bidirectional decision support in task performances.

For integration of physiological data capture the CAI Pack was equipped with a PCMCIA Dual Serial I/O card (Socket Communication, Newark, CA), which provided two serial communications ports. One port was utilized for the data from the pulse oximeter and the second port was used for the remaining physiological sensors in the VPack network.

Within the CAI Pack a stand-alone software module was developed to capture the physiological sensor data packages. The software module was developed using Visual Basic (Microsoft, Redmond, WA) on the Windows platform. The software polls the serial ports and receives data every 200 ms. The received data package is validated by the predefined data structures and tagged with a time stamp. Thus acceptable data characteristics are identified within the software. Only when a complete data package is received and there is no data loss or alteration, can this data package be analyzed and used to calculate physiological values.

3.4 Data Archive and Analysis

The sensor values are archived to a customized database within the CAI Pack for automated analysis and backup. The data was validated first to compare the physiological data with a predefined threshold, and acceptable ranges of deviations beyond this threshold. The customized VPack software produces CSV data files including all the acquired sensor data along with a time stamp. Each sensor's data stream is organized in a column delineated format. Analysis of the data is done to detect deviations and generate alarms. These alarms are visually displayed via a software interface and are color coded to facilitate quick analysis.

Fig. 2 VPack software interface

3.5 Communication Stack

Presentation and communication of the data values was done in two venues: direct display to the user in the suit and remote access to secure geographic locations. The data from the various sensors was displayed numerically within the software interface which provides color coded buttons that change color when values move into caution or dangerous physiological zones (Fig. 2). The software interface was displayed via the CAI Pack to a head mounted display (Micro Optical Corp) within the helmet of the space suit.

Remote access to the VPack software interface was possible from the simulation habitat using the established wireless mesh network and VNC software. Dual access to the data stream provided confirmation of sensor functionality as well as confirmation that the subject in the suit was not strained during terrain navigation and exploration.

4 Results and Analysis

This system was deployed in field tests at Johnson Space Center, Houston, TX and additional rollouts at the simulated MARS environment in Flagstaff, AZ with NASA's RATS team. Additional simulated data routing with wireless networks were done in collaboration with the Canadian Space Agency during simulated EVA sessions on Devon Island, Northern Canada. These environments were used as the MARS analog for terrestrial exploration with minimal wireless communication resources. Each trial was conducted with the data maintaining confidentiality of the source and thus was not labeled to identify the individual being monitored from within the suit. The data was displayed on the helmet up display and was periodically reviewed by the subject during the EVA traverse of the terrain. Periodic visual reference of the data display by the subject was acceptable and provided an objective reference to the level of exertion as tasks were performed during terrain exploration exercises. The secondary remote display of the data was transmitted to a simulated habitat environment and was received wirelessly from the subject. This real-time observation of physiological data trends allowed the support clinical observers to trend the level of physical exertion made by the subject during physical activities. Bidirectional communication to the subject by voice was carried out to confirm degree of comfort during EVA terrain exploration and task performance.

In contrast, the current EMU uses electrocardiographic leads to provide only monitoring of heart rate and rhythm [5]. During the current study customized software integrations were made to include heart rate value displays as an overlay to the software that displayed map of the terrain that the subject was exploring. This interoperability was achieved with limited success since the numerical values required closer observation by the subject due to the font size. Instead, modifications were made such that the VPack software display used color coded buttons. Display of color schemes was easier to interpret by the subject with a quick glance and not be distracted to achieve task completion. Clinical standards were used to identify the accepted values and program the system software with defined threshold ranges such that any values outside of this range from the sensors on the subject would change the color of the display buttons to provide a quick alert. The advantage of the system defined in this study is that the data can provide a mechanism to trend the exertion experienced by the subject but future development will allow the capability to calculate the metabolic exertion during EVA such that a balance can be maintained between quality of the microenvironment in the suit and the energy expended by the subject. The second generation is to integrate a series of skin temperatures along the length of the subject's torso to better track the level of the cooling mechanism and integrate a skin temperature in the axilla region to represent analogy of body core temperature. Additional efforts in the second generation is to integrate analysis of air in vicinity of the subject's mouth in order to analyze respiratory rate, expired carbon dioxide (CO_2), and inspired CO_2. The inspired CO_2 level will reflect the quality of the air in the helmet region. Supplementing these efforts are efforts to deploy comparative analysis of data from two independent

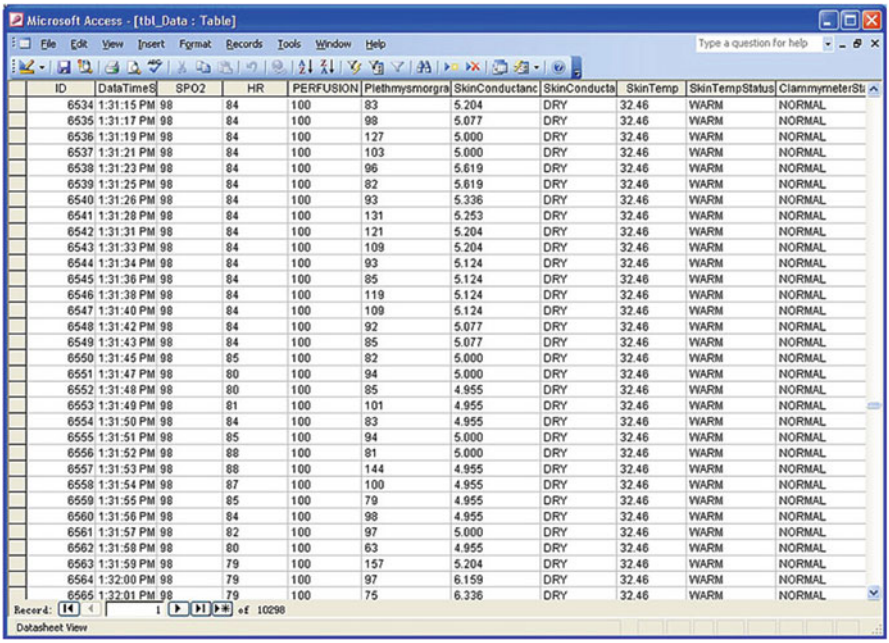

Fig. 3 Sample of pilot data acquired with sensor network and archived in an access database

VPack systems in multiple subjects carrying out EVA activity so that a comparison of physiological monitoring could be carried out more objectively.

A sample segment of test data stream as archived in the database is illustrated in Fig. 3. The data packets from each sensor are column separated and archived in ASCII format or raw data file so that just data is presented. With a fixed column format the data from each variable sensor is always archived into the same column. This data archiving allows automation in data trend analysis and application towards human factor studies.

5 Discussion

This manuscript provides a description of a sensor network that has been integrated into hardware and software architecture to effectively monitor an astronaut's physiology within the confined environment of the space suit during EVA activity. The overall practice of crew health with long-duration space flight is anticipated to involve autonomous practices. Some of the challenges facing medical care have been identified to include resource constraints relative to mass and volume; inclusion of expert medical professionals amongst the crew; and lack of extensive training that can anticipate possible illnesses or emergencies during flight [6].

Continued efforts of the medical support system are oriented towards further defining the levels of possible risk factors impacting health of the crew and development of countermeasures to ensure effective health monitoring.

Additional adjustments to WSN for space environments are possible with adopting a hierarchical architecture to the network that is highly flexible and allows the integration of diverse applications [7]. Sensors (e.g., medical, system and environment monitoring) would need to operate under a unified communication platform. Interoperability among multiple wireless platforms and the ability to have plug-and-play capability would be crucial. At the crew monitoring level WSN will be in one hierarchy, which communicates with upper level hierarchy to convey information to servers for more processing. An elevated upper hierarchy could be at the space habitat level and at the EVAs stage where higher bandwidth technology is adopted in order to aggregate traffic from different lower level networks. Hierarchies operate independently with their own routing protocols and data delivery services. Additionally, they communicate with upper hierarchies in a multidomain environment to transfer the information for further processing. Figure 4 outlines the general topological architecture.

The network would also implement information-centric protocols to be able to send information rather than all the data sensed by sensors. Information is defined in this context as interpretation, what the expert matter would like to infer from the data the sensor is sampling. This will make the network take advantage of pervasive computing and make local decisions and as a consequence reduce the amount of data that will be transmitted. Since communication consumes energy more than computational capacity there will be a direct impact on energy efficiency of the system. Information centric protocols have been explored by researchers and an effective solution is adding a middleware layer that manages data processing and filtering [8]. The Bionet middleware, for instance, would provide

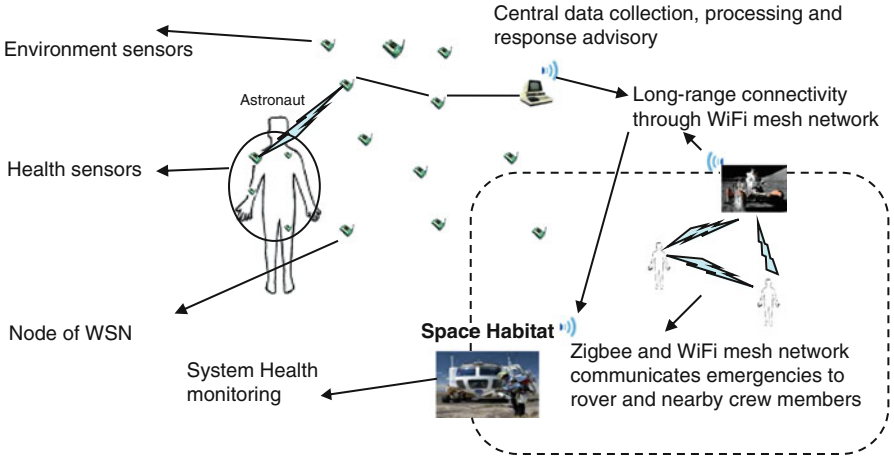

Fig. 4 Conceptual architecture of sensor network in space applications

hardware abstraction (HAB) for sensor motes (e.g., MicaZ, IRIS). Irrespective of the underlying communication network (WiFi, Ethernet, sensor), the middleware represents a common application-network, seamlessly exchanging data between sensors and application.

WSN were not designed to support quality of service (QoS) or managing emergency data traffic with stringent requirements [8]. However, the exploration of sensor technology use in health and system monitoring during space missions has created the need to support data exchange traffic priorities and differentiated resource allocation methods. The network requires capacity to adapt dynamically within time-sensitive circumstances (e.g., emergency situation). Supporting QoS at the MAC layer is critical for achieving differentiated traffic services since the Medium Access Control (MAC) layer arbitrates the wireless medium (policing) among multiple competing nodes and contributes a significant delay to the overall delay. Although national policies such as IEEE 802.11e Enhanced Distributed Channel Access (EDCA) standard provide QoS support, it needs crucial enhancements for time-sensitive and emergency applications (http://standards.ieee.org/getieee802/download/802.11e-2005.pdf). A smart preemption algorithm was developed to enable emergency data routing traffic to preempt traffic of other routine application (similar to ambulance on the road) [9]. We proposed a modification to the standard inter-frame delays where the emergency class of data traffic has unique and smaller access-wait and slot times in comparison with data being routed by all other activities in classes.

Acknowledgments This project was supported under a coop agreement grant with the Space Partnership Development division of NASA's Marshall Space Flight Center, Huntsville, AL. We acknowledge valuable input of content and expert consultation from Dr. Benhaddou in preparing this manuscript.

References

1. Williams DR (2002) Bioastronautics: optimizing human performance through research and medical innovations. Nutrition 18:794–796
2. Houtchens BA (1993) Medical-care systems for long-duration space missions. Clin Chem 39(1):13–21
3. Terster (1998) Model-directed autonomous systems, (presentation) James Kurien, Autonomous Systems Group, Computational Sciences Division, NASA-Ames, June 1998
4. Dorland WD (2002) Health management for NASA second-generation reusable launch vehicles. AI Signal Research Inc. 256-551-0008. Article in Sep 2002 MFPT Forum
5. Newman DJ (2000) Life in extreme environments: how will humans perform on Mars? Gravit Space Biol Bull 13(2):35–48
6. Risin D (2009) Risk of inability to adequately treat an ill or injured crew member. In: McPhee JC, Charles JB (eds) Human health and performance risks of space exploration missions. Johnson Space Center, Houston, Texas pp 239–252

7. Benhaddou D, Balakrishnan M, Yuan X, Chen J, Rungta M, Barton R, Yang H (2009) Wireless sensor networks for space applications: network architecture and protocol enhancements. Sens Transducers J 7:203–212, Special Issue "MEMS: From Micro Devices to Wireless Systems"
8. Zhong R, Hail MA, Hellbruck H (2013) CCN-WSN – a lightweight, flexible content-centric networking protocol for wireless sensor networks. In: IEEE eighth international conference on intelligent sensors, sensor networks and information processing, pp 123–128
9. Balakrishnan M et al (2008) In-channel service preemptions for emergency medium access in healthcare sensor networks. In: MILCOM 2008, IEEE, San Diego, November 2008

Part III
Vehicular Applications

AGORA: A Versatile Framework for the Development of Intelligent Transportation System Applications

Mohammad Ali Salahuddin and Ala Al-Fuqaha

Abstract Advances in communication and computational technologies have made it possible not only to talk or text anywhere but also play high-resolution interactive games on the go. Our goal is to leverage these wireless and cellular communication and high performance computing capabilities, to design and develop a framework for the Transportation Network. Such a network would be part of an Intelligent Transportation System (ITS), which could save lives, money, non-renewable resources, and time.

In this chapter we present AGORA, a novel framework for ITS. Here, vehicles and pedestrians equipped with wireless or cellular devices are interconnected to each other via AGORA infrastructure and its cloud services. AGORA cloud services offer a wide range of safety, efficiency, and infotainment applications for vehicular users and pedestrians. AGORA also provides an XML-based development platform for designing, developing, and deploying new rule based AGORA applications. We will present a detailed discussion of the hardware and software components of AGORA architecture, followed by a thorough discussion of the suite of accompanying ITS applications. We will also discuss the AGORA development environment for extending AGORA ITS applications. We will end this chapter with the installation procedure for AGORA and its development environment.

1 Introduction

In 2009 alone, there were over 40,000 fatalities including pedestrians, in the USA [1]. Also, in 2009 Americans consumed 138,496,176,000 gallons of motor gasoline [2], resulting in emission of 2.66×10^{12} pounds of carbon dioxide into the environment. Moreover, traffic congestions led to $100 billion lost in wasted fuel and time [3]. These daunting numbers are an incentive to improve the safety and efficiency of the transportation network, while reducing ecological footprint and

M.A. Salahuddin (✉)
Université du Québec à Montréal, Montreal, Quebec, Canada
e-mail: mohammad.salahuddin@ieee.org

A. Al-Fuqaha
Western Michigan University Kalamazoo, Michigan, USA

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_8

increasing personal safety and productivity. Today government agencies, industries, and academia are investing resources in the emerging area of Intelligent Transportation Systems (ITS) to reduce human fatality rate and increase efficiency of vehicles, while conserving time and resources in our antiquated transportation network.

Traditional transportation systems encompassed radio broadcasts and variable message signs for road hazards, which evolved into electronic systems, such as electronic toll payment. ITS leverage state-of-the-art wireless communication and mobile computing technologies, to build an interconnected network of vehicles and infrastructure, using vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communications. A vehicle in ITS is equipped with an on-board unit (OBU), which consists of positioning systems, processing units, and radio transceivers. ITS infrastructure are composed of road-side unit (RSU), typically installed alongside the road, with processing and communication capabilities. Numerous ITS applications have been proposed that can be classified into safety, efficiency, infotainment, and convenience applications. These applications rely on the V2I and V2V communications to provide real-time, interactive services to reduce accidents and perform efficiency tasks, such as traffic congestion management.

In this chapter we present, AGORA, a versatile framework for ITS applications. It provides a suite of safety, efficiency, and infotainment applications. Safety applications are high priority applications, geared to reduce traffic and pedestrian accidents, e.g. cooperative collision avoidance, pedestrian alerts, incident management, weather advisory, hazard alerts, road sign notification, variable message sign, etc. Efficiency applications encompass intelligent traffic flow management and navigation, vehicle carbon footprint, gas consumption, etc. Infotainment applications include information and entertainment applications, such as gas prices, instant messaging, VoIP, gaming, parking space finder, etc. A thorough analysis of various ITS applications is discussed in [4] and a detailed classification and taxonomy of ITS applications is presented in [5]. Furthermore, AGORA is also extensible, based on an XML-based Rule Engine. It can be used to design, develop, and deploy custom applications using an Integrated Development Environment (IDE) and a Software Development Kit (SDK).

AGORA framework consists of Regional Managers (RM), Super Nodes (SN), Traffic Management Centers (TMC) cloud services, Vehicle Integrated Systems (VIS), and Mobile Integrated Systems (MIS) as illustrated in Fig. 1.

Vehicles (VIS) and pedestrians (MIS) interact with AGORA Super Nodes and Regional Managers to receive and update their neighborhood information and to establish V2V and vehicle-to-mobile (V2M) communications. V2M communication in AGORA uniquely integrates wireless and cellular networks to facilitate ITS applications across these platforms. This has a twofold advantage: (1) pedestrians equipped with smart phones can use AGORA applications and (2) vehicles equipped with smart devices can also use vital AGORA applications without the need for additional OBU. TMC provide additional safety and efficiency applications, such as variable messages, advisory alerts, and infotainment applications, such as weather summary and gas prices. TMC also provides a web-based interface for transportation system authorities to advertise various alerts, generate tomographic reports for road conditions, and perform vehicle tracking.

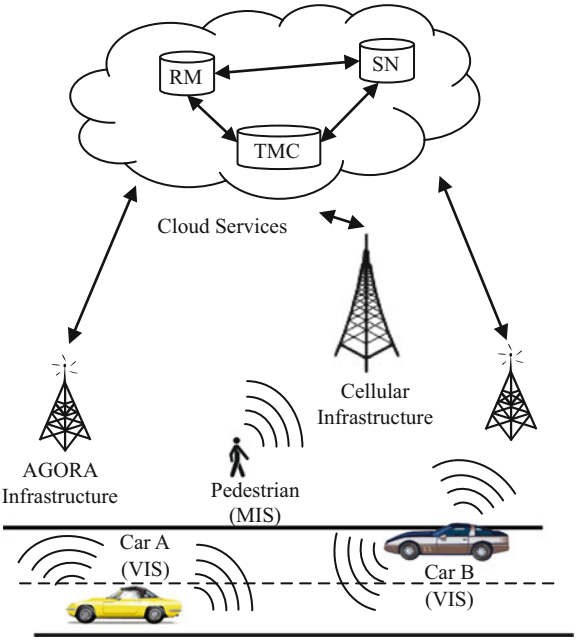

Fig. 1 AGORA framework for ITS with cloud services

The rest of this chapter is organized as follows. Section 2 presents the architecture of AGORA along with its hardware and software components. Section 3 discusses the AGORA suite of applications and its development environment.

2 Architecture

AGORA is a framework is a Connected Vehicle technology that offers a set of applications for the ITS. The AGORA infrastructure consists of hardware and software components. The hardware components consist of OBU and RSU. The software components include Super Nodes (SN), Regional Managers (RM), TMC, VIS, and MIS that work together to increase the safety and efficiency of the transportation network.

The underlying communication infrastructure of AGORA has been implemented as a mesh network and uses the Quadrature Division Multiple Access (QDMA) technology. Though its inherent high speed mobility support makes it an ideal communication infrastructure for ITS, AGORA software components can be used with any other communication platform, such as WiMAX, IEEE 802.11p, 3G/4G LTE, etc. We have further extended AGORA, by integrating the vehicular and cellular environments. This enables pedestrians and vehicular users enabled with wireless and/ or smart cellular devices to leverage AGORA applications without the need for additional OBU.

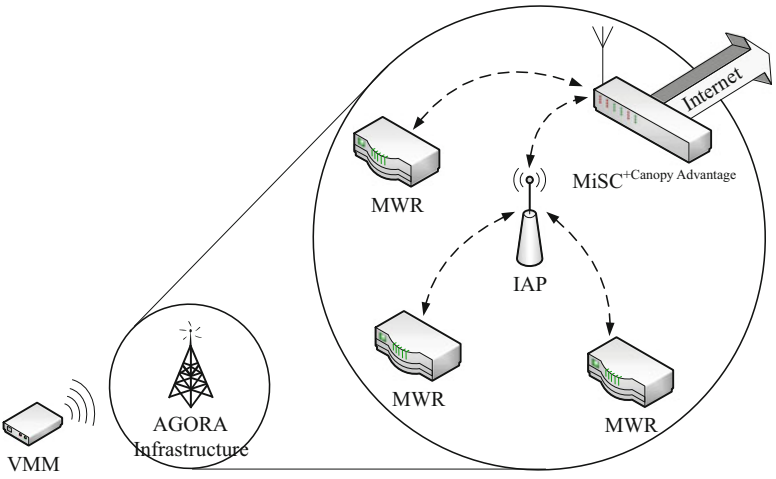

Fig. 2 VMM in OBU connects to RSU in AGORA infrastructure

In this section, we will present a detailed description of the hardware and software components of the AGORA architecture.

2.1 Hardware Components

AGORA hardware is divided into RSU and OBU components. RSU consists of Intelligent Access Points (IAP), Mesh Wireless Routers (MWR), Mobile Internet Switching Controllers (MiSC), and Canopy Advantage Access Point (AP), as illustrated in Fig. 2.

The RSU components, shown in Fig. 3, are deployed and configured into a 2.4 GHz mesh network. The MWR enables OBU components to connect to the Internet via the IAPs and/or the switching controllers. The Vehicle Mounted Modem (VMM) in the OBU of a vehicle connects to the MWR in the AGORA infrastructure. The MWR is deployed to increase the range between IAPs and VMM while simultaneously increasing the spectral efficiency of the network [6]. The IAPs and MWRs are fixed infrastructure components, providing ITS vehicles in the coverage area access to the wired network via MiSC with Canopy Advantage Access Point [6]. The MiSC enables network management functions, such as provisioning, management, authentication, configuration, fault management, monitoring, and reporting [6].

The OBU components, as shown in Fig. 4, consist of a Global Positioning System (GPS) device, an On-Board Diagnostic System (OBD) scanner that connects to OBD-II interface of vehicles manufactured in 1996 or later, a Windows based nettop interfaced with a touch screen display and a VMM. The vehicle's geographical

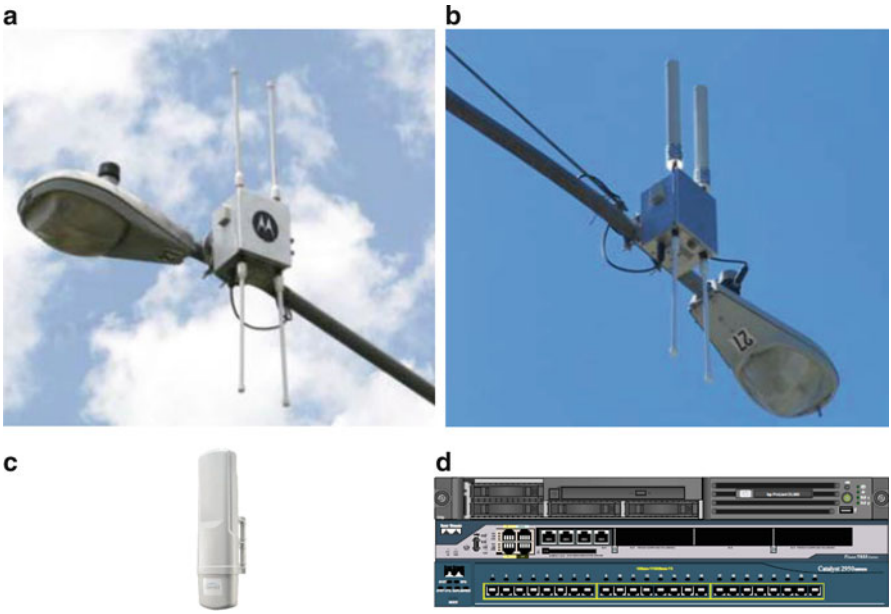

Fig. 3 AGORA RSU components. (**a**) Intelligent Access Point (IAP). (**b**) Mesh Wireless Router (MWR). (**c**) Canopy Advantage Access Point. (**d**) Mobile Internet Switching Controller (MiSC)

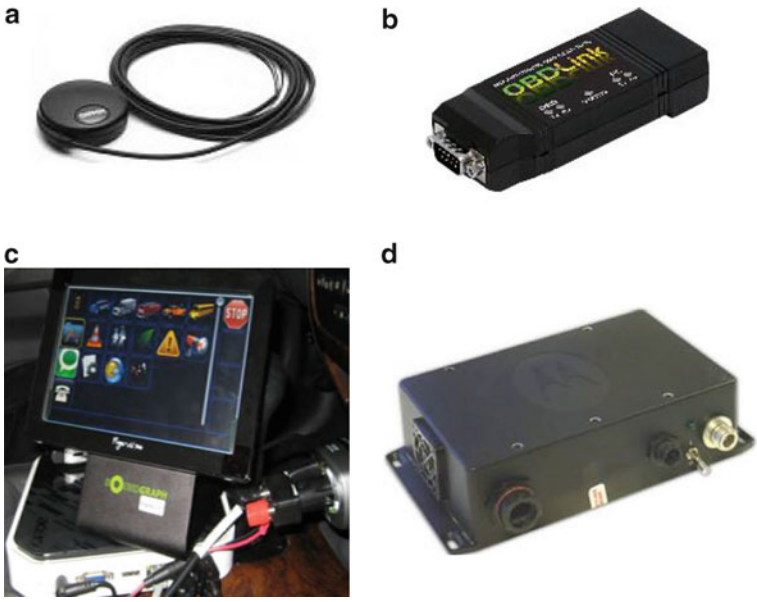

Fig. 4 AGORA OBU components. (**a**) Global Positioning System. (**b**) On-Board Diagnostic System Scanner. (**c**) Nettop. (**d**) Vehicle Mounted Modem

information is gathered by the GPS device and internal operational information, such as revolutions per minute (RPM), speed, fuel consumption, etc., via the OBD scanner. The nettop is a cost-effective, power-efficient, small form factor computer, with basic computing capabilities, interacting with the vehicle's users through the touch screen display. The VMM enables secure, high speed connectivity to the 2.4 GHz mesh network. The underlying mesh network and its VMM supports up to 6 Mbps burst data rate at speeds in excess of 100 mph, enabling V2I and V2V communication. V2M communication occurs when cellular nodes, such as pedestrians equipped with a smart phone, connect to the AGORA framework using the cellular network with no additional equipment.

2.2 Software Components

AGORA software components consist of server-side and client-side applications. The server-side applications are composed of Super Nodes (SN), Regional Managers (RM), and web servers hosting the TMC. The client-side applications are classified into VIS and MIS, for the vehicular and cellular users in AGORA. Figure 5 illustrates the integration of the client-side applications on with the server-side applications and in the following subsections we will discuss their details.

2.2.1 Server-Side Components

In AGORA the region of interest is divided into centroids, as illustrated in Fig. 5. The centroids are locations with co-ordinates. A centroid set might contain one or more centroids, and is typically associated with a geographical region. An Super Node keeps an inventory of a set of centroids to maintain a global view of the available centroid sets. These centroid sets are allocated to Regional Managers by Super Nodes. An SN governs and facilitates multiple RM. When an SN receives a ping from an RM, it allocates any unused centroid to the RM and updates the global view. If an SN peer has already allocated the centroid, then a rollback is performed. An RM maintains a database of vehicular and cellular nodes in its centroid. The user interface for the SN and RM components is shown in Fig. 6.

As illustrated in Fig. 5, vehicular and pedestrian nodes interact with SN and RM to establish connection with AGORA infrastructure and use the numerous services it offers. Initially, vehicular and pedestrians nodes submit a query over TCP, containing the latitude and longitude of their current geographical location, to a random SN. It matches the geographical location of the node to the closest RM and replies with the corresponding RM(s) IP address. Next, the node initiates a TCP connection with the RM to retrieve neighborhood information. The RM replies with a list of neighbors and their contact information. The nodes populate their neighborhood list and establish peer-to-peer (P2P) communication. The sequence of actions to establish V2V and/or V2M communication is depicted in Fig. 7.

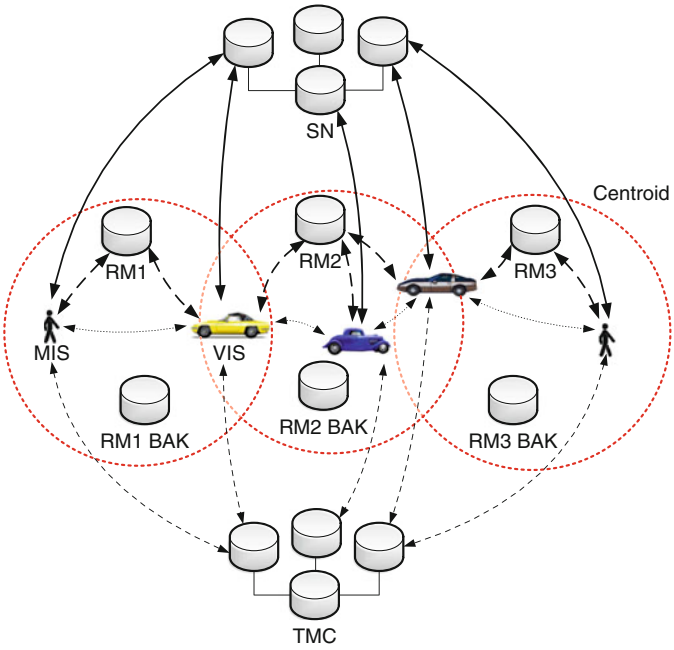

Fig. 5 Integration of client-side and server-side applications in AGORA

TMC is a repository of transportation network data collected from vehicles and pedestrians. TMC offers various web services that vehicles and/or pedestrians can leverage to report and retrieve transportation network and environment specific information. Communication with TMC is over TCP to leverage its inherent reliability and guarantee of service.

The TMC offers a web interface that can be used by transportation system authorities, to retrieve visual reports. Currently, TMC reports contain node tracking and tomographic reports, as illustrated in Fig. 8, based on road condition monitoring of potholes, dead animals, and other road hazards.

TMC is also used to broadcast or unicast alerts using the Advisory Alert application (A3), depicted in Fig. 9, such as amber alerts or other variable messages. The transportation authority can also use the Vehicle's Wallet application, shown in Fig. 9, to appropriately bill users for road infrastructure tolls, such as parking and road tolls.

Furthermore, a rich set of APIs are provided with AGORA to extend the functionality of TMC. Figure 10 illustrates the architecture of TMC. Services exposed by TMC adhere to strict Service Level Agreement (SLA) requirements and offer scalability and fault tolerance, leveraging clustered deployment on a cloud computing platform.

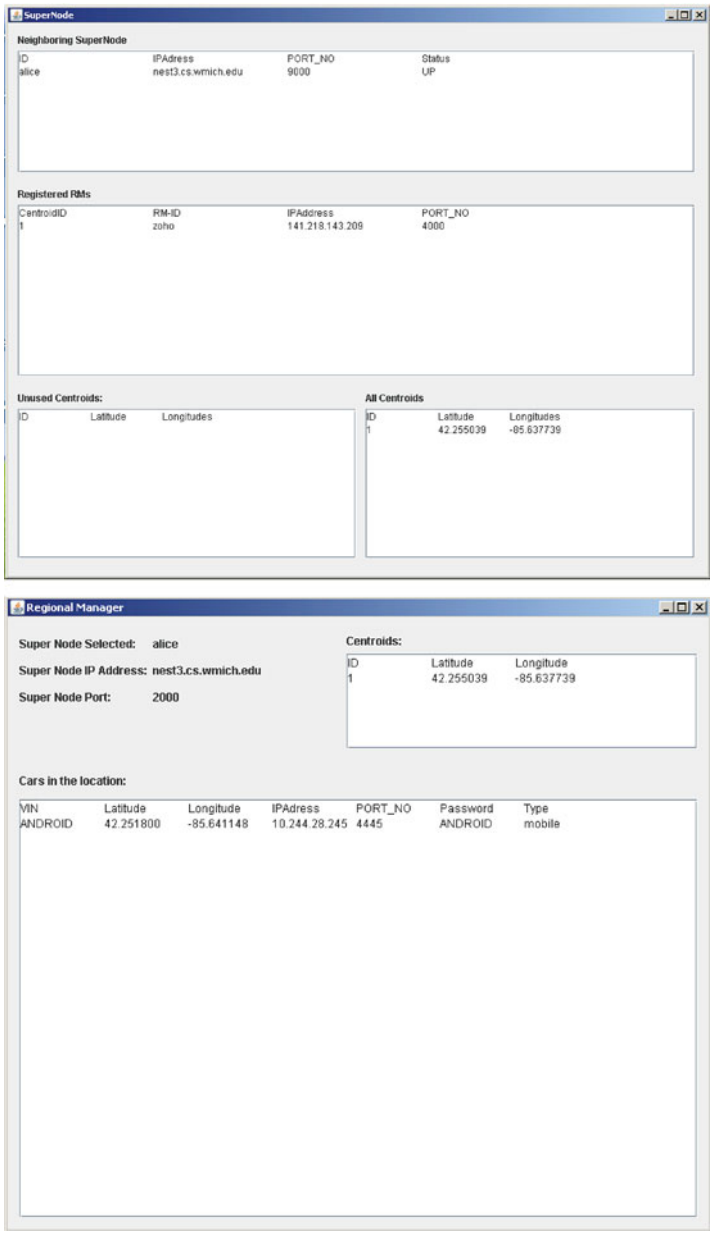

Fig. 6 Super Node and Regional Manager user interface

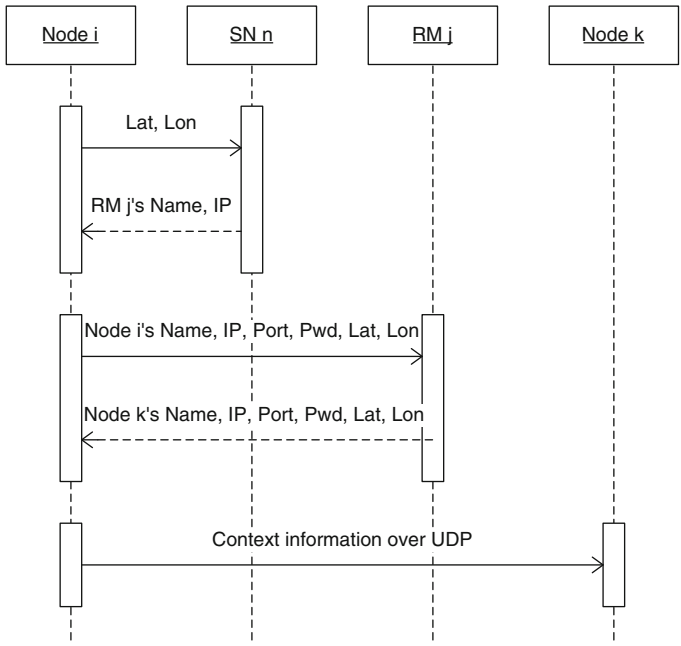

Fig. 7 Sequence diagram for vehicle/pedestrian connection initiation with AGORA

Several measures have been taken to build AGORA to be a scalable, load-balanced, fault-tolerant framework with low latency applications. Some of these are listed as follows:

1. First, the Regional Managers, Backup of Regional Managers and Super Nodes have been organized in a hierarchical manner to leverage the inherent benefits of a tiered architecture.
2. Second, multiplicity in the number of SN, RM and Backup of RM (RM BAK) introduces redundancy, distributes work load, and provides low latency for application requests.
3. Third, centroid sets in regions of high traffic density can be blocked from containing more than two centroids.
4. Fourth, AGORA V2V and V2M packets are encrypted and communicated over UDP to leverage the inherent low packet delay.

2.2.2 Client-Side Components

AGORA framework has two interfaces for its vehicular and cellular users, the VIS and the MIS, respectively. They are XML configurable interfaces, hosting numerous safety, efficiency, and infotainment applications. Some example applications are

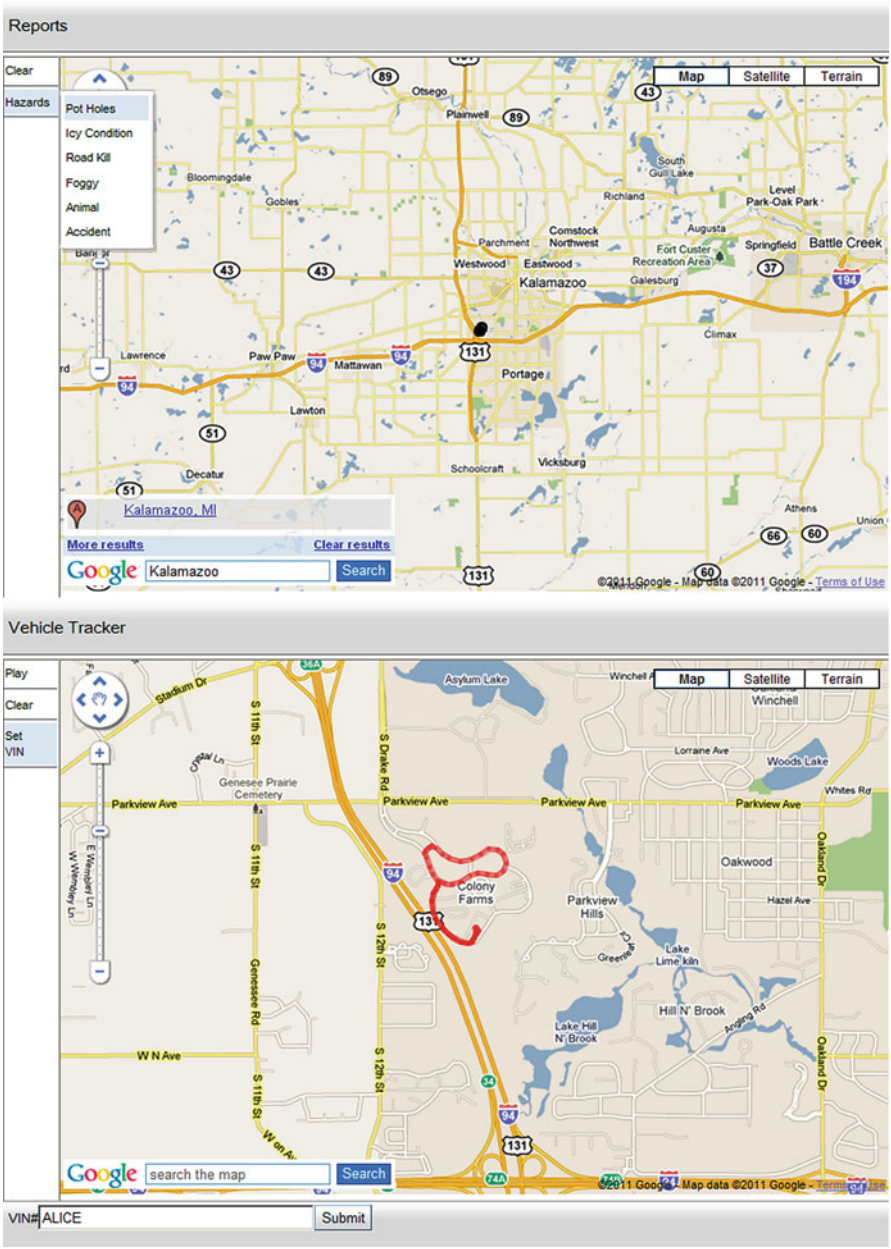

Fig. 8 TMC tomographic report and vehicle tracking services

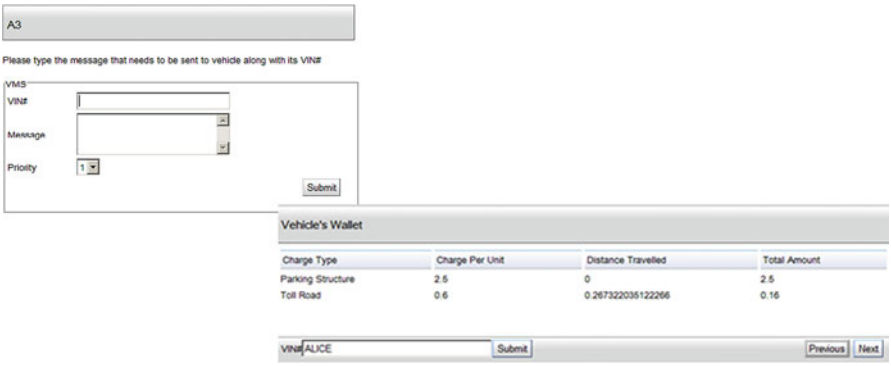

Fig. 9 TMC Advisory Alert and Vehicle's Wallet applications

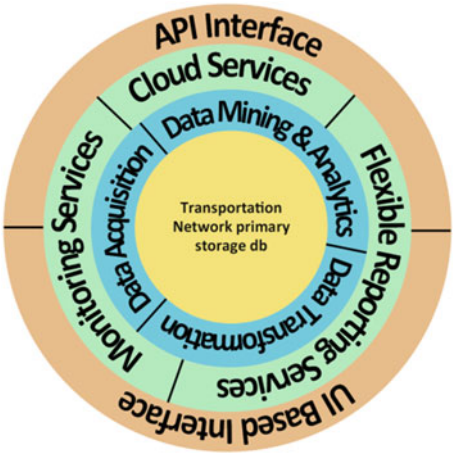

Fig. 10 Traffic Management Center (TMC) Architecture

contextual information, road sign notification, incident management, advisory alert, carbon emission footprint, VoIP calls for emergency response, voice synthesis for hands free operation, etc.

VIS, presented in Fig. 11, consists of context awareness, Sphinx, and VoIP applications, with display areas for variable message sign and dynamic and static road signs. It also consists of user created applications, under Applications, discussed in Sect. 3.

The real-time context awareness application is an integral part of VIS and MIS, which depicts the proximity of neighboring vehicular nodes. Neighboring vehicles within a safe distance are visualized as green blinking squares, whereas alerts for possible collision and close proximity vehicles are denoted by red blinking squares. The granularity of the context information can be represented in two modes, coarse and fine. Figure 12 illustrates the VIS of Car A, of Fig. 1, depicting the pedestrian and Car B, of Fig. 1. The pedestrian is a dynamic alert in the Dynamic & Static Road

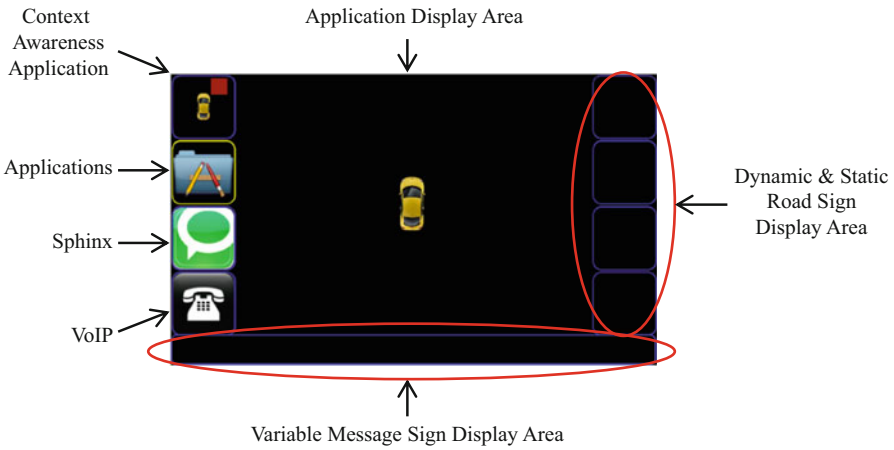

Fig. 11 Vehicle Integrated System (VIS) interface

Fig. 12 VIS on Car A in Fig. 1

Sign Display Area and Car B is a blinking green square on the rear right corner in coarse context. In this case, the pedestrian in Fig. 1 must be advertising itself as a pedestrian using the Pedestrian application available on the MIS, discussed later.

The context awareness application is initiated when vehicle 1 receives contextual data from neighboring vehicles, say vehicle 2. The context information contains vehicle 2's latitude (lat2), longitude (lon2), and heading. Vehicle 1 uses this context information to calculate the bearing and distance to vehicle 2. The bearing, illustrated in Fig. 13, is computed using Eq. (1).

$$\theta = a\ \tan 2\Big(\sin(\Delta\text{lon}) \times \cos(\text{lat2}), \cos(\text{lat1}) \times \sin(\text{lat2}) - \sin(\text{lat1}) \times \cos(\text{lat2}) \times \cos(\Delta\text{lon})\Big) \quad (1)$$

where θ represents the direction (in compass degrees) and $\Delta\text{lon} = \text{lon2} - \text{lon1}$.

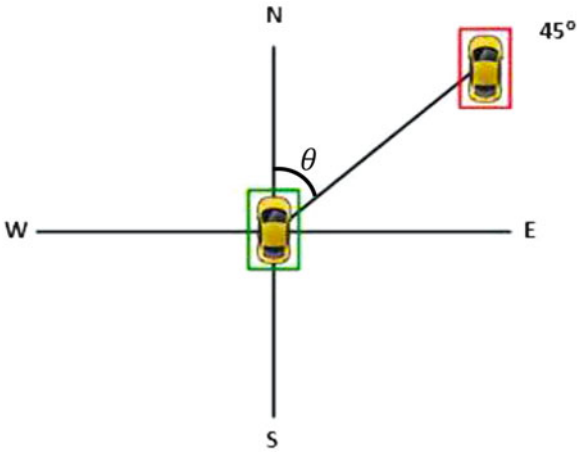

Fig. 13 Bearing between two vehicles

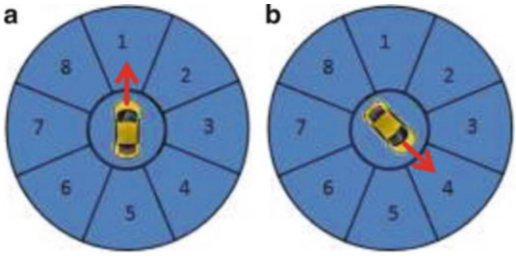

Fig. 14 Coarse context locations for vehicle (**a**) heading north and (**b**) southeast

The Haversine function, Eq. (2), is used to calculate the great-circle distance, Eq. (3), and determine distance, Eq. (4), between the two vehicles.

$$a = \sin\left(\frac{\Delta\text{lat}}{2}\right)^2 + \cos(\text{lat1}) \times \cos(\text{lat2}) \times \sin\left(\frac{\Delta\text{lon}}{2}\right)^2 \tag{2}$$

$$c = 2 \times a \tan 2\left(\sqrt{a}, \sqrt{1-a}\right) \tag{3}$$

$$d = R \times c \tag{4}$$

where $\Delta\text{lat} = \text{lat2} - \text{lat1}$ and $R = 6{,}371$ km, earth's mean radius.

Vehicle 1 uses the bearing and distance to vehicle 2 to deduce the context location of vehicle 2, that is, the location of vehicle 2 with respect to vehicle 1. The context locations in the coarse context awareness application are illustrated in Fig. 14, and here, context location 1 pertains to bearings in the range of 337.5–22.5 on a compass. Therefore, if vehicle 2's context location is 1 and vehicle 1 is heading north, then the coarse context awareness application would graphically depict vehicle 2 by blinking square one, of Fig. 15, in the Application Display Area.

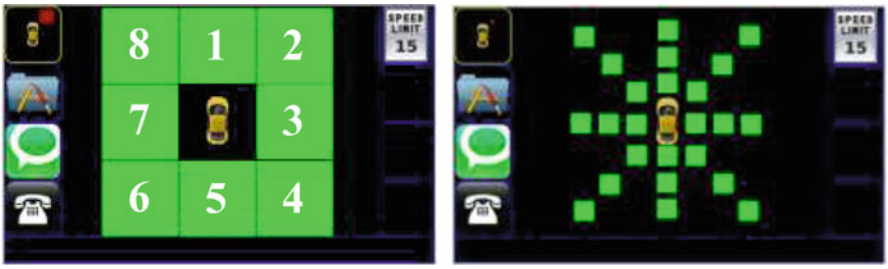

Fig. 15 Coarse and fine blink locations

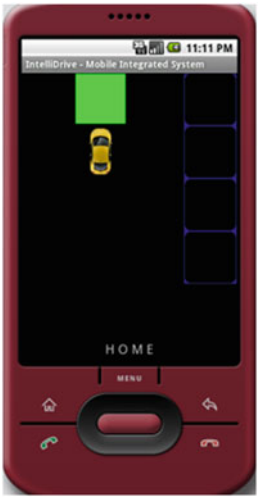

Fig. 16 Context Awareness application on MIS, cellular counterpart of VIS, in coarse mode

The color of the blinking square, green or red, would depend on the proximity of the vehicle. However, if vehicle 1 was heading southeast, as shown in Fig. 14b, and vehicle 2 was directly in front of it, then in coarse context, position 4 would be the blink location. It is important to note that context location accounts for the heading of the vehicle hosting the context awareness application.

The coarse and fine grain modes of the context awareness application differ in the number of context locations. Coarse and fine grain offer 8 and 24 context locations, respectively, illustrated in Fig. 15. Consequentially, fine grain context awareness can represent vehicles that are geographically farther away.

Once the context location and its corresponding blink location have been identified, it is stored in the local hash table of the host vehicle. Every 500 ms, the GUI thread checks all the neighbors in the hash table for their context location information and generates a list of blink locations, to be turned on. The context awareness application is also available on MIS, the cellular counterpart of VIS in AGORA, illustrated in Fig. 16.

Sphinx [7] offers hands free operation of VIS, providing a safe way to interact with the visual interface. Sphinx is speech recognition library developed by the

Table 1 Current list of VIS commands recognized by Sphinx

Words	VIS application
Application	Applications
Coarse	Coarse context
Fine	Fine context
Call	VoIP call
Voice	Sphinx

Sphinx group at Carnegie Mellon University. It allows the driver to switch between VIS applications using simple single word voice commands. When the user clicks the start button on the application, the system activates Sphinx and waits for the user to speak one of the recognized words. When the system hears a word that it recognizes, its switches focus to the associated application. Sphinx continues to operate until the user clicks the stop button. Table 1 delineates the current list of words supported by VIS to change between applications.

VIS also offers a VoIP application, which is essential to promote safety. It allows the system to send and receive voice of IP (VoIP) calls, for example to and from an emergency dispatch center. To activate a VoIP call, the user clicks the phone icon on the lower left corner of the VIS interface. VIS VoIP application uses MjSip [8] library to initiate a VoIP connection. MjSip is an open source java based implementation of the session initiation protocol (SIP) stack, which is the IETF (Internet Engineering Task Force) standard for multimedia sessions. Once the connection is established, the voice data is processed using Java Media Framework (JMF). The user ends the VoIP call by pressing the hang-up icon in the Application Display Area.

The Variable Message Sign Display Area in the VIS shows alerts and advertisements received from the TMC, via its web interface, as discussed earlier. The Dynamic & Static Road Sign Display Area is used by the road sign notification application to give real-time feedback of static road signs, such as stop sign, pedestrian crossing sign, speed limit sign, etc., and dynamic alerts, such as an approaching school bus, police car, ambulance, fire truck, pedestrian, etc. This application uses an XML file for retrieving static signs information.

We use a custom Road-Sign Capture tool, shown in Fig. 17, to populate the XML file with the static road sign information, for the region of interest. As illustrated, the XML file stores the list of static road signs and their longitude, latitude, heading, bearing, and display priority. A comprehensive static sign lists can be obtained from the Department of Transportation (DoT).

In the following section, we will discuss some user created AGORA applications. These applications are available on the VIS and the MIS, where appropriate.

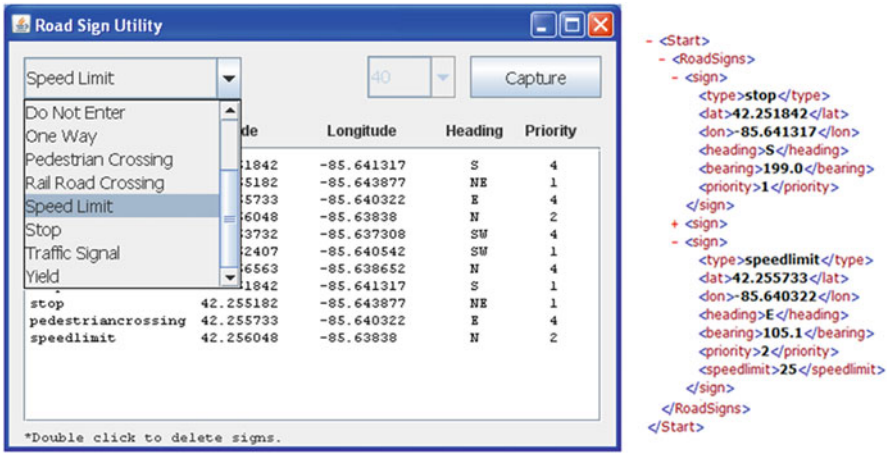

Fig. 17 Road-Sign Capture tool and an example XML file

Fig. 18 AGORA applications in VIS

3 AGORA Applications

AGORA offers several ITS applications for VIS and MIS, as shown in Fig. 18. ITS applications can be categorized under safety, efficiency, and infotainment. Safety applications are aimed at preventing damaging incidents on the road infrastructure with respect to life and property. Efficiency applications increase the efficiency of the vehicles and the transportation network. Infotainment applications can ease mundane tasks, such as finding dining restaurants, checking gas prices, or weather.

Fig. 19 (**a**) Pedestrian alert, (**b**) Government 2.0, and (**c**) Advisory Alert applications on MIS

3.1 Safety Applications

Some safety applications provided in AGORA include pedestrian alert, Government 2.0, advisory alert, etc. The Government 2.0 safety application, shown in Fig. 19b, can be used to report hazardous road conditions, based on current geographical location, to the TMC, which stores it in a database. Consequentially, safety applications like the Advisory Alert application, in Fig. 19c, are used to retrieve hazardous road conditions from the TMC based on the geographical location. AGORA applications like, Government 2.0 and Advisory Alert, use TMC hosted web services to push or pull alerts to and from the TMC.

The pedestrian alert application, illustrated in Fig. 19a, can be used by pedestrians, e.g. joggers, visually impaired, etc. to directly advertise their presence to other neighboring AGORA nodes. This information is piggy-backed on context awareness application messages. For example, the pedestrian alert application for the pedestrian in Fig. 1 must be playing, for the VIS of Car A to be alerted of the pedestrian by the pedestrian sign displayed in the Dynamic & Static Road Sign Display Area, as illustrated previously in Fig. 12. Similar XML configurable advertising applications are included in AGORA, such as Police, Ambulance, Fire Truck, Tow Truck, School Bus, and Smart Cones.

It is important to note that AGORA accounts for security of application usage by using an RFID Tag based User Restriction application, which authorizes application usage based on user privileges, as illustrated in Fig. 20. It uses information from RFID tag for authorization from the TMC. The RFID tag is embedded, for example

Fig. 20 RFID Tag based User Restriction safety application in VIS

in a key fob. The user will scan the fob, the RFID tag reader [9] will retrieve the user information and the RFID based User Restriction application will transmit the data to the TMC. The TMC will process and return the restriction type (access level). Once the authorization is complete, the appropriate application set becomes enabled for the corresponding user.

3.2 Efficiency Applications

A suite of efficiency applications have been developed for AGORA's VIS and, where possible, for MIS. These applications include Traffic Alerts, Ambulance Alert, Fire Truck Alert, etc. The Traffic Alerts application, shown in Fig. 21a, pulls road and transportation network information, such as construction delays, congestion, road closures, etc. The application uses TMC web services to pull data based on geographical location of the vehicle. This enables drivers to avoid these areas by seeking alternate routes and reducing congestion and traffic in already backlogged areas.

Other efficiency applications include applications such as the Ambulance Alert Application, in Fig. 21b, which can increase the response time of ambulances. The ambulance can advertise itself and traffic on its path can be alerted preemptively and diverted to allow clear passage for the emergency and first responder vehicles, like ambulance, fire trucks, and police.

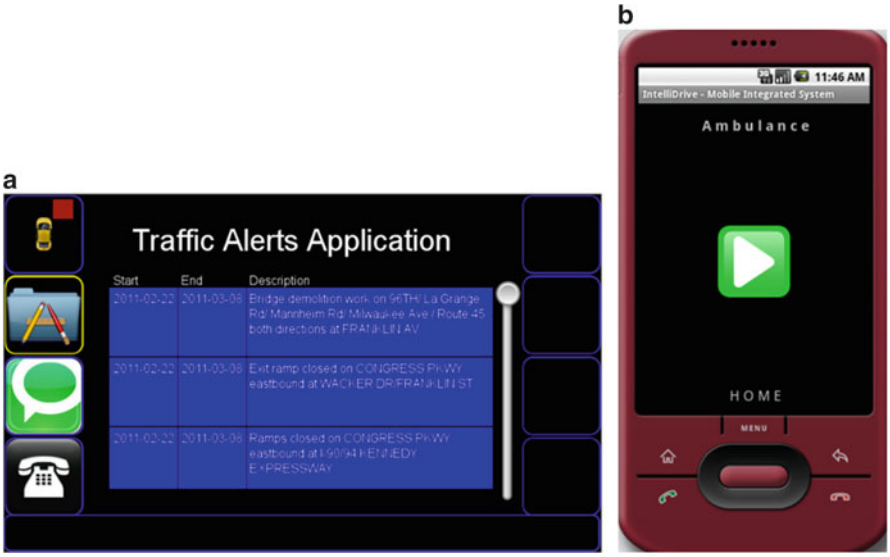

Fig. 21 (**a**) Traffic Alerts and (**b**) Ambulance Alert as efficiency applications in AGORA

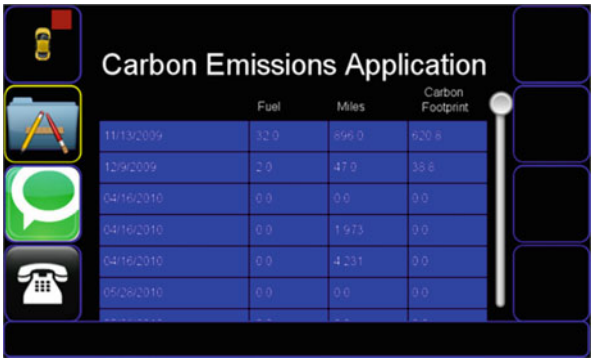

Fig. 22 Carbon Emissions infotainment application in VIS

3.3 Infotainment Applications

Various infotainment applications also accompany the AGORA suite of applications for VIS and MIS. These applications can provide information, convenience, and entertainment for AGORA users. Some of these applications include Carbon Emissions, Gas Prices, and Weather.

The Carbon Emissions application is an information application, illustrated in Fig. 22, tracks the amount of carbon emitted in the atmosphere by the vehicle. The tracking is displayed in the Application Display Area in a trip-by-trip format.

The Carbon Emissions Tracker calculates the miles traveled, fuel used, and carbon foot print for each trip. A trip is the time from which the engine is started,

Fig. 23 (**a**) Local Weather and (**b**) Fuel Locations infotainment applications in MIS

to the time it shuts off. The number of miles traveled is calculated using the GPS device and the OBU. Every 10 s, the GPS is polled for the new location of the car. The distance between the new latitude and longitude location and the old latitude and longitude location is then calculated and added to the running distance total. The amount of fuel consumed, mass air flow, and speed are obtained from the OBD and used to compute the instantaneous miles per gallon and consequentially the carbon footprint. This is stored for a trip in an XML file for storage and retrieval. The values automatically populate the grid in the Application display area, when the application starts, so no interaction is needed by the driver.

Weather and gas price applications provide vehicular or cellular users convenience in visualizing the weather forecast for the next 7 days or the list of the five cheapest gas stations, based on the current geographical location, as shown in Fig. 23a and b, respectively.

3.4 Development Environment and SDK

AGORA provides a solution that delivers rich location and context aware services to its users. AGORA client-side application technology is based on the use of "rules" to synthesize location and context aware applications. The idea is to enable the rapid and visual composition of location and context aware services based on "rules" in the form "IF conditions THEN actions," as illustrated in Fig. 24, where an XML file pairs the user defined conditions and actions.

AGORA provides a Development Environment, which follows the plug-in software model, enabling composition of audio-visual applications. Furthermore,

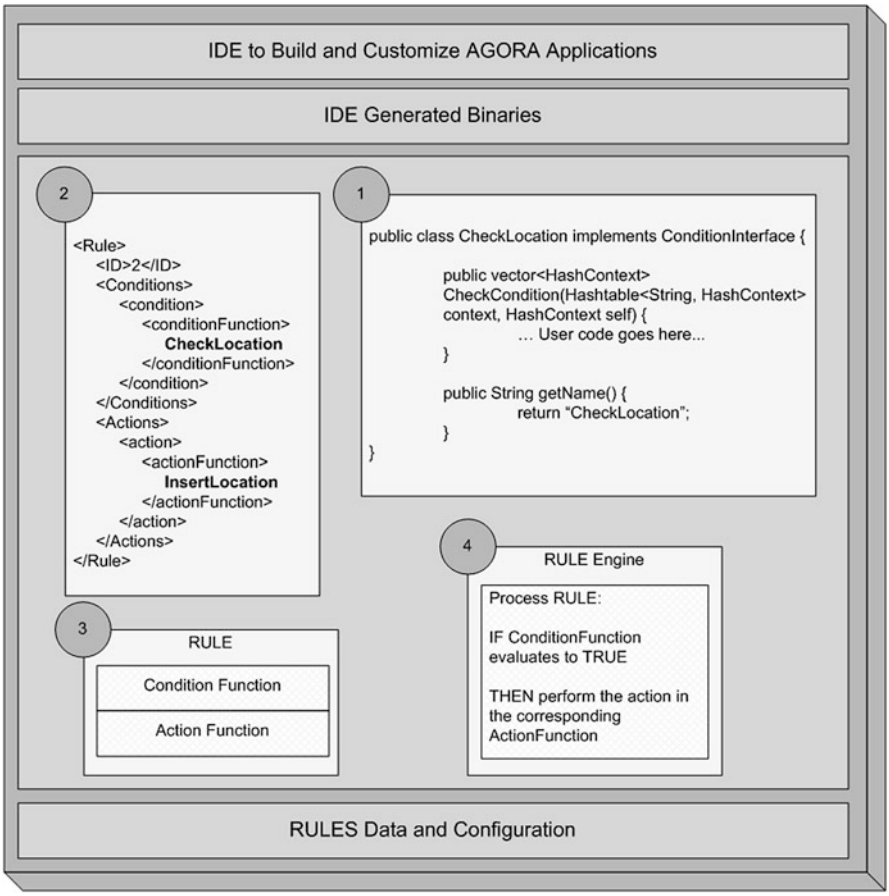

Fig. 24 AGORA Application Development Environment and Rule Engine

a complete SDK is provided for the development of more advanced applications. AGORA provides a set of vehicular applications out-of-the-box. However, users can build customized applications using AGORA's development environment and SDK.

AGORA applications consist of three components, Business Logic Rules, Audio-Visual, Braille-Code Interface Elements and Mapping Logic Rules. Business Logic Rules describe the core functionality of the application as a set of rules. A rule is composed of antecedent and consequent parts. The antecedent part returns references to the data items that fulfill the described conditions. The consequent part executes actions that can communicate, assert, or retract facts as needed. Audio-Visual and Braille-Code Interface Elements dictate the user interface of the application. Applications can be built to have a variety of interfaces including Audio-only, Visual-only, Audio-Visual or Braille-Code interfaces. This part of the

application is compiled separately and is linked to the Business and Mapping Logic parts of the application using Java Reflections technology. The Mapping Logic Rules pertain to the refresh rate of the audio-visual and Braille-Code elements that are associated with the application. There is provision to update these rules dynamically through TMC.

These rules enable the rapid development of applications fitting custom needs in AGORA client-side applications. AGORA system specifications, design documents, installation guide, development environment setup, and source code can be downloaded from http://www.cs.wmich.edu/~alfuqaha/AGORA/.

Acknowledgements This project was funded by Michigan Department of Transportation (MDoT) and undertaken by the Computer Networks, Embedded Systems and Telecommunications (NEST) Research Lab, Department of Computer, Western Michigan University. Research and development contributions came from Jamie Lynn Groos, Mrinal Khanvilkar, Vinay Babu Gavirangaswamy, and Jonathon Klobucar.

References

1. United States Census Bureau, U.S. Department of Commerce. Motor Vehicle Accidents and Fatalities – The 2012 Statistical Abstract – U.S. Census Bureau, 2012 [Online]. Available: http://www.census.gov/compendia/statab/2012/tables/12s1103.pdf. Accessed 5 May 2013
2. Energy Information Administration. 2010 Gasoline Consumption. American Fuels, 26 February 2011 [Online]. Available: http://americanfuels.blogspot.com/2011/02/2010-gasoline-consumption.html. Accessed 5 May 2013
3. Stoller G (2012) New report: road congestion wastes 1.9 billion gallons of gas. USA Today, 25 March 2012 [Online]. Available: http://usatoday30.usatoday.com/money/industries/energy/story/2012-03-25/wasted-fuel-report/53776164/1. Accessed 5 May 2013
4. Khaled Y, Tsukada M, Santa J, Ernst T (2009) On the design of efficient vehicular applications. In: IEEE vehicular technology conference (VTC), Barcelona
5. Dar K, Bakhouya M, Gaber J, Wack M (2010) Wireless communication technologies for ITS applications. IEEE Commun Mag 48(5):156–162
6. Motorola (2006) MOTOMESH 1.2 Network Setup and Installation Guide. Motorola, Motorola Inc., Schaumburg, Illinois, USA. http://meshsupport.motorolasolutions.com/Documents/Download/484
7. CMU Sphinx, Open Source Toolkit for Speech Recognition. Carnegie Mellon University [Online]. Available: http://cmusphinx.sourceforge.net/. Accessed 8 May 2013
8. MjSip. 22 April 2012 [Online]. Available: www.mjsip.org. Accessed 8 May 2013
9. Phidgets Inc. – Unique and Easy to Use USB Interfaces. Phidgets, Inc., 2012 [Online]. Available: http://www.phidgets.com/. Accessed 7 May 2013

Model, Analysis, and Improvements for Inter-Vehicle Communication Using One-Hop Periodic Broadcasting Based on the 802.11p Protocol

Tseesuren Batsuuri, Reinder J. Bril, and Johan J. Lukkien

Abstract Many future vehicle safety applications will rely on one-hop periodic broadcast communication (oPBC). The key technology for supporting this communication system is the new standard IEEE 802.11p which employs the carrier sense multiple access/collision avoidance (CSMA/CA) mechanism to resolve channel access competition. In this work, we first aim at understanding the behavior of such oPBC under varying load conditions by considering three important quality aspects of vehicle safety applications: reliability, fairness, and delay. Second, we investigate possible improvements of these quality aspects. We start with a clear mathematical model which gives the foundation for making an accurate simulation model as well as for defining new appropriate metrics to judge the aforementioned quality aspects. We evaluate oPBC with a strictly periodic broadcasting scheme, i.e., each vehicle broadcasts messages in a strictly periodic manner. The evaluation reveals that the hidden terminal, or Hidden Node (HN), problem is the main cause of various quality degradations especially when the network is unsaturated. To be more specific, the HN problem reduces the message reception ratio (i.e., reliability degradation) and causes unfair message reception ratios for vehicles (i.e., fairness degradation). Moreover, it causes long-lasting consecutive message losses (i.e., delay degradation) between vehicles while they are encountering each other, i.e., entering their Communication Ranges (CRs). In some serious cases, a certain vehicle could not successfully deliver any of its messages to a particularly destination vehicle throughout an entire encounter interval of these two vehicles. We propose three simple, but effective broadcasting schemes, to alleviate the impact of the HN problem. Though these solutions do not affect the message reception ratio (i.e., reliability) of the entire network, they do improve the fairness and delay aspects. These solutions are fully compatible with the IEEE 802.11p standard, i.e., they are application-level solutions and can be easily introduced in practice.

T. Batsuuri • R.J. Bril (✉) • J.J. Lukkien
Department of Mathematics and Computer Science, Eindhoven
University of Technology, Eindhoven, The Netherlands
e-mail: r.j.bril@TUE.nl

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_9

1 Introduction

A rapid progress in mobile and wireless technologies in the last decade enables a wide spectrum of applications in the intelligent transportation system (ITS) domain targeting vehicle safety, transportation efficiency, and driver comfort. In recent years, many industry/government consortiums are formed around the world to carry out projects to investigate such applications: the Vehicle Safety Communications consortium in the USA [1], the Car2Car Communication consortium in Europe [2], and the Internet ITS consortium in Japan [3]. As a result of these efforts, many interesting vehicle safety application scenarios are identified and their communication requirements are carefully examined. It is now becoming clear that most of these applications will rely on broadcast communication that comes in two flavors: event-driven and time-driven. In the event-driven (or emergency) case, a vehicle starts broadcasting a safety message for a certain duration periodically when a hazardous situation is detected and, hence, these messages are not sent in a normal situation. In the time-driven case, each vehicle continuously performs one-hop periodic broadcast communication (oPBC) to pro-actively deliver a beacon message with its status information (e.g., position, speed) to the neighboring vehicles. The key idea of such oPBC is to make each vehicle aware of its vicinity such that future vehicle safety applications running on the vehicle will leverage this information to detect any hazardous situation in a timely manner. A lane change advisor and a forward collision warning application [1] are two typical examples that rely on this oPBC. These applications require a frequency of ten messages per second with a maximum no message interval (or a tolerance time window) of [0.3 s, 1.0 s] [1, 4, 5]. In addition, these applications pose a strict fairness requirement on oPBC [6, 7], where each vehicle should have equal opportunity for using the shared channel. In this type of system, message loss is unavoidable (we explain the causes below); however, it must not be the case that one or a few vehicles take all the loss, because this would result in these vehicles becoming invisible (from a communication point of view) to their surrounding vehicles.

In this work, we focus on this oPBC from a vehicle safety application perspective. Particularly, we are interested in oPBC which is addressed in the IEEE 802.11p [8], IEEE 1609 standards [9–12], and SAE J2735 [13]. The 802.11p standard has been designed specifically for inter-vehicle communication. Besides the regular support for higher-layer protocols like IP, the 802.11p medium access control (MAC) supports a short message protocol called WSMP (WAVE short message protocol, IEEE 1609, where WAVE stands for wireless access in vehicular environments). Among other uses, this WSMP protocol together with the SAE J2735 addresses the transmission of basic safety message (BSM) also known as a beacon message that is used by a vehicle to inform other vehicles about its status and condition. The BSM (in the rest of paper, we simply call it message) is sent periodically, in broadcast mode, with a typical frequency of 10 Hz.

In general, the members of the 802.11 family, where the 802.11p is one of the newest members, of wireless standards support two communication modes: a managed mode called *point coordination function (PCF)* where a base station

manages access to the channel and an ad-hoc mode called *distributed coordination function (DCF)* where stations collaborate to manage channel access [14]. In DCF, stations employ the CSMA/CA mechanism to resolve channel access competition. For point-to-point communication, stations repeatedly perform channel sensing followed by a random Back-off (Bf) period selected from an increasing Contention Window (CW). Bf is used to reduce the probability of a contention problem which occurs when two or more stations that exist in each other's Communication Range (CR) incidentally happen to start transmission at the same time causing collisions. In addition, Request To Send/Clear To Send (RTS/CTS) signaling is used to resolve the hidden terminal, or Hidden Node (HN), problem which occurs when two stations that are outside each other's CR have overlapping transmissions in time interfering with their common neighbors in the intersection of the CRs. On top of this, a MAC level acknowledgement can be used to resolve the remaining message losses. An initial channel access delay, namely arbitration inter frame space (AIFS), allows discriminating among several priority classes. When stations broadcast messages rather than sending point-to-point the situation in DCF is quite different. First, CW from which a Bf period is drawn is fixed, and Bf is at most done once [14]. Second, RTS/CTS signaling and MAC layer acknowledgement do not work since there is no particular destination for a message. As a result, when all stations use broadcast-based communication, the collision problems, i.e., the contention and the HN problems increase. Figure 1 gives an overview of the 802.11p communication behavior in broadcast mode and illustrates the collision problems.

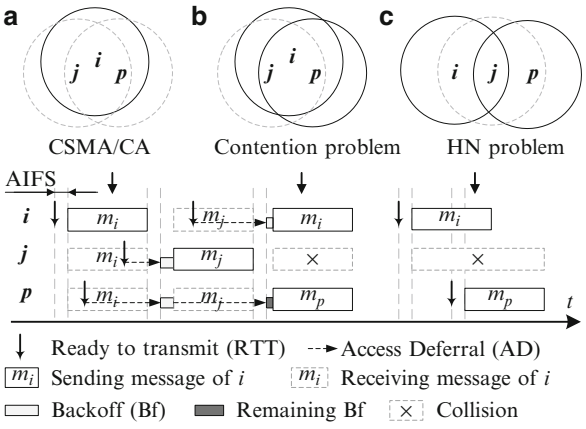

Fig. 1 The 802.11p MAC CSMA/CA in broadcast mode. (**a**) shows the contention protocol that serializes three stations. A station, which wants to transmit, first must check if the channel is free for a duration of Arbitration Inter Frame Space (AIFS). In case the channel is found active during this duration, the intended transmission is deferred and a Bf is performed. (**b**) shows a message collision due to a contention problem which happens if incidentally, two stations start transmission at the same time (mainly because their Bf periods expire at the same time), i.e., a Neighboring Node (NN) collision. (**c**) shows a message collision due to the Hidden Node (HN) problem, i.e., a HN collision: two stations that cannot sense each other's transmission may cause message collisions in the intersection of their Communication Ranges (CRs)

The purpose of our research is to understand the behavior of this oPBC based on the 802.11p DCF. We want to understand message losses due to the contention and HN problems under varying load conditions. In particular we want to understand oPBC by considering three quality aspects which are important for vehicle safety applications: *reliability* (i.e., successful message reception ratio), *fairness* (i.e., distribution of successful message reception ratio over vehicles), and *delay*[1] (i.e., the longest interval in which a vehicle that is in the CR of another vehicle doesn't receive a message from that latter vehicle.). In addition, we want to investigate possible improvements.

Our first step is to develop a mathematical model of the 802.11p behavior under oPBC. This serves three purposes. First, it makes the discussion unambiguous. Second, it gives the foundation for a simulation model and third, it allows us to develop new relevant metrics (criteria) to judge communication quality for the above aspects. Standard performance metrics like a successful message reception ratio of the entire network or an average end-to-end delay fall short in this environment. Our second step is to simulate according to this model and to determine the values of the given metrics. This gives us insight in shortcomings of this oPBC and gives directions for improvements.

The contributions of this work are fourfold. First, we give a novel timing model of oPBC under the 802.11p DCF. Second, we define several new metrics for judging the quality of oPBC. Third, we simulate oPBC under several different circumstances showing problems and shortcomings. Namely, the simulation reveals that the HN problem is the main cause of various quality degradations when the network is unsaturated. It reduces the message reception ratio of the entire network (i.e., *reliability* degradation) and causes unfair message reception ratios for vehicles (i.e., *fairness* degradation). Moreover, it causes long-lasting consecutive message losses (i.e., *delay* degradation) between vehicles while they are encountering each other, i.e., entering their CRs. In some serious cases, a certain vehicle could not successfully deliver any of its messages to a particularly destination vehicle throughout an entire encounter interval of these two vehicles. Fourth, we identify application-level improvements. We propose three simple, but effective broadcasting schemes, to alleviate the impact of the HN problem. These solutions are fully compatible with the IEEE 802.11p standard, i.e., they are application-level solutions and can be easily introduced in practice.

The remainder of the paper is organized as follows. Section 2 introduces the mathematical model of oPBC. Section 3 presents an evaluation study of oPBC by means of simulations which are carried out according to the model, and Sect. 4 presents our solutions for the collision problems. Section 5 covers recent studies most relevant to our work. Finally, Sect. 6 gives a summary of the main results and presents an outlook on future work. In addition, Appendix A gives a detailed overview of CSMA/CA in broadcast mode, and Appendix B gives a short overview of a signal reception model which is chosen for our simulation model.

[1] In this paper,"No Message Interval" is also used to denote "delay."

2 Models of oPBC

In this section, we introduce our model of oPBC. Based on the model, we show how the appropriate metrics are defined for evaluating the three quality aspects of oPBC (i.e., *reliability*, *fairness*, and *delay*).

2.1 Two Models

In our analysis of communicating vehicles we encounter two aspects: a simulation of the movement of the vehicles and a simulation of the behavior of the wireless communication as a function of the position of the vehicles. Thus we have the *traffic model* which yields the position of vehicles as a function of time and the *communication model* that describes the communication events between vehicles as a function of time and vehicle location. Hence, the communication model uses the traffic model's output as one of its input parameters.

The interface between the two models is formed by the location of the vehicles. Together with the radio channel model this yields the *neighborhood* structure viz., a set of vehicles that each vehicle can transmit to or receive from at any point in time. The traffic model can be very advanced, even to the extent that life traces are simulated [15, 16]. In this work we are not concerned, however, with the traffic model. For our simulations we stick to a simple highway model, represented as a stretch of several kilometers with three lanes per direction and periodic boundary conditions (which makes it, in fact, a loop). Speeds per lane are assumed to be fixed. In our simulations, the main concern of the traffic model is to simulate with a small enough time step to have a realistic and sufficiently accurate description for the communication model. The motivation for this restriction is that we want to study just the communication model under varying load conditions.

2.2 The Communication Model

The communication model consists of two parts. First, we describe the communication of the 802.11p standard and the radio channel model that generates the events. Second we model timing and events in the system consisting of communicating vehicles to define the concepts of interest.

2.2.1 The 802.11p Communication and the Radio Channel Model

We restrict ourselves to describing the broadcast mode of the 802.11p MAC. In Appendix A, Fig. 21 gives an extended flowchart of the broadcast procedure taken

Table 1 The 802.11p parameter settings

Parameters	Values
Date rate	6 Mbps
A slot duration	13 μs
AIFS	6 slots
CW size	7 slots
Preamble length	40 μs
Antenna gain	0 dB
Antenna height	1.5 m
Noise floor(nF)	−99 dBm
Power sense threshold($PsTh$)	−92 dBm
Carrier sense threshold ($CsTh$)	−85 dBm
SINR threshold ($SrTh$)	8 dB

from [17], and Fig. 22 describes the corresponding state machine diagram. In our simulation model, every vehicle is implemented according to this state machine. Besides, we take a Signal to Interference plus Noise Ratio (SINR) based signal reception model of the updated NS-2 implementation of the 802.11p [18] and a brief description of this model is given in Appendix B. In addition, we choose the Two-Ray Ground (TRG) signal propagation model, because we study the effect of message collisions, which are properly covered by this model, rather than environmental impacts (e.g., weather conditions or interference from other objects). The main configuration parameters of the 802.11p and the TRG model are chosen as in Table 1.

2.2.2 The Timing Model

We assume a set V of N vehicles $v_1, v_2, \ldots v_N$ periodically broadcasting messages. The behavior of the system is described as a series of events happening at certain times. As a convention we use a superscript to denote a kth occurrence or instance. For example, $e^{(k)}$ denotes the kth occurrence of an event e and $m_i^{(k)}$ denotes the kth message of v_i. In addition, we often do not name the event but only the time of occurrence using a similar notation, as explained next.

The activation time $a_i^{(k)}$ is the time at which v_i becomes ready to broadcast $m_i^{(k)}$. The start time $s_i^{(k)}$ and finish time $f_i^{(k)}$ are the times at which v_i actually starts and finishes the transmission of message $m_i^{(k)}$, respectively. Note, from a receiver vehicle's perspective, the start time and the finish time at which the vehicle starts and finishes receiving the message $m_i^{(k)}$ are $s_i^{(k)} + \delta$ and $f_i^{(k)} + \delta$, respectively. δ is an air propagation delay that is relatively small,[2] therefore we neglect this in our model.

[2] $\delta \ll 1\mu s$ [19, 20].

The transmission interval $tI_i^{(k)}$ of message $m_i^{(k)}$ is defined as

$$tI_i^{(k)} \stackrel{\text{def}}{=} [s_i^{(k)}, f_i^{(k)}) \,. \tag{1}$$

We require that

$$a_i^{(k)} < s_i^{(k)} \leq f_i^{(k)} \leq a_i^{(k+1)} \tag{2}$$

holds. Message transmission is assumed to be periodic. If a message is not sent at all or is delayed such that the remaining part of the interval is not enough for successful completion, we say that the message is dropped. This may mean a partial message transmission or, in the extreme case, no transmission at all ($s_i^{(k)} = f_i^{(k)}$). In both cases, we define $f_i^{(k)} = a_i^{(k+1)}$ and we take that as the condition of message dropping.

Moreover, we define transmission power $Pt_i(t)$ of vehicle v_i and its reception power at vehicle v_j as $Pr_{ij}(t)$ and cumulative reception power $cPr_j(t)$ of vehicle v_j at time t. Note, we always assume that $i \neq j$ holds whenever we talk about two vehicles v_i and v_j. We require that $Pt_i(t) > 0$ holds during $tI_i^{(k)}$ and its value is determined by the application. $Pr_{ij}(t)$ is determined by a given signal propagation model, by $Pt_i(t)$ and by the distance between sender and receiver at time t. $cPr_j(t)$ is determined by all receiving signal strengths at v_j at time t plus a noise floor, nF, as follows:

$$cPr_j(t) = nF + \sum_{v_i} \{Pr_{ij}(t) | Pr_{ij}(t) \geq PsTh\}, \tag{3}$$

where $PsTh$ is a Power Sense threshold of the receiver. Given these notions, we define the neighborhood of a vehicle. At any time t, each vehicle v_i has a target neighbor set of other vehicles, $Nb_i(t)$, where $v_j \in Nb_i(t)$ means that v_j is in the CR of v_i at time t. It is defined as follows:

$$v_j \in Nb_i(t) \stackrel{\text{def}}{=} \frac{Pr_{ij}(t)}{nF} \geq SrTh, \tag{4}$$

where $SrTh$ is an SINR threshold for receiving the message successfully. Note, CR is the reception range, the places where the message could be received disregarding interference of other stations. A necessary condition for receiving a message is that the receiving vehicle must be in the CR of the sending vehicle for the duration of the message transmission. A sufficient condition for a message reception is that the receiving signal power must be equal to or greater than $SrTh$ with respect to the cumulative power of all other signals for the entire duration of the message transmission. This is defined as follows:

$$\forall t : t \in tI_i^{(k)} \ \wedge \ \frac{Pr_{ij}(t)}{(cPr_j(t) - Pr_{ij}(t))} \geq SrTh. \tag{5}$$

We extend the concept of a neighborhood to intervals by

$$\downarrow Nb_i(I) = \bigcap_{t \in I} Nb_i(t) \ . \tag{6}$$

This interval represents all vehicles that have been in the CR of vehicle v_i during the entire interval I. Changes of neighbor sets are represented by enter and leave events. Entering time $e_{ji}^{(k)}$ is the time at which v_j enters the CR of v_i for the kth time while leaving time $l_{ji}^{(k)}$ is the time at which v_j leaves the CR of v_i for the kth time. The kth encounter interval $eI_{ij}^{(k)}$ of v_j with v_i is defined as

$$eI_{ij}^{(k)} \stackrel{\text{def}}{=} [e_{ji}^{(k)}, l_{ji}^{(k)}) \ . \tag{7}$$

During $eI_{ij}^{(k)}$ we say that there is a link from i to j and we call that the kth such link. Figure 2 gives a schematic of these notions.

Message Loss

The most important concern is whether messages are actually received by vehicles that could receive them. Considering message $m_i^{(k)}$ there are three reasons why another vehicle v_j might not receive it.

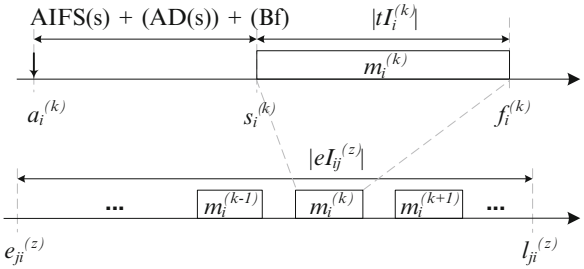

Fig. 2 Vehicle v_j enters the CR of v_i at $e_{ji}^{(z)}$ and leaves it at $l_{ji}^{(z)}$. During this encounter interval, v_j receives a sequence of messages from v_i. A vehicle v_i becomes ready to broadcast its kth message at $a_i^{(k)}$ but it actually starts the transmission at $s_i^{(k)}$ and finishes at $f_i^{(k)}$. The distance between $a_i^{(k)}$ and $s_i^{(k)}$ depends on the channel condition. In the best case, it can be only an AIFS. In the worst case, it can be multiple AIFSs + multiple ADs (Access Deferrals) + Bf

- *(OOR) Out Of Range.* In order for a vehicle v_j to receive $m_i^{(k)}$ it must be in the neighborhood of v_i for the duration of the transmission. When $v_j \notin \downarrow Nb_i(tI_i^{(k)})$, v_j does not receive $m_i^{(k)}$.
- *(MD) Message Dropping.* This happens, as described above, if the back-off interval becomes so long that the message transfer time does not fit in the remaining part of the period. In our model this is equivalent to

$$f_i^{(k)} = a_i^{(k+1)} . \tag{8}$$

  No vehicle will receive message $m_i^{(k)}$.
- *(MC) Message Collision.* The message is transmitted but not received by v_j since other vehicles may transmit at the same time to v_j and their interferences are strong enough to corrupt the receiving message of v_i. This is defined as follows:

$$\exists t : t \in tI_i^{(k)} \ \wedge \ \frac{Pr_{ij}(t)}{(cPr_j(t) - Pr_{ij}(t))} < SrTh. \tag{9}$$

Given these reasons for loss we define the *transmission condition* of message $m_i^{(k)}$ and, accordingly, the *reception condition* of $m_i^{(k)}$ by a vehicle v_j as follows:

$$Tc_i^{(k)} = \begin{cases} MD & \text{if (8)} \\ XMT & \text{otherwise} \end{cases} \tag{10}$$

$$Rc_{ij}^{(k)} = \begin{cases} OOR & \text{if } v_j \notin \downarrow Nb_i(tI_i^{(k)}) \\ MC & \text{if } v_j \in \downarrow Nb_i(tI_i^{(k)}) \ \wedge \ (9) \\ Tc_i^{(k)} & \text{otherwise.} \end{cases} \tag{11}$$

If $Tc_i^{(k)} = XMT$, message $m_i^{(k)}$ is broadcast successfully. If $Rc_{ij}^{(k)} = XMT$, the message is received by vehicle v_j at time $f_i^{(k)}$ successfully.

Metrics

We define the most appropriate metrics that can judge the communication quality for the three most important aspects of the safety applications: *reliability*, *fairness*, and *delay*.

For the *reliability* aspect, we use the fraction of successfully delivered messages (*SMR*, successful message ratio). This concept can be refined to links between vehicles and to individual messages. To start we define the number of received messages from v_i by v_j in a given interval, as well as the number of times that such message could have been received. The ratio is the SMR in that interval.

$$Rs_{ij}(I) = |\{k \mid tI_i^{(k)} \subseteq I \wedge Rc_{ij}^{(k)} = XMT\}| \tag{12}$$

$$Ns_{ij}(I) = |\{k \mid tI_i^{(k)} \subseteq I \wedge Tc_i^{(k)} = XMT \wedge v_j \in \downarrow Nb_i(tI_i^{(k)})\}| \tag{13}$$

$$SMR_{ij}(I) = \begin{cases} \frac{Rs_{ij}(I)}{Ns_{ij}(I)} & \text{if } Ns_{ij}(I) > 0 \\ 0 & \text{if } Ns_{ij}(I) = 0 \end{cases} \tag{14}$$

Generalizing this by summing over the receiving vehicles gives the SMR of v_i in an interval.

$$SMR_i(I) = \begin{cases} \frac{\sum_{v_j} Rs_{ij}(I)}{\sum_{v_j} Ns_{ij}(I)} & \text{if } \sum_{v_j} Ns_{ij}(I) > 0 \\ 0 & \text{if } \sum_{v_j} Ns_{ij}(I) = 0 \end{cases} \tag{15}$$

As a special case, $SMR_i(tI_i^{(k)})$ is the *SMR* of $m_i^{(k)}$. Again, generalizing by summing over the sending vehicles we obtain the *SMR* of the entire network during that interval.

$$SMR(I) = \begin{cases} \frac{\sum_{v_i,v_j} Rs_{ij}(I)}{\sum_{v_i,v_j} Ns_{ij}(I)} & \text{if } \sum_{v_i,v_j} Ns_{ij}(I) > 0 \\ 0 & \text{if } \sum_{v_i,v_j} Ns_{ij}(I) = 0 \end{cases} \tag{16}$$

At the network level an interesting question is: how does $SMR([0,T))$, where T represents a time of consideration, behave as a function of vehicle density?

From the *fairness* perspective, the behavior of individual vehicles is more important than the average. This is why we also analyze SMR_i to see whether losses are distributed evenly (or fairly) over the vehicles. The cumulative distribution function shows this; a fair distribution would give a transition from 0 to 1 within a short interval.

$$cdfSMR(I,x) = \frac{|\{v_j \mid SMR_j(I) \leq x\}|}{N}, \text{ for } 0 \leq x \leq 1 \tag{17}$$

In addition, plotting SMR_i as a function of time gives insight in the visibility of v_i for other vehicles.

Finally, from the *delay* perspective, an important further question is how losses of a particular vehicle are distributed in time and across vehicles: do losses happen in sequences and do they affect the same links? To that end we define the concept of a "No Message Interval" between two vehicles during a given interval I which is the length of the longest subinterval of I without a successful message transmission. In addition, the "First Delay" is the length of the longest initial subinterval and represents a delay in discovery in case we apply it to an encounter interval.

$$NoM_{ij}(I) = \sup\{|J| \mid J \subseteq I \wedge Rs_{ij}(J) = 0\} \tag{18}$$

$$FD_{ij}([a,b)) = \sup\{x \mid [a,a+x) \subseteq [a,b) \wedge Rs_{ij}([a,a+x)) = 0\} \tag{19}$$

In our analysis, we look at genuine *NoM* and *FD*, viz., those that correspond to encounter intervals. These are examined as a function of their length and plotted as a density (histogram) or as a cumulative distribution.

3 Evaluation of oPBC Under CSMA/CA Coordination

In this section, we evaluate oPBC using the newly defined metrics. We implemented a simulator according to the model and verified its correctness against the updated NS-2 implementation of 802.11p [18]. The following subsections describe the simulation setup, results, and analysis.

3.1 Simulation Setup

For the purpose of this evaluation, two different scenarios are simulated. In the first scenario (single domain (SD)), vehicles are deployed at fixed locations within a single CR viz., all vehicles can receive each other's messages. This scenario allows us to study the collisions caused only by the contention problem, i.e., NN collisions since there are no HNs. In the second scenario (multi domain (MD)), vehicles are deployed on a 3 km long highway with three lanes per direction. This scenario allows us to study both HN and NN collisions. By having these two scenarios, we can compare the impact of these two types of collisions. The vehicles at the three lanes have fixed velocities of 20, 30, and 40 m/s, respectively. In both scenarios, different inter-vehicle spacings are used in order to create different *Vehicle Densities* (*VD*). We assume a single channel, a fixed broadcasting period and initially, a random phasing within this period as

$$a_i^{(k)} \stackrel{\text{def}}{=} \phi_i + kT_i. \tag{20}$$

Thus, each v_i has a broadcasting period $T_i \in \mathbb{R}^+$ and an initial broadcasting phase $\phi_i \in \mathbb{R}^+$, where ϕ_i is uniformly selected from an interval of $[0, T_i)$. Moreover, we assume the same signal strength, the same broadcasting period, and the same message size fixed over time for all vehicles. Table 2 presents the values of the simulation parameters.

Table 2 Simulation settings

Parameters	Values
Message size	555 bytes
CR	300 m
Broadcasting period (T)	0.1 s
Simulation length	60 s

3.2 Simulation Results and Analysis

First, we study the *reliability* by means of the successful message reception ratio metric, i.e., *SMR*([0,60)). The *SMR* of the overall network with respect to *VD* of the SD and MD cases are shown in Fig. 3. For each different *VD* case, we performed ten simulations with a different random seed for selecting the initial phases. Figure 3 presents the average values of these simulations with a confidence interval of 99 %. In addition, the theoretical maximum *SMR* (*maxSMR*) is plotted to show the upper boundary. This *maxSMR* is given by

$$maxSMR(VD) = \begin{cases} 1 & \text{if } VD \leq SP \\ SP/VD & \text{otherwise} \end{cases}, \tag{21}$$

where *SP* is the channel saturation point, i.e., the maximum capacity of the channel in terms of the number of vehicles that can fit in one period duration without any overlap in time for broadcasting. SP is given as

$$SP = \frac{T}{T_s + T_d}, \tag{22}$$

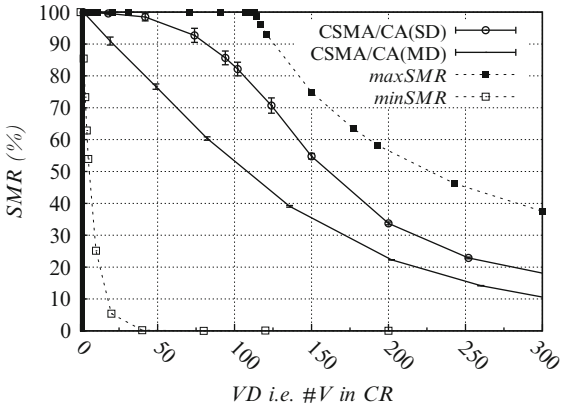

Fig. 3 Successful Message Ratio (*SMR*) of the entire network with respect to the vehicle density (*VD*) shows the communication reliability. *maxSMR* is the maximum possible *SMR* calculated analytically. *minSMR* is the minimum possible *SMR* obtained by means of simulations in which all vehicles have approximately the same phases for broadcasting. SD (Single Domain) case shows *SMR* degradation only due to the contention problem, where all the vehicles are deployed at fixed locations within a single CR, therefore, there are no HNs. MD (Multi-Domain) shows *SMR* degradation due to both the contention and HN problems, where all vehicles are deployed on a 3 km long highway with three lanes per direction. Both cases show average values of ten simulations with a 99 % confidence interval

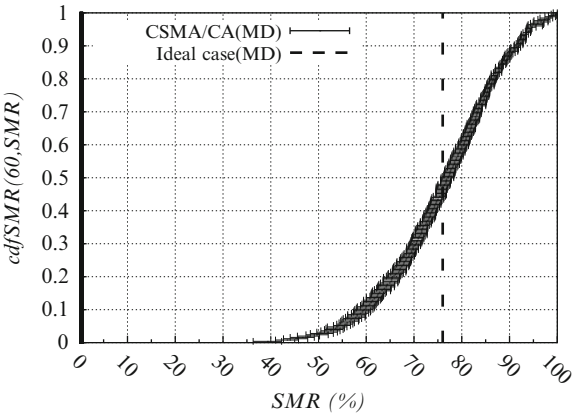

Fig. 4 This shows the fairness through CDF of vehicles by their *SMR*, when *VD* is about 50. In the graph, a point indicates that y % of vehicles have at most x % *SMR*. The result shown is an average of ten simulations with a confidence interval of 99 %. In an ideal fair case, the dashed line is expected where all vehicle should have the same *SMR* that is equal to SMR of the entire network

where T_s is the inter-frame space, i.e., an *AIFS* duration, and T_d is the time to transmit a single message. *When all vehicles are optimally synchronized over the period for broadcasting, the SMR should approach this maxSMR level.* Besides, we obtained the minimum possible SMR level (*minSMR*) by means of simulations in which we defined approximately the same phases for all vehicles.

From Fig. 3, the HN problem appears to be the main cause of *SMR* degradation when the network is unsaturated. Once the network load exceeds its maximum capacity, the NN collisions start occurring in bursts, thus yielding lower *SMR*.

We now continue our study at individual vehicle level to investigate the *fairness*. Here, we select an unsaturated network condition where the traffic density is sparse, i.e., *VD* is about 50 vehicles (that corresponds to about 85 vehicles per km over six lanes in our settings). Figure 4 shows a relatively unfair distribution of message receptions over vehicles where some vehicles have a high *SMR* whereas others have a relatively low *SMR*. In an ideal fair case, the dashed line is expected where the distance between the best and worst cases should be close to 0, i.e., a transition from 0 to 1 within a short interval. In this case, however, the distance between the best case and worst cases is approximately 65 %.

Figure 5 shows how *SMR* of an individual vehicle evolves over a time interval of 15 s. We take only an interval of 15 s from a simulation of 60 s for better visibility. The graph shows two specific cases of vehicles with the best *SMR* and the worst *SMR*. From this graph, we can conclude that vehicles remain in one condition (a certain *SMR* level) for a relatively long period of time if we ignore some fluctuations there. In addition, we can say that the best and the worst vehicles performances are clearly distinguished.

Figures 6 and 7 show the impact of the collision problems on the *delay* aspects at the link level through a cumulative distribution of links by their *NoM* and a histogram of links by their *FD*, respectively. During 60 s of simulation, approximately 53,000 links are established in total. Note, the link is a one-way relationship. Some vehicles join the CR of a vehicle whereas some may leave the CR due to the relative speed between the vehicle and its neighbors. From Fig. 6,

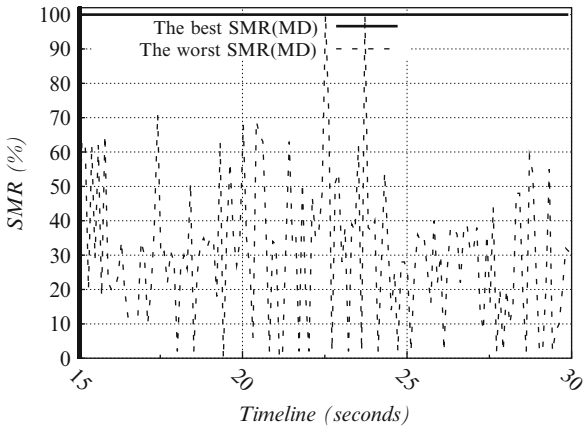

Fig. 5 *SMR* fluctuations over time of two individual vehicles that experience the best and the worst SMR, respectively, when *VD* is approximately 50. It shows that there is a clear difference between the best and the worst cases. The graph shows the results of an arbitrary simulation

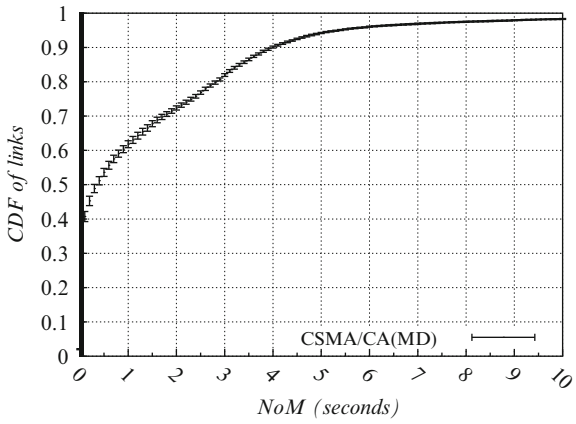

Fig. 6 CDF of links by their *NoM* (the longest no message interval), when *VD* is approximately 50. In the graph, a point indicates that y % of links have at most x seconds of *NoM*. The result shown is an average of ten simulations with a confidence interval of 99 %

we can see that almost 30 % of links experience more than 1 s of *NoM*. This implies that a certain vehicle does not receive a sequence of messages from another vehicle although the vehicle could have received these messages in the absence of interferences. From Fig. 7, many vehicles, i.e., about 350±50 are seen that did not even discover some of their one-hop neighboring vehicles for their entire encounter interval.

3.3 Summary of the Evaluation

This section summarizes the main findings of the evaluation. We conclude the following:

- The HN problem is the main cause of the *SMR* degradation when the network is unsaturated. Once it is saturated, the NN problem reduces the *SMR* dramatically. Therefore, the latter one is more a network congestion problem. In fact, this

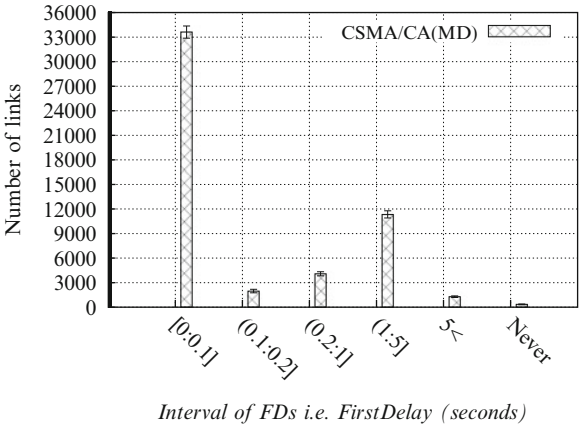

Fig. 7 Distribution of links by their *FD* (delay to discover a new neighbor), when *VD* is approximately 50. The graph presents 5 different intervals of *FD*. The last interval "Never" means that some vehicles never discover its neighbors. The number of links for "5<" and "Never" are 1280±95 and 350±50, respectively. The result shown is an average of ten simulations with a confidence interval of 99 %

congestion problem is well known and addressed in many works, e.g., [5, 7], and [21]. The main approaches are to reduce beacon generation rate, beacon size, or to reduce the CR which are indeed all derived from (22).

- The impact of the HN problem is clearly revealed by an unfair *SMR* distribution and the delay characteristics such as long-lasting no message interval and first delay.
- The aforementioned quality impacts are mainly due to synchronized HNs, because vehicles traveling on a highway, particularly those traveling in the same direction could have a rather static topology for a relatively long period, i.e., in the order of multiple seconds.[3] In that topology, some vehicles could be incidentally synchronized as HNs which leads to a systematic message loss. It can be explained by a simple scenario where two vehicles are moving forward in the same direction as illustrated in Fig. 8. Assume further that the difference between the message broadcasting phases (ϕ) of the vehicles is less than a message transmission time, and they do not have any contenders (i.e., no vehicle in the CR with the same phase). In that scenario, both vehicles would remain broadcasting nearly at the same time and the messages would always collide at receiving vehicles in the intersection of the CRs as described in Fig. 1c.

4 Solution Study on the Collision Problem in oPBC

We are interested in the collision problems, particularly the HN problem, when the traffic density is moderate or sparse, i.e., when the network is unsaturated. We assume that in that situation message loss is even more serious in terms of vehicle

[3]When CR is 300 m, two vehicles approaching each other from the opposite directions with a relative speed of 80 m/s will have an encounter interval of 7.5 s.

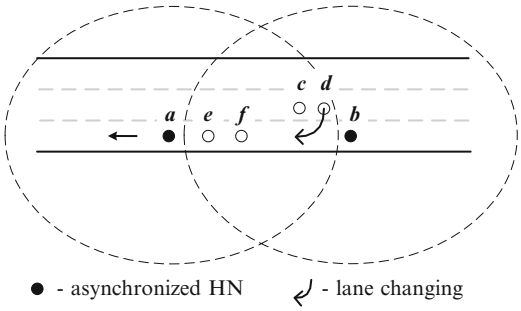

Fig. 8 This figure illustrates two examples of dangerous situations caused by synchronized HNs. Vehicles "a" and "b" are synchronized HNs. In the picture, vehicle "d" is changing lanes assuming it is safe to do so. Because "b is synchronized with "a," the driver of "d" is not informed about "b." Another case is a forward collision situation, where vehicle "a" is slowing down but "e" and "f" are not aware of this

safety since the vehicles can have relatively high speeds. Therefore, such traffic conditions should have even stricter requirements on the communication. In the following sections, we look into three broadcasting schemes that can alleviate the impact of the collision problems. The key idea of these schemes is to break the synchronization between vehicles as much as possible to prevent systematic message loss.

4.1 Scheme-1: Elastic Scheme

The first approach is what we call the elastic scheme in which the initial phase of broadcasting is changed at a regular basis. In this scheme, the message activation time is defined as follows:

$$a_i^{(k)} \stackrel{\text{def}}{=} \begin{cases} \phi_i & \text{if } k = 0 \\ a_i^{(k-1)} + T_i & \text{if } k > 0 \wedge (k + \phi_{ei}) \bmod er_i \neq 0 \\ a_i^{(k-1)} + r(2T_i) & \text{if } k > 0 \wedge (k + \phi_{ei}) \bmod er_i = 0 \end{cases} \tag{23}$$

where er_i is the elastic rate that defines how often the phase should be changed and r is a function that returns a random value within the given interval. This value defines how much the phase should be changed. ϕ_{ei} is a phase for starting elasticity and it is given as $\phi_{ei} = \lfloor r(er_i) \rfloor$. To keep the expected number of generated messages the same as the strict periodic scheme, $2T_i$ is selected as the interval. The worst case delay between two messages is $2T_i$.

Figures 9, 10, 11, and 12 show the results of this scheme in which we use the same er for all vehicles. From these graphs, we can make several interesting observations.

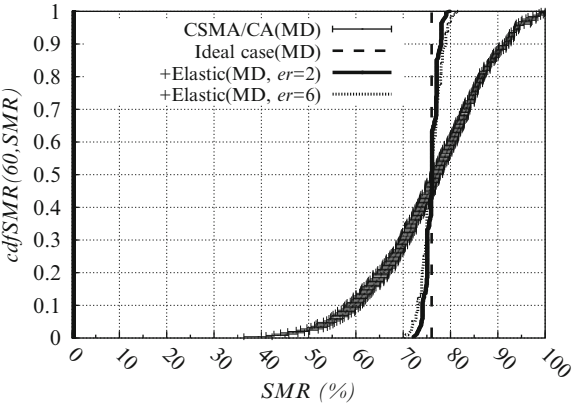

Fig. 9 The elastic scheme improves the fairness drastically, when *VD* is approximately 50. The result shown is an average of ten simulations with a confidence interval of 99 %. For the elastic scheme, the confidence interval is ± 0.3

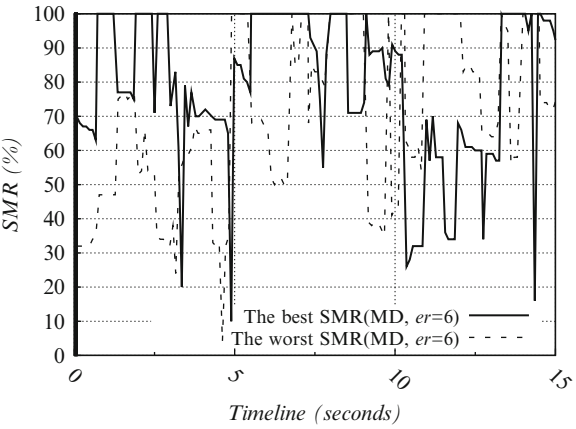

Fig. 10 *SMR* fluctuations over time of two individual vehicles that experience the best and the worst *SMR*, respectively, when *VD* is approximately 50. Compared to the pure CSMA/CA case, it is now hard to see any difference between the best and the worst cases. The graph shows the results of an arbitrary simulation

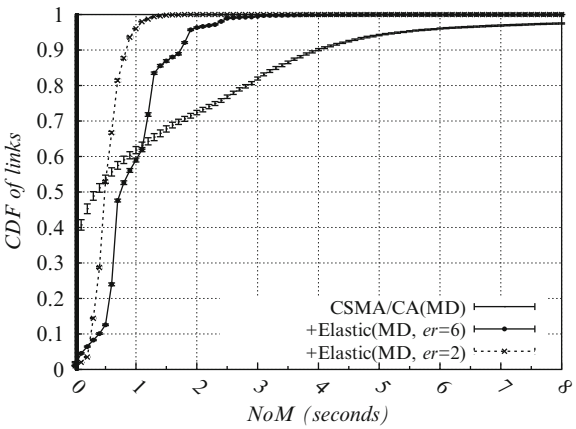

Fig. 11 The elastic scheme improves the *NoM* significantly, when *VD* is approximately 50. The result shown is an average of ten simulations with a confidence interval of 99 %

First, it is clearly seen that the more often the phase is changed, the better the elastic scheme improves the fairness and the delay characteristics. Particularly, the fairness is improved drastically even at the higher value of er. The reason for

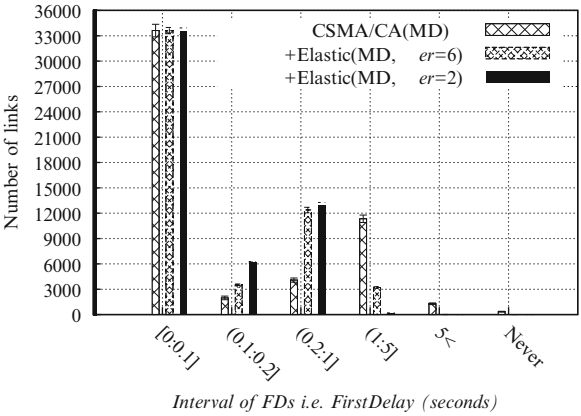

Fig. 12 The elastic scheme improves *FD* significantly, when *VD* is approximately 50. In case of "er=6," the number of cases for "5<" and "Never" are 15±7 and 0, respectively. In case of "er=2," the number of links for "5<" and "Never" are both 0. The result shown is an average of ten simulations with a confidence interval of 99 %

this result is the frequent change of phasing in the elastic scheme which affects the channel condition of the vehicle. Under the frequent change of the channel condition, the lifetime of a synchronized period of the vehicles (also a period of favorable channel condition of the vehicle) becomes shorter, i.e., highly likely to be at most the er period. Figures 10 and 11 reflect the effect of the short living synchronization when er is 6; we see somewhat discrete and step-like effects. As a result, each vehicle experiences more or less the same fluctuating channel conditions in the long run. From Figs. 11 and 12, though the EJ scheme shows a dramatic improvement on the fairness even at the higher value of er, it indeed needs the lower value of er to improve the delay significantly.

Second, in Fig. 9, we can see that the elastic scheme does not affect SMR of the entire network. It only affects SMRs of individual vehicles. For example, in the case of pure CSMA/CA, roughly half of the vehicles shows SMRs between 75–100 %, while the other half shows SMRs between 40–75 %. But, in the case of elastic scheme, this is completely changed and all vehicles show more or less the same SMRs that is closer to SMR of the entire network. This is also seen in Fig. 10 where the difference between the best and the worst cases are now hardly distinguishable.

4.2 Scheme-2: Jitter Scheme

The second approach is what we call the jitter scheme in which the activation time is defined as

$$a_i^{(k)} = \phi_i + kT_i + AJ_i - r(2AJ_i), \tag{24}$$

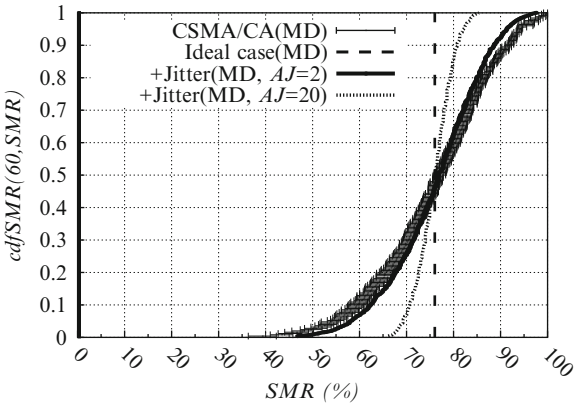

Fig. 13 The jitter scheme improves the fairness, when VD is approximately 50. However, it shows that a small jitter does not help much for improving the fairness. The result shown is an average of ten simulations with 99 % confidence interval. For the jitter scheme, the confidence interval is ± 1.0

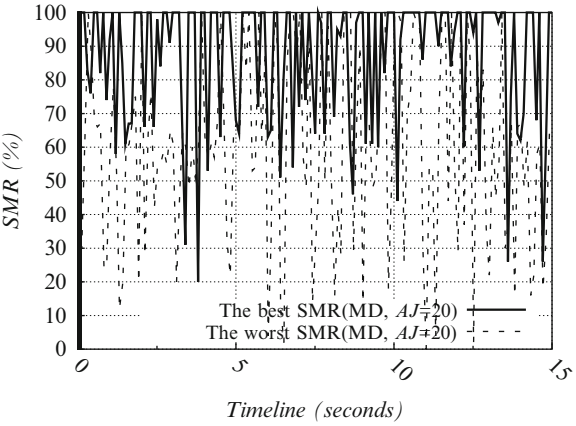

Fig. 14 *SMR* fluctuations over time of two individual vehicles that experience the best and the worst *SMR*, respectively, when VD is approximately 50. Compared to the elastic scheme, it looks more volatile. The graph shows the results of an arbitrary simulation

where AJ_i is an activation jitter and we give an indication of AJ in terms of the number of message transmit times (i.e., $AJ = n \leftrightarrow AJ = nT_d$). The worst delays between messages of this scheme, therefore, is equal to $T_i + 2AJ_i$. Note that successive message activation times should be increasing. If due to a large negative jitter, possibly in combination with a small random period, a next message activation time would be "earlier" or too close to the previous message, simply a very short interval between messages activations is used.

Again, we can make a number of observations. First, similar as the elastic scheme, the jitter scheme improves the *fairness* and the *delay* characteristics as shown in Figs. 13, 14, 15, and 16. We chose the same AJ for all vehicles. The bigger AJ is chosen, the better the jitter scheme works. Note that a small jitter size does not show much improvement. Compared to the elastic scheme, the jitter scheme needs a bigger jitter size to improve the fairness though a small jitter size already works pretty well on the delay characteristics. This indeed makes sense, because, in the jitter scheme, the channel condition of a vehicle does not change completely compared to the elastic scheme. Let's say there are two vehicles synchronized with

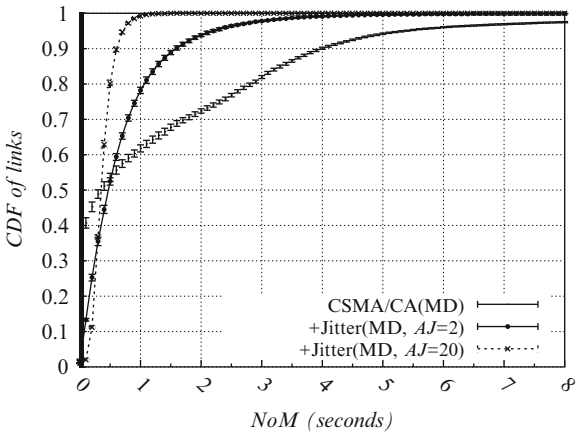

Fig. 15 The jitter scheme improves the *NoM* significantly, when *VD* is approximately 50. The jitter scheme performs better than the elastic scheme for this metric. In case of *AJ*=20, 60 % of the links have less than 0.5 s of NoM. The result shown is an average of ten simulations with a confidence interval of 99 %

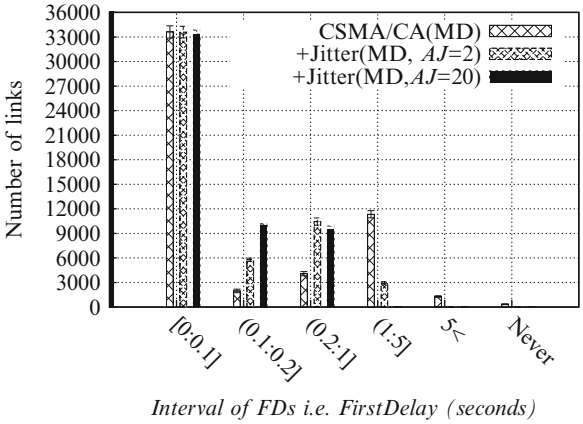

Fig. 16 The jitter scheme improves the *FD* significantly, when *VD* is approximately 50. In case of $AJ = 2$, the number of cases for "5<" and "Never" are 31 ± 6 and 29 ± 9, respectively. In case of $AJ = 20$, the number of cases for "5<" and "Never" are both 0. It shows that the jitter scheme performs better than the elastic scheme. In case of $AJ = 20$, the number of the links in an interval of (0.2:1] is much lower. The result shown is an average of ten simulations with a confidence interval of 99 %

each other causing message collisions on their receivers. For the elastic scheme, we showed that the lifetime of such synchronization becomes relatively short. But, in the jitter scheme, the two vehicles would remain synchronized during their entire encounter interval. The jitter only sometimes helps to prevent the message collisions happening. In addition, we can say that the jitter scheme works better than the elastic scheme on the delay characteristics. Particularly, from Fig. 16, we learn that the number of links on a 0.2–1 s interval is much lower than the elastic scheme result.

4.3 Scheme-3: Elastic + Jitter (EJ) Scheme

In addition to the previous two schemes, we also look into a third approach which is a combination of the elastic and the jitter schemes. We call this scheme the EJ scheme and it is defined as

$$a_i^{(k)} \stackrel{\text{def}}{=} \begin{cases} \phi_i & \text{if } k = 0 \\ a_i^{(k-1)} + T_i + AJ_i - r(2AJ_i) & \text{if } k > 0,\ (k + \phi_e) \bmod er_i \neq 0 \\ a_i^{(k-1)} + r(2T_i) + AJ_i - r(2AJ_i) & \text{if } k > 0,\ k \bmod er_i = 0 \end{cases} \tag{25}$$

As hoped, this solution outperforms both previous schemes as shown in Figs. 17, 18, 19, and 20. This third solution features the advantages of both schemes. Similar to the elastic scheme, the fairness is improved drastically by the EJ scheme. Similar to the jitter scheme, the EJ scheme improves the delay characteristics to a greater extent.

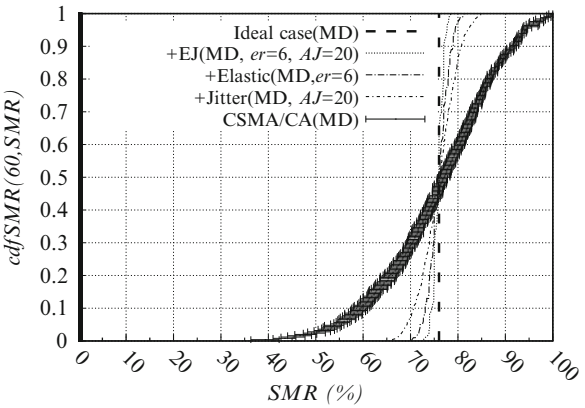

Fig. 17 The EJ scheme outperforms the jitter scheme and it is slightly better than the elastic scheme for improving the fairness. The result shown is an average of ten simulations with a confidence interval of 99 %. For the elastic scheme, the confidence interval is ±0.3

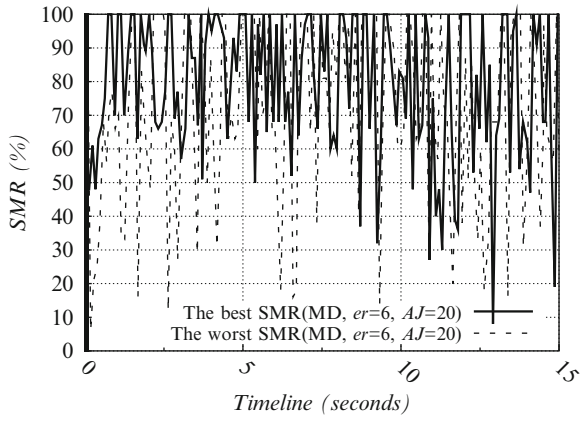

Fig. 18 *SMR* fluctuations over time of two individual vehicles that experience the best and the worst *SMR*, respectively, when VD is approximately 50. The graph shows the results of an arbitrary simulation

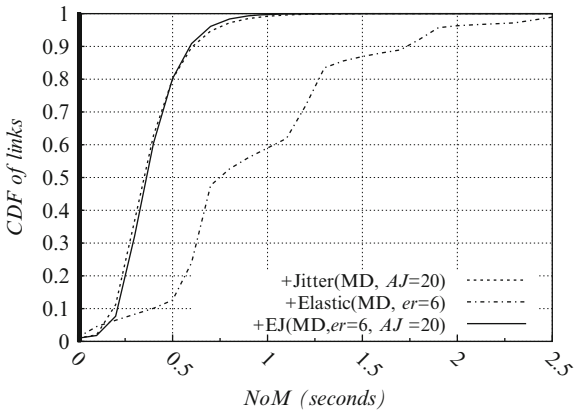

Fig. 19 The EJ scheme improves the delay characteristic by reducing *NoM* similar as the jitter scheme when VD is approximately 50. The result shown is an average of ten simulations with a confidence interval of 99 %

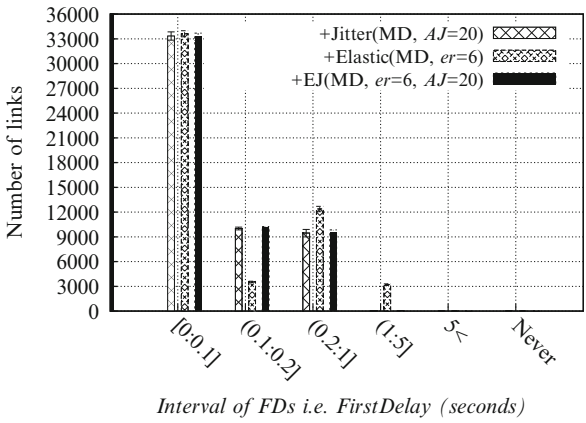

Fig. 20 The EJ scheme improves the delay characteristics similar as the jitter scheme. when VD is approximately 50. In case of EJ scheme, the number of links for "5<" and "Never" are both 0, respectively. The result shown is an average of ten simulations with a confidence interval of 99 %

5 Related Works

Performance studies of oPBC under CSMA/CA are carried out in many works. Here, we discuss the most relevant to our work. Computer simulation studies are done in [17, 22, 23], analytical models for analyzing performance of oPBC are proposed in [19, 20], and a real-world experiment is carried out in [4]. In [22], J. Yin et al. measured message throughput and latency under an unsaturated network condition. The throughput is defined as the long-run average percentage of single-hop neighbors who successfully receive a broadcast message (*SMR* of the entire network). The latency is defined as the long-run average time elapsing between sending a message at the source and successfully receiving $(f_i^{(k)} - s_i^{(k)} + \delta)$ that message at a single-hop neighbor. They conclude that the latency is reasonably good whereas the throughput is moderate and needs improvement. It is not clear from their work what caused the moderate throughput. In [17], K. Bilstrup et al. analyzed a highway scenario with periodic broadcast of beacon messages. They particularly

studied the "channel access time" ($s_i^{(k)} - a_i^{(k)}$) under saturated network conditions showing that a specific vehicle is forced to drop over 80 % of its beacon messages because no channel access was possible before the next message was generated. As a result, they confirmed that some vehicles become invisible to surrounding vehicles for up to 10 s. Notably, the analysis is done at vehicle level rather than entire network level. They did not, however, consider the reception of messages. We think that "channel access time" is not a sufficient metric since the successful channel access does not guarantee the successful message delivery due to the collision and environmental problems. Also, it is more relevant to consider unsaturated conditions.[4] In [23], T. Kuge et al. point out the impact of the HN problem on successful message reception in oPBC under unsaturated network conditions. They used a metric that is defined as "the rate of the messages which are received correctly against all the messages that are generated during measurement time (*SMR* of the entire network)." They confirmed that the number of message collisions that are caused by the HN problem is higher than that are caused by the contention problem. Moreover, they showed that changing CW size helps to reduce the collisions caused by contention. But, it has no effect on the collisions caused by the HN problem. Instead, they showed that the collision rate can be reduced to up to one-third of the conventional CSMA scheme by using a spread spectrum (SS) scheme. They do not provide any analysis on delay characteristics.

Xiaomin Ma et al. propose an analytic model in [19] for analyzing performance of emergency message transmission as well as oPBC. In their model, they take the 802.11 backoff counting process, the HN problem, unsaturated network, fading channel, and mobility into account. They use two metrics: a message reception rate that is defined as the ratio of the number of messages successfully received by all vehicles within the range of a sending vehicle to the number of messages transmitted (*SMR* of the entire network), and a message transmission delay that is the average delay a message experiences between the time at which the message is generated and the time at which the message is successfully received ($f_i^{(k)} - a_i^{(k)} + \delta$). With their model, they confirmed that the message delivery delay (less than 2 ms) meets the requirement of the safety applications (500 ms). However, the obtained message reception rates fail to meet reliability requirements for the safety critical messaging. A. Vinel et al. propose an analytic model in [20] for analyzing performance of oPBC under both saturated and unsaturated network conditions by two metrics: beacon message successful reception probability (*SMR* of the entire network) and mean beacon message transmission delay that is the time from the moment the beacon was issued until it has been transmitted ($f_i^{(k)} - a_i^{(k)}$). With the model, they found that the delay requirements are met, but the probability of successful message reception is rather low in typical scenarios. It is not clear whether they considered the HN problem in their model.

[4]There are more vehicles in a communication range for a saturated network condition than for an unsaturated network condition. A saturated network condition is most likely a traffic jam situation, where vehicles have a much lower speed compared to a normal traffic situation. As a result, there is a lower likelihood of a big accident with serious casualties.

A very interesting real-world experiment is performed by F. Bai et al. in [4]. The work studied the environmental impact (e.g., fading, doppler, multi-path effect) on reliability of the 802.11p by an experimental setup that includes a fleet of only three vehicles equipped with the 802.11p communication system. With this setup, the probability of having the collision problems is almost 0. In that sense, our work is complementary to their work since we study the impact of the collision problems (i.e., the other main reason of message loss besides the environmental impacts) on the quality of oPBC. The work judges the reliability of 802.11p using distribution of consecutive message losses (similar to NoM; remember that NoM is the longest no message interval) together with the general metric "message delivery ratio." The message delivery ratio is calculated as a ratio of the number of data messages received at the receiver to total number of messages transmitted at the sender within some pre-defined time window ($SMR_{ij}(I)$). The work shows that the reliability of the 802.11p is adequate in a wide variety of traffic environments. Most importantly, they observed that the message losses do not occur systematically viz., there are almost no consecutive message losses between two certain vehicles even under the harsh freeway traffic environment.

All in all, we conclude the following: Two common metrics are widely used in these works. The first one is *SMR* of the entire network and the second one is an end-to-end delay (e.g., some define it as $f_i^{(k)} - s_i^{(k)} + \delta$ and some define it as $f_i^{(k)} - a_i^{(k)} + \delta$ or as $f_i^{(k)} - a_i^{(k)}$). These common metrics do not show or quantify the exact causes of a certain problem (e.g., the contention and HN problems). Namely, *SMR* of the entire network cannot show the fairness aspect and the end-to-end delay metric cannot show no message interval between two vehicles which is more relevant to vehicle safety applications that rely on oPBC. In addition, most of these works lack a formal model.

6 Concluding Remarks

In this final section, we summarize the main findings and achievements of this work. In addition, we shed some light on future research directions.

6.1 Main Contributions

We regard the following four results as the main contributions of the paper. Among these, the biggest contribution of the work are the repairing schemes for improving the quality of oPBC.

First, a novel timing model of oPBC in the context of the 802.11p MAC has been defined. This model gives the foundation for a simulation model and for defining new more appropriate metrics to judge the communication quality from

the perspective of safety applications. Among others, we introduce concepts of "neighborhood of a vehicle," i.e., set of other vehicles that could receive its messages and "link," i.e., an encounter interval that starts when a vehicle B enters the Communication Range (CR) of another vehicle A and ends when vehicle B leaves the CR of vehicle A.

Second, new metrics based on the above concepts have been defined for judging the quality of oPBC in three of the most important aspects: *SMR* for the *reliability*, cumulative distribution of *SMR* per vehicle for the *fairness*, No Message interval (*NoM*) (i.e., the longest interval in which no message is successfully delivered in a link), and First Delay (*FD*) (initial NoM) for the *delay* aspects, respectively.

Third, an evaluation of oPBC is performed according to the simulation model under a strict periodic broadcasting scheme, i.e., each vehicle broadcasts messages in a strictly periodic manner. The evaluation reveals that the HN problem is the main cause of various quality degradations especially when the network is unsaturated. Once the network is saturated, the contention problem already reduces the *SMR* dramatically. A detailed evaluation is conducted on an unsaturated network condition where the traffic condition is sparse, i.e., 85 vehicles/km on a highway with three lanes per direction. We selected such traffic condition because we assume that it is a typical highway condition. Besides, this condition is even more stringent in terms of vehicle safety since then vehicles have relatively high speeds. Such traffic condition, therefore, will have even stricter requirements on the communication quality, particularly, on the delay aspect. The evaluation results suggest that the HN problem causes unfair *SMR* distribution where the difference between the best and worst vehicles by their *SMR* is 65 % in a simulation. Moreover, it causes long-lasting consecutive message losses in a link of two vehicles. In some cases, a certain vehicle could not successfully deliver any of its messages to a particularly destination vehicle throughout an entire link interval. These quality issues are mainly due to synchronized HNs that can occur under the strict periodic scheme.

Fourth, we propose three simple but effective broadcasting schemes (i.e., elastic scheme, jitter scheme and a combination of these two schemes) to alleviate the impact of the HN problem. Though the three solutions do not affect the SMR (or reliability aspect) of the entire network, they do show significant improvements on the fairness and the delay aspects. Particularly, the combined scheme features the advantages of the other two schemes for improving the communication quality in terms of the fairness and the delay aspects. These solutions are fully compatible with the 802.11p, i.e., they are application-level solutions and can therefore be easily introduced in practice.

6.2 Future Directions

We are projecting the following future directions as our next steps:

- investigation of further broadcasting schemes (e.g., location and direction based scheduling) particularly for avoiding the collision problems;
- investigation of further combinations of different broadcasting schemes and their compatibility to each other (e.g., location based + jitter + predictive coding scheme);
- investigation of more adaptive schemes for the network congestion problem (e.g., dynamic message size or dynamic channel allocation i.e., switching between control and service channel);
- testing the proposed solutions with realistic traffic simulations and different traffic conditions (e.g., real traces of highway or urban roads);
- generalization of the solutions to other domains (e.g., Mobile Ad-hoc Network or Wireless Sensor Networks).

Acknowledgements We would like to thank the Strategic Platform for Intelligent Traffic Systems (SPITS) project for funding this work (spits-project.com). SPITS is a Dutch project, tasked with creating ITS concepts that can improve mobility and safety.

Appendices

A CSMA/CA

This section explains how CSMA/CA operates in periodic broadcast mode. Figure 21 shows a basic flowchart and Fig. 22 gives a corresponding state machine diagram. According to CSMA/CA, when a station becomes ready to broadcast (Ready To Transmit (RTT)), the station must first check the channel for a duration of AIFS. If the channel has been idle for longer than AIFS, the station starts its transmission immediately. If the channel is busy or becomes busy during AIFS, the station must wait for the channel to become idle. 802.11 refers to this wait as *Access Deferral*. If access is deferred, the station first waits for the channel to become idle for AIFS again. If the channel is idle, the station must perform Bf procedure by starting a Bf timer which is set to a random number drawn from an interval of {0,1,. . .,CW}. The timer has the granularity of a slot time and is decremented every time when the channel is sensed to be idle for a slot time. The timer is stopped in case the channel becomes busy and the decrementing process is resumed when the channel becomes idle again (i.e., idle for a duration of AIFS). The station is allowed to transmit its message when the Bf timer reaches zero. Depending on the channel condition, a station may experience multiple AIFS plus access deferrals. Note that the Bf counting is at most done once. If a new message arrives from the upper layer, then the current message must be dropped and the new message transmission will start.

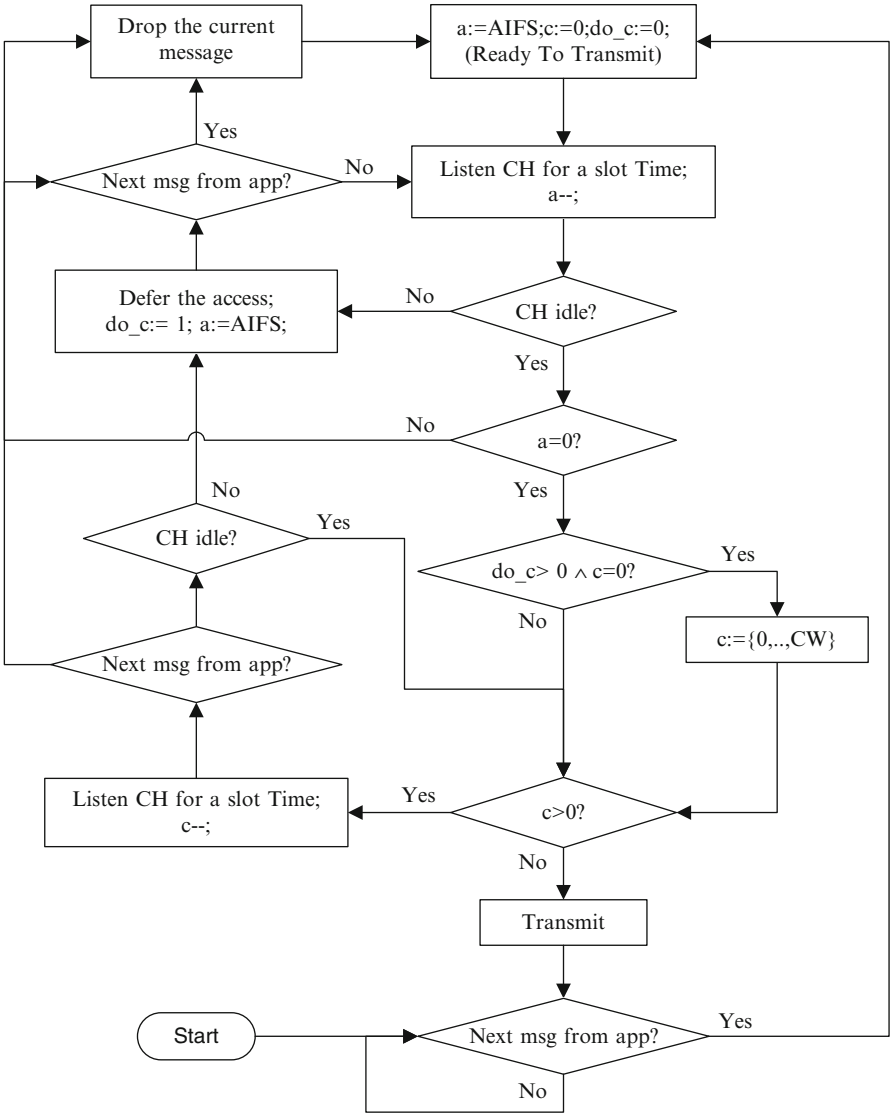

Fig. 21 CSMA/CA procedure in periodic broadcast mode

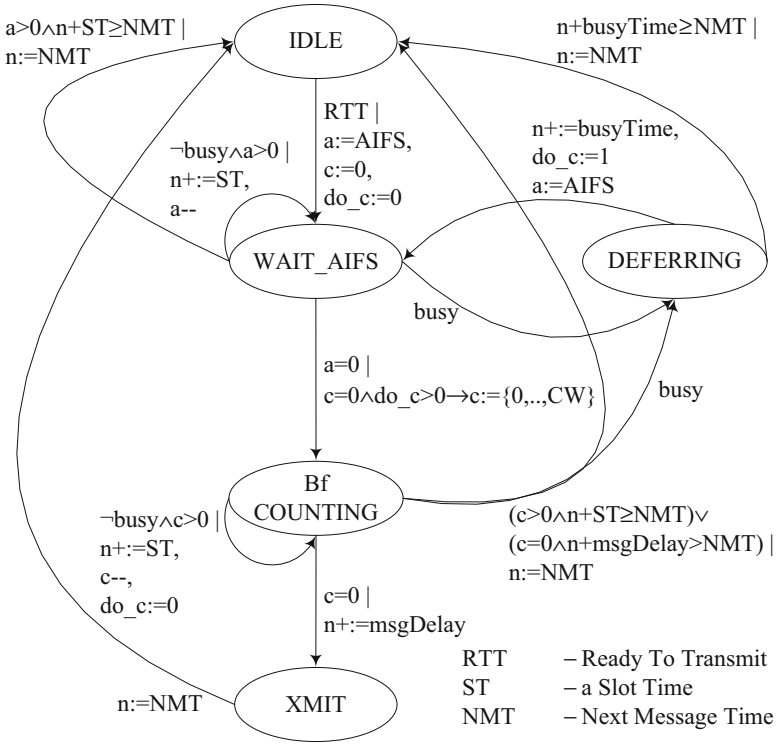

Fig. 22 CSMA/CA state machine

As shown in the state machine diagram in Fig. 22, each station is in one of the five states: IDLE, WAIT_AIFS, DEFERRING, Bf COUNTING, and XMIT. The station is in the IDLE state when it is neither in transmission nor RTT. In this diagram, we use several variables and constants to show the state transition conditions and timing changes.

- The “n” holds the current time.
- The “a” holds the current value of the AIFS counting.
- The “c” holds the current value of the Bf counting.
- The “do_c” is a boolean variable to indicate whether the station should perform the Bf counting.
- The “busyTime” is the duration of the channel being busy.
- The “msgDelay” is the duration of one message transmission.
- The “Next Message Time (NMT)” is the time at which the station becomes ready to transmit its next message.
- The “busy” to indicate the busy channel.
- The “AIFS” is the number of slots for the AIFS waiting and it is given by the standard.

- The "CW" is the number of slots for the Bf and it is given by the standard.
- The "a Slot Time (ST)" is the duration of one slot time and it is given by the standard.

B Signal Reception Model

There are generally two methods used for physical layer (or PHY) modeling in network simulations, namely SINR threshold based and Bit Error Rate (BER) based [22]. Under the former method, the receiver accepts the message when the computed SINR value is above the SINR threshold for a particular modulation scheme. The method based on BER decides whether or not a message is received successfully based on the message length and bit error rate deduced by the pre-computed BER versus SINR curve for every modulation scheme at the receiver.

In our case, the signal reception model is taken from the NS-2 implementation of 802.11p [18], where the signal reception decision is based on SINR ratio. In this model, three basic signal threshold concepts play a role. The first one is a Power Sense Threshold, *PsTh*.[5] If any receiving signal is equal to or greater than *PsTh*, the PHY will try to decode the signal. In addition, any signal equal to or greater than *PsTh* is considered to be strong enough to interfere with any other signal. Therefore, such signal is added into the cumulative signal level of the receiver. The second threshold is the SINR threshold (or receiving threshold), *SrTh*. To receive a message successfully, the preamble must be received successfully. While a station is not transmitting nor receiving any signal, namely the PHY is constantly searching for a preamble and if a new signal arrives with a signal strength that is equal to or greater than *SrTh* with respect to all other interfering signals plus noise, then the station will start receiving the signal as the preamble. If this SINR ratio holds for the entire duration of the preamble length, the PHY can finish the preamble reception successfully. Once the preamble is received successfully, the PHY will inform the MAC layer that the PHY is receiving a message and will continue until the complete payload has been received regardless of whether the signal level is greater or less than *SrTh*, unless the frame capturing feature is enabled. Note, the preamble includes information about payload (the length of payload, modulation scheme etc.). In addition, if the receiving signal strength becomes less than *SrTh* during the payload reception, PHY layer puts an error in the payload such that when the MAC layer checks the CRC of the received message it will not send the message to the application layer.

The third threshold is a carrier sense threshold, *CsTh*. If the cumulative power level of the receiver is equal to or greater than *CsTh*, then the PHY layer will inform

[5] Some concepts are rather misleading. In the document provided there are an RX threshold and a carrier sense threshold. But in the implementation there are an SINR threshold, a power sense threshold, and a carrier sense threshold.

the MAC layer that the channel is busy. From this, we can see that the MAC layer is informed about the channel status in two different ways. First, no matter what if the cumulative power is equal to or greater than *CsTh*, then the MAC is informed the channel is busy. Second, if the PHY layer successfully receives a preamble viz., start receiving a payload, the MAC layer is informed the channel is busy.

References

1. Final report, Vehicle safety communications project, Technical Report DOT HS 810 591, 2006
2. The CAR 2 CAR Communication Consortium. http://www.car-to-car.org/. 2011
3. Internet ITS consortium. http://www.internetits.org/. 2011.
4. Bai F, Krishnan H (2006) Reliability analysis of DSRC wireless communication for vehicle safety applications. In: Intelligent transportation systems conference, 2006. ITSC '06. IEEE, pp 355–362
5. Robinson C, Caveney D, Caminiti L, Baliga G, Laberteaux K, Kumar P (2007) Efficient message composition and coding for cooperative vehicular safety applications. IEEE Trans Veh Technol 56(6):3244–3255
6. Mittag J, Schmidt-Eisenlohr F, Killat M, Torrent-Moreno M, Hartenstein H (2010) MAC layer and scalability aspects of vehicular communication networks. In: VANET: vehicular applications and inter-networking technologies, pp 219–269
7. Torrent-Moreno M, Mittag J, Santi P, Hartenstein H (2009) Vehicle-to-vehicle communication: fair transmit power control for safety-critical information. IEEE Trans Veh Technol 58(7):3684–3703
8. IEEE draft 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 : wireless access in vehicular environments, IEEE unapproved draft Std P802.11p/D11.0, 2010
9. IEEE trial-use standard for wireless access in vehicular environments (WAVE) - resource manager, IEEE Std 1609.1, pp 1–63, 2006
10. IEEE trial-use standard for wireless access in vehicular environments - security services for applications and management messages, IEEE Std 1609.2, pp 1–105, 2006
11. IEEE trial-use standard for wireless access in vehicular environments (WAVE) - networking services, IEEE Std 1609.3, pp 1–87, 2007
12. IEEE trial-use standard for wireless access in vehicular environments (WAVE) - multi-channel operation, IEEE Std 1609.4, pp 1–74, 2006
13. Dedicated short range communications (DSRC) message set dictionary, SAE Standard J2735, 2009
14. 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, IEEE Std 802.11-1997, pp 1–445, 1997
15. Choffnes DR, Bustamante FE (2005) An integrated mobility and traffic model for vehicular wireless networks. In: Proceedings of the 2^{nd} ACM international workshop on vehicular ad hoc networks, pp 69–78
16. Baumann R, Heimlicher S, May M. (2007) Towards realistic mobility models for vehicular Ad-hoc networks. In: 2007 Mobile networking for vehicular environments, pp 73–78
17. Bilstrup K, Uhlemann E, Strom E, Bilstrup U (2008)Evaluation of the IEEE 802.11p MAC method for vehicle-to-vehicle communication. In: Vehicular technology conference, 2008. VTC 2008-Fall. IEEE 68^{th}, pp 1–5

18. Chen Q, Schmidt-Eisenlohr F, Jiang D, Torrent-Moreno M, Delgrossi L, Hartenstein H (2007) Overhaul of IEEE 802.11 modeling and simulation in NS-2. In: Proceedings of the 10^{th} ACM symposium on modeling, analysis, and simulation of wireless and mobile systems, pp 159–168
19. Ma X, Chen X (2007) Delay and broadcast reception rates of highway safety applications in vehicular ad hoc networks. 2007 Mobile networking for vehicular environments, pp 85–90
20. Vinel A, Staehle D, Turlikov A (2009) Study of beaconing for car-to-car communication in vehicular ad-hoc networks. In: Communications workshops, 2009. ICC workshops 2009. IEEE international conference on, pp 1–5
21. Yang L, Guo J, Wu Y (2008) Channel adaptive one hop broadcasting for VANETs. In: Intelligent transportation systems, 2008. ITSC 2008. 11^{th} international IEEE conference on, pp 369–374
22. Yin J, ElBatt T, Yeung G, Ryu B, Habermas S, Krishnan H, Talty T (2004) Performance evaluation of safety applications over DSRC vehicular ad hoc networks. In: VANET '04: Proceedings of the 1^{st} ACM international workshop on vehicular ad hoc networks, pp 1–9
23. Kuge T, Ohno K, Itami M (2009) An analysis of performance degradation caused by hidden terminal and its improvement in inter-vehicle communication. In: Intelligent transport systems telecommunications,(ITST), 2009 9^{th} international conference on, pp 482–485

A Survey of Security and Privacy in Connected Vehicles

Lotfi Ben Othmane, Harold Weffers, Mohd Murtadha Mohamad, and Marko Wolf

Abstract Electronic control units (ECUs) of a vehicle control the behavior of its devices—e.g., break and engine. They communicate through the in-vehicle network. Vehicles communicate with other vehicles and road side units (RSUs) through vehicular ad-hoc networks (VANets), with personal devices through wireless personal area networks (WPANs), and with service center systems through cellular networks. A vehicle that uses an external network, in addition to the in-vehicle network, is called connected vehicle.

A connected vehicle could benefit from smart mobility applications: applications that use information generated by vehicles, e.g., cooperative adaptive cruise control. However, connecting in-vehicle network, VANet, WPAN, and cellular network increases the count and complexity of threats to vehicles, which makes developing security and privacy solutions for connected vehicles more challenging.

In this work we provide a taxonomy for security and privacy aspects of connected vehicle. The aspects are: security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. We use the taxonomy to classify the main threats to connected vehicles, and existing solutions that address the threats. We also report about the (only) approach for verifying security and privacy architecture of connected vehicle that we found in the literature. The taxonomy and survey could be used by security architects to develop security solutions for smart mobility applications.

L. Ben Othmane (✉) • H. Weffers
Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, The Netherlands
e-mail: l.ben.othmane@tue.nl; h.t.g.weffers@tue.nl

M.M. Mohamad
Faculty of Computer Science and Information System, Universiti Teknologi Malaysia, Johor, Malaysia
e-mail: murtadha@utm.my

M. Wolf
ESCRYPT GmbH–Embedded Security, Munich, Germany
e-mail: marko.wolf@escrypt.com

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_10

1 Introduction

Each vehicle uses a set of sensors and electronic control units (ECUs) to collect data about the vehicle's behavior and environment, and to control the functionalities of the vehicle. ECUs (of a vehicle) collaborate by exchanging messages; they compose an *in-vehicle network* (a.k.a. on-Board network). Figure 1 depicts an example of architecture of in-vehicle network.

Vehicles, in a road, exchange messages with neighboring (close by) vehicles and with road side units (RSUs); they communicate directly—without intermediate node(s), or indirectly—through intermediate node(s). Each vehicle communicates with the neighboring RSUs to inform them about itself (the information may include location, speed, and heading) and to get traffic conditions of the road. Vehicles and RSUs compose *vehicular ad-hoc networks (VANets)* [1] (a.k.a. inter-vehicle network). Members of a VANet exchange information, for example, to have a shared knowledge about the traffic.

A wireless personal area network (WPAN) is composed of personal devices[1] that communicate through short range (from few centimeters to 100 m) wireless technologies, such as Bluetooth, near field communication (NFC), and Infrared (IR). WPAN gateway, of a vehicle, enables exchange of messages between personal devices (e.g., personal digital assistances (PDAs) and iPods) and in-vehicle ECUs. For example, a driver controls the lights, windshield wipers, air flow, and heat of his/her vehicle through a Bluetooth-enabled headset [2], or even starts remotely its engine and unlocks its doors using his/her PDA.

Vehicles, equipped with mobile devices, exchange messages with service centers (SCs) through cellular network (a.k.a. mobile network); they provide data about their locations, behavior, and environment. An *SC* is a remote office that receives

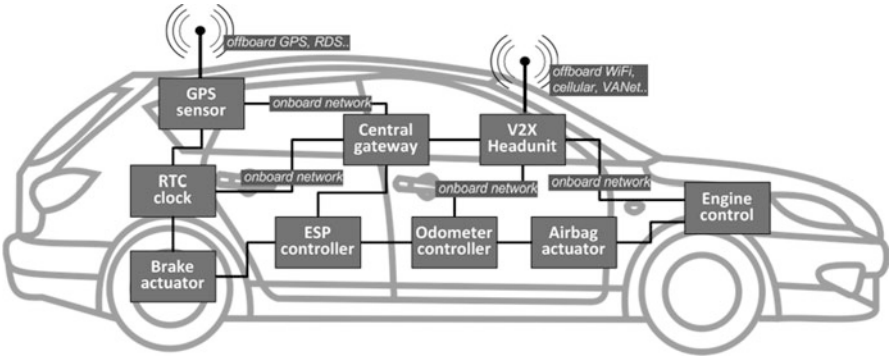

Fig. 1 Example of architecture of in-vehicle network

[1] Infotainment devices are in most cases connected to the in-vehicle network; they are not members of the WPAN.

and sends data to vehicles or RSUs in order to assist and provide services to drivers, vehicle owners, and the public community. Example of SCs are fleet management systems. The *cellular network* enables communications of devices that have wireless communication capabilities (e.g., mobile phones) with mobile and land phones (cf. [3]).

Several applications, such as cooperative adaptive cruise control, remote firmware update, e-call, and remote diagnostic of vehicles use the integration of the four networks of a connected vehicle (Sect. 2.2 describes several applications). These applications are called *smart mobility applications*: applications that use data collected from vehicles to improve the use of vehicles and the safety and comfort of drivers, and to rationalize the use of public infrastructure.

A *connected vehicle* is a vehicle whose ECUs communicate through an in-vehicle network, and it communicates with neighboring vehicles and RSUs through VANets, with personal devices through WPAN, and with service providers (SPs) and SCs through cellular network.[2] A connected vehicle is equipped with an on-board unit (OBU): a device for communicating a vehicle with other entities through VANets, WPAN, cellular network, and routing messages to/from ECUs of the vehicle.[3]

Connected vehicles enable the use of intelligent transportation systems (ITSs). ITSs support the efficient and safely use of transport infrastructure and means (cars, trains, planes, ships) to facilitate the mobility of human and goods through the use of information and communication technologies [4].[4]

An attacker who aims to change the behavior of a unconnected vehicle needs to be able to physically access to its devices, or access to its communication bus, or be able to install malicious code in a device connected to the in-vehicle network. Vehicle manufacturers have overlooked security of vehicles [5]. There is a common assumption, by the vehicle manufacturers, that it is highly unlikely that potential attackers could acquire one of these capabilities. The assumption is not valid anymore because vehicles become connected to other vehicles, to personal devices, and to SCs.

Connected vehicles offer more capabilities for the attacker to compose complex attacks. An attacker could connect to the in-vehicle network of a target connected vehicle without the need for any of the capabilities listed above. For example, an attacker who has remote access to ECUs of a target vehicle could inject in the in-vehicle network messages to increase the speed of the vehicle.

[2] We do not enumerate all (possible) communication mediums—e.g., satellite communication—for vehicles. Instead, we discuss the networks that are commonly used and reported (in the literature) to impact the security and privacy of vehicles.

[3] An OBU in general—e.g., OBU dedicated to VANets—does not act as a gateway to the in-vehicle network.

[4] Terms smart mobility and ITS are often considered similar in the literature. In this work, we use vehicle to refer to car and truck when we discuss smart mobility and all transport means when we discuss ITS.

Threats to connected vehicles have financial, privacy, and safety impacts. The description of the impact follows:

- *Financial*: A company could collect the location data of the vehicles of its competitors and extract sensitive information, such as the address of their customers and the performance of their fleets. The information gives the company business advantages. Other examples include: stealing pay content—e.g., infotainment, and location-based services; misuse pay-as-you-drive systems—e.g., car insurance, car taxes, car rental, and warranty or maintenance plans; and Intellectual Property (IP) theft of firmware.
- *Privacy*: A company could collect the location and in-vehicle data of drivers and use them to provide services for the drivers, and the public community. Disclosing the information, which is private information, to parties without the consent of their owners is a privacy violation.[5]
- *Safety*: A malicious attacker could remotely control the behavior of a target vehicle and inject messages in its in-vehicle network to lock its break system and make it crash with another vehicle or an obstacle. An easier attack is to misinform the driver.

In the last decade, several threat analysis, security solutions, and security and privacy architectures addressing a specific network type (i.e., either in-vehicle network, or VANet, or WPAN, or cellular network) have been proposed in the literature. In this chapter, we propose a taxonomy of security and privacy aspects for connected vehicles. The aspects are: security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. We survey the threats to connected vehicles and classify them using the taxonomy. We survey also existing solutions that address the threats[6] and classify them using the taxonomy. We also report on the only solution for verifying security and privacy of connected vehicles that we found in the literature. The taxonomy and survey could be used by security architects to develop security solutions for smart mobility applications.

This chapter is organized as follows. Section 2 describes how connected vehicles collaborate in an ITS and describes a set of smart mobility applications. Section 3 proposes and describes a taxonomy for security and privacy aspects of connected vehicles. Section 4 surveys the security and privacy threats to connected vehicles and classifies them based on our taxonomy. Section 5 surveys the security and privacy solutions for connected vehicles and classifies them based on our taxonomy. Section 6 reports on verifying security and privacy solutions for connected vehicles. Section 7 concludes the chapter.

[5]In the USA and Europe privacy violation is prohibited by the law.

[6]We survey security and privacy threats and solutions for *only* connected vehicles, which may not include all security and privacy threats and solutions for smart mobility applications or ITS.

2 Overview of the System Architecture and Its Applications

In this section we describe an example of ITS and show how connected vehicles benefit from the system. We also describe a set of smart mobility applications that could be used by connected vehicles.

2.1 Overview of ITS and Smart Mobility Applications

Figure 2 shows an ITS composed of two connected cars, a back office (BO)—e.g., a fleet management system, an SP—e.g., e-call service, a personal device, a mobile device, and an RSU. Each car is equipped with ECUs that collaborate to regulate the functionality of the vehicle by exchanging messages through the in-vehicle network. For example, the vehicle dynamics control system (VDCS) of a vehicle assists the driver in over-steering, under-steering and roll-over situations: it uses the steering wheel angle and other relevant sensors data available in the in-vehicle network to decide when to apply the brakes of the individual wheels or adjust the engine torque [6].

Both cars communicate through VANet. Nodes of a VANet communicate using wireless access for vehicular environments (WAVE) [7]—a standard for inter-vehicle communication. They exchange information in order to, for example, avoid

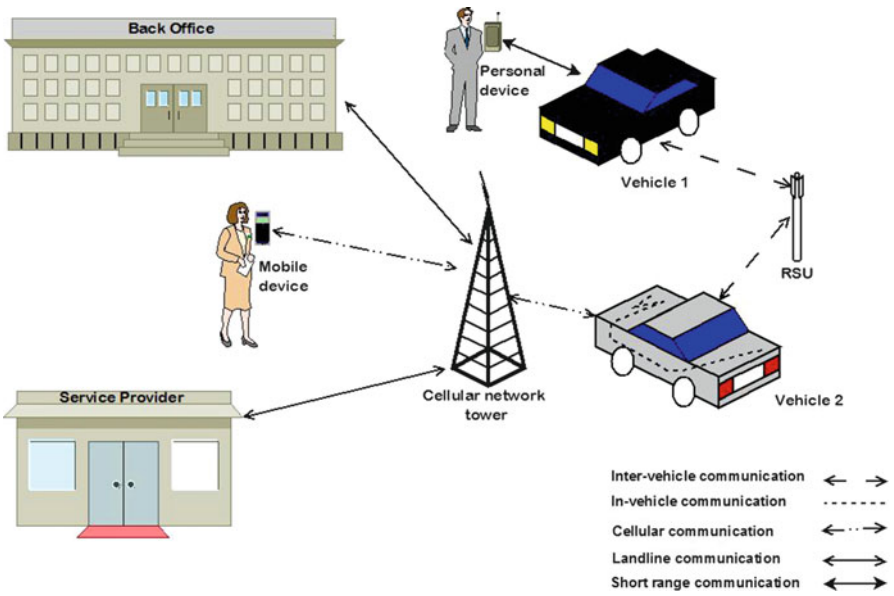

Fig. 2 Example of an intelligent transportation system (ITS)

collision. A vehicle communicates with the neighboring RSUs using WAVE to provide data about its own activities (e.g., location, speed, and heading) and get traffic conditions related to its location.

Vehicle 1 communicates with a personal device through WPAN; the driver uses his/her PDA to remotely start his/her vehicle. Vehicle 2 communicates with BO and SP through cellular network. For instance, Vehicle 2 sends data about its location to a fleet owner system, a BO, and communicates with an e-call provider [8], an SP, in case of an emergency. A second example, the car owner of Vehicle 2 uses a mobile device to inquire about the location of his/her car.

2.2 Smart Mobility Applications

Data collected from vehicles is useful for individuals, businesses, and public organizations. In the following we describe examples of smart mobility applications.

Connected vehicles could exchange messages with neighboring vehicles through VANets and adjust their speed to keep safe distance with neighboring vehicles. A *Cooperative adaptive cruise control* [9] application—of a vehicle—exchanges messages with the neighboring vehicles and sends appropriate messages to the ECUs of the vehicle to adjust its speed. Cooperative adaptive cruise control application is different from adaptive cruise control application since the latter is limited to the use of radar installed on the vehicle to detect predecessor vehicles, obstacles, and other moving objects. It computes the safe distances between the vehicle and its predecessors and adjusts the speed of the vehicle accordingly. This application works only when predecessor vehicles are in the line of sight of the radar.

Autonomous vehicles are self-driving vehicles—i.e., robots. They sense their environment and navigate by their own. A human can choose the destination, but may not perform the driving. There are currently several prototypes of autonomous vehicles. For instance, Google [10] reported recently that it is testing a prototype of autonomous vehicle in the traffic—e.g., to drive a blind to the destination she/he chooses.

Connected vehicles use VANets to share information with their neighboring vehicles, such as their location and speed. A *Cooperative collision avoidance* [11] application identifies potential collision situations and either informs the driver or acts automatically to avoid the collision. A vehicle involved in a collision, could also send notifications to other vehicles, which otherwise would drive into the crashed vehicle. This helps to avoid cascade accidents (i.e., an accident that involves many vehicles).

Companies who own fleet of vehicles use *fleet management* applications to track the locations and activities of their vehicles. Fleet managers get actual real-time information about their vehicles, including their locations, fuel levels, and speeds. The information is used for scheduling the business activities, and helps to quickly respond to events, such as accidents.

Table 1 Classification of the smart mobility applications into networks they use

Smart mobility application	In-vehicle network	VANet	Cellular network	Personal area network
Cooperative adaptive cruise control	x	x		
Autonomous vehicles	x	x	x	
Cooperative collision avoidance	x	x		
Fleet management	x		x	
Remote vehicle control	x		x	x
Traffic management		x	x	

Owners (or drivers) of vehicles equipped with personal devices, such as PDAs and mobile phones, could use *remote vehicle control* applications to, for example, start/stop the vehicle engine, lock/unlock the doors, and monitor remotely the speed, mileage, and location of their vehicles.

Currently, there is a growing traffic congestion problem in the main cities in the world. Connected vehicles could send data about their locations and speeds to RSUs through VANets. RSUs could collect the information and use *Traffic management* applications to aggregate them to generate traffic conditions data which it disseminates to neighboring vehicles. Traffic management applications improve the traffic flow on a road by reducing congestion problem and travel time [11].

Table 1 shows for each smart mobility application, that we describe above, the networks it uses. A connected vehicle can benefit from a specific application only if it can communicate with other parties through the required network(s). (Recall that each vehicle has an in-vehicle network.) As a final note, we refer the interested readers on smart mobility applications to Kargiannis et al. [12].

Note that ITS and smart mobility applications have broader scope than connected vehicles. Figure 2 shows an ITS, which includes connected vehicles and other entities as described above. Connected vehicle is limited to one entity: a single vehicle. This chapter is limited to connected vehicle.

3 Taxonomy for Security and Privacy Aspects of Connected Vehicles

We use a simple taxonomy of security and privacy aspects of connected vehicles, summarized in Fig. 3, which uses six classes: security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. The description of the aspects follows.

- *Security of communication links*: They are the medium for transmitting and receiving data between two or more entities. There are four types of

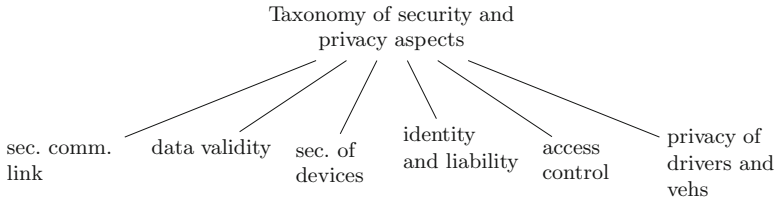

Fig. 3 Taxonomy of security and privacy aspects of connected vehicles (We use comm. for communication or communicating, vehs. for vehicles, sec. for security, and btw. for between)

communications: (1) communication between ECUs, and between ECUs and OBUs through the in-vehicle network; (2) communication between vehicles, and between vehicles and RSUs through VANets; (3) communication of vehicles with personal devices through WPAN; and (4) communication between vehicles and SPs and SCs through cellular network.

- *Data validity*: Data are information generated, manipulated, transmitted, and received by OBUs (of vehicles), ECUs, RSUs, SPs, and SCs. Data include in-vehicle data, location data, and aggregated data. (Aggregated data are computed using aggregation function, such as average and sum, the aggregation functions use data collected by a vehicle from its neighbors.)
- *Security of devices*: Devices are electronic components (hardware) of vehicles—i.e., ECUs and OBUs. They control the behavior of vehicles. A device has two parts: (1) hardware: the ECU or OBU; and (2) firmware: programs that run on ECUs or OBUs.
- *Identity and liability*: It refers to binding an entity to a specific information or event. Identity refers to a characteristic of an entity, which distinguishes it from other entities. Liability refers to the ability to prove that a specific entity (vehicle, driver, and car owner)—or a set of entities—is responsible for a specific event or a set of events; that is, the entity cannot repudiate the responsibility for a specified event.
- *Access control*: It refers to enforcing rules for access or deny to certain functions or data for identified entities.[7]
- *Privacy of drivers and vehicles*: It is the right of an entity to be able to control when, how, to what extent, and for what purpose information about themselves is shared with others (cf. [13]), and to determine the degree to which the entity will interact with its environment (cf. [14]). Privacy of drivers refers to the right of the driver to control the access and use of his/her personal data. Privacy of vehicles refers to controlling access to the identities of communicating vehicles.

[7]The identity of an unidentified entity is anonymous. Anonymous entities have access to public resources, if any.

4 Taxonomy of Threats to Connected Vehicles

In the following we describe the assets of the attackers and the threats to connected vehicles.

4.1 Assets of the System

An *asset* is a system resource that shall be protected by a security policy or a security mechanism [14]. It is either a component of the system or data consumed or produced by the system. Table 2 describes the assets to connected vehicle.

A connected vehicle exchanges data with other vehicles, RSUs, personal devices, SPs, and SCs. The data types are: in-vehicle data, VANet data, ECU firmware, and OBU firmware. Table 3 provides a classification of these data types into networks—i.e., in-vehicle network, VANet, WPAN, and cellular network—they use. The goal is to show that the data could be used through several networks. This contrasts the possibility of using the data in the case of unconnected vehicles. For example, in-vehicle data is available only in the in-vehicle network for the case of unconnected vehicle. However, it could be exchanged with other parties through VANet, WPANs, and cellular network in the case of connected vehicles.

Table 2 Assets of connected vehicle

Asset	Description
In-vehicle data	Data extracted from the in-vehicle network, such as speed, turning light status, and fuel level
VANet data	Data of messages exchanged between vehicles, which may include vehicle locations. They could be used for applications, such as collision warning and traffic condition warning
ECU	In-vehicle device that controls the behavior of a component of a vehicle (e.g., break, engine) and collaborates with other ECUs to regulate the behavior of the vehicle. ECUs include GPS device: a device that provides the location of the vehicle at a specific time
OBU	A device for communicating a vehicle with other entities through VANets, WPAN, or cellular network and exchanging data with ECUs of the vehicle
ECU firmware	Programs that controls the behavior of the ECU
OBU firmware	Programs that controls the behavior of the OBU

Table 3 Classification of data used by connected vehicle into the networks they use (We use "net." for network)

Data types	In-vehicle net.	VANet	WPAN	Cellular net.
In-vehicle data	x	x	x	x
VANet data	x	x		x
ECU firmware	x			
OBU firmware		x	x	x

4.2 Attackers and Their Capabilities

*Attacker*s are entities (e.g., human, robots) who intend to compromise an asset (or many assets) and have capabilities to perform threats to the system. The main attackers for connected vehicle are [15]:

- *Greedy driver*—deviates from the protocol for gains; e.g., sends data about congestion ahead, so other cars change to alternate to long routes, and frees the road.
- *Snoop*—profiles drivers through aggregating data and violating privacy of drivers.
- *Prankster*—seeking fame hacker; e.g., trick two vehicles who use a collision avoidance application in order to slow down one driver and speed up the second. Thus, the vehicles may crash.
- *Insiders*—loads malicious software on a vehicle.
- *Malicious*—attempts to cause harm; e.g., manipulate deceleration warning system to create gridlock before detonating a bomb.

An attacker could be a car owner, a driver, a mechanic, or a government officer; so, they could have one or many attacker capabilities. The capabilities could be used to potentially compromise a target asset. Table 4 lists the capabilities that an attacker uses to perform the threats on assets of connected vehicle.

4.3 Threats

*Threat*s are circumstances and events that may harm an information system through unauthorized access, destruction, disclosure, modification of data, and/or denial of service [14]. Threats are performed by attackers who compromise system's resources.

We classify threats to connected vehicles into six aspects: security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. The classification uses the taxonomy described in the previous section.

Table 4 Possible capabilities of attacker to a connected vehicle

Capability code	Attacker capability
Cap001	Physical access to ECU
Cap002	Remote access to ECU
Cap003	Remote control of ECU
Cap004	Physical access to OBU
Cap005	Remote access to OBU
Cap006	Remote control of OBU
Cap007	Update of ECU firmware
Cap008	Update of OBU firmware
Cap009	Access the communication link between ECUs
Cap010	Control of the communication link between ECUs
Cap011	Access to the communication link between two vehicles
Cap012	Control of the communication link between two vehicles
Cap013	Access to the communication link between a vehicle and the SC
Cap014	Control of the communication link between a vehicle and the SC
Cap015	Access to the communication link between a vehicle and a personal device
Cap016	Control of the communication link between a vehicle and a personal device

4.3.1 Threats to the Communication Links

Connected vehicles and RSUs exchange messages through VANets; ECUs of a vehicle exchange in-vehicle data through the in-vehicle network; a vehicle exchange data with personal devices through WPAN, and with SPs and SC through cellular network. The threats to messages exchanged between the communicating parties in the four networks are similar. We describe in the following the main threats (cf. [15]).

A snoop could **eavesdrop the communication between two parties** without their consent. The two parties could be, for example, two devices of a same vehicle, two vehicles, or a vehicle and an SP or SC. For example, an attacker installs a mobile phone spy software (e.g., [16]) on the OBU of his/her victim's vehicle and gets periodically the data available in the OBU.

A vehicle of a greedy driver, who is required to forward messages it receives to other parties, could **drop the messages** (a.k.a. message suppression) it receives. For example, a prankster may receive congestion alerts from neighboring vehicles, but, does not forward them to its neighbors (i.e., close by vehicles) [15]; they will have to wait in the traffic.

A prankster could **fabricate messages** (i.e., create false messages) and send them to other vehicles or devices of the vehicle [5]. For example, a greedy driver makes his/her car pose as emergency vehicle and sends alert messages to its neighbors. The attack allows the driver to speed up their own trip [15].

A malicious attacker could **alter the messages** being exchanged between two parties (after it intercepts them). The attacker may receive messages from one

party, change their content, and forward them to the other communicating party. For example, a greedy vehicle receives messages about the improvement of traffic conditions. It changes the messages to worsening traffic conditions and broadcasts them to its neighbors.

A malicious attacker may receive messages from close by vehicles. They could perform **replay attack** by changing the time stamp of the messages and broadcasting them multiple times to close by vehicles. For example, a malicious attacker could intercept a message broadcasted by a vehicle and replay it to jam the communication channels of close by vehicles.

4.3.2 Threats to Data Validity

Smart mobility applications rely on the *validity* of the data (the data are correct—not false, and accurate) they use. False data, sent by malicious vehicles, could have severe safety impacts such as accidents. In the following, we describe a set of threats to data validity (a.k.a. data falsification).

The attacker disseminates **bogus data** (i.e., false data) in the network to affect the behavior of other vehicles [17]. For instance, an attacker could send information about bad traffic conditions. Vehicles receiving the messages try to find alternate paths; thus, the attacker frees the path for himself.

An attacker could **cheat in aggregating data** they receive from their neighbors. For example, in traffic management application, a vehicle collects information from its neighbors about the count of vehicles in their proximities—e.g., in 300 m radius. The data shall be used to compute the count of vehicles in a wider area—e.g., 2 km road length. The attacker sends false aggregate data instead of the data it computes using the data it receives from its neighbors.

An attacker who controls two communicating vehicles can tunnel packets broadcasted in one location to another, thus, disseminating wrong information: they perform a **wormhole attack** on the VANet [17]. For example, the attacker could tunnel traffic information messages from a vehicle in a crowded zone to vehicles in another zone. This misleads traffic management application which collects the information and broadcasts it to other vehicles.

4.3.3 Threats to Devices

An attacker who aim to compromise the data used by a smart mobility application (i.e., vehicle location, in-vehicle data, or VANets data) through (unauthorized) altering the behavior of the devices (GPS devices, ECUs, or OBU) of a target vehicle, needs to have either physical access or remote access to the device. The description of the main threats ECUs and OBUs follows.

A malicious attacker who has physical access to a device can **tamper the device hardware** by modifying or deactivating physical sensors, memories, and hardware functionalities. For example, an attacker changes circuits of a device to extract sensitive data, such as cryptographic keys.

A malicious attacker who has physical access to a device can **tamper the device software** by modifying its firmware. For example, an attacker modifies the firmware of a device to send false data to other vehicles.

A malicious attacker who has access to the devices of a target vehicle could **replace a device** of the vehicle with a compromised device to, for example, prevent it from working properly. For example, a mechanic—insider attacker—replaces a night vision device of a military vehicle so it does not work properly [18].

A malicious attacker injects dummy messages or jam the communication channel to stop the communication between a vehicle and other parties; and prevent important messages from reaching the vehicle [17]; that is, perform a **denial of service (DoS)** attack. For example, a malicious attacker provokes an accident and performs a DoS attack to prevent the appropriate deceleration warnings from reaching other drivers; they create a cascade crash on the highway [15].

A malicious attacker could inject messages to **remotely control the vehicle**. For example, an attacker installs malicious code on OBU accessible through cellular network and remotely start the vehicle.

An attacker could change the software installed on a device. They change the device behavior by **tampering the device firmware**. For example, an attacker can change the firmware of his/her OBU, so the vehicle does not send speed exceeding a specified limit. The information is, for example, used by the car insurance of the attacker to evaluate their driving behavior.

An attacker could perform an **unauthorized over the air update of the device** by installing remotely their own software on the device. For instance, a malicious attacker could attack the over-the-air (OTA) diagnostic and firmware update procedure; they install their own software on the device [19].

4.3.4 Threats to Identity and Liability of Vehicles

In the following we describe a set of threats to identity and liability of vehicles. Members of VANets: vehicles and RSUs; members of in-vehicle networks: ECUs, GPS, and an OBU; members of WPAN: personal devices and an OBU; members of cellular networks: handheld devices (HHD), SPs, and SCs, use identity information when communicating. Attackers may exploit the information to perform attacks. In the following we describe the main threats.

A malicious attacker uses identity information of another vehicle to impersonate (i.e., pretend to be) it [17]; that is, perform a **masquerade attack**. For example, a vehicle could impersonate another vehicle and sends information, such as its own location, to the fleet owner application.

A prankster or a malicious attacker could use multiple identities to deceive the communicating parties; that is, perform a **Sybil attack**. For example, a malicious vehicle pretends to be multiple other vehicles. Reported data from the vehicle appears to arrive from a large number of distinct vehicles, and hence it is considered credible [20].

An attacker sends information to an entity, but performs **repudiation attack**: denies responsibility of sending specific messages in the case of a probe. For example, an attacker sends emergency vehicle warnings, so they could bypass other vehicles, but deny the act in the case of a probe.

4.3.5 Threats to Access Control

An attacker could modify data or copy code of OBUs or ECUs: they perform **unauthorized access** to data and code. For example, a malicious attacker, who wants to reduce his/her insurance premium, could modify the aggregated data collected and computed by the OBU of his/her vehicle, before sending them to pay-as-you-drive insurance service.

4.3.6 Threats to Privacy of Vehicles and Drivers

The main threats for the privacy of vehicles and drivers are location privacy and driving behavior privacy. A snoop could perform a **location privacy attack** through collecting time series data about the location of a vehicle. The data could be obtained from messages the vehicle broadcasts in VANets, or from messages the vehicle sends to an SC through cellular network. For instance, a competing company could spy on its competitor by tracking the location of its vehicles. The information allows the company to extract information about the customers of the competitor.

An attacker could violate the **driving behavior privacy** of a driver by collecting time series data about the use of a driver of his/her vehicle, and use the data to infer and extract private information. For example, the police gets access to a database of in-vehicle data collected by an SC and extracts information about drivers who have speed violations (a.k.a. big brother threat).

5 Taxonomy of Security and Privacy Solutions for Connected Vehicles

In the following we describe a set of solutions that we found in the literature that address threats to connected vehicles.

5.1 Solutions to Assure Secure Communication Between Parties

This subsection describes solutions for secure communication in in-vehicle network, secure communication in VANets, secure communication between vehicles and personal devices, and secure communication between vehicles and SPs or SC.

5.1.1 Solutions for Secure Communication in In-Vehicle Network

Wolf et al. [21] propose to secure the in-vehicle communications between ECUs by a kind of hybrid encryption scheme, which combines symmetric and asymmetric encryption. Thus, for efficiency reasons, communication authenticity and confidentiality between ECUs can be ensured by a symmetric cipher together with a shared secret key. The secure distribution of the shared key to new ECUs can be done by an asymmetric encryption scheme, that checks the validity of the new ECU's certificate before securely handing the shared key. In order to extend this scheme to different in-vehicle networks executed in parallel (e.g., controller area network (CAN) and local interconnect network (LIN)), it can make reuse of the central gateway, which already routes the messages between different in-vehicle networks. Thus, the central gateway handles also the decryption and re-encryption of messages of different in-vehicle networks with different shared keys. Moreover, the central gateway can also add and remove new ECUs by running the asymmetric ECU authentication. The security mechanisms for secure communication are:

- *ECU authentication*: Each ECU sends its certificate and identity information to the gateway which authenticates the ECU; the gateway verifies whether the certificate of the ECU is signed by an authorized original equipment manufacturer (OEM). An authenticated ECU receives a symmetric key, which it uses to encrypt messages it broadcasts in the in-vehicle network.
- *Message encryption*: An ECU uses the symmetric key it gets from the authentication process to communicate with other ECUs. It is not practical, and not safe, to use asymmetric encryption schemes to encrypt all messages because of the time and capacity constraints of the in-vehicle network. For example, a vehicle could crash with neighboring vehicles, if the duration required for the break engine to receive the request from the driver and decrypt it exceeds a safety delay.
- *Gateway firewalls*: The certificate of an ECU includes an authorization code, used by the gateway of the in-vehicle network, to authorize the ECU to communicate with ECUs from another bus network.

A first practical realization of the proposed scheme has been done by the EVITA project (cf. [22]).

Bar-El [23] developed a framework for security services for in-vehicle network,[8] which consists of a set of modules installed on the ECUs and several applications that use the modules. The Framework includes the following components: a tool to generate and distribute session keys[9] for ECUs; a library for encrypting and authenticating messages exchanged between ECUs; a central secure storage for sensitive information; a library to assure that each ECU is (upon the boot) about to execute a code that was approved and intended to be executed; and a tool to provide a trusted time. The applications that are included in the framework are: secure ECUs code update enabler, secure logging enabler, authorization enabler, theft prevention enabler, and secure feature activation enabler.

5.1.2 Solutions for Secure Communication in VANets

In the following, we describe IEEE1609.2 standard [24] and research solutions proposed for securing the communication between nodes of a VANet. IEEE 1609.2 Standard addresses the threats to message eavesdropping, modification, and replay; and minimizes unauthorized identity disclosures. The standard defines the format and processing of secure messages, and a digital certificate (a document binding a public key to an entity described by the identity information, such as name and address) format to limit the use of bandwidth.

The standard specifies 4 security and privacy services for the network and applications that run over WAVE, which are:

- confidentiality—assuring that only the recipients of the massage can read it;
- authenticity—confirming of origin of the message;
- integrity—assuring that the message has not been changed while in transit between parties;
- anonymity—broadcasted message should not leak information to unauthorized recipients identifying the vehicle originating the messages.[10]

The standard recommends using elliptic curve integrated encryption scheme (ECIES) algorithm to encrypt messages exchanged between vehicles and elliptic curve digital signature algorithm (ECDSA) to sign the messages.[11] The minimum recommended size of encryption keys is 256 bits, of signature keys is 224, and of the certification authority (CA) keys is 256 bits. (CA is an entity that issues digital certificates. A digital certificate certifies the ownership of a public and private key pair to the subject of the certificate.) It recommends the use of

[8] The authors use the term intra-vehicle network to refer to in-vehicle network.

[9] Session keys are symmetric keys shared by communicating parties and are valid for a duration of the communication.

[10] This definition of anonymity is specific to the standard.

[11] Encryption assures confidentiality of messages. Digital signature assures integrity and authenticity of messages.

advanced encryption standard (AES)-counter with cipher block chaining message authentication code (CBC-MAC) (CCM), as a bulk encryption algorithm, for authenticating-then-encrypting messages they exchange. The standard recommends minimizing exchange of data that uniquely identify the recipient or would allow an unauthorized recipient to identify the sender of the messages.

Now, we summarize some research solutions for secure communication in VANets. Kargl et al. [25], Papadimitratos et al. [26], and Raya and Hubaux [17, 27] propose the use of digital certificate to assure integrity of messages and authenticate vehicles in the context of European research project SeVeCom [28]. They propose secure group communication,[12] key bootstrapping, and key revocation approaches. Secure communication approach assures confidentiality of messages exchanged between a group of vehicles. Key bootstrapping is the process of creating private keys and related digital certificate for vehicles, and for setting up the vehicles with the keys and certificates. Key revocation is the process of disabling the use of specified certificates.

Papadimitratos et al. [26] evaluate the security properties required by cooperative driving applications[13] and found that most of the applications require authentication, and integrity, but not confidentiality.[14]

Note that the European Telecommunications Standards Institute (ETSI) is currently working on standards for protecting data exchanged between vehicles, and between vehicles and RSUs in VANets [29].

5.1.3 Solution for Secure Communication Between Vehicles and Personal Devices

An in-vehicle WPAN is composed of a set of personal devices (users' short range wireless devices) and an in-vehicle access point (iVAP): a gateway between the in-vehicle network and the personal devices. The iVAP could be the OBU.

Mahmud and Shanker [2] propose an architecture for in-vehicle secure WPAN and a mechanism for generating and distributing secret keys for personal devices. The in-vehicle secure WPAN is composed of a coordinator, which is the iVAP, and personal devices. To join the network, a personal device D_i registers with the iVAP using a trusted link (i.e., a man-in-the-middle attacker cannot eavesdrop the link) which is either a peer-to-peer wired link, e.g., universal serial bus (USB), or very short range (i.e., few centimeters) wireless (VSRW) link, e.g., IR and NFC. The key distribution approach is adopted by NIST for distributing Bluetooth link keys in its revised guide for Bluetooth security [30].

[12] Secure group communication is communication between a group whose members share a secret key, which they use to encrypt messages they exchange.

[13] A cooperative driving application allows a set of vehicles, members of a VANet, to coordinate their actions by exchanging data through a VANet.

[14] Confidentiality is required by few applications, such as platooning: vehicles follow each other such that they do not collide.

The iVAP prompts the user for the (authentication) password upon connecting a new personal device D_i to the iVAP. If the authorization succeeds, the iVAP generates n secret keys[15] K_{ij} and sends them to the device, which stores them for future use. The iVAP (resp. personal device D_i) maintains a counter C_i that stores the number of times it communicates with personal device D_i (resp. iVAP); that is, the number of communication sessions. Each time D_i communicates with the iVAP, it uses the secret key K_{ij}, where index j is the value C_i; then, it removes the key from its store. iVAP sends new n secret keys to D_i when they reach the n^{th} communication session. To revoke a key k_{ij}, iVAP needs only to remove it from its store.

Devices composing a WPAN, in most of the cases, communicate using the Bluetooth technology. The two encryption algorithms supported by Bluetooth are: E_0 and advanced encryption standard-counter with cipher block chaining message authentication code (CBC-MAC) (AES-CCM) [31]. Note that, Lu et al. found an attack that can recover the encryption key used by algorithm E_0 in 2^{38} computations [32].

5.1.4 Solutions for Secure Communication Between Vehicles and Service Centers

We describe in this subsection the use of secure sockets layer (SSL) protocol and the use of wireless transport layer security (WTLS) protocol for communicating vehicles with SCs.[16]

We discuss in the following SSL [33] protocol. The protocol is used by applications to assure secure transport communication between two entities. It uses a reliable transport protocol, such as transport control protocol (TCP) [34]. (A transport protocol is reliable if it assures ordering of received messages, removing of duplicate messages, and preventing loss of messages.) The protocol assures confidentiality, integrity, and authenticity of messages. It has 4 sub-protocols:

- *Handshake protocol*: Two parties use the protocol to create a secure communication session. It is a sequence of steps that allows a server and a client to authenticate each other, to negotiate a cipher-spec., and to share keys before sending or receiving data. The cipher-spec. lists the cryptographic specifications that both parties agree on: key agreement algorithm, symmetric encryption algorithm, message authentication code (MAC) algorithm, and length of the secret keys.[17]

[15] Value of n may depend on the memory of the device.

[16] In this section we use mobile device to refer to OBU that has mobile device communication and processing capabilities. The reason is that the work that we refer to is about mobile device and not explicitly OBU; but, in general it applies to OBU.

[17] The reader may consult Katz and Lindell [31] for background on cryptography—if needed.

- *Application data protocol*: Application data are fragmented, compressed, and protected using the encryption and MAC algorithms defined in the current cipher-spec.
- *Change cipher-spec. protocol*: It signals a change of the cipher-spec. that both parties will use; both parties could send a change cipher-spec. message to notify the other party that subsequent messages will be protected under the just-negotiated cipher-spec.
- *Alert protocol*: It signals a warning or error during the handshake and up to the closure of the session. A party that sends a fatal error message to its partner closes immediately the session, after sending this record. The message gives a reason for the session closure. A party receiving a warning message decides whether to close the session or to keep it.

OBU has limited bandwidth (rate of sending data), processing speed, and memory size. The device shows a security processing[18] gap if the processing, memory, and bandwidth capabilities of the OBU are smaller than the security processing requirements [35]. Security processing gap causes loss of data that the OBU is required to send to other parties.

There are several issues that limit the use of SSL by mobile devices. For instance, SSL requires the use of a reliable transport protocol, such as TCP; however, mobile devices use also unreliable transport protocol user datagram protocol (UDP) for short message service (SMS). Also, mobile devices have limited bandwidth, and limited processing and memory capabilities. Furthermore, there are restrictions on exporting and using strong cryptography outside the USA.

WTLS [36] protocol was created to address the issues related to the use of SSL by mobile devices. The protocol is designed to provide confidentiality and integrity of messages, and authentication of communicating applications [36]. The protocol specification is similar to SSL; it provides the same functionalities and has the same sub-protocols: Handshake protocol, Application protocol, Change cipher-spec. protocol, and Alert protocol.

WTLS protocol addresses the issue of the need to support unreliable transport protocol, such as UDP. WTLS messages include the sequence number mode and key refresh rate, in plain text. The sequence number mode indicates the scheme used to communicate the sequence numbers (e.g., send in plain text). The information is used by the receiving party to enforce reliability of messages, if needed. The refresh rate indicates the frequency of updating the cryptographic specifications used by the communicating parties.

The protocol addresses the issues of bandwidth, processing, and memory limitations through the following measures: compress the application data, use of a digital certificate format shorter than X509.x format [37],[19] truncate the MAC of messages [38], use of weak cryptographic algorithms, and use of short secret keys [38].

[18] Security processing is computations for the purpose of security, such as data encryption and signature.

[19] WTLS supports both its own digital certificate format and X509.x format [37].

Most of the changes to SSL made WTLS vulnerable to several attacks [38]. For instance, the use of unauthenticated alert messages (messages sent through the Alert protocol) allows truncation attack: remove arbitrary packets from the data stream without the detection of the receiving party. Saarinen [39] describes 8 possible attacks on WTLS protocol. Note that some of these attacks are inherent to the protocol, while other are related to the use of weak cryptographic algorithms, or use of short keys.

5.2 Solutions for Detecting False Data

We discuss in this subsection solutions for detecting malicious data, wormhole attacks, and for secure data aggregation for VANets.

5.2.1 Detecting Malicious Data

Nodes use sensors to collect in-vehicle data and location data. They share the information with their neighbors, which may propagate it to their neighbors. Nodes collect data from their neighbors which could be somehow redundant.[20] Golle et al. [40] propose a model for detecting malicious data based on: assuming that data obtained from several source about a fact (e.g., the location of a vehicle or his/her speed) is redundant; assuming that an attack involving few malicious nodes is more likely than an attack that requires collision of a large number of nodes; and use of physical rules, when available, such as a node can have only one location at a time.

5.2.2 Protection Against Wormhole Attack

A wormhole attack on VANet causes a vehicle, member of a VANet, to use data *of* vehicles that are not its neighbors. Su and Boppana [41] propose neighbor verification by overhearing (NEVO): a protocol for verifying the neighbors of any vehicle. The protocol is a sequence of three messages between two neighbor nodes i and j. First, node i broadcasts a message probe query (PQ). Second, after node j receives the PQ from node i, it rebroadcasts the message as its probe forward (PF). Third, node j sends node i a probe reply (PR) message, which contains the processing delay, δ. Next, node i verifies if node j is its neighbor; it checks if equation $t_{\text{oh}} - tx_j - \delta \leq 2R/s_p$ holds, where t_{oh} is the time delay for node i to

[20]The difference between the reported locations of a vehicle by two other vehicles is smaller than an error margin.

overhear node j forwarding its own PQ message, tx_j is the transmission time for the forwarded message by node j, R is the radio signal propagation range, and s_p is the radio signal propagation speed.

There are two main issues with using NEVO protocol in VANet. First, the solution assumes that the processing delay, δ, is small compared to propagation and transmission delays. A node j could deceive node i by sending it a false value for δ that passes the test, and not the true delay, which may not pass the test. Second, the protocol does not consider the mobility of vehicles. Vehicles mobility requires considering the time of taking the measurements and the lifetime of the measurements.

Shokri et al. [42] propose another protocol for verifying the neighbors of any node of a network by verifying consistency of the information collected by neighboring vehicles. The protocol consists of three phases. In the first phase, ranging, every node of the network exchanges messages with its neighbors to evaluate its own distance to its neighbors. It creates a neighbors table that includes the computed distances.

In the second phase, neighbors table exchange, every node shares with its neighbors its own neighbors table. Next, each node creates a table that includes the distances between its neighbors, in addition to its own distance to its neighbors.

In the third phase, link verification, each node A performs three consistency tests to verify whether a reported neighbor is a true neighbor or not. The first test, link symmetry test, uses the integrated neighbors table to test whether the distance from node i to node j is similar to the distance from node j to node i.[21] The second test, maximum range test, checks if the distance between two neighbors is less than the maximum range of the radio transmission of vehicles. The third test, quadrilateral test, checks for every neighbor B of a node A, if there are two nodes C and D, such that (A, B, C, D) is a four-clique (the four nodes are neighbors to each other), and is a convex.[22]

This protocol has the same issue as the previous one; it does not consider the mobility of vehicles.

5.2.3 Secure Data Aggregation for VANets

Several smart mobility applications, such as co-operative detection of traffic jams, require the collection of information from big set of vehicles, e.g., vehicles running in a distance of 10 km. These applications require that each vehicle aggregates the information it receives from its neighbors (combine the information using

[21] The similarity test considers an error threshold in checking the equality of the computed distances.

[22] Quadrilateral is a polygon with four edges and four vertices or corners. Quadrilateral (A, B, C, D) is convex if the product of the cross products $(\overrightarrow{AB} \times \overrightarrow{BC})(\overrightarrow{BC} \times \overrightarrow{CD})(\overrightarrow{CD} \times \overrightarrow{DA})(\overrightarrow{DA} \times \overrightarrow{AB})$ is positive.

an aggregation function, such as average and sum) and disseminates it in larger area [43]. However, the integrity of aggregated information cannot be easily verified.

Dietzel et al. [43] propose an approach for secure aggregation, where each vehicle strategically selects signed atomic reports (i.e., data that are not the result of an aggregation function) from other vehicles and computes a trust in the correctness of the aggregated data. First, the aggregate area is divided into n squares. Then, the vehicle selects reports of vehicles located at the borders of the aggregate area, and n reports, each is an arbitrary report of a vehicle from each square. (The length of the sides of the squares, which defines the number of squares n, affects the granularity of the aggregate data.) Next, the reports are aggregated. The vehicle uses fuzzy reasoning [44] to compute a value describing the quality of selected atomic reports. The output value is a crisp value (e.g., very good, good, acceptable, bad, very bad) indicating the trust of the vehicle in the aggregate value.

5.3 Solutions for Tampering with Devices

In the following we describe four solutions that address devices tampering threats. The solutions are: use of devices that include a hardware security module (HSM), a virtualization platform for devices, a secure OTA firmware update, and a solution for DoS attack on OBU. Note that there are several other works that advocate the use of tamper resistance devices but did not provide solutions, e.g., [45].

5.3.1 Use of Devices That Include Hardware Security Module

The objective of project EVITA [46] is to design, verify, and prototype architecture for in-vehicle network, such that security-relevant components are protected against tampering, and sensitive data are protected against compromise. The partners developed three HSMs [47]:

- full HSM protects high-performance ECUs (e.g., OBU, central gateway) by providing a high-performance symmetric and asymmetric cryptographic engine for efficient in-vehicle and VANet communications,
- medium HSM is foreseen to protect some central multi-purpose ECUs (e.g., engine control, immobilizer) and hence is similar to full HSM except it has less processing performance and does not have a hardware asymmetric cryptographic engine,
- light HSM secures communication between small but critical ECUs, sensors, and actuators of a vehicle (e.g., pedal sensors, brake actuators, GPS or clock controller) through the use of a symmetric cryptographic engine.

The partners developed also software components for HSM and ECUs and integrated their libraries with AUTOSAR. AUTOSAR is an open and standardized automotive software architecture for ECUs used by automobile manufacturers, suppliers, and tool developers [48].

For secure update and regular verification of the device's firmware (i.e., secure boot) and some limited counterfeit protection, there already exists a cost-efficient automotive-capable security hardware extension (SHE). SHE is an official OEM-controlled specification [49] and is already available from several dedicated semiconductor manufacturers.

5.3.2 Over-the-Air Firmware Update

Companies managing smart mobility applications and vehicle manufacturers use OTA firmware update to fix bugs and improve the functioning of their software installed in OBUs and ECUs. Idrees et al. [50] propose a secure protocol for OTA firmware update for the devices of a vehicle. The main steps of the protocol are:

1. The service station remotely diagnoses the ECUs of the vehicle.
2. The service station sends a request for approving the updates of the firmware of a set of ECUs to the owner of the vehicle human–machine interface.[23]
3. For each ECU, the protocol terminates if the customer denies the request; otherwise, it proceeds with the following steps:
   (a) The station switches the ECU from application mode to reprogramming mode.
   (b) The service station requests from the OEM the key for decrypting the firmware of the ECU and gets the key encrypted using the public key of the ECU, which it forwards to the ECU.
   (c) The ECU downloads the firmware, encrypted with the key that it received from the OEM, from the service station in parts. The size of each part does not exceed the maximum length of the messages of the in-vehicle network.
   (d) The ECU verifies the authenticity of the firmware.[24] It proceeds with installing the firmware, deleting the old firmware, and switching the ECU back to application mode if the verification succeeds. It terminates otherwise.

   The protocol could be used, for example, by service stations.

5.3.3 Protection Against Denial of Service Attack

IEEE 1609.2 standard proposes the use of ECDSA for signing messages exchanged between vehicles in VANet. However, the verification of a single ECDSA signature requires 7 ms of computation on the typical OBU proposed by the standard [51]. Attacker can create and send invalid signatures in very short time. We call *arrival*

[23] The human–machine interface could be, for example, a mobile phone, or a device installed in the vehicle.

[24] It checks if a computed ECU configuration register (ECR) (a signature of the firmware) is equal to the reference ECR received from the OEM.

time, the average duration between the arrival of two messages to the OBU, and *processing time*, the average time required by the OBU to verify the signature of received messages. A DoS attacker could exploit the difference between processing time and arrival time to compromise the availability of the OBU: flood the OBU with invalid messages.

Studer et al. [51] propose a hybrid authentication mechanism named VANET authentication using signature and TELSA++ (VAST). The mechanism uses digital signature algorithm ECDSA and TELSA++, an extension for timed efficient stream loss-tolerant authentication protocol (TELSA). TELSA++ uses symmetric cryptography with delayed and periodic key disclosure for authenticating messages. The signature could be used, when needed, for non-repudiation of messages, e.g., in case the driver must be alerted about the received message.

5.4 Solutions for Vehicle Identity and Liability

Hubaux et al. propose electronic license plates (ELPs) [45] to identify vehicles. An ELP could be a digital certificate granted by authorized governmental agency. Example of use of ELPs includes paying toll roads: vehicles send their ELPs and other required information to an automated toll collection system. ELPs could also be used to authenticate vehicles in parkings.

Wolf [52] describes another general, holistic approach for designing, securing, implementing, and applying an ELP for various existing and upcoming vehicular application scenarios.

5.5 Solution for Access Control

The project open vehicular secure platform (OVERSEE) [53] develops an open standardized in-vehicle software and communication platform, which enables sharing of the platform's computing resources and its internal (e.g., in-vehicle network) and external communication capabilities (e.g., wireless fidelity (Wi-fi), cellular, VANet) for vehicular applications, while assuring mutual isolation and strong access control of applications, platform resources, services, communication interfaces, and data.

This approach empowers also third parties to develop, download, and install vehicular applications, while OVERSEE ensures that such third party applications cannot harm each other or any other in-vehicle IT system regardless whether an application malfunction is caused by an application failure (IT safety) or an application vulnerability exploited through a hacker or a computer virus (IT security).

The underlying protection approach is based on virtualization together with a strong access control mechanism and other central security functionalities (e.g.,

encryption, digital signature), which in turn are protected by an automotive-capable hardware security anchor (i.e., use of EVITA HSM).

5.6 Solutions for Privacy Protection

We describe in this subsection a set of solutions for protecting privacy of vehicles in VANets and privacy of drivers whose data are collected by SCs.

5.6.1 Privacy of Vehicles in VANets

Several smart mobility applications require authenticating messages exchanged between collaborating vehicles, such as cooperative collision avoidance. Papadimitratos et al. [26] propose message authentication using pseudonyms (fictitious names used to perform a particular role) to protect their privacy, instead of the identities of communicating vehicles. The description of the solution follows. Each vehicle generates a set of key pairs and sends the public keys to a corresponding CA. The CA generates a pseudonym for each key pair, signs each key pair, to produce certificates, and sends the certificates to the vehicle. The vehicle uses the key pairs to authenticate its messages sequentially. It uses one pair for a period of time, discards it, and uses the next key pair. The vehicle, while using a set of certificates, can require the next set of signed certificates from the CA. Changing pseudonyms makes it difficult for an adversary to link messages from the same vehicles and track its movements.

5.6.2 Privacy of Drivers Whose Data Are Collected by Service Centers

Project privacy enabled capability in co-operative systems and safety applications (PRECIOSA) [54] aims to design an architecture for policy enforcement in cooperative intelligent transport systems (CITSs). The architecture protects privacy of drivers, whose data are collected and used by SCs. Thus, a user associates with his/her data a set of policies for using the data. The policy includes entities that access the data (i.e., processors), purposes of using the data, retention period of data, and authorized operations—e.g., averaging, summation, and atomic reading. Kargl et al. [55] developed a privacy-enforcing runtime architecture for assuring enforcement of privacy policy in the trusted domain—a domain composed of hosts that are trusted to enforce the policies.

Kung et al. [56] advocate privacy-by-design approach for protecting privacy of drivers and vehicles.[25] The principles of privacy by design are:

- data minimization—the collection of personal data should be strict minimum;
- privacy enforcement—the operations of applications that use the personal data should provide maximum protection of data;
- transparency—the application should provide the stakeholders the way it ensure protection of the private data.

The authors discuss also a software engineering process that implements privacy-by-design principles.

Kost et al. [57] propose the use of an ITS privacy ontologies[26] to evaluate privacy violations of a model, representing an ITS application. That is, check the compliance of the model with privacy constraints—e.g., minimization of the use, access, and collection of personal data.

6 Approaches for Verifying the Security Properties of Connected Vehicles

Manual verification of security and privacy solutions for a connected vehicle is difficult because of the complexity of the network used by the vehicle. An attacker, who could remotely connect to a vehicle—e.g., through a cellular network—can inject messages in the in-vehicle bus (the link connecting the vehicle's ECUs) of the victims' vehicle. They do not need to physically connect to the in-vehicle bus to inject messages—a capability required to attack an unconnected vehicle [5].

Verifying the security and privacy properties of a connected vehicle requires the use of an automated mechanism. The only work on the subject that we found in the literature is automated verification of real time software (AVATAR) [58]: an environment for modeling and verifying real-time embedded systems. It is implemented by TTool [59]. TTool is an open source, general purpose modeling language for systems engineering applications supporting the specification, analysis, design, verification, and validation of a broad range of complex systems, which may include hardware, software, information, processes, personnel, and facilities. TTool supports systems modeling language (SysML) [60].

[25] The paper addresses ITS applications which, as we discussed in Sect. 2, includes connected vehicles.

[26] An ontology is a representation of knowledge as a set of concepts and the relationships between them in a specific domain.

The verification process is as follows. First, a user models the (to be verified) system using TTool. The model is a set of SysML diagrams extended with annotations describing the implementation of the security solution and the security properties that the system shall assure (e.g., a secret key is confidential and accessible to only communicating parties, not the attacker). Next, the user activates the security verification of the model—they click the appropriate button of TTool. AVATAR uses the security verification toolkit *ProVerif* [61]. TTool translates SysML diagrams, including the security properties, to the specification language of ProVerif. ProVerif verifies whether the security property is satisfied or not. If a security requirement is not satisfied, TTool provides traces that show how an attacker could exploit the system and compromise the security property.

Pedroza et al. [19] use AVATAR to verify firmware update (FU) protocol, developed by Idrees et al. [50]. They modeled the system using SysML diagrams, specified the security properties, and verified several security requirements including the confidentiality of the communication between the OEM, source of firmware update, and the ECU destination of the update. The experiment is a real use case of AVATAR.

Although AVATAR authors claim that it is intended to verify embedded system, TTool has been used only to verify the security of protocols (e.g., communication protocol, and key distribution protocol). ProVerif uses Dolev–Yao [62] attack model which focuses on the security of the communication between parties; Dolev–Yao attack model does not include attacks, such as controlling the behavior of one of the legitimate parties.

7 Summary

A *connected vehicle* is a vehicle whose ECUs communicate through an in-vehicle network; and it communicates with neighboring vehicles and RSUs through VANets, with personal devices through WPAN, and with SCs and SPs through cellular network. An attacker on connected vehicles has more capabilities to perform threats than an attacker on unconnected vehicles. Developing security and privacy solutions for smart mobility applications (applications that use information generated by vehicles; e.g., cooperative adaptive cruise control) requires considering threats to the system that integrates in-vehicle network, VANets, WPANs, and cellular network.

This work proposes a taxonomy for security and privacy aspects of connected vehicles, which are: security of communication links, data validity, security of devices, identity and liability, access control, and privacy of drivers and vehicles. It describes and classifies—based on the taxonomy—the threats to connected vehicles, describes and classifies the solutions proposed in the literature that address the threats, and reports on the only approach (that we found in the literature) for verifying security and privacy in connected vehicles.

The goal of the survey is to provide security architects of smart mobility applications with an initial repository of threats to connected vehicles and solutions that mitigate the threats. We believe that, currently, there are no "strong" solutions for verifying the security and privacy architecture of smart mobility applications.

Acknowledgements This work is supported by the Dutch national HTAS innovation program; HTAS being an acronym for High Tech Automotive Systems. More information on this innovation program is accessible via the document http://www.htas.nl/files/pdf%20bestanden/HTAS_Innovatie_Programma_-_september_2007[2].pdf. Any opinions expressed in this chapter are those of the authors and do not necessarily reflect those of Dutch national HTAS innovation program.

The authors thank Dr. Arno Spinner, from The Federal Highway Research Institute (BASt), Germany, and Pelin Anguin, from Purdue University, for providing valuable comments on an earlier draft of this book chapter.

References

1. Brooks R, Sander S, Deng J, Taiber J (2009) Automobile security concerns. IEEE Veh Technol Mag 4(2):52–64
2. Mahmud S, Shanker S (2006) In-vehicle secure wireless personal area network (swpan). IEEE Trans Veh Technol 55(3):1051–1061
3. Zhang J, Stojmenovic I (2005) Cellular networks. In: M. Gill (ed) Handbook of security, vol I, Part 2. Wiley, New York, pp 654–663
4. Intelligent Transport Systems (ITS) Communications Architecture (2010) The European Telecommunications Standards Institute (ETSI) Std. ETSI EN 302 665, Rev. V1.1.1, 09 2010. http://www.webapp.etsi.org/workprogram/Report_WorkItem.asp?WKI_ID=28554
5. Koscher K, Czeskis A, Roesner F, Patel S, Kohno T, Checkoway S, McCoy D, Kantor B, Anderson D, Shacham H, Savage S (2010) Experimental security analysis of a modern automobile. In: Proceedings of IEEE symposium on security and privacy, San Diego, CA, May 2010, pp 447–462
6. Johansson KH, Torngren M, Nielsen L (2005) Vehicle applications of controller area network. In: D Hristu-Varsakelis, W Levine (eds) Handbook of networked and embedded control systems. Springer, New York, pp 741–765
7. Uzcategui R, Acosta-Marum G (2009) Wave: a tutorial. IEEE Commun Mag 47(5):126–133
8. (2011) ecall: Time saved = lives saved. http://www.ec.europa.eu/information_society/activities/esafety/ecall/index_en.htm
9. van Arem B, van Driel C, Visser R (2006) The impact of cooperative adaptive cruise control on traffic-flow characteristics. IEEE Trans Intell Transp Syst 7(4):429–436
10. Markoff J Google cars drive themselves, in traffic. http://www.nytimes.com/2010/10/10/science/10google.html?_r=1&hp=&pagewanted=all
11. Kihl M (2009) Vehicular network applications and services. Vehicular networks techniques, standards, and applications. Auerbach Publications, Boston, pp 21–39
12. Karagiannis G, Altintas O, Ekici E, Heijenk G, Jarupan B, Lin K, Weil T (2011) Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions. IEEE Commun Surv Tutorials 13(4):584–616
13. Westin A (1967) Privacy and freedom. Atheneum, New York
14. Shirey R (2007) Internet security glossary, Version 2. RFC 4949 (Informational). http://www.ietf.org/rfc/rfc4949.txt
15. Parno B, Perrig A (2005) Challenges in securing vehicular networks. In: Workshop on hot topics in networks (HotNets-IV), College Park, Nov 2005. http://www.sparrow.ece.cmu.edu/~parno/pubs/vehicles.pdf

16. (2012) Mobile phone spy cell phone monitoring and tracking system. http://www.mobilephonespyx.com/
17. Raya M, Hubaux J-P (2007) Securing vehicular ad hoc networks. J Comput Secur 15(1):39–68
18. Wolf M, Weimerskirch A, Wollinger TJ (2007) State of the art: embedding security in vehicles. EURASIP J Embed Syst 2007:074706
19. Pedroza G, Idrees M, Apvrille L, Roudier Y (2011) A formal methodology applied to secure over-the-air automotive applications. In: 2011 IEEE Vehicular technology conference (VTC Fall), Sept 2011, pp 1–5
20. Zhou T, Choudhury RR, Ning P, Chakrabarty K (2007) Privacy-preserving detection of sybil attacks in vehicular ad hoc networks. In: The 4th annual international conference on mobile and ubiquitous systems: computing, networking and services, Philadelphia, Aug 2007, pp 1–8. http://www.dx.doi.org/10.1109/MOBIQ.2007.4451013
21. Wolf M, Weimerskirch A, Paar C (2004) Security in automotive bus systems. In: Workshop on embedded security in cars (escar)'04, Bochum, Germany, Nov 2004
22. Schweppe H, Idrees S, Roudier Y, Weyl B, Khayari RE, Henniger O, Scheuermann D, Pedroza G, Apvrille L, Seudie H, Platzdasch H, Sall M (2011) Deliverable d3.3: secure on-board protocols specification. Technical report, July 2011. http://www.evita-project.org/Deliverables/EVITAD3.3.pdf
23. Bar-El H (2009) Intra-vehicle information security framework. In: Proceedings of the 7th ESCAR embedded security in cars conference, Dïsseldorf, Germany, Nov 2009
24. IEEE (2006) Trial-use standard for wireless access in vehicular environments - security services for applications and management messages. IEEE Std. http://www.ieeexplore.ieee.org/servlet/opac?punumber=11000
25. Kargl F, Papadimitratos P, Buttyan L, Müter M, Wiedersheim B, Schoch E, Thong T-V, Calandriello G, Held A, Kung A, Hubaux J-P (2008) Secure vehicular communication systems: implementation, performance, and research challenges. IEEE Commun Mag, 46(11):110–118
26. Papadimitratos P, Buttyan L, Holczer T, Schoch E, Freudiger J, Raya M, Ma Z, Kargl F, Kung A, Hubaux J-P (2008) Secure vehicular communication systems: design and architecture. IEEE Commun Mag 46(11):100–109
27. Raya M, Hubaux J-P (2005) The security of vehicular ad hoc networks. In: The 3rd ACM workshop on security of Ad Hoc and sensor networks, series SASN '05, Alexandria, VA, Nov 2005, pp 11–21
28. (2011) Secure vehicle communication. http://www.sevecom.org/Pages/Publications.html
29. Randall S, Houmb S-H (2012) Experience in developing standards for cooperative systems. In: Workshop personal data protection and security aspects related to its applications, Brussels
30. Padgette J, Scarfone K, Chen L (2010) Guide to bluetooth security: recommendations of the national institute of standards and technology. National Institute of Standards and Technology (US), Gaithersburg
31. Katz J, Lindell, Y (2007) Introduction to modern cryptography. Chapman & Hall/CRC, Boca Raton
32. Lu Y, Meier W, Vaudenay S (2005) The conditional correlation attack: a practical attack on bluetooth encryption. In The 25th annual international conference on advances in cryptology, series CRYPTO'05. Springer, Santa Barbara, pp 97–117. http://www.dx.doi.org/10.1007/11535218_7
33. Freier A, Karlton P, Kocher P (2011) The secure sockets layer (SSL) protocol version 3.0, internet engineering task force (IETF) Std. http://www.tools.ietf.org/html/rfc6101
34. Postel J (1981) Transmission control protocol, Std. RFC793. http://www.tools.ietf.org/html/rfc793
35. Ravi S, Raghunathan A, Kocher P, Hattangady S (2004) Security in embedded systems: design challenges. ACM Trans Embed Comput Syst 3(3):461–491
36. L. Wireless Application Protocol Forum. Wireless transport layer security, Std, 2001. http://www.openmobilealliance.org/wapdocs/wap-261-wtls-20010406-a.pdf
37. Housley R, Ford W, Polk W, Solo D (1999) Internet X.509 public key infrastructure certificate and CRL profile, Std. rfc2459. http://www.ietf.org/rfc/rfc2459.txt

38. Jormalainen S, Laine J (1999) Security in WTLS. http://www.hut.fi/jtlaine2/wtls/
39. Saarinen M-JO (1999) Attacks against the WAP WTLS protocol. In: The IFIP TC6/TC11 joint working conference on secure information networks: communications and multimedia security, Leuven, Belgium, Sept 1999, pp 209–215. http://www.dl.acm.org/citation.cfm?id=647800.736984
40. Golle P, Greene D, Staddon J (2004) Detecting and correcting malicious data in VANETs. In: Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks, series VANET '04, Philadelphia, Oct 2004, pp 29–37. http://www.doi.acm.org/10.1145/1023875.1023881
41. Su X, Boppana R (2008) Mitigating wormhole attacks using passive monitoring in mobile ad hoc networks. In: IEEE global telecommunications conference, 2008. IEEE GLOBECOM 2008, New Orleans, Dec 2008, pp 1–5
42. Shokri R, Poturalski M, Ravot G, Papadimitratos P, Hubaux J-P (2009) A practical secure neighbor verification protocol for wireless sensor networks. In: Proceedings of the 2nd ACM conference on wireless network security, series WiSec '09, New York, pp 193–200. http://www.doi.acm.org/10.1145/1514274.1514302
43. Dietzel S, Schoch E, Könings B, Weber M, Kargl F (2010) Resilient secure aggregation for vehicular networks. Netw Mag Glob Internetw 24(1):26–31. http://www.dx.doi.org/10.1109/MNET.2010.5395780
44. Zadeh LA (1975) Fuzzy logic and approximate reasoning. Synthese 30:407–428. doi:10.1007/BF00485052. http://www.dx.doi.org/10.1007/BF00485052
45. Hubaux J, Capkun S, Luo J (2004) The security and privacy of smart vehicles. IEEE Secur Privacy 2(3):49–55
46. (2012) Evita project: E-safety vehicle intrusion protected applications. European commission research grant fp7-ict-224275. www.evita-project.org
47. Apvrille L, Khayari RE, Henniger O, Roudier Y, Schweppe H, Seudié H, Weyl B, Wolf M (2010) Secure automotive on-board electronics network architecture. In: FISITA 2010 world automotive congress, Budapest, Hungary, May–June 2010
48. (2012) Autosar. http://www.autosar.org/
49. Hersteller Initiative Software - Security Working Group (2009) SHE-functional specification v1.1, rev 439
50. Idrees MS, Schweppe H, Roudier Y, Wolf M, Scheuermann D, Henniger O (2011) Secure automotive on-board protocols: a case of over-the-air firmware updates. In Proceedings of the 3rd international conference on Communication technologies for vehicles. Springer, Berlin/Heidelberg, pp 224–238. http://www.dl.acm.org/citation.cfm?id=1987310.1987333
51. Studer A, Bai F, Bellur B, Perrig A (2009) Flexible, extensible, and efficient vanet authentication. J Commun Networks 11(6):574–588
52. Wolf M (2010) A secure and privacy-preserving electronic license plate. In Automotive: safety & security, Stuttgart, Germany, 21–23 June 2010
53. (2012) Oversee. https://www.oversee-project.com/
54. (2012) Preciosa-privacy enabled capability in co-operative systems and safety applications. http://www.preciosa-project.org/
55. Kargl F, Schaub F, Dietzel S (2010) Mandatory enforcement of privacy policies using trusted computing principles. In: AAAI spring symposium: intelligent information privacy management, Stanford, CA, Mar 2010
56. Kung A, Freytag J, Kargl F (2011) Privacy-by-design in ITS applications. In: 2011 IEEE international symposium on a world of wireless, mobile and multimedia networks (WoWMoM), Lucca, Italy, June 2011, pp 1–6
57. Kost M, Freytag J-C, Kargl F, Kung A (2011) Privacy verification using ontologies. In: The 1st international workshop on privacy by design, Vienna, Austria, Aug 2011, pp 627–632
58. Pedroza G, Apvrille L, Knorreck D (2011) AVATAR: a SysML environment for the formal verification of safety and security properties. In: The 11th annual international conference on new technologies of distributed systems (NOTERE), Paris, France, Mar 2011, pp 1–10
59. TTool - an open-source UML and SysML toolkit. http://www.ttool.telecom-paristech.fr/

60. Object Management Group Inc (OMG) (2010) OMG systems modeling language (OMG SysML). http://www.sysml.org/docs/specs/OMGSysML-v1.2-10-06-02.pdf
61. Blanchet B (2009) Automatic verification of correspondences for security protocols. J Comput Secur 17(4):363–434. http://www.dl.acm.org/citation.cfm?id=1576303.1576304
62. Dolev D, Yao AC (1981) On the security of public key protocols. Technical report, Stanford

Erratum to:

Wireless Sensor and Mobile Ad-Hoc Networks

D. Benhaddou, A. Al-Fuqaha (eds.)

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_2

DOI 10.1007/978-1-4939-2468-4_11

The print and online versions of the book contain some errors, and the corrections to these versions are given on the following pages:

1. The forename of the first author M. Ayyash was incorrect in the original publication. The name has been corrected to read **'Moussa Ayyash'**

2. Affiliation of the third author M. Anan was incorrect in the original publication. The affiliation has been corrected as below:
   Alfaisal University, Riyadh, Saudi Arabia

The publisher apologizes to the authors and readers for these errors.

The online version of the original chapter can be found at
http://dx.doi.org/10.1007/978-1-4939-2468-4_2

D. Benhaddou, A. Al-Fuqaha (eds.), *Wireless Sensor and Mobile Ad-Hoc Networks*,
DOI 10.1007/978-1-4939-2468-4_11