Dipartimento di Ingegneria Elettrica, Elettronica e Informatica

PhD in Systems, Energy, Computer and Telecommunication Engineering

XXIX Cycle

Real-Time Networks for Robotics and Industrial applications: Research Challenges and Novel Solutions

Ing. Gaetano Patti

Coordinator
Prof. Paolo Arena

Tutor
Prof. Lucia Lo Bello

Abstract

The increasing adoption of smart sensors and actuators enables industrial applications to process data in a decentralized way. In this context, communication networks play a key role, as industrial automation applications require network architectures and communication protocols, wireless and wired, able to support the interaction between multiple devices not only in a reliable way, but also guaranteeing the meeting of real-time constrains of the supported applications. The distributed processing among coordinated automation devices includes the cooperation of robot teams. However, cooperative robot networks impose additional communication constraints to those required in the automation context.

This thesis originates from the collaboration with STMicroelectronics and targets mechanisms, algorithms and protocols that meet the communication requirements of cooperative robot applications in the industrial automation scenario. With the aim to avoid the definition of "yet another protocol", the work mainly focuses on the existing communication technologies adopted in the industrial context. In particular, innovative mechanisms, algorithm and protocols built upon standard communication technologies are investigated, with the aim to meet both the specific requirements imposed by cooperative robot applications (e.g., mobility and low latency) and those that are typical of industrial automation networks.

In the thesis, innovative solutions for several communication technologies, both wired and wireless, are presented and described. Among the network technologies addressed, EtherCAT, Bluetooth Low En-

ergy, Sub-GHz Communications, the IEEE 802.15.4e and IEEE 802.11 standards. Evaluations through simulations, analysis and experiments on proof-of-concept implementations are reported, which prove the effectiveness and the suitability of the proposed solutions.

Acknowledgements

I wish to express my sincere gratitude to the people who supported me during these years as a PhD candidate.

I would like to thank STMicroelectronics, in particular, Ing. Nunzio Abbate, Ing. Alessandro Faulisi and Ing. Marco Branciforte, for their confidence and the support they provided me, not only materially but also through their novel ideas and suggestions for my research activities.

Many thanks to Prof. Giovanni Muscato, with whom I had the pleasure to collaborate during these three years.

Many thanks to Prof. Johan Akerberg for the hospitality and for giving me the possibility to spent a period of time conducting research activities at ABB AB, Corporate Research, Sweden.

Most importantly, I wish to extend my warmest and sincerest gratitude to my advisor Prof. Lucia Lo Bello for the opportunity to live this amazing experience. I am grateful to her for the advices, the time and the efforts she invested in my professional education and for the challenging and inspiring conversations thanks to which we got ambitious professional goals.

Contents

List of Acronyms

AIFS	Arbitration InterFrame Space
API	Application Program Interface
ARQ	Automatic Repeat reQuest
BLE	Bluetooth Low Energy
CAN	Controller Area Network
CAP	Contention Access Period
CCA	Clear Channel Assessment
CCCD	Client Characteristic Configuration Descriptor
CFP	Contention-Free Period
CPS	Cyber-Physical Systems
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DCC	Dynamic Channel Configuration
DMR	Deadline Miss Ratio
DSME	Deterministic and Synchronous Multi-channel Extension
E2ED	End-to-End Delay
EB	Enhanced Beacon
EDCA	Enhanced Distributed Channel Access
FIFO	First-In First-Out
GTS	Guaranteed Time Slots
HCCA	Hybrid Coordination Function Controlled Channel Access

IWSN	Industrial Wireless Sensor Networks
LLDN	Low-Latency Deterministic Network
MAC	Medium Access Control
MCCA	Mesh Coordination Function Controlled Channel Access
MCU	MicroController Unit
NLM	Network Link Matrix
NSP	Network Saturation Point
PAN	Personal Area Network
PCA	Priority Channel Access
PLR	Packet Loss Ratio
RSSI	Received Signal Strength Indicator
RTE	Real-Time Ethernet
SPI	Serial Peripheral Interface
ST	Scheduled Traffic
TAS	Time-Aware Shaper
TDMA	Time Division Multiple Access
TSCH	Time Slotted Channel Hopping
WCRT	Worst Case Response Time

List of Publications

Included publications

- **Publication A.** L. Lo Bello, E. Bini, G. Patti, "Priority-Driven Swapping-Based Scheduling of Aperiodic Real-Time Messages Over EtherCAT Networks", *IEEE Transactions on Industrial Informatics*, vol. 11, issue 3, pp. 741-751, Aug. 2014.

- **Publication B.** G. Alderisi, G. Patti, O. Mirabella, L. Lo Bello, "Simulative assessments of the IEEE 802.15.4e DSME and TSCH in realistic process automation scenarios", *Proc. of the IEEE 13th International Conference on Industrial Informatics (INDIN)*, Cambridge, UK, July 2015.

- **Publication C.** G. Patti, L. Lo Bello, "A Priority-Aware Multichannel Adaptive Framework for the IEEE 802.15.4e-LLDN", *IEEE Transactions on Industrial Electronics*, vol. 63, no. 10, pp. 6360-6370, Oct. 2016.

- **Publication D.** G. Patti, G. Muscato, N. Abbate, L. Lo Bello, "Towards Low-datarate Communications for Cooperative Mobile Robots", *Proc. of the IEEE World Conference on Factory Communication Systems (WFCS)*, Palma de Mallorca, Spain, 05/2015.

- **Publication E.** G. Patti, G. Alderisi, L. Lo Bello, "SchedWiFi: An Innovative Approach to support Scheduled Traffic in Ad-

hoc Industrial IEEE 802.11 networks", *Proc. of the 20th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Luxembourg, Sept. 2015.

- **Publication F.** G. Patti, L. Leonardi, L. Lo Bello, "A Bluetooth Low Energy real-time protocol for industrial wireless mesh networks", *Proc. of the 42nd Annual Conference of IEEE Industrial Electronics Society (IECON)*, Florence, Italy, Oct. 2016.

Publications not included

- M. Ashjaei, G. Patti, M. Behnam, T. Nolte, G. Alderisi, L. Lo Bello, "Schedulability Analysis of Ethernet Audio Video Bridging Networks with Scheduled Traffic Support", *Real-Time Systems - The International Journal of Time-Critical Computing Systems* , accepted Jan. 2017, to appear.

- G. Patti, G. Alderisi, L. Lo Bello, "Introducing multi-level communication in the IEEE 802.15.4e protocol: the MultiChannel-LLDN", *Proc. of the IEEE 19th International Conference on Emerging Technologies and Factory Automation (ETFA)*, Barcelona, Spain, 09/2014.

- G. Patti, G. Alderisi, "An approach towards adaptivity in wireless sensor networks", *Proc. of the International Conference on Numerical Analysis and Applied Mathematics (ICNAAM)*, Rhodes, Greece, 09/2014.

- L. Lo Bello, G. Patti, G. Alderisi, V.D. Patti, O. Mirabella, "A Flexible Mechanism for Efficient Transmission of Aperiodic Real-Time Messages over EtherCAT networks", *Proc. of the IEEE International Workshop on Factory Communication Systems (WFCS)*, Tolouse, France, IEEE, 05/2014.

Chapter 1

Introduction

In the last years the increasing spread of smart sensors, able to acquire data from multiple transducers, processing it and take the appropriate decisions (such, for instance, the activation of one or multiple actuators), has opened new frontiers for the investigation of more and more complex industrial applications. In fact, the capability of such devices to process data in a decentralized way provides a noteworthy potential for distributing on multiple nodes the processing activities to perform complex tasks. In this context, communications play a very important role, as they enable the coordination and the cooperation of the actors of automation applications (e.g., sensors, actuators, robotic arms, etc).

The coordinated action of sensors and actuators in the automation field requires network architectures and communication protocols, both wireless and wired, able to support the interaction between multiple devices not only in a reliable way, but also guaranteeing the meeting of the real-time constrains of the supported applications.

In this context of distributed processing among coordinated automation devices, teams of cooperating robots come into play. This is a major improvement in robotics, as cooperation enables robots to go beyond the limitations of individuals, providing the possibility to perform more complex tasks [1] than those that each single robot can

perform alone.

This thesis stems from industrial needs and, more specifically, from the collaboration with STMicroelectronics, with the aim of enabling the network technologies that are typically adopted in industrial automation to support also cooperative robot applications in the industrial automation scenario.

The heterogeneity of the supported applications and their requirements imposes the investigation of innovative communication systems able to fulfill all the application needs. For instance, the integration of multiple mobile robots in an automation network also requires mobility support together with other features such as, real-time behavior and reliability.

In the industrial automation, the flexibility and adaptivity of communication systems has also gained a remarkable interest. For instance, Germany started the so-called fourth industrial revolution (Industrie 4.0 [2]), which foresees the Internet connectivity of industrial assets [3]. According to the "Industrie 4.0" vision, industry shall develop networks integrating machinery, sensors, plants with nodes able to autonomously exchange information, trigger actions, and cooperate with each other without human intervention. This not only applies to field devices or machines, but also to the mobile robots that, for instance, automatically transports various types of goods on the factory floor or handle materials in automated manufacturing systems.

The typical requirements of industrial automation communications are summarized in the following.

- **Reliability:** Automation networks have to provide a suitable reliability level for the transmissions, i.e., a low error probability. To cope with this requirement, some of the mechanisms adopted are retransmissions (with or without acknowledgement), relaying (i.e., the transmission of a message over multiple paths from the source to the destination), and Automatic Repeat reQuests (ARQs) to detect errors.

- **Fault-tolerance:** The node services have to be always available, as a stop of some hours entails high costs for the industry. For this reason, fault-tolerance mechanisms (such as, node redundancy and recovery, replication of functions) have to be provided.

- **Real-Time:** Networks have to meet the timing constraints of the supported applications. This means that the end-to-end delays of messages, i.e., the delivery times from the source to the destination, have to be bounded and predictable. Hence, suitable deadline-aware Medium Access Control (MAC) mechanisms to avoid unpredictability in communications and scheduling policies have to be investigated.

The above mentioned requirements have a general significance in automation networks. However, applications impose different constraints. In particular, cooperative multi-robot applications require networks able to support additional constraints, e.g.,

- **Low message latency:** Due to the fast dynamics of this kind of systems, messages not only have to be delivered within a certain deadline (real-time requirement), but also need low message latencies (i.e., from hundreds of microseconds to hundreds of milliseconds).

- **Scalability:** Cooperative multi-robot networks are typically realized with a high number of nodes. However, some of the most adopted medium access mechanisms in industrial automation are TDMA-based. Such a mechanism is well known to be not very scalable, as a high number of nodes entails a high number of timeslots in the communication cycle, thus increasing the message latency.

- **Mobility:** The introduction of mobile robots in the industrial scenario entails additional requirements on the networks. In particular, the standard wireless communication technologies, such as the IEEE 802.11e, IEEE 802.15.4e, etc., do not specifically

support node mobility. For these reasons, several mechanisms, which modify the standard specification, were proposed [4], [5]. However, the support for mobility should not clash with the other requirements described above.

- **Support to multiple traffic classes:** The integration of the Internet technologies in robotics applications and also in industrial networks requires the support for the transmission of multiple traffic types, with different timing constraints, in an efficient way and without compromising the transmission of the control traffic (that is generally the most critical one).

This thesis proposes innovative mechanisms that build upon the existing communication technologies used in automation and enable them to support the needs of real-time industrial applications, including the ones based on teams of cooperating robots. The aim of the work is to meet the requirements of mobility, low latency and scalability that are imposed by cooperative robot applications and those of reliability, fault-tolerance and real-time that are typical of industrial automation networks. Innovative solutions are investigated and novel mechanisms, protocols and algorithms for different technologies are proposed.

1.1 Research challenges and methodology

The research challenges addressed in this thesis are various and refer to both wired and wireless technologies.

1.1.1 Introducing the support for aperiodic real-time traffic over EtherCAT networks

The EtherCAT protocol is a real-time Ethernet standard adopted in multiple automation scenarios (substation automation in smart grids, motion control in factories, robotics) that provides high bandwidth

and meets the requirements of industrial real-time communications. Among the RTE protocols, the EtherCAT standard is suitable for motion control and closed-loop control applications, which require very short cycle times. One important limitation for event-driven robotics applications is that, as EtherCAT was specifically devised for periodic traffic, aperiodic real-time transmissions are far from being efficient and entail long cycle times.

To overcome this limitation, the first research challenge of this thesis is therefore introducing the support for aperiodic real-time traffic over EtherCAT networks. The solution proposed is a general framework for priority-driven swapping-based scheduling of aperiodic real-time messages over EtherCAT networks, which uniformly covers both dynamic and static priority and allows for very short cycle times.

Chapter 2 provides a description of the priority-driven swapping framework, a schedulability analysis for both static priority and dynamic priority scheduling, and simulative assessments, obtained through OMNeT++ simulations.

1.1.2 Low datarate wireless technologies for real-time communication in automation and robotics

The second research challenge regards the suitability of extending industrial low datarate wireless technologies for supporting cooperative robots applications. The research activities start with an assessment of existing wireless communication technologies, in order to evaluate their suitability for supporting cooperative robot applications. The focus is on the protocols of the IEEE 802.15.4e standard, as they are novel and promising. Then, the research work continues with the investigation of mechanisms to improve the scalability, the fault-tolerance, and to introduce message prioritization in the Low-Latency Deterministic Network (LLDN) protocol. Next, a low-datarate MAC protocol specifically devised for cooperative mobile robot applications, is defined and a proof-of-concept implementation of the above

mentioned protocol on STMicroelectronics Sub-GHz devices (i.e., devices operating on frequencies lower than 1 GHz) is presented.

Step 1. Performance assessment of the IEEE 802.15.4e protocols. This step addresses the assessment of the wireless protocols defined in the IEEE 802.15.4e standard for low datarate Industrial Wireless Sensor Networks (IWSNs). The IEEE 802.15.4 standard is not able to cope with the requirements which are found in many application domains that require low latency, robustness, and determinism. For this reason, the IEEE 802.15.4e amendment was introduced, which provides novel MAC-layer profiles that are optimized for a broad range of application domains, including process automation. The first step focuses on two of these profiles, i.e., the Deterministic and Synchronous Multi-channel Extension (DSME) and the Time Slotted Channel Hopping (TSCH). The aim of this work is twofold. First, assessing their behavior in realistic process automation scenarios. Second, comparing their performance in terms of end-to-end delay, reliability and scalability. The ultimate aim of the work is identifying the limits of those protocols, thus paving the way to further work addressing suitable approaches to tackle them.

Chapter 3 presents the description of the protocols, defined in the IEEE 802.15.4e standard, the assessed scenario and simulative results.

Step 2. Improving the existing technologies: the Low-Latency Deterministic networks. The LLDN protocol is intended for factory automation applications that require very low latency and large networks, such as automotive manufacturing and robotics. However, LLDN does not provide priority support to properly deal with real-time traffic or dynamic channel configuration capabilities to cope with unreliable channels. Moreover, it offers a limited scalability, as the cycle time grows linearly with the number of network nodes. This thesis therefore proposes the priority-aware multi-channel adaptive (PriMulA) framework, which introduces in the LLDN priority-aware

scheduling, multichannel communication, adaptive channel selection, and channel blacklisting. PriMulA supports a higher number of network nodes than the LLDN protocol while keeping short cycle times. In addition, PriMulA avoids deadline miss and improves the network reliability. It maintains the interoperability with LLDN standard nodes and can be implemented on commercial off-the-shelf devices.

Chapter 4 presents the PriMuA framework, the schedulability analysis, comparative simulations to assess the performance, and a proof-of-concept implementation.

Step 3. A novel solution: the RoboMAC protocol. The PriMulA framework meets the requirements of real-time, reliability, fault-tolerance, low latency and scalability imposed by the industrial and cooperating robot applications. For this reason it is a suitable solution for the applications that do not require mobility. However, PriMulA is not suitable for mobile robot applications, as it does not support the node mobility. Moreover, recent cooperative robot applications envisage the support of low datarate communication technologies, as this choice is beneficial in terms of energy consumption, weight reduction and integration with WSNs. For these reasons, the research work targets an innovative low datarate protocol, specifically devised for cooperative mobile robots applications, that operates on low datarate networks and is able to provide bounded latencies, mobility and scalability.

Chapter 5 presents the RoboMAC protocol and its implementation on Sub-GHz devices produced by STMicroelectronics. Experimental assessments prove the protocol suitability for cooperating mobile robot applications.

1.1.3 Consumer wireless technologies for real-time communication in automation and robotics

The third research challenge is the introduction of a mechanism to support real-time transmissions over wireless mesh networks. The activities foresees two steps related to the IEEE 802.11 protocol and the Bluetooth Low Energy protocol, respectively.

Step 1. Introducing scheduled traffic over IEEE 802.11 ad-hoc networks. The spread use of IEEE 802.11 networks in industrial automation raise the need to support multiple traffic classes with different requirements. The work targets a novel approach, called SchedWiFi, that provides flexible support to the scheduled traffic class, i.e., a high priority traffic class that is transmitted according to a fixed schedule, over IEEE 802.11 ad-hoc industrial networks. SchedWiFi operates on the IEEE 802.11n physical layer, thus providing high datarate, and supports multiple traffic classes with different priorities. SchedWiFi modifies the Enhanced Distributed Channel Access (EDCA) mechanism allowing to transmit scheduled traffic without requiring any predefined superframe structure, or timeslots, thus allowing for more flexible schedule of non-ST traffic.

Chapter 6 presents the SchedWiFi protocol and a simulative performance assessment in realistic scenarios.

Step 2. Introducing real-time communication in Bluetooth Low Energy mesh networks. The large diffusion of hand-held devices opens new frontiers in the human interaction with the industrial plant and its robots. The Bluetooth Low Energy (BLE) protocol is an attractive solution for implementing low-cost IWSN with reduced energy consumption and high flexibility. However, the current BLE standard does not provide real-time support for data packets and is limited to a star topology. This thesis presents RT-BLE, a real-time protocol for industrial wireless mesh networks that

is developed on top of the BLE and overcomes these limitations.

Chapter 7 describes the RT-BLE protocol and provides a timing analysis, a proof-of concept implementation on STMicroelectronics BlueNRG-MS devices, analytical and experimental results.

Chapter 8 gives the conclusions of this thesis and provides the hints for future works.

Chapter 2

Scheduling of Aperiodic Real-Time Messages over EtherCAT Networks

The integration of heterogeneous applications with different information flows and requirements requires networks capable to support multi-service communications. In particular, modern industrial networks must offer support for both time-driven and event-driven control applications. In time-driven applications, messages are periodically transmitted and control actions are taken at constant rate [6,7], while in event-driven applications, messages are transmitted when one or more trigger events occur (e.g., if the controlled variable exceeds a given threshold) [8,9]. For example, closed-loop control applications typically generate periodic messages with deadlines of approximately 1 ms [7]. However, these applications may also require the transmission of aperiodic real-time messages, that have to be accommodated in the overall traffic schedule without affecting periodic messages. Aperiodic real-time transmissions are also found in Cyber-Physical Systems (CPSs), as they typically operate in unpredictable environments [10–12].

Recently, Real-Time Ethernet (RTE) technologies [13] have be-

come increasingly popular, as they offer high bandwidth, meet the requirements of industrial real-time communication and allow for vertical integration of the different levels in the automation pyramid [14]. Recent literature highlighted the properties of industrial Ethernet networks able to support various traffic classes [15] and temporal constraints [16]. One example is Profinet IRT, whose bandwidth management and scheduling are addressed in [17] and [18], respectively. Another interesting real-time Ethernet protocol is TTEthernet [19], which offers three different traffic classes to support the temporal and bandwidth constraints of a broad range of applications.

This chapter focuses on the EtherCAT protocol, which is included in both the IEC 61158 [20] and IEC 61784 [13] standards. EtherCAT is suitable for motion control and closed-loop control applications, which require very short cycle times, where the cycle time is defined as the time necessary to exchange the input/output data between the controller and all the networked devices once [21, 22].

EtherCAT provides a daisy-chain topology and a master/slave architecture in which the master periodically transmits a standard Ethernet frame that embeds an EtherCAT frame containing multiple *telegrams* (as shown in Figure2.1). Slaves read and/or write data in the telegram by processing the frame "on-the-fly", so when a byte arrives to a slave it is processed and transmitted to the next slave without waiting for the complete reception of the Ethernet frame. The last slave in the chain transmits the frame back to the master by exploiting the full-duplex capability of Ethernet.

As shown in Fig.2.1, the EtherCAT frame used for transmitting process data contains one or multiple telegrams, which start with a header containing the command code (i.e., read, write or read/write), the addressing fields, and the payload length. Telegrams end with a Working Counter (WKC) field, which is incremented by the slaves every time they successfully read and/or write data into the telegram. The WKC is also used for error detection.

In order to allow slaves to transmit periodic real-time traffic, the EtherCAT standard provides polling-based mechanisms that can also

be adopted for the transmission of aperiodic real-time messages. For instance, the master may send EtherCAT frames containing at least one telegram for each slave that might have aperiodic real-time traffic to transmit.

However, such a mechanism would entail long cycle times, as the master must provide room in the EtherCAT frame for any slave that has the potential to transmit aperiodic real-time traffic, regardless of whether such a slave actually has traffic to transmit. For instance, in a network with 20 slaves, each with the potential for transmitting 32 byte long aperiodic real-time messages, the master should provide 20 telegrams for each cycle. This would result in an increase in the cycle time duration of 70.4 μs, with the probability that most of the time such telegrams would not actually be used, due to the event-driven nature of aperiodic traffic generation.

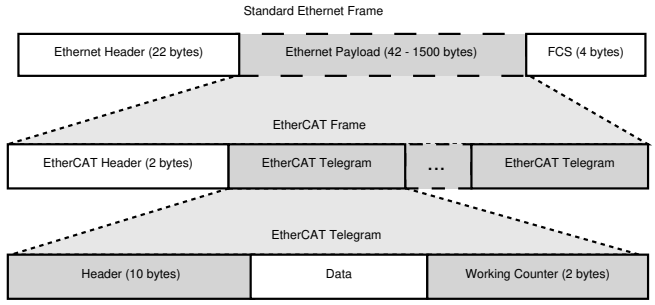

Figure 2.1: Structure of an EtherCAT frame containing multiple telegrams

In addition to the cycle time increase, the need for handling aperiodic real-time messages by a polling mechanism performed by the master would also introduce, in the case of event-triggered control applications, a bandwidth waste that would reduce the main advantage of the event-triggered control, that is, the low bandwidth demand.

Some works in the recent literature address methods to reduce the cycle times in the EtherCAT networks. In [23] periodic and aperiodic real-time traffic is scheduled by the master, which provides

guaranteed bandwidth for the slave transmissions. In [24] a switch operating at the EtherCAT telegram level is proposed, but such a solution does not focus on aperiodic traffic scheduling.

The work in [25] proposes an arbitration-based access scheme for aperiodic real-time messages which introduces new aperiodic telegrams that are contented by the slaves for transmitting aperiodic messages. The slave with the highest priority among the ones competing for the aperiodic telegrams will overwrite "on-the-fly" the incoming aperiodic message.

The mechanism in [25] foresees that the master transmits a copy of the aperiodic message received. In this way, the slave that transmitted the message realizes that it was successful and so the message can be safely removed from its queue. Conversely, the other slaves know that they did not succeed and must try again. This approach reduces the cycle time for transmitting aperiodic messages compared with the EtherCAT standard, but still suffers from some limitations. The main one is that low priority messages may experience long delays due to the interference from high priority ones, with a potential for starvation.

The same authors in [26] added the capability for embedding multiple aperiodic messages in one telegram. In this new approach, a slave has to receive a cumulative "acknowledgement" message from the master before removing the transmitted message from the local queue. The purpose of such a message is twofold. It is a notification of successful transmission and also a way to allow a slave to remove the aperiodic message from its local queue.

Inspired by the Slot Swapping Protocol (SSP) [27–29], in [30] dynamic priorities are exploited to swap between an incoming lower priority aperiodic message and a higher priority message pending at the current slave. This is achieved by defining a new telegram type, contented among the slaves that have an aperiodic real-time message to transmit.

Contention is ruled by comparing the absolute deadlines of the messages according to the EDF algorithm. When the message con-

tained in the incoming telegram is less urgent and therefore loses the contention, it is swapped and replaced by the message with the most urgent deadline in the local queue. Swapping does not entail message loss, because the slave which has swapped the incoming message will be in charge of transmitting it whenever possible.

Contributions of this chapter In this chapter a Priority-driven swapping-based mechanism to deal with the problem of providing support for aperiodic real-time traffic over EtherCAT networks is presented. As previously explained, the EtherCAT standard does not offer this support. The Priority-driven swapping-based approach combines the arbitration mechanism in [26] with the deadline-driven swapping proposed in [30]. In particular, this work extends the work in [30], which only addresses EDF scheduling, in a general framework for priority-driven swapping-based scheduling of aperiodic real-time messages over EtherCAT networks, which uniformly covers both dynamic and static priority. The proposed mechanism allows slaves to transmit multiple aperiodic real-time messages in a single EtherCAT frame, while maintaining compatibility with the EtherCAT standard and achieving cycle times in the order of $100\mu s$.

The target is to achieve short cycle times while ensuring the schedulability of aperiodic real-time messages. For this reason, the chapter provides schedulability conditions that enable the network designer to configure the network parameters (e.g., the number of aperiodic telegrams in the EtherCAT frame) so as to avoid deadline miss. The Priority-driven Swapping here proposed is implementable with minor modifications in the EtherCAT protocol state machine and maintains compatibility with EtherCAT standard devices, so there is a clear potential for industrial exploitation.

This chapter is organized as follows. Section 2.1 describes the Priority-driven Swapping approach here proposed and discusses its implementation. Sect. 2.2 presents the timing analysis of the approach, under static and dynamic priorities, respectively. Sect. 2.3 deals with simulative assessments of the Priority-driven Swapping

approach and discusses its performance. Finally, Sect. 2.4 concludes the chapter and gives hints for future work.

2.1 The Priority-driven Swapping

In the Priority-driven Swapping approach here proposed to efficiently support aperiodic traffic under both static and dynamic priority scheduling, a novel telegram has been introduced. The aperiodic telegram, which is shown in Figure 2.2, is contented among the slaves according to a preemptive policy that enables the slaves to send aperiodic messages when needed. This is possible as the daisy-chain topology of EtherCAT allows for message preemption by changing "on-the-fly" the telegram payload of an incoming frame when the frames traverses a slave.

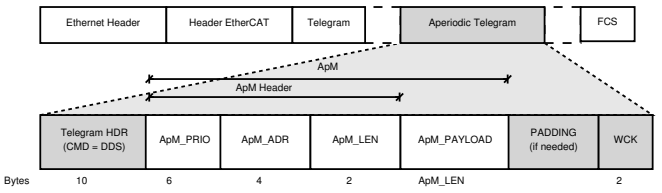

Figure 2.2: Aperiodic Telegram structure

Each slave maintains a local queue of aperiodic messages (ApMs), ordered according to a priority. The network can work under either static or dynamic priority scheduling, but the choice has to be made during the configuration phase, through a suitable setting.

In order to maintain compatibility with the EtherCAT standard, the aperiodic telegram is mapped into the standard EtherCAT telegram as shown in Table 2.1.

The CMD field of the telegram header contains the 0x10 value which indicates an aperiodic telegram, while the ADR field contains the address of the slave that generated the last received ApM. The other fields remain the same as specified in the EtherCAT standard.

Table 2.1: Aperiodic Telegram fields.

Data Field	Data Type	Description/Value
CMD	Unsigned8	Command: (PdS) 0x10
IDX	Unsigned8	Index
ADR	DWORD	Slave address of the last ApM
LEN	Unsigned11	DATA field Length
RESERVED	Unsigned3	0x00
C	Unsigned1	Circulating frame
NEXT	Unsigned1	0 if the last telegram in the frame
IRQ	WORD	Reserved for future use
DATA	OctectString	Data
WKC	WORD	Working Counter

The DATA field contains an ApM, which is the aperiodic message containing the data transmitted by a slave. The ApM is composed of an ApM header and a payload. The header fields are specified as follows:

- ApM_PRIO: the message priority.
- ApM_ADR: the address of the slave that transmitted the ApM.
- ApM_LEN: the ApM payload length in bytes.

The master periodically transmits one EtherCAT frame containing both standard telegrams for the periodic data and one or multiple aperiodic telegrams. A slave, with a ready-to-transmit ApM, upon receiving the aperiodic telegram, stores the local ApM with the highest priority (M_{loc}) in the output buffer (Out_Buf) and the incoming ApM (M_{in}) in the input buffer (In_Buf). When the first byte of M_{in} arrives, the contention starts and the slave compares byte by byte the priorities of M_{loc} and M_{in}. The message with the highest priority value is transmitted, while the other one is stored in the slave local queue.

For priority comparison, due to serial communication, the six bytes of the ApM_PRIO field are encoded and transmitted from the

most to the least significant byte. The comparison works as follows. The i-th byte of the priority of the incoming message M_{in} (i.e., the ApM_PRIO field of the ApM), $B_{i,in}$, for i=0...5, is compared with the corresponding byte of the ApM_DL field of the ApM of the local message M_{loc}, $B_{i,loc}$. The incoming message M_{in} is swapped if the following inequality (2.1) holds (here we assume that the lower the value, the higher the priority):

$$B_{i,loc} < B_{i,in} \tag{2.1}$$

otherwise M_{in} is forwarded to the next slave. No swap occurs if the incoming ApM has the same priority as the local ApM. If according to inequality (2.1) a swap occurs, while the local message M_{loc} is transmitted to the next slave and removed from the local queue, the swapped message has to be entirely received and is then inserted in the local queue according to its priority. Such a mechanism cannot be implemented in the upper layer of the Data Link, as it requires "on-the-fly" processing of the frame.

Compared to the EtherCAT standard [20], the Priority-driven swapping provides the possibility for slaves to transmit aperiodic messages without the need for the master to provide room for each slave with a potential for transmitting, thus reducing the cycle time. In fact, a single aperiodic telegram can be contented among all the slaves that intend to transmit an aperiodic message. If compared to the mechanism proposed in [26], the Priority-driven swapping provides two advantages. First, it allows a slave to embed more aperiodic telegrams into a single EtherCAT frame. This is possible because the slave which has swapped the incoming message will be in charge of transmitting it whenever possible. Second, thanks to this mechanism, a slave which transmitted an ApM can remove it from the local queue right after completing the transmission without waiting for an "acknowledgement" message, as no aperiodic message will be lost due to preemption from other messages.

This is not the case for the CAN-like approach in [26], where a slave must wait for the acknowledgement message sent by the master

before removing the transmitted message from the local queue, so the same slave cannot send more than one aperiodic message in the same telegram. The second advantage compared to [26], which only addresses static priorities in a CAN-like fashion, is that the Priority-driven swapping uniformly covers both dynamic and static priority scheduling, therefore, at the configuration step, the most appropriate scheduling policy for the application at hand can be chosen on a case-by-case basis.

An example of dynamic priority scheduling is the Earliest Deadline First (EDF) algorithm [31], in which the messages with closer absolute deadlines preempt those with less imminent ones. To implement EDF in the Priority-driven swapping approach here proposed, the absolute deadline counts the microseconds elapsed since January 1, 2000 (the date refers to the EtherCAT system time [32]) and a slave generating an ApM calculates the absolute deadline by adding to the system time the relative deadline of the message received from the upper layers. Such a mechanism relies on the clock synchronization, which is also provided with the EtherCAT standard with an accuracy in the order of $100ns$ [32].

2.1.1 Implementing the Priority-driven Swapping

Each EtherCAT slave consists of three layers, i.e.:

- Application layer, which contains the Host Controller. This can be implemented, for instance, in a microcontroller.
- Data Link Layer, which contains the EtherCAT Slave Controller (ESC). This can be implemented either in hardware (FPGA, ASIC) or in software [33].
- Physical Layer, which implements the physical interface to the network.

The Priority-driven Swapping approach requires an additional module in the ESC whose main components, shown in Fig. 2.3, are

a prioritized local queue containing the local ApMs, two buffers for the incoming and outcoming ApMs, and an 8-bit comparator. All these components can be implemented in hardware in the case of FPGA/ASIC ESC. The hardware implementation of the buffers and the comparator is simple and, as far as the prioritized local queue is concerned, in the literature several hardware implementations of prioritized local queues are proposed [34] (e.g., Shift register prioritized queue or Systolic array prioritized queue). The module can be also implemented in software in the case of software ESC, as shown in [33]. Some slight modification in the EtherCAT protocol state machine has also to be added in order to handle the novel telegram type.

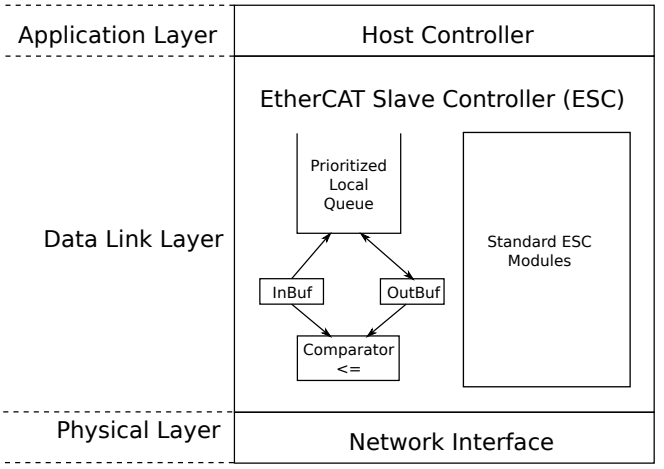

Figure 2.3: Modules implementing the Priority-driven Swapping

2.2 Analytic Assessment

2.2.1 Frame propagation and Timing

In EtherCAT networks, if the data to be embedded in the Ethernet frame exceed the maximum payload size (i.e., 1500 bytes), multiple

Ethernet frames will have to be transmitted by the master to complete a cycle. However, this chapter addresses the case of very short cycle times (e.g., in the order of $100\mu s$), and a maximum payload Ethernet frame has a cycle time of $150\mu s$. Hence, here only a single frame cycle is considered.

Figure 2.4 illustrates the propagation of the Ethernet frame and the related terminology and notations (also summarized in Table 2.2). The Figure shows a scenario with an Ethernet frame that is transmitted by the master to a chain of three slaves, and then goes back to the master according to the daisy chain topology. In Figure 2.4 the master is represented twice to illustrate separately the transmission (top-side) and the reception (bottom-side). The propagation of the Ethernet header and the *Frame Check Sequence* (FCS) field is drawn in light gray.

The Ethernet payload is drawn in two shades of gray (gray/dark gray). In the payload, we highlight in dark gray the EtherCAT telegrams (three in the figure) dedicated to the transmission of aperiodic messages (ApMs). All the notations are described in Table 2.2.

The master periodically sends an Ethernet frame, with period P. Each frame then propagates to the slaves through the network. Slaves are labeled following the frame reception order, so slave 1 is the one that receives the frame first, while slave m is the last. The time elapsing from the transmission of a frame by the master to its reception, as a response [20], is equal to the time needed by the signal to propagate through the medium (T_{pr}), plus the time needed by the m slaves to process the frame (T_{de}).

From Figure 2.4 it is possible to observe that the slaves process the frame "on-the-fly", i.e., each frame starts to be transmitted before it has been fully received from the preceding node. This allows a low end-to-end latency and enables the transmission of an ApM even if the external event generating the ApM arrives during the reception of the Ethernet frame. At any slave, the instant at which the highest priority ApM is transmitted is the start of the reception of the next aperiodic telegram (here called start time). Then, the availability of

Table 2.2: Summary of notation

Symbol	Definition
m	The number of slaves in the chain
ℓ_k	Length of the cable connecting the k-th slave to the $(k+1)$-th slave. ℓ_0 is the length of the cable connecting the master to slave 1, while ℓ_m the length of the cable connecting slave m back to the master.
P	The transmission period of the Ethernet frame.
T_c	The minimum cycle time, i.e., the minimum time taken by all the network nodes to exchange their input/output data once. In Figure2.4, $T_c = T_{et} + T_{ec} + T_{pr} + T_{de} + T_{if}$.
T_{et}	The sum of the transmission times of the Ethernet header and Frame Check Sequence (FCS) fields ($\mathsf{a} + \mathsf{c}$ in Fig. 2.4).
T_{ec}	The time necessary to transmit the EtherCAT frame.
u	The propagation delay per unit of length over the medium (that is 5 ns/m according to the EtherCAT standard).
T_{pr}	The propagation delay over the communication medium that is equal to $T_{pr} = u\sum_{k=0}^{m} \ell_k$.
T_{sv}	the time taken by each slave to process the frame.
Δ_k	the delay from the reception of a frame at slave k to the reception of the same frame at the master, that is $(m-k+1)T_{sv} + u\sum_{i=k}^{m} \ell_i$.
T_{de}	The frame delay that is mT_{sv}.
T_{if}	The inter-frame gap, i.e., the time between the end of the transmission of an Ethernet frame and the start of the transmission of the next one.
p	Number of aperiodic telegrams in the frame ($p = 3$ in Fig. 2.4).
T_{ap}	The time between the start of the transmission of the Ethernet frame and the start of the transmission of the first aperiodic telegram ($\mathsf{a} + \mathsf{b} - p\mathsf{q}$ in Fig. 2.4).
n	The number of ApMs.
π_i	The index of the slave where the i-th ApM is generated.
$\mathsf{pri}(i)$	Priority of the i-th ApM.
T_i	The minimum interarrival time between two consecutive instances of the i-th ApM.
D_i	The relative deadline of the i-th ApM.
S	The time for receiving an aperiodic telegram (q in Fig. 2.4).
A	Time elapsing from the start of the reception of the first aperiodic telegram to the end of the reception of the frame ($pS + \mathsf{c}$ in Fig. 2.4).

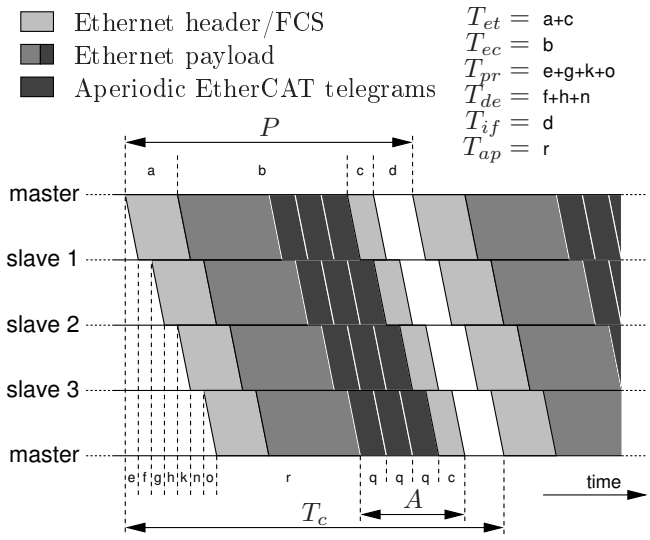

Figure 2.4: EtherCAT frame processing sequence

aperiodic telegrams must be carefully analyzed, as illustrated in the following.

2.2.2 Timing Analysis

In real-time systems, in which the focus is on meeting the deadlines in the worst case, the analysis is made by determining the worst case for both the resource availability and the resource request [35]. In the analysis the same number of aperiodic telegrams for each EtherCAT frame is assumed.

In this context, the resource is the start time of an aperiodic telegram. Hence, the worst case for the resource availability is determined by computing the minimum number $s(t)$ of start times of aperiodic telegrams which may occur in any interval of length t. As illustrated in Fig. 2.5, the worst-case interval (that is the one containing the minimum number of start times of aperiodic telegrams) begins when the last aperiodic telegram has just started. From this

instant (please refer to Fig. 2.5 where the black dots denote the start time of the aperiodic telegrams over time), the time a slave may have to wait before another aperiodic telegram starts is $P-(p-1)S$. Then p aperiodic telegrams will start, spaced S, and the pattern will repeat with period P.

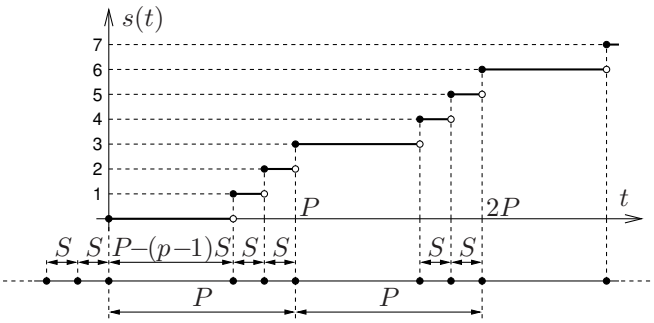

Figure 2.5: Example of the minimum number of start times $s(t)$, with $p = 3$ aperiodic telegrams in a frame.

The formal expression of $s(t)$ can then be written as follows

$$s(t) = \sum_{j=1}^{p} \left\lfloor \frac{t + (j-1)S}{P} \right\rfloor , \tag{2.2}$$

which accounts for the fact that:

- the start times of the p aperiodic telegrams in the same frame are separated by S;
- consecutive aperiodic telegrams at the same position in the frame are separated by P.

In the example shown in Figure 2.5, we have $p = 3$ aperiodic telegrams in a frame.

2.2.3 Response-time analysis

For computing the longest response time of an ApM, we adopt the classic busy-interval approach [36], which was extended to the event-stream task model [37] by Ritcher et al [38]. To apply this method, it is necessary to compute the longest time $w(N)$ a slave has to wait to see N consecutive start times of aperiodic messages. By this definition, $w(N)$ is given by

$$w(N) = \sup\{x \in \mathbb{R} : s(x) < N\}. \tag{2.3}$$

By definition, $w(N)$ then is the longest time interval in which less than N aperiodic telegrams may start.

In the special case of $s(t)$ of (2.2), consecutive aperiodic telegrams in the same frame are separated by S, while the last aperiodic telegram in the frame and the first one in the next frame are separated by $P-(p-1)S$, so that aperiodic telegrams at the same position in the frame are separated exactly by P. As also illustrated in Figure 2.5, this condition implies that $w(N)$ increases by $P-(p-1)S$ when N is a multiple of p, while it increases by S at all other values of N (not multiple of p). Hence $w(N)$ can be written as

$$w(N) = (Q+1)P - (p-1-Z)S \tag{2.4}$$

with Q and Z, respectively, quotient and remainder of the Euclidean division of $N-1$ by p, that is $N-1 = Qp+Z$ with $Z \in \{0, 1, \ldots, p-1\}$. For example, if $p = 3$ (as illustrated in Fig. 2.5) and $N = 1$, then the result of the Euclidean division is $Q = 0$ and $Z = 0$. For this choice, from (2.4) we find $w(1) = P - 2S$, which correctly is the longest separation between two consecutive aperiodic telegrams. If, for example, $N = 3$, then $Q = 0, Z = 2$ and $w(3) = P$. With these definitions in mind, we can compute the response time of an ApM, by applying the classical busy-period approach. First, we define $I_j(t)$ as the largest number of messages of the j-th ApM which can be generated in any interval of length t. For example, in the classical example of sporadic ApM with minimum interarrival time T_j, we have

$I_j(t) = \left\lceil \frac{t}{T_j} \right\rceil$. Then, we compute the largest number N_i of aperiodic telegrams, which may be needed to transmit the i-th ApM, as the smallest fixed point of the following iteration

$$\begin{cases} N_i^{(0)} & = 1 \\ N_i^{(k+1)} & = 1 + \sum_{\mathsf{pri}(j)>\mathsf{pri}(i)} I_j(w(N_i^{(k)})) \\ & \quad + \sum_{\substack{\mathsf{pri}(j)=\mathsf{pri}(i) \\ \pi_j<\pi_i}} I_j(w(N_i^{(k)})) \\ & \quad + \sum_{\substack{\mathsf{pri}(j)=\mathsf{pri}(i) \\ \pi_j=\pi_i, j\neq i}} 1. \end{cases} \tag{2.5}$$

The expression of $N_i^{(k+1)}$ accounts for the fact that in the time interval $w(N_i^{(k)})$ we must consider the following possible sources of interference:

- the ApMs with higher priority, represented by the first sum with j such that $\mathsf{pri}(j) > \mathsf{pri}(i)$;
- the ApMs with the same priority as the i-th, but generated at preceding slaves (as described earlier, an incoming ApM with the same priority of the ApM at the local slave is not swapped), represented by the sum over $\mathsf{pri}(j) = \mathsf{pri}(i), \pi_j < \pi_i$;
- the ApMs at the same slave with the same priority, which, however, may interfere only once, since ApMs with the same priority are scheduled FIFO within the same slave (the last term in the sum of (2.5)).

Note that, if messages have minimum interarrival time T_j, the above described iterative definition is proved to converge [36] if the maximum number of needed aperiodic telegrams is less than the available ones, that is

$$\sum_{\mathsf{pri}(j)>\mathsf{pri}(i)} \frac{1}{T_j} < \frac{p}{P}. \tag{2.6}$$

Once N_i is computed, the response time (R_i) of the i-th ApM is

$$R_i = \Delta_{\pi_i} + w(N_i) + A \tag{2.7}$$

with the following interpretation:

- Δ_{π_i} is the maximum time from the slave π_i to the master;
- $w(N_i)$ is the time the i-th ApM may have to wait to have one telegram available;
- A is the time from the reception of the aperiodic telegram at the master to the instant the message is actually read.

2.2.4 EDF analysis

If the ApMs have a deadline D_i between their release at the slave and the delivery time at the master, and ApM priorities are assigned based on EDF, so it is then possible to tighten the analysis by adapting the classic EDF guarantee test [39].

Theorem 1. *A set of n aperiodic messages (ApMs), each one triggered by events with minimum interarrival time T_i are guaranteed to be received at the master within D_i from their release time when scheduled by EDF, if:*

$$\forall t > 0, \quad \sum_{i=1}^{n} \max\left(0, \left\lfloor \frac{t - (D_i - \Delta_{\pi_i} - A)}{T_i} \right\rfloor + 1\right) \leq s(t). \qquad (2.8)$$

Proof. Since the priority of ApMs is assigned according to EDF, we adapt the guarantee test based on the demand bound function [39]. Differently than a classic CPU scheduling problem, the resource to be scheduled is not the CPU time but rather the aperiodic telegrams. As also illustrated by the dark gray band in Figure 2.4, the same aperiodic telegram is available at different times for different slaves. To remove this slave-dependent time shift, we set the common reference time at the reception side of the master (illustrated as the horizontal line at the bottom of Figure 2.4). The arrival of any ApM at time t at slave k can be equivalently represented by the arrival of the same message at time $t + \Delta_k$ at the master. Then, if the i-th ApM has

deadline D_i from the arrival at slave π_i to the reception at the master, it can be equivalently represented by a message arriving directly at the master with deadline $D_i - \Delta_{\pi_i}$. In addition, the assumption that the latest time to transmit the ApMs is at the start of the reception of the incoming aperiodic telegram, while the reading of the aperiodic message is made only at the end of the reception of the entire frame at the master, implies that all ApM deadlines must be decremented by an amount $A = T_{et} + T_{ec} - T_{ap}$ (also represented in Figure 2.4 by $p\mathsf{q} + \mathsf{c}$). In conclusion, the demand bound function [39], which in this case is *the largest possible number of ApMs with arrival properly translated to the reference time at the master and deadline within an interval of length* t, is given by the following expression:

$$\sum_{i=1}^{n} \max\left(0, \left\lfloor \frac{t - (D_i - \Delta_{\pi_i} - A)}{T_i} \right\rfloor + 1\right) \tag{2.9}$$

Since the minimum possible number of aperiodic telegrams available in any interval of length t is represented by $s(t)$, then Condition (2.8) holds, because EDF can schedule any task set for which the demand bound function does not exceed the available resource for all $t \geq 0$. Hence, the Theorem is proved. □

Condition (2.8) also implies the necessary condition

$$\sum_{i=1}^{n} \frac{1}{T_i} \leq \frac{p}{P} \tag{2.10}$$

which is equivalent to the classic necessary condition of a non-overloaded processor in CPU scheduling problems.

Theorem 1 provides a test which is not practical. In the next Lemma, following the idea by Baruah et al [40], we reduce the set in which the inequality of (2.8) needs to be tested to a finite one.

Lemma 1. *A set of n aperiodic messages (ApMs), each one triggered by events with minimum interarrival time T_i are guaranteed to be received at the master within D_i from their release time when scheduled*

by EDF, if:

$$\sum_{i=1}^{n} \frac{1}{T_i} < \frac{p}{P} \tag{2.11}$$

and

$$\forall t \in \mathcal{D}, \quad \sum_{i=1}^{n} \max\left(0, \left\lfloor \frac{t-\phi_i}{T_i} \right\rfloor\right) \leq s(t) \tag{2.12}$$

with

$$\phi_i = D_i - \Delta_{\pi_i} - A - T_i \tag{2.13}$$

$$L^* = \max_{\ell=0,\ldots,n} \frac{\frac{p}{P}(P-(p-1)S) - \sum_{i=1}^{\ell} \frac{\phi_i}{T_i}}{\frac{p}{P} - \sum_{i=1}^{\ell} \frac{1}{T_i}} \tag{2.14}$$

$$\mathcal{D} = \{d : d = \phi_i + kT_i, i = 1, \ldots, n, k \in \mathbb{N}, d < L^*\}, \tag{2.15}$$

and assuming that ApMs are sorted by an increasing value of ϕ_i.

Proof. We observe that $s(t)$ of (2.2) can be lower bounded as follows

$$\begin{aligned} s(t) \geq & \sum_{j=1}^{p} \left(\frac{t+(j-1)S}{P} - 1 \right) = \frac{p}{P}(t-P) + \frac{S}{P} \sum_{j=1}^{p} (j-1) \\ = & \frac{p}{P}(t-P) + \frac{S}{P} p(p-1) = \frac{p}{P}(t-(P-(p-1)S)). \end{aligned}$$

The left-hand side (LHS) of (2.8), which we denote for simplicity $\mathsf{dbf}(t)$ to recall the *demand bound function* of EDF scheduling [39], can be upper bounded as follows

$$\mathsf{dbf}(t) \leq \sum_{i=1}^{n} \max\left(0, \frac{t-\phi_i}{T_i}\right) = \max_{\ell=0,\ldots,n} \sum_{i=1}^{\ell} \frac{t-\phi_i}{T_i},$$

since the ordering $\phi_1 \leq \phi_2 \leq \ldots \leq \phi_n$ holds. From the lower bound of $s(t)$ and the upper bound of $\mathsf{dbt}(t)$, it follows that $\mathsf{dbt}(t) \leq s(t)$

for all t satisfying the next condition

$$\mathsf{dbf}(t) \leq \max_{\ell=0,\ldots,n} \sum_{i=1}^{\ell} \frac{t-\phi_i}{T_i} \leq \frac{p}{P}(t-(P-(p-1)S)) \leq s(t)$$

$$\forall \ell = 0,\ldots,n, \quad \sum_{i=1}^{\ell} \frac{t-\phi_i}{T_i} \leq \frac{p}{P}(t-(P-(p-1)S))$$

$$\forall \ell = 0,\ldots,n, \ \frac{p}{P}(P-(p-1)S) - \sum_{i=1}^{\ell} \frac{\phi_i}{T_i} \leq t\left(\frac{p}{P} - \sum_{i=1}^{\ell} \frac{1}{T_i}\right)$$

$$\forall \ell = 0,\ldots,n, \ t \geq \frac{\frac{p}{P}(P-(p-1)S) - \sum_{i=1}^{\ell} \frac{\phi_i}{T_i}}{\frac{p}{P} - \sum_{i=1}^{\ell} \frac{1}{T_i}}$$

$$t \geq \max_{\ell=0,\ldots,n} \frac{\frac{p}{P}(P-(p-1)S) - \sum_{i=1}^{\ell} \frac{\phi_i}{T_i}}{\frac{p}{P} - \sum_{i=1}^{\ell} \frac{1}{T_i}} = L^*.$$

Note that dividing LHS and right-hand side (RHS) by $\frac{p}{P} - \sum_{i=1}^{\ell} \frac{1}{T_i}$ is possible, thanks to the hypothesis of (2.10).

Then $\forall t \geq L^*$ is always $\mathsf{dbf}(t) \leq s(t)$. If, instead, $t < L^*$ the condition (2.8) needs to be explicitly checked. However, it is sufficient to check it only at the discontinuity of the LHS, which are the absolute deadlines of all ApMs not larger than L^*, all contained in $\mathcal{D}$. □

2.3 Simulative assessments

2.3.1 Simulation Model Assessment

To assess the performance of the proposed approach, a suitable simulation model was developed using the OMNeT++ framework. In the simulation model two kinds of nodes are implemented, i.e., the EtherCAT Master and the EtherCAT Slave. The EtherCAT Master is composed of a *MasterDLL* module and a *MasterPHY* module. The first module periodically generates Ethernet frames and collects statistics. The MasterPHY transmits the frame in one-byte long

packets and transmits each byte every $0.08\mu s$ (i.e., the byte time of the 100Mb/s Ethernet). In this way, the timing of the simulation model is compliant with that of the EtherCAT standard. The EtherCAT Slave module provides several functionalities. Among them, the EtherCAT DLL, which supports both the periodic telegrams foreseen by the standard and the aperiodic telegrams of the proposed Priority-driven swapping approach, the forwarding mechanisms for incoming packets, the read/write and management functions of the ApMs local queue, and the slave Application layer.

The simulation model was assessed by comparing the EtherCAT timing parameters calculated as in Sect. 2.2 with those obtained in the simulation. In particular, for the simulations, a typical networked control system for motion control was simulated. The system consists of one controller (the master), 5 devices (the slaves), 6 joints and 4 wheels. Three slaves are in charge of 2 axes each, and the other two slaves manage 4 wheels (2 wheels for each slave). For the joint control 48-byte data is cyclically exchanged with a period of $50\mu s$, a realistic value for these applications [41]. The wheels are controlled in an event-triggered way, so the traffic is sporadic (i.e., characterized by a minimum interarrival time). The data transmission period for the wheel control is $500\mu s$ [42] and data size at the application layer is 32 bytes. The wheel control traffic is randomly generated between $500\mu s$ and $1000\mu s$ using a uniform distribution. The relative deadline D_i is chosen equal to the minimum interarrival time (i.e., $500\mu s$). The slaves may also transmit event notification messages characterized by $T_i = 1000\mu s$ and $D_i = 1000\mu s$. The distance between two consecutive network nodes is 2m, so the overall distance covered by the Ethernet frames is 20m. The relevant simulation parameters and timing are summarized in Table 2.3.

In the simulation, the slaves 1 and 2 manage the wheels, while the slaves 4,5, and 6 are in charge of the 6 joints. Table 2.4 shows the ApMs parameters. An ApM set indicates the ApMs with the same interarrival time T_i and the same relative deadline D_i, and Δ_k is the delay from the reception of a frame at slave k to the reception

Table 2.3: Simulation I - Network parameters and timings

Net. params	Value/range
T_{sv}	$1\mu s$
T_{pr}	$50ns$
T_c	$46.33\mu s$
P	$41.28\mu s$
A	$3.84\mu s$
p	1
Payload size of an aperiodic telegram	44 bytes
Number of periodic telegrams	7
Payload size of a periodic telegram	48 bytes

of the same frame at the master (as defined in Table 2.2). With these parameters, the conditions of Lemma 1 are met. Therefore, the scheduling of the ApM set is feasible.

Table 2.4: Sporadic Traffic parameters of simulation I

ApMs Set	Slave π_i	T_i	D_i	Δ_k
1	1	$500\mu s$	$500\mu s$	$5.040\mu s$
1	2	$500\mu s$	$500\mu s$	$4.030\mu s$
2	1	$1000\mu s$	$1000\mu s$	$5.040\mu s$
2	2	$1000\mu s$	$1000\mu s$	$4.030\mu s$
2	3	$1000\mu s$	$1000\mu s$	$3.020\mu s$
2	4	$1000\mu s$	$1000\mu s$	$2.010\mu s$
2	5	$1000\mu s$	$1000\mu s$	$1.000\mu s$

A simulation run of 10s, corresponding to a generation of about 20000 ApMs, was performed and simulations were repeated 5 times varying the seed of the random generators. The calculated cycle time is $46.33\mu s$ as required from the application. Results show that the response time values obtained by the simulation match the ones calculated using the analysis described in Section 2.2. In Figure 2.6 the Cumulative Percentage Distribution of the message response times, defined as the percentage of ApMs with the response time lower than a given response time value (i.e., the value of x-axis), is drawn.

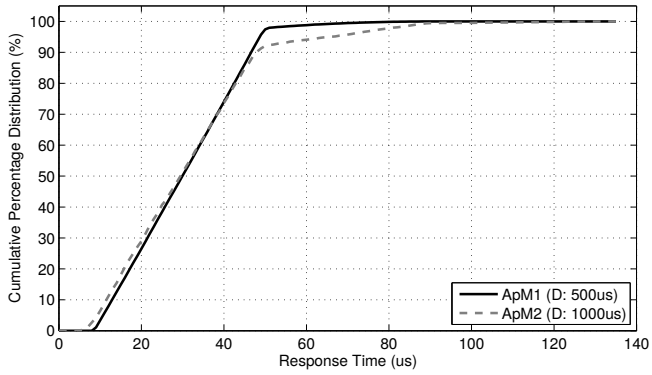

Figure 2.6: Cumulative Percentage Distribution of the ApM Response Times.

This corresponds to the results of the analysis that proved the feasibility of the considered ApM set (i.e., that the ApM response times are always lower than their relative deadlines) in the addressed scenario. The same simulation was performed using static priorities. Results also match with the values obtained with the analysis in Sect. 2.2.3.

2.3.2 Cycle Time assessment

To assess the cycle times which can be reached using the proposed approach and to compare it with the standard EtherCAT, a set of simulations were performed. The goal of these simulations is to find the minimum number of aperiodic telegrams required to transmit the ApMs while maintaining the Deadline Miss Ratio (DMR) (i.e., the number of deadline misses over the number of generated ApMs) lower than 0.1%.

Each slave generated ApMs with three different priorities (or deadlines in the case of EDF scheduling). ApMs were generated with exponentially distributed random generation periods with a given mean (λ), while the ApMs deadlines were uniformly distributed. Ta-

ble 2.5 shows the simulation parameters. Lambda is given as an interval because for each simulation run different lambda values were used. This choice was made in order to evaluate the protocol performance with varying mean generation periods.

To have realistic frame sizes, 20 periodic telegrams were transmitted, in addition to the aperiodic telegrams, every cycle. This entails an increase of $70.4\mu s$ in the cycle time. Each simulation run was repeated 5 times varying the seed. In each simulation the simulated time was chosen to collect statistics over 50000 ApMs.

Table 2.5: Parameters of Simulation II

Parameters	**Values/Range**
Number of slaves	10
Number of periodic telegrams	20
Payload size of a periodic telegram	32 bytes
Number of aperiodic telegrams (p)	from 1 to 8
Payload size of an aperiodic telegram	32 bytes
ApMs deadlines	$300\mu s, 600\mu s, 900\mu s$
λ	from $186\mu s$ to $1515\mu s$
Repetitions	5 repetitions varying the seed

Increasing the number of aperiodic telegrams embedded in the EtherCAT frame entails an increase in the cycle time. For the simulated scenario, the cycle time values as a function of the number of aperiodic telegrams are shown in Table 2.6.

Table 2.6: Cycle Time as a function of the number of aperiodic telegrams in the EtherCAT frame

p	T_c	p	T_c
1	$87.62\mu s$	5	$101.70\mu s$
2	$91.14\mu s$	6	$105.22\mu s$
3	$94.66\mu s$	7	$108.74\mu s$
4	$98.18\mu s$	8	$112.26\mu s$

Figure 2.7 compares the simulation results obtained by the standard EtherCAT, the Priority-driven Swapping with dynamic priori-

ties (PdS-EDF), and the Priority-driven Swapping with static priorities (PdS-SP).

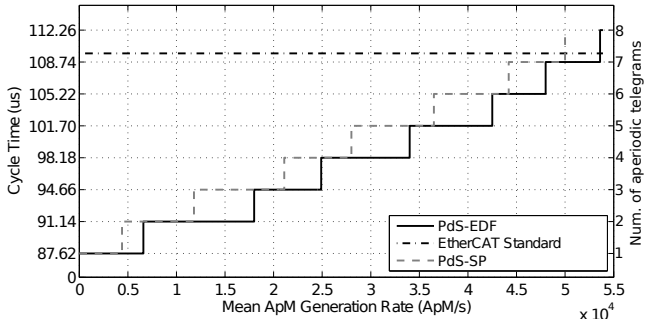

Figure 2.7: Cycle Time vs. Mean ApM Generation Rate

In the EtherCAT standard simulation, one periodic telegram with 20 bytes of payload is transmitted by each slave, so the cycle time is constant in Figure 2.7 (i.e., 109.70μs). Conversely, in the PdS-EDF and in the PdS-SP the cycle time varies as a function of the aperiodic workload. This is an advantage of the two mechanisms here proposed over the EtherCAT standard. In fact, while the latter provides a constant cycle time, that is higher than 109 us in Figure 2.7, independently of the actual aperiodic workload, the two Priority-driven Swapping approaches obtain a cycle time that depends on the aperiodic workload. As a result, under low aperiodic workloads, the cycle time is also low. As shown in Figure 2.7, the two Priority-driven Swapping approaches experience a cycle time value comparable with that of the EtherCAT standard only under a high aperiodic workload, i.e., 53700 aperiodic messages per second. The Priority-driven Swapping (PdS) shows the same trend with both static (PdS-SP) and dynamic priorities (PdS-EDF), with the difference that with the same number of aperiodic telegrams embedded in one EtherCAT frame, and so with the same cycle time, the PdS-EDF supports a higher generation rate for aperiodic real-time messages than the PdS-SP. This is because in the PdS-SP the low-priority messages experience

longer delays due to the interference from the messages with higher priority. We highlight that in both the Priority-driven Swapping approaches the number of ApMs in the slave queue is always lower than 10 ApMs, so the local queues were never saturated. Summarizing, the Priority-driven Swapping offers better performance than the EtherCAT standard in terms of reduction of the cycle time for the aperiodic traffic.

2.3.3 Comparison with the CAN-Like

In this simulation a comparison between the Priority-driven Swapping with another approach proposed in the literature [26], here called *¨CAN-Like¨*, is performed. In particular, the response times in a defined scenario are compared.

In the CAN-Like an aperiodic telegram, called *MARB*, is embedded in the EtherCAT frame. Slaves transmit aperiodic messages in the MARB using a ¨bytewise¨ arbitration (similar to the CAN protocol). The CAN-Like approach allows the possibility to transmit multiple aperiodic messages in a MARB (but only one for each slave). Moreover, as the CAN-Like does not provide any swapping mechanism, it requires an additional telegram to inform the slaves about the received messages (i.e., the MDBT Telegram). So, if the slave messages are received by the master, then the slaves can remove the pending messages from their local queue. This prevents the master from embedding multiple MARBs in the same EtherCAT frame and also prevents the slaves from transmitting multiple aperiodic messages within a single MARB.

In order to compare the two protocols a network with 10 slaves was simulated. Each slave generates aperiodic messages according to an exponential distribution with mean λ. A simulation was performed using static priorities. The priority of each message was assigned randomly with a uniform distribution between three ranges, as shown in Table 2.7. As the CAN-Like approach does not provide the possibility to generate messages with the same priorities (as the

priority identifies the message), the index of each slave is subtracted to the priority, so different slaves generate messages with different priority (e.g., slave 10 subtracts 10 to the priority of its messages). The same assignment has been used for the Priority-driven Swapping here proposed. In Table 2.7 the parameters of this simulation are shown.

Table 2.7: Parameters of Simulation III

Parameters	**Values/Range**
Number of slaves	10
Number of periodic teleg.	20
Payload size (periodic teleg.)	32 bytes
Number of aperiodic teleg. (p)	4 (PdS), 1 (CAN-Like)
Payload size (aperiodic teleg.)	32 bytes (PdS), 50 bytes (CAN-Like)
Repetitions	5 repetitions varying the seed
λ	$1162\mu s$ (i.e., 8600 messages/s)
Aperiodic message priorities	High:$[600, 609]\ \mu s$, Medium:$[900, 909]\ \mu s$, Low:$[1200, 1209]\ \mu s$

A tuning of the MARB length (for CAN-Like) and of the number of aperiodic telegrams (for PdS) was performed in order to choose the best simulation parameters regarding the response times. For the CAN-Like a MARB of 50 bytes was set, so that 2 aperiodic messages per cycle can be transmitted (increasing the MARB size does not result in better performance). The mean generation period for the aperiodic messages is $1162\mu s$, which corresponds to a workload of 8600 messages/s, so as not to saturate the network. The Cumulative Percentage Distribution of response times is shown in Figure 2.8.

Taking into account 80% of the aperiodic messages, the Priority-driven Swapping (PdS) obtains response times lower than $100\mu s$, while the response times obtained by the CAN-Like are higher than $100\mu s$, but remain lower than $200\mu s$. Considering the whole set of messages, the maximum response time for the highest priority messages is $214\mu s$ for the PdS and $532\mu s$ for the CAN-Like, while the maximum response time for the lowest priority messages is $406\mu s$ for

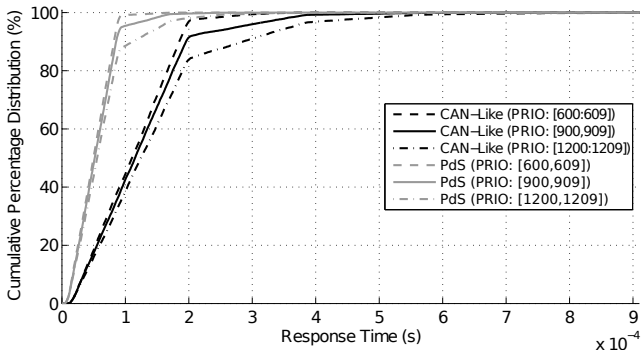

Figure 2.8: Cumulative Percentage Distribution of Response Times (CAN-Like vs. PdS)

the PdS and 879μs for the CAN-Like. The Priority-driven Swapping provides lower response time than the CAN-Like thanks to the capability for the same slave to transmit multiple aperiodic telegrams in the same EtherCAT frame.

2.4 Conclusions

In this work a Priority-driven swapping mechanism to deal with the problem of providing support for aperiodic real-time traffic over EtherCAT networks was presented. The proposed mechanism provides the possibility for slaves to transmit aperiodic real-time messages under static and dynamic priorities, respectively, while maintaining the compatibility with the EtherCAT standard. The proposed approach is suitable for event-triggered control applications, as the aperiodic messages can be transmitted while maintaining short cycle times (i.e., in the order of 100μs). This work proposed a general analysis for the Priority-driven scheduling based on response times which can be used to assess the feasibility of a static priority message set. As far as dynamic priorities are concerned, the case of EDF scheduling was analytically assessed and the relevant schedulability

conditions were derived. An extensive simulative assessments performed through the OMNeT++ framework is also provided. Future works will address the possibility to embed multiple ApMs within a single aperiodic telegram so as to further reduce the cycle time.

Chapter 3

Simulative assessments of the IEEE 802.15.4e DSME and TSCH protocols

Industrial Wireless Sensor Networks (IWSN) consist of a number of sensor devices that gather data from the physical environment and send it to controllers. IWSNs are found in many application domains, such as factory automation, distributed process control, real-time monitoring, and so on. Typical requirements for IWSNs are low latency, robustness and determinism for transmission of small-size packets [43, 44]. In recent years, many standard protocols to support IWSNs in different application domains, i.e., WirelessHART [45], ISA100.11a [46], Bluetooth [47], IEEE 802.15.4e [48] appeared and were evaluated in different industrial automation scenarios [49–55]. The IEEE 802.15.4 [56] standard proved to be unable to address the typical industrial application requirements. For this reason, in 2012, an amendment was introduced, i.e., the IEEE 802.15.4e standard, which offers new MAC-layer profiles that are optimized for several typical automation domain, including process automation. The IEEE 802.15.4e standard presents three profiles, i.e., the Low Latency Deterministic Network (LLDN), the Deterministic and Synchronous

Multi-channel Extension (DSME) and the Time Slotted Channel Hopping (TSCH). In [48] a set of reference application domains is defined for each profile. LLDN targets very low-latency applications, such as those commonly found in robotics, automotive manufacturing and milling machines. DSME is designed for industrial, commercial and healthcare applications, such as process automation, smart metering, and telemedicine. TSCH is intended for process automation applications. Both the DSME and the TSCH are suitable for process automation applications. Unlike other industrial protocols (e.g., WirelessHART and ISA100.11a), the protocols defined in the IEEE 802.15.4e amendment [48] are interoperable with the original IEEE 802.15.4 standard [56], while they provide the real-time behavior required for industrial applications. For this reason, this chapter focuses on these two profiles with a twofold aim. First, to assess their behavior in realistic process automation scenarios, highlighting their strong and weak points. Second, to compare their performance in terms of delay, reliability and scalability. One goal of the work is identifying which of the two protocols is the best choice in given operating scenarios based on some parameters such as, the number of nodes, the channel noise, and so on. Moreover, the chapter also intends to highlight the limits of those protocols, thus paving the way to the design of novel approaches able to solve them.

This chapter is organized as follows. Section 3.1 presents related work, while Section 3.2 provides an overview of the profiles defined in the IEEE 802.15.4e amendment. Section 3.3 describes the realistic application scenarios and presents and discuss the results obtained. Finally, Section 3.4 gives the conclusions and outlines future work.

3.1 Related Work

The IEEE 802.15.4e [48] amendment provides support to apply the IEEE 802.15.4 standard to application domains that require specific properties, such as reliability, low latency and scalability. Several

works in the literature deal with the IEEE 802.15.4e, but most of them focus on only one of the three profiles provided in the amendment [48]. In [43] the performance of the LLDN protocol are evaluated in a simple scenario and an extension of the standard retransmission mode is also presented. The work considers the LLDN protocol only. In [57] a simulative assessment of an improved version of the IEEE 802.15.4 standard is presented. Although in the paper such an extension is called IEEE 802.15.4e protocol, it is just an early version of the LLDN protocol. The works in [58] and [59] propose some improvements to the LLDN protocol. The first one suggests to use redundant transmissions in IEEE 802.15.4e network to reduce the effect of packet loss. The second provides a two-level network in which several sub-networks operate at the same time on different channels, thus enhancing the LLDN scalability. In [60] the performance, in terms of packet delivery ratio and delay, of the DSME are evaluated through simulations for a wireless hospital room scenario. As in the amendment [48] e-health and patient monitoring are mentioned as reference applications for the DSME protocol, such scenarios are not suitable for a comparative evaluation between the DSME and the other two profiles described in the standard [48]. In [61] the performance of the DSME protocol in the presence of IEEE 802.11 interference are assessed. The outcome is that the DSME protocol is more reliable, under WLAN interference, than the IEEE 802.15.4. In [62] and [63] the performance of DSME are compared, through simulations, to those of the IEEE 802.15.4 protocol, in terms of energy consumption and throughput, respectively. In [62] a novel energy efficient implementation of the DSME protocol (called ELPIDA) is presented and compared with the native DSME approach and the IEEE 802.15.4 protocol. The results proved that the power consumption of DSME or ELPIDA is always lower than that of the IEEE 802.15.4 protocol. Although the chosen simulation scenario of [62] and [63] is typical of a process automation applications, and so also TSCH protocol would have been a suitable candidate, both works do not address comparative assessment with the TSCH. The focus of the work in [64] is the

formation process of TSCH networks, especially as far as the joining time (i.e., the total time taken by a new node to join the network) is concerned, but nothing is said about the DSME and LLDN. In the work it is proved that using more channel offsets significantly reduces the joining time, but may be beneficial only when the network density is extremely high. In [65] the advantages offered by the TSCH protocol are compared to the features provided by the WirelessHART standard, while in [66] the performance of coordinated sampled listening (CSL), which is a feature presented in [48] that foresees that a transmitter node sends wakeup frames before sending a data, are investigated and compared to that of TSCH.

Comparing with related works, this work provides a twofold contribution. First, a performance evaluation of the DSME and TSCH protocols under a realistic process automation scenario based on [44] and [67], with the aim of addressing both scalability and reliability. In particular, we assessed the effect of retransmissions on the end-to-end delay of the two protocols and their reliability under different channel propagation parameters. Moreover, scalability was assessed evaluating the impact on the end-to-end delay of increasing the number of network nodes. Second, a comparative evaluation of the two protocols to highlight the strengths and the weaknesses of each of them. To the best of our knowledge there is no previous work that presents a comparative assessments of the DSME and the TSCH protocols in a realistic process automation scenario. The LLDN is not addressed in this work because, as specified in [48], it is not designed for process automation applications.

3.2 Overview of the 802.15.4e standard

In this Section an overview of the IEEE 802.15.4e amendment is presented. About the LLDN protocol just a brief introduction is provided, as this work focuses in on the TSCH and DSME protocols only.

3.2.1 Low Latency Determinstc Network (LLDN)

The IEEE 802.15.4e-LLDN protocol is specifically devised for industrial applications requiring low latency, such as those found in manufacturing, in robotics, etc. According to the LLDN profile, time is divided in superframes (SFs), that repeat one after the other in a regular way. A SF consists of several timeslots and each node has assigned one or multiple timeslots in which it is allowed to transmit. Each timeslot is large enough to accommodate the transmission of one packet. The LLDN protocol provides a star topology in which nodes transmit using a Time Division Multiple Access (TDMA) mechanism. According to the LLDN protocol the network is managed by the PAN coordinator, which is responsible for the network configuration and node synchronization.

3.2.2 Deterministic and Synchronous Multi-channel Extension (DSME) protocol

The IEEE 802.15.4e-DSME protocol [48] is designed for meshed networks and allows for a very efficient allocation of the available resources by exploiting both time and frequency multiplexing. The DSME extends to 15 the number of Guaranteed Time Slots (GTS) per superframe that was provided by the IEEE 802.15.4. One additional slot is used for sending the beacon frame.

The medium access is based on a specific time structure, called multi-superframe (multi-SF). Each multi-SF consists of a set of SFs and each SF consists of several consecutive timeslots. Each SF is composed of three parts: a) An Enhanced Beacon (EB); b) The Contention Access Period (CAP), c) The Contention-Free Period (CFP). During the CAP, that is made up of at most 8 timeslots in which monitoring periodic data, urgent or non-periodic data can be sent, the network devices compete for channel access using a slotted CSMA/CA. The CAP is mandatory just for the first SF in a multi-SF. In fact, the *CAP reduction* flag can be enabled to allow all the

superframes (but the first one) to skip the CAP. A beacon timeslot is scheduled at the beginning of each SF. Such a timeslot is used by the PAN coordinator to send the beacon frame in the first SF only. The other beacon timeslots are used by the other nodes that expect to receive packets, i.e., *coordinator nodes*.

The CFP is made up of a set of multi-channel GTS (called DSME-GTSs), which are used for communication between two devices and are characterized by pair-wise assignments of channels and time slots. The CFP follows immediately after the CAP and extends to the end of the superframe. A device in a DSME network can play one of the roles of PAN coordinator, coordinator or end node.

- The PAN coordinator usually is the sink node of the network and sends EB every beacon interval (BI). There is one PAN coordinator for each network. The EB sent by the PAN coordinator contains a specific DSME PAN descriptor, which includes information about time (to synchronize the network nodes), channel hopping and timeslots.

- The coordinator is the sink node for some of the network nodes. A coordinator sends a beacon at least once per multi-SF, in the beacon slot, in order to declare its presence in the network. It acts as a relay for the nodes that cannot reach the PAN coordinator directly. In the same network multiple coordinators nodes are allowed.

- End nodes produce data and send it to the relevant coordinator (or to the PAN coordinator, if it is close enough). After receiving a beacon, an end node can compete for transmitting the data during the CAP or ask for a DSME-GTS.

The duration of the SFs and the structure of the multi-SF depends on the following parameters:

- The MultiSuperframe Order (MO), which describes the multi-SF length.

- The Beacon Order (BO), which indicates the number of SFs any coordinator has to wait to transmit a beacon.
- The Superframe Order (SO), which represents the length of the active portion of the superframe, which is equal to the superframe length as the standard [48] does not foresee an inactive portion in the SF.
- Beacon Interval (BI), which is the time interval between two consecutive Enhanced Beacon frames sent by the PAN coordinator.

The number of superframes in a multi-SF is given by formula (3.1):

$$numberOfsuperframes = 2^{(MO-SO)}. \tag{3.1}$$

The number of multi-superframes in a beacon interval can be obtained as in formula (3.2):

$$numberOfMultisuperframes = 2^{(BO-MO)}. \tag{3.2}$$

The superframe duration (SD), in symbols (in the 2.45GHz PHY Layer the standard specifies that 62500 QPSK-symbols per second are transmitted), is given in formula (3.3):

$$SD = aBaseSuperframeDuration * 2^{SO} \tag{3.3}$$

where $aBaseSuperframeDuration$ is the number of symbols forming a superframe when SO is equal to 0. The multi-superframe duration (MD), in symbols, is given in formula (3.4):

$$MD = aBaseSuperframeDuration * 2^{MO}. \tag{3.4}$$

The BI duration, in symbols, is shown in formula (3.5):

$$BI = aBaseSuperframeDuration * 2^{BO}. \tag{3.5}$$

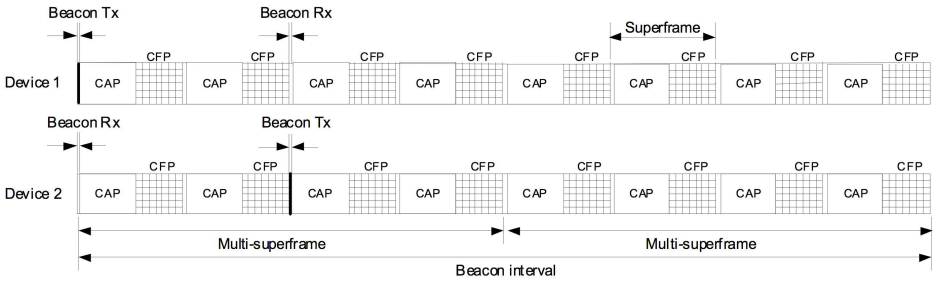

Figure 3.1: The DSME multi-SF structure shown in the IEEE 802.15.4e [48].

The value of MO, SO, and superframe duration, SD, are related as in formula (3.6):

$$0 \leq SO \leq BO \leq 14. \tag{3.6}$$

In Figure 3.1 the structure of two consecutive multi-SFs, seen from the perspective of two generic coordinator devices, named Device 1 and Device2, is shown. The Device2 receives the EB during the beacon timeslot in the first SF and then is able to send an EB itself during the beacon timeslot in the third SF. During a BI, multiple multi-SFs can be scheduled. The multi-SF structure is defined by the coordinators, which periodically transmit an EB with the DSME PAN descriptor.

The DSME implements channel hopping. The series of channels used at each slot is referred as the hopping sequence. The transmitter switches to the channel used by the receiver in order to send a data frame. If the receiver successfully receives the data frame, it sends an acknowledgement frame (ACK) to the transmitter on the same channel. Group ACK are also foreseen by the standard, as follows. A coordinator allocates two timeslots for acknowledgement purposes. The first, here called TACK1, is used to acknowledge all the data frames received from the first GTS in the superframe to the TACK1 timeslot, while the second, called TACK2, is used to acknowledge all the data frames received after the TACK1 frame, but before the

transmission of the TACK2 frame.

The maximum number of devices that can act as a receiver, including the PAN coordinator, is given by the total number of SFs in the Beacon Interval (BI), corresponding to the number of available beacon slots.

3.2.3 Time Slotted Channel Hopping (TSCH)

The IEEE 802.15.4e-TSCH is suitable for multi-hop networks where multi-channel communication allows for an efficient use of the available resources [64]. Nodes synchronize periodically, on a periodic SF that consists of a sequence of contiguous timeslots. In each timeslot a node is allowed to send a maximum-size data frame and receives the relevant acknowledgement (ACK). In order for the PAN coordinator to advertise the presence of the network, an Enhanced Beacon (EB) is sent. The kind of information contained in the EB is the same as the one in the EB of the DSME, described in the previous subsection.

In a TSCH network, the superframe concept is replaced with that of slotframe. The slotframe contains either contention-based or contention-free timeslots. The main difference between the slotframe and the superframe is that the network nodes are supposed to share a common notion of time, so the slotframe automatically repeats without requiring beacon frames to initiate communications. The timeslots assignment to the devices within the slotframe may initially be communicated through a beacon, but usually they are configured by the higher network layers when the device joins the network. In Fig. 3.2 an example of TSCH slotframe structure is shown. In this example, among the five timeslots of each slotframe, only three are assigned, to node A, B and C respectively, while the remaining two timeslots are empty.

In TSCH all the transmissions are direct. If a single transmission attempt fails or the ACK is not received within a predefined timeout, the data frame is deferred to the next time slot assigned to the same <*sender, destination*> pair of nodes. The device can retry data

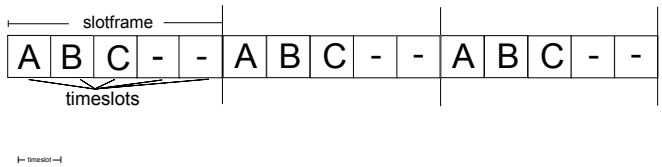

Figure 3.2: The TSCH slotframe structure.

transmission and waits for the ACK up to a given maximum number of times.

The TSCH combines time slotted access with multi-channel transmission and channel hopping capabilities. The hopping sequence is defined by an ID, the sequence length and an ordered list of channels. This default sequence is a pseudo-random list of all the channels available at the physical layer. A link between communicating devices is represented by a pair *<timeslot, channel offset>*. Shared links are intentionally assigned to multiple devices for transmission. This can lead to collisions and result in a transmission failure detected by a missing ACK. In order to reduce the probability of repeated collisions, a retransmission backoff algorithm is implemented for shared links. This backoff algorithm has the following properties:

- The wait for a retransmission backoff applies only to the transmission on shared links, there is no wait for transmission on dedicated links.

- In case of transmission failure on a shared link, a device initializes the backoff exponent (BE)(which is a parameter used for determining the backoff delay) to a given minimum, called $macMinBE$.

- The retransmission backoff is calculated in terms of the number of shared transmission links. The device shall delay for a random number of shared link in the range 0 to $2^{BE} - 1$ before attempting a retransmission on a shared link.

- For each successive failure on a shared link, the device should increase BE, up to a given maximum value, i.e., $macMaxBE$. According to [48], the value of $macMaxBE$ for TSCH is chosen in the range 3 to 8.

- A successful transmission in a shared link resets the backoff window to the minimum value.

- In dedicated links, Clear Channel Assessment (CCA) may be used to promote coexistence with other users of the radio channel. In this case, backoff delay is not foreseen.

3.3 Simulative Assessment

In this section the simulative and comparative assessment between TSCH and DSME is described. The assessment focuses on the process automation domain. The used simulator is developed using the OMNet++ simulation environment. The INETMANET framework was adopted for the wireless channel and the IEEE 802.15.4 physical layer, while the MAC layers were developed from scratch. The simulation models were validated analytically by comparing the simulation timing with those analytically calculated. Following the findings in [44], in a process automation context the network contains a maximum of 50 nodes. The purpose of this assessment is to compare the two protocols in order to evaluate reliability, scalability and maximum delay. Note that processing delays are excluded from the simulations as they are implementation-dependent.

Two scenarios are considered, called Scenario A and B, respectively. Scenario A is adopted to assess the reliability and delay performance of the two protocols, while Scenario B is used to evaluate their scalability.

3.3.1 Reliability and delay assessment

To assess both the reliability of the two protocols under study and the impact of retransmissions on the end-to-end delay, a network with 10 sensor nodes and a PAN coordinator that acts as sink node for all the data frame sent from the sensors was simulated. Different kinds of sensors, taken from realistic scenarios, are considered. For monitoring applications, there are pressure (PS) and temperature sensors (TS). For closed-loop control, there are torque sensors (TqS) and for interlocking and control there are flow (FS) and proximity sensors (PxS). In both protocols, two sensor nodes have a twofold role: they act as sensor nodes, sending data to the PAN coordinator, and as coordinator nodes, collecting data from other sensors that cannot reach the PAN coordinator directly and forwarding them to the PAN coordinator in a multihop way. In both simulations a single possible retransmission for each message is assumed. Nodes are placed in a sensing area of 50 x 50 meters. Both the PAN coordinator and the relay nodes are placed in the sensing area to be in the range of the sensor nodes that are directly connected to them, while the sensor nodes are randomly placed following a uniform distribution. The network data rate is 250 Kbps.

Table 3.1 shows the traffic characterization. The first column shows the sensor name, which includes the sensor type, so the PS-1 node is the first pressure sensor, the TS-1 is the first temperature sensor and so on. The second column shows the payload at the application layer in bytes, while in the third column the period at which the data are sent by the relevant sensor node is shown. The last column shows the magnitude order of the maximum delay allowed by the relevant application.

In this simulation the Packet Loss Ratio (PLR) and the End-to-End Delay (E2ED) are assessed.

The PLR is defined as the number of lost or corrupted messages over the overall number of messages transmitted by the sensors nodes

Table 3.1: Traffic characterization

Sensor Nodes	Payload	Period	Delay
Monitoring and supervision			
PS-1	50 bytes	1 s	*ms*
TS-1	40 bytes	5 s	*s*
Closed loop control			
TqS-1	50 bytes	200 ms	*ms*
TqS-2	50 bytes	200 ms	*ms*
TqS-3	50 bytes	500 ms	*ms*
Interlocking and control			
FS-1	50 bytes	250 ms	*ms*
FS-2	50 bytes	250 ms	*ms*
FS-3	50 bytes	500 ms	*ms*
PxS-1	25 bytes	150 ms	*ms*
PxS-2	25 bytes	250 ms	*ms*

(at the application layer). The PLR is calculated as in Formula (3.7)

$$PLR = \left(1 - \frac{NumRxMessages}{NumTxMessages}\right) \times 100 \tag{3.7}$$

where *NumRxMessages* is the number of messages correctly received, while *NumTxMessages* is the overall number of messages transmitted by the sensor nodes.

The E2ED is the time a messages takes from its generation in the source application to its arrival in the sink application. It is calculated as in Formula (3.8)

$$E2ED = ArrivalTime - GenTime. \tag{3.8}$$

In this scenario the Log-normal Shadowing propagation model is adopted. The assessment was performed with different propagation parameters measured in the industrial environment in [68]. The adopted parameters are shown in Table 3.2, where $PL(d_0)$ is the path loss measured over the reference distance d_0, n is the path-loss exponent and σ is the standard deviation.

Table 3.2: Propagation Parameters

Run no.	d_0	$PL(d_0)[dB]$	n	$\sigma[dB]$
0	15m	63.57	2.40	4.97
1	15m	63.57	2.77	5.42
2	15m	63.57	3.44	8.63

In both the DSME and TSCH networks each node is assigned one slot for the message transmission and one for the retransmission. Hence, the configuration parameters were heuristically chosen in order to obtain the lowest superframe period, while taking into account the above assumption. The slot position does not influence the results, as simulations are repeated with different random message generation start times of the nodes and the application which generates messages is not synchronized with the superframe slots. The radio transmission power was set to 1 mW, sensitivity is set to -85 dBm. The simulation parameters for the DSME and TSCH networks are shown in Table 3.3

Table 3.3: Simulation parameters

Parameter	**Value/Range**
DSME Network	
Beacon Order (BO)	4
Multisuperframe Order (MO)	3
Superframe Order (SO)	2
Group Ack	enabled
CAP Reduction	enabled
Slot duration	3.84 ms
TSCH Network	
Num. of slot in a Superframe	20
Slot duration	10 ms

Each simulation run has a duration of 600s to collect a significant number of data. Moreover simulations are repeated three times varying the seed for the random number generation, which influences the nodes position, the application start times of the nodes and the

log-normal pathloss model.

The PLR results, presented in Table 3.4, show that with the three different propagation parameters the packet loss ratio is always lower than 0.01%. Such a result demonstrates that the protocols behave similarly as far as reliability is concerned.

Table 3.4: Packet Loss Ratio results

Run no.	**PLR DSME**	**PLR TSCH**
0	0%	0%
1	0.005%	0%
2	0.01%	0.005%

Fig 3.3 shows the Cumulative Percentage Distribution (CPD) of the end-to-end delay, defined as the percentage of messages with the delay lower than a given value (i.e., the relevant x-axis value), for the run no. 2. The results of the other runs are quite similar to these ones.

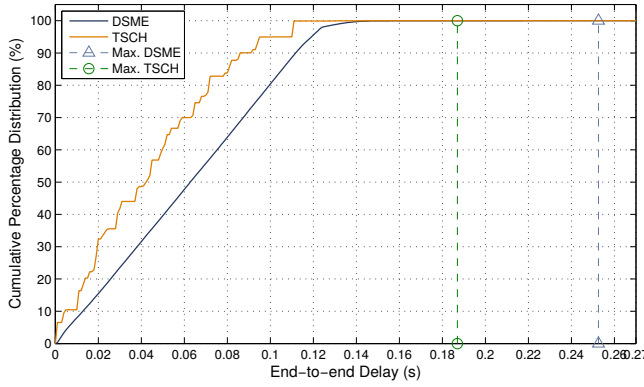

Figure 3.3: End-to-end delay for the run no. 2 in Table 3.2.

The results in Fig. 3.3 show that in this scenario the TSCH protocol provides significantly lower delays than the DSME protocol. In fact, 70% of messages in the TSCH simulation obtain delays lower

than 61ms, while the same percentage of nodes in the DSME simulation has end-to-end delay values close to 90ms.

The maximum end-to-end delay (the vertical dashed lines in Fig. 3.3) is 187ms in the TSCH simulation and 253ms in the DSME one. The difference between the two maximum end-to-end delays is mainly due to the fixed and static multi-superframe structure of DSME, which presents more GTSs than those required, hence introducing a higher delay.

The order of magnitude of these values is the same, or lower, than the order of magnitude of the maximum delay allowed by the application (Table 3.1). This confirms that both DSME and TSCH are suitable for process automation, although TSCH provides lower end-to-end delays.

Note that the stepped trend of the TSCH distribution is due to the long timeslot duration (i.e., 10 ms).

3.3.2 Scalability Assessment

To assess the scalability of the two protocols, a network in which each sensor node periodically transmits a 70-byte message with period equal to 500ms was deployed. The aim of this simulation is to evaluate and compare the delays when increasing the number of nodes in the network.

In both protocols four sensor nodes have a twofold role. They act as sensor nodes, sending data to the PAN coordinator, and as coordinator nodes, collecting data from the sensors that cannot reach the PAN coordinator directly and forwarding them to the PAN coordinator. The messages with an end-to-end delay higher than their period are here called late messages.

For both the protocols, the configurations were designed tuning the relevant parameters (BO, SO, MO, etc.) so as to avoid that the message end-to-end delay is higher than the message generation period.

The simulation parameters for the DSME protocol are shown in

Table 3.5. The last column of the Table shows the multi-superframe duration, which depends on the number of superframes in a multi-superframe and on the superframe length. In all the simulations, the DSME-GTS duration is equal to 3.84 ms. The number of superframes in the multi-superframe can be increased as a power of two (Formula (3.1)). This, in turn, entails that the number of slots increases as a multiple of 16. In the DSME simulations one GTS for data transmission is required for each sensor node and retransmissions are handled in dedicated slots (DSME-GTSR). Hence, each single node requires from one to two slots, so the higher the number of nodes in the network, the higher the number of DSME-GTS required, that in turn entails a higher multi-superframe duration. Such a value is important to understand the timings of messages. In fact, in this scenario a message is multi-hop, i.e., it is first transmitted from the sensor node to a coordinator node and then to the PAN coordinator. As a result, if the multi-superframe duration is higher than half of the message period (i.e. 500ms), the message delay may be higher than the message generation period. In the simulation with 50 nodes this value is close to half of the message period. As all the DSME-GTS are assigned, there is no room for retransmissions. In the other cases (i.e., with 10 and 30 nodes), the choice to enable the group ACK is mandatory, as the messages have to be transmitted within the multi-superframe, otherwise they would arrive to the PAN coordinator too late.

Table 3.5: Simulation parameters for DSME

# of nodes	**BO**	**MO**	**SO**	**GroupACK**	**MultiSF Period**
10	4	3	2	enabled	122.88ms
30	4	3	2	enabled	122.88ms
50	5	4	2	disabled	245.76ms

The configuration for the TSCH protocol is shown in Table 3.6. Even in this configuration a slot is assigned to each node, therefore the superframe period is equal to the number of nodes multiplied by the slot size (i.e., 10ms). Looking at Table 3.6, in the networks

with 10 and 30 nodes the superframe periods are lower than the sampling periods in Table 3.1. This allows message retransmissions, as a message has a chance to be retransmitted once without being late. On the contrary, in the 50-nodes configuration, if a message is retransmitted it will interfere with the transmission of the next messages, as in this configuration all the slots are used. For this reason in the 50-node configuration retransmissions are not allowed.

Table 3.6: Simulation parameters for TSCH

#. of Nodes	**#. of Slots**	**Superframe Period**
10	10	100ms
30	30	300ms
50	50	500ms

The end-to-end delay results for the network with 10 nodes, 30 nodes and 50 nodes are shown in Fig. 3.4, in Fig. 3.5, and in Fig. 3.6, respectively.

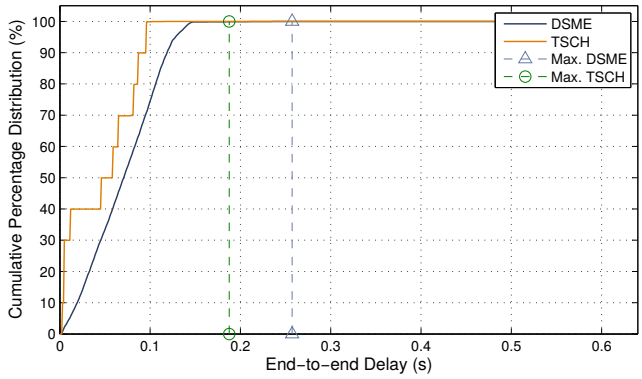

Figure 3.4: End-to-end delay: 10 nodes.

In Fig. 3.4, the case of a network with 10 nodes shows that the TSCH protocol performs better than the DSME. In fact 99% of messages present delays lower than 100ms, while the DSME approaches

150ms. The maximum delay for the TSCH is lower than 200 ms, while for the DSME is close to 260ms.

In fact, the DSME superframe fixed structure entails a higher delay for messages to be transmitted, as the multi-superframe grows exponentially with the number of superframes.

Increasing the number of nodes in the network, results start changing. In Fig. 3.5, that refers to 30 nodes, while the delay distribution is in favor of the TSCH protocol, the maximum delays, represented by the vertical dashed lines, show that the TSCH presents a higher maximum delay than the DSME. This is because the TSCH does not have slots dedicated to retransmissions, thus retransmitted messages interfere with the regular transmissions. In the case of the DSME, the message lateness is due to the number of retransmission slots assigned to a coordinator, which is chosen lower than the number of nodes connected with each coordinator (as a higher number of retransmission slots would entail adding more superframes in the multi-superframe thus introducing delays higher than the message periods). Hence, the messages that need to be retransmitted will wait for the next DSME-GTSR in the next multi-superframe.

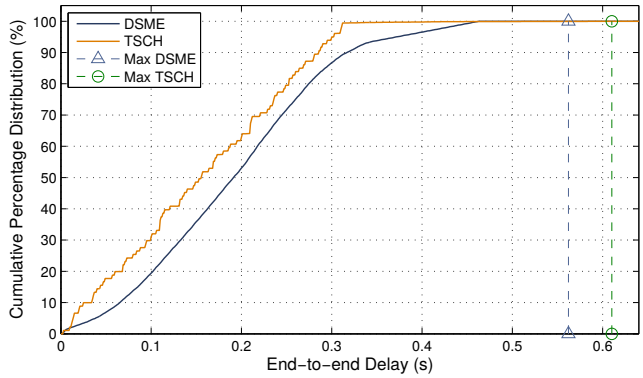

Figure 3.5: End-to-end delay: 30 nodes.

In the simulation with 50 nodes (in Fig. 3.6) the delays of the DSME are lower than the TSCH ones. In particular with the DSME,

80% of messages experience a delay lower than 313ms, while with the TSCH delays are lower than 418ms.

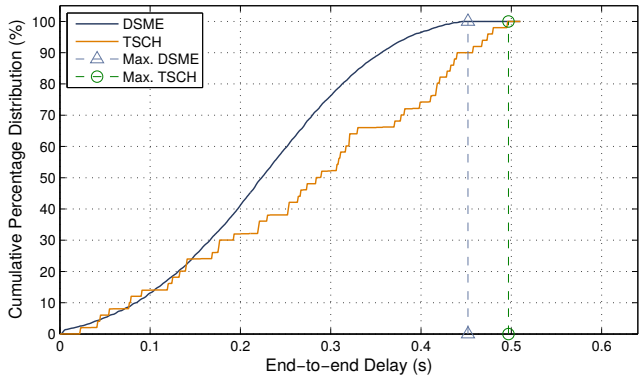

Figure 3.6: End-to-end delay: 50 nodes

Scalability results show that the fixed-structure of DSME penalizes the networks in which the number of nodes is lower than 30. In fact, using the group ACK, retransmissions occur in dedicated DSME-GTSR located at the end of the superframe (this entails a high number of slots and a long waiting period for regular message transmissions) and this will entail high delays. The adoption of the classical ACK mechanism of the IEEE 802.15.4 is allowed, but it is not suitable for these scenarios, as retransmissions cannot be accommodated in the same multi-superframe (as specified in the standard [48]).

On the contrary, if the number of nodes is higher than 30, the DSME performs better compared to the TSCH, as the latter provides a longer superframe period than the DSME multi-superframe duration.

As far as scalability is concerned, TSCH performs better than DSME when the number of nodes is lower than 30, as there are no dedicated retransmission slots, so a lost message will be retransmitted in the next available slot for the same node that failed the transmission. However, this entails that retransmitted messages will interfere

with the regular message transmission, as in the case of retransmission, the regular message transmission will be delayed to make room for the retransmission. This is an advantage when the number of nodes is low, as the superframe period is low too and there is time left for retransmissions. When the number of nodes is higher than 30, the superframe period increases, thus decreasing the possibility of retransmissions.

3.4 Conclusions and future works

This work presented a performance assessment of two of the three IEEE 802.15.4e profiles, i.e., the DSME and TSCH. According to the IEEE 802.15.4e amendment [48], process automation represents the reference application domain for both the profiles. The evaluation is performed in a realistic process automation scenario and the performance metrics are reliability, delay and scalability. As far as reliability is concerned, both protocols proved to be robust towards channel noise. In general, TSCH offers better end-to-end delays than DSME. In particular, simulation results show that DSME perform better than TSCH in terms of end-to-end latency only when the number of nodes is higher than 30. This result derives from the rigid structure of the DSME multi-superframes, in which the number of superframes in multi-superframes grows as a power of 2 and the number of slots in a superframe is equal to 16. When the number of nodes is low, such a structure leads to an overprovisioning of DSME-GTS that not only affects the bandwidth efficiency, as more slots than those actually needed are used, but also determines long delays for the messages to be transmitted. On the contrary, when the number of nodes grows, all the DSME-GTS in the multi-superframe are used, thus improving the bandwidth efficiency, and the DSME delay values are better than those of TSCH, thanks to the smaller size of the DSME slots compared to the TSCH ones.

Conversely, the TSCH has a more flexible slotframe structure than

DSME, as a single timeslot can be inserted in or removed from the superframe. However, timeslots have to be larger than in DSME, in order to accommodate the ACK. This means that in networks with a high number of nodes, the superframe period grows faster in TSCH than in DSME. Moreover, another limitation is that the IEEE 802.15.4e standard does not provide group ACK methods for the TSCH.

Ongoing work extends this study to address possible approaches to overcome the limits of DSME or TSCH previously mentioned. Moreover, the support provided by the two protocols to event-driven traffic will be investigated. A timing analysis of the two protocols is also in progress.

Chapter 4

A Priority-Aware Multichannel Adaptive Framework for the IEEE 802.15.4e-LLDN

Industrial wireless sensor networks (IWSNs) require bounded message latency [69–71], high reliability [72–74], scalability [75–77] and, in some cases, very short cycle times. As such requirements were not adequately addressed by the IEEE Std 802.15.4-2011, the IEEE 802.15.4e amendment [48] was proposed, which defines three Media Access Control (MAC) protocols, i.e., the Low Latency Deterministic Network (LLDN), the Deterministic and Synchronous Multi-channel Extension (DSME) and the Time Slotted Channel Hopping (TSCH). These protocols are quite different, as they target different industrial application domains. DSME fits well general industrial applications that do not need low latency, but require deterministic latency, flexibility, high reliability, efficiency, scalability, and robustness. The TSCH instead meets the requirements of process control applications, while the typical applications addressed by the LLDN are those found in factory automation (such as, automotive manufacturing), in which

a large number of devices observe and control the production. These applications, which require very low latency and large networks with many sensors (even more than 100), could not be supported by DSME or TSCH. In fact, the DSME protocol proved to be able to achieve low latency (i.e., in the order of hundreds of milliseconds) by using the Ultra Wide Band physical layer, which operates at 850 kbps [60]. However, with the most common IEEE 802.15.4 physical layer operating at 250 kbps, low latency is difficult to obtain with the DSME for two reasons. First, the DSME MAC frames have a high overhead (as typically they are the same as the original IEEE 802.15.4 protocol). Second, the DSME provides a constrained superframe structure composed of sixteen timeslots, one for the beacon and the others for the Contention Access and the Contention Free Period, respectively. If more timeslots are needed, an entire superframe has to be added in the multi-superframe and this increases latency. As far as the TSCH is concerned, some assessments were recently proposed in the literature [78], [79]. In particular, TSCH proved to be not suitable to support applications that require low latency (or round-trip time) and, more in general, applications with fast dynamics. In fact, the TSCH uses the classical IEEE 802.15.4 MAC frames, which entail a high overhead and therefore a high cycle time [78]. Conversely, in the LLDN low latency transmissions are achieved introducing MAC frames that provide a minimum MAC overhead of 3 bytes (1 byte for the header, 2 bytes for the FCS).

However, the LLDN protocol has some limitations. First, it does not provide any message prioritization mechanism. Second, it suffers from scalability problems, as the cycle time (i.e., the time in which all nodes can transmit once) grows linearly with the number of network nodes. Finally, LLDN does not provide mechanisms for dynamic channel configuration, so if the current channel becomes unreliable, there is no way to dynamically switch to another channel. To overcome these limitations, this work proposes the Priority-aware Multichannel Adaptive (PriMulA) framework, which introduces in LLDN priority-aware scheduling, multichannel communication and dynamic

channel configuration. Thanks to the multichannel communication, PriMulA allows for an increase in the number of nodes in the LLDN network while minimizing the superframe augmentation. PriMulA therefore enables the deployment of larger LLDN networks for applications requiring low latency (e.g., round-trip time in the order of tens of milliseconds), such as robotic and automotive manufacturing [48]. PriMulA improves the network reliability providing a mechanism for adaptive channel selection and blacklisting and, thanks to the introduction of message priority, PriMulA avoids deadline miss. Finally, PriMulA maintains the interoperability with LLDN standard nodes and can be implemented on COTS devices.

The chapter is organized as follows. Sect. 4.1 presents related work, while Sect. 4.2 overviews the LLDN protocol. Sect. 4.3 describes the PriMulA framework. Sect. 4.4 presents a schedulability analysis for real-time messages. Sect. 4.5 provides comparative assessments of LLDN, PriMulA and another approach, obtained through OMNeT++ simulations. Sect. 4.6 presents an implementation of PriMulA on off-the-shelf devices. Finally, Sect. 4.7 concludes the chapter and gives hints for future work.

4.1 Related Work

Several works in the literature present solutions to address the requirements of IWSNs. Shen at al. [71] propose a MAC protocol to support high priority traffic over IWSNs, while Yan et al. [70] investigate the planning of the superframe structure of slotted MAC protocols. To improve scalability and delays, Tung et al. [77] suggest multiple transceivers in a single node. This approach reduces delays, as multiple nodes can transmit at the same time, but increases complexity and costs. Toscano and Lo Bello [80], [76] propose a multi-channel approach for the original IEEE 802.15.4 beacon-enabled protocol with the twofold aim of avoiding beacon collisions and allowing multiple devices to communicate at the same time in networks with multiple

nodes. Mechanisms to enhance reliability in IWSN are proposed in the literature [81–83]. Dobslaw et al. [74] present and assess a low-complexity scheduling algorithm to guarantee bounded end-to-end reliability, while Willig et al. [84] address relaying and the problem of scheduling relayers and retransmission slots for periodic flows in a Time Division Multiple Access (TDMA)-based network. To improve slot efficiency, Yang et al. [85] present a shared slot scheduling for retransmission in IWSNs. Unlike such works, PriMulA addresses three objectives, i.e., the ability to cope with message deadlines, reliability, and scalability, with the aim of achieving all of them in an integrated way. As far as LLDN is concerned, Dariz et al. [86] propose an enhancement of the LLDN to reduce the cycle time and handle different traffic classes, which provides for different timeslot lengths according to the data to be transmitted. However, the approach has the drawback of introducing heavy modifications in the LLDN standard. In the IEEE 802.15.4k [87] standard a mechanism for the transmission of critical event messages, called a Priority Channel Access (PCA), was specified. The PCA allows nodes to transmit critical event messages during the Contention Access Period using a modified Carrier Sense Multiple Access (CSMA) mechanism, which, on average, reduces the backoff duration compared to that of the plain CSMA. However, unlike PriMulA, the PCA does not provide guarantees on the maximum message delay. The MC-LLDN proposed by Patti et al. [59] provides a two-level network, in which a Higher Level Network (HLN), made up of nodes that are directly connected to the Personal Area Network (PAN)-Coordinator, and several sub-networks operate at the same time on different channels. Such an approach improves the cycle time up to 44% in a network with 100 nodes. However, the MC-LLDN cannot cope with message deadlines, as there is no way to prioritize messages based on their different temporal constraints. In addition, in MC-LLDN the channels for the HLN and the sub-networks are statically allocated. Consequently, if one such channel becomes permanently unreliable (e.g., due to interference), the relevant sub-network would be unreachable or, if the affected channel is

the HLN one, the entire network would be disrupted. The PriMulA framework is the solution for addressing all these aspects in a unified way.

4.2 LLDN: an overview

In the LLDN protocol, the network time is organized in cyclically scheduled superframes made up of equally sized timeslots, as shown in Fig. 4.1. The LLDN superframe starts with the transmission of an

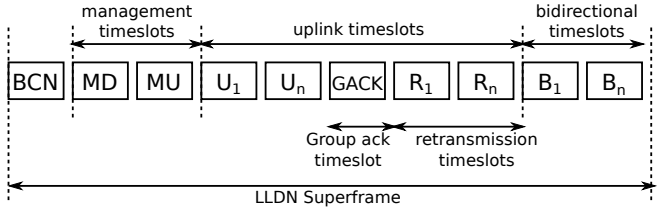

Figure 4.1: The LLDN Superframe (redrawn version from [48])

.

LL-Beacon frame (BCN) from the PAN-Coordinator. The beacon is used for synchronization purposes and contains information about the superframe structure and the network state (i.e., number of timeslots, acknowledgement information, timeslot direction, etc.). The LLDN global state can change at any time and this information is contained in the beacon frame, hence, if the beacon is lost, the transmission of the end nodes is compromised for the entire cycle. The beacon is optionally followed by two management timeslots, for downlink (MD) and uplink (MU) transmission. The MU timeslot is contended among the end-nodes through a CSMA mechanism. The remaining part of the superframe contains uplink (U) and bidirectional (B) timeslots for data transmission. The uplink ones are used by the end-nodes to transmit data to the PAN-Coordinator (some timeslots can also be reserved for retransmissions (R)), while in the bidirectional ones

both the PAN-Coordinator and the end-nodes can transmit. The separated group ack timeslot, which provides a bitmap to acknowledge the successful transmissions of the current cycle, is used. Retransmission timeslots come at the end of the uplink timeslots, as shown in Fig. 4.1, and are not preassigned to nodes, as their assignment is regulated by a distributed algorithm defined in the standard, which depends on the cumulative ack transmitted in each cycle. On each cycle, a node can retransmit the unacknowledged messages at most once. The duration of a timeslot (T_{ts}) is defined in [48] as a function of the maximum expected data payload which can be transmitted by a node plus the overhead introduced by the physical and the MAC layer, as in

$$T_{ts} = \frac{(o+n)sb + \mathsf{IFSpace}}{symbolRate}, \tag{4.1}$$

where o is the overhead (in bytes) introduced by the physical and MAC layer, n is the maximum expected data payload, sb is the number of symbols per byte, and IFSpace is the inter-frame space (i.e., 12 symbols for frames shorter than 18 bytes, 40 symbols otherwise).

The number of timeslots in a superframe (N) multiplied by the timeslot duration gives the cycle time (T_s)

$$T_s = N \times T_{ts}. \tag{4.2}$$

As in the LLDN each node has at least one timeslot assigned, the cycle time grows linearly with the number of nodes. This increases the cycle times in networks with a very large number of nodes. For this reason, in [59] the MC-LLDN was presented. In the MC-LLDN all the nodes in the sub-networks send data to the relevant sub-coordinator. Each sub-coordinator uses a single timeslot to transmit all the data collected from all the nodes belonging to the sub-network, therefore the timeslot size has to be large enough to accommodate in one timeslot all the messages generated within a sub-network. Conversely, in PriMulA the timeslot size can be smaller, as the framework provides the nodes with the ability to embed a configurable number (Ω) of PriMulA messages in a single LL-Data frame (i.e., the standard data

frame transmitted from the MAC layer). Combining this feature with message priorities, the timeslot size in PriMulA is set large enough to allow for the transmission of the Ω highest priority messages only. The reduction of the timeslot size in PriMulA shortens the cycle time, which depends on both the number of timeslots and their size, as shown in (4.2). As a result, when compared with LLDN and MC-LLDN, PriMulA can support applications with shorter message generation periods.

4.3 The Priority-based Multichannel-LLDN

The PriMulA framework here proposed, comparing with the standard LLDN, aims to provide several novel features, i.e.,

- Support for meeting the message deadlines, through priority-aware scheduling;
- Shorter cycle times and improved scalability, through a combination of hierarchical topology, multichannel communication and message priority;
- Increased network reliability, through dynamic channel selection.

4.3.1 Priority-aware scheduling

In PriMulA, each node maintains at the MAC level a queue of outgoing messages ordered by priority. To efficiently support priority scheduling, PriMulA introduces a novel message, called a PriMulA message, that is embedded in the payload of a standard LL-Data frame. This message consists of the message priority and payload. The PRIO field is encoded in one byte to achieve a good tradeoff between the number of priorities and the overhead of each message, so up to 256 different priorities can be handled. However, a larger PRIO field would not affect the effectiveness of the proposed approach.

In PriMulA, messages are periodically generated by the application, with fixed period (P). For each message, the application defines a lifetime, called a relative deadline (D). D is the maximum time interval, measured at the application layer, within which a generated message has to be consumed. Both P and D of a message depend on the supported application. The relative deadline can be shorter than or equal to period, or greater than the period. PriMuLa adopts a fixed priority assignment, which assigns priorities to messages according to their relative deadlines, i.e., the shorter the relative deadline, the higher the priority.

4.3.2 Multichannel communication

PriMulA provides for a hierarchical topology, as shown in Fig. 4.2, where an HLN with three sub-networks is drawn. Node 0 is the PAN-

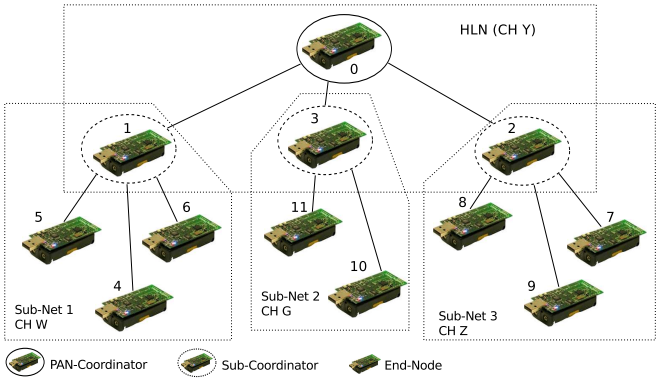

Figure 4.2: Example of logical topology.

Coordinator and operates on channel Y, while Nodes 1, 2 and 3 are sub-coordinators and operate on channels W, G and Z, respectively. The sub-networks work in parallel and the sub-coordinators switch between the HLN and their sub-network in fixed timeslots. As shown in Fig. 4.3, PriMulA provides multiple superframes, one for the HLN and one for each sub-network. All the superframes contain the same

number of timeslots and all the timeslots have the same size. In timeslot 1, the PAN-Coordinator transmits its beacon frame to synchronize all nodes in the HLN and Nodes 1, 2 and 3 receive it on channel Y. In timeslot 2, the sub-coordinators switch to their own channels and transmit their beacon. In the next timeslots, the nodes transmit their data to the relevant sub-coordinator. Each node is allowed to transmit up to Ω messages in priority order. Ω is heuristically calculated as a tradeoff between the cycle time and the estimated delay of the messages. Ω calculation is described in detail afterwards in this Section. On timeslot 4, the sub-coordinator 3 switches back to the HLN channel to transmit to the PAN-Coordinator the Ω PriMulA messages with the highest priority gathered from its sub-network nodes. On timeslots 5 and 6 the sub-coordinators 2 and 1, respectively, do the same.

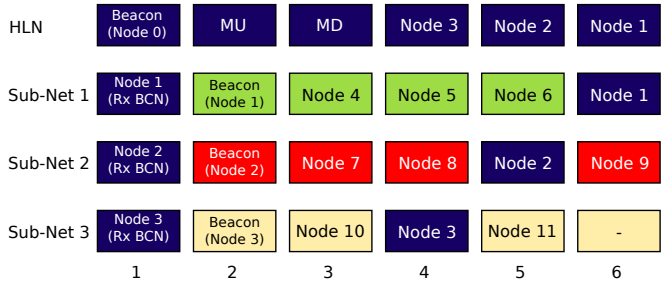

Figure 4.3: Example of superframes in the PriMulA network.

In PriMulA, the minimum number of timeslots (N_{min}) that must be provided in each superframe to allow the periodical transmission of all nodes but the PAN-Coordinator can be calculated as in

$$N_{min} = max_{i=0...C} \left(E_{\mathsf{HLN}} + 2, E_i + 2 \right), \tag{4.3}$$

where C is the number of sub-coordinators (hence, the number of sub-networks), E_{HLN} represents the number of nodes in the HLN (excluding the PAN-Coordinator), while E_i is the number of nodes in the ith sub-network (including the sub-coordinator). Therefore, there is

one timeslot for each node in the HLN, one timeslot for each node in the ith subnetwork and in both cases two additional timeslots are needed, one for the beacon transmission from the PAN-Coordinator and one for the beacon transmission from sub-coordinators, respectively. Such a formula assumes the minimal configuration, i.e., with no management timeslots. The number of subnetworks (C) is heuristically set as a tradeoff between reliability and cycle time. In fact, a high C value limits the number of available channels to be used if the current channel becomes unreliable, while a small C value entails a high number of nodes per subnetwork, a high number of messages to be handled by a sub-coordinator and, consequently, high delays. The number of subnetworks must be in the range [1,15], as the IEEE 802.15.4 Physical Layer [56] provides only 16 channels and one of them is used by the HLN. The number of nodes per subnetwork (E) is also heuristically set. As all the superframes must have the same length, the nodes should be evenly distributed over the subnetworks whenever possible. The timeslot size (in seconds) is determined in the same way as in (4.1), calculating the maximum expected payload for an LL-Data frame (n) as in

$$n = (\mathsf{PriMulAHdr} + \phi)\,\Omega, \tag{4.4}$$

where PriMulAHdr is the PriMulA message header length, ϕ is the maximum expected length of the PriMulA message payload (that depends on the supported application) and Ω is the maximum number of PriMulA messages which can be embedded in a single LL-Data frame. The Ω value is chosen as a tradeoff between cycle time and message delays. On one hand, Equation 4.4 shows that the timeslot length grows with Ω, so a larger Ω entails a longer cycle time. On the other hand, a large Ω value allows to transmit more messages in one timeslot, thus reducing the waiting times of the messages queued in the nodes. As the standard IEEE 802.15.4 physical layer provides for a maximum payload of 127 bytes, Ω has to be set so as to not exceed the maximum payload allowed. Hence, condition (4.5) must

hold

$$\Omega \in \aleph : 1 \leq \Omega \leq \left\lfloor \frac{127 - o}{(\mathsf{PriMulAHdr} + \phi)} \right\rfloor, \tag{4.5}$$

where $o = 3$ is the MAC overhead (in bytes) and the right-hand side indicates the maximum number of PriMulA messages that can be transmitted without exceeding the maximum allowed payload. Once the message generation patterns of the nodes are known, and the number of subnetworks as well as the total number of nodes for each subnetwork are set, Ω is chosen as the largest value that maximizes the throughput of the subnetwork with the heaviest load. The Ω value has then to be fed into the timing analysis presented in Sect. 4.4. If the message set is found schedulable, the Ω value is confirmed, otherwise a lower value is chosen (as previously explained) and the timing analysis is run again.

4.3.3 Dynamic channel configuration and blacklisting

In PriMulA, during the network configuration, the PAN-Coordinator assigns to each sub-coordinator the channel number of the relevant sub-network. This operation complies with the standard [48], that foresees a generic *Configuration Parameters* field in the command data frame [48] that is exchanged during the configuration phase. To avoid cross-interference [88] between contiguous channels, the PAN-Coordinator first assigns to each sub-network, starting from the HLN, odd-numbered channels in ascending order, and then assigns the even-numbered channels in descending order. These steps maintain the HLN channel far enough from the other sub-network channels to reduce the chance for cross-interference. After the configuration, as foreseen in the LLDN standard [48], the network switches to the *Online State*, in which nodes start data transmission. During this phase, if the channel of a sub-network becomes unreliable, the sub-coordinator changes its state to *Configuration State* and transmits to the PAN-Coordinator, in the uplink management timeslot, a *Config-*

uration status frame, i.e., a standard command frame indicating that a node requires to be reconfigured. Note that to detect an unreliable channel and activate the Dynamic Channel Configuration (DCC) mechanism, any approach provided in the literature, such as [89], can be adopted. The PAN-Coordinator is aware that the node requiring configuration is a sub-coordinator, therefore it assigns a new channel (chosen from the available ones) to the sub-coordinator that required it and blacklists the unreliable channel. If there are no more available channels, the PAN-Coordinator picks up from the blacklist the least recently blacklisted channel and assigns it to the sub-coordinator. As during the channel switching phase the end-devices of the sub-network that is subject to interference do not receive any beacon, they switch to the *discovery state*. Once the sub-coordinator is assigned a new channel, it also switches to the *discovery state* to find and configure its sub-network nodes. The DCC mechanism here proposed only involves the sub-coordinators and the PAN-Coordinator, so it is transparent to the end-nodes and therefore is compatible with the normal operation of standard LLDN end-nodes.

4.4 Schedulability analysis

In PriMulA, in the general case, each message is sent twice, i.e., from the end-node to the sub-coordinator and from the sub-coordinator to the PAN-Coordinator. Each node can generate multiple messages belonging to different flows with different (or equal) priorities. The messages within the node are scheduled according to their priority, which depends on the flow relative deadline. If two messages have the same priority (i.e., the same relative deadline), they are transmitted in *First-In First-Out* (FIFO) order. The maximum time RT_i taken by a message of the ith PriMulA flow sent from an end-node to arrive to the PAN-Coordinator is calculated as

$$RT_i = T_{q1_i} + T_{q2_i} + 2T_{x_i}, \tag{4.6}$$

where T_{q1_i} and T_{q2_i} are the maximum queuing delays experienced by the message in the end-node and in the sub-coordinator, respectively, while T_{x_i} is the message transmission time. In the case where an end-node is directly connected to the PAN-Coordinator, RT_i is given by $RT_i = T_{q1_i} + T_{x_i}$.

The analysis here presented assumes no retransmissions, but it can be extended to consider retransmissions using the separated group ack defined by the standard [48] and described in Sect 4.2. The separated group ack allows messages to be retransmitted once within the same superframe. To calculate the WCRT in this case, a retransmission timeslot for each uplink timeslot has to be assumed. Moreover, in the worst case all the messages are retransmitted, following the same order as the transmissions. Under these hypotheses, the analysis here presented can be applied assuming that messages will be transmitted only in the uplink retransmissions timeslots. Table 4.1 summarizes the notations used in the chapter. The ith periodic flow is schedulable (i.e., all the messages of the ith flow will arrive to the PAN-Coordinator within their deadline) if condition (4.7) holds

$$RT_i \leq D_i, \tag{4.7}$$

where D_i is the ith flow relative deadline. As it was explained in Sect. 4.3.1, the relative deadlines are constraints imposed by the application timing. In the following, a timing analysis for calculating T_{q1_i}, T_{q2_i}, and T_{x_i} is presented.

4.4.1 Response-time analysis

First, we assume that T_{x_i} is equal to the timeslot duration (T_{ts}), as the latter is configured to allow the complete transmission plus the inter-frame space. Under this assumption, the worst case response time for a message of the ith flow is given by

$$RT_i = T_{q1_i} + T_{q2_i} + 2T_{ts}. \tag{4.8}$$

Table 4.1: PriMulA notation

Symbol	Definition
RT_i	The maximum response time of a PriMulA message of the ith flow.
T_{q1_i}	The maximum time that a PriMulA message of the ith flow waits to be transmitted in the end-node queue.
T_{q2_i}	The maximum time that a PriMulA message of the ith flow waits to be transmitted in the sub-coordinator queue.
T_{x_i}	The transmission time of the ith message.
D_i	The relative deadline of a PriMulA message in the ith flow.
T_{ts}	The timeslot duration.
$s_j(t)$	The maximum number of messages that the jth node can transmit in the $[0, t)$ interval.
T_s	The superframe duration or cycle time.
N	The number of timeslots within a superframe.
$z^{(j)}$	The zth timeslot assigned to the jth node.
$p_z^{(j)}$	The index of the zth timeslot assigned to the jth node.
$z_{worst}^{(j)}$	The zth timeslot assigned to the jth node, which provides the worst case response time for a message if the message arrives shortly after the beginning of this timeslot.
$p_{z_{worst}}^{(j)}$	The index of the z_{worst} timeslot assigned to the jth node.
Γ_j	The number of timeslots assigned to the jth node in a superframe.
Ω	The maximum number of PriMulA messages that a node can transmit within a timeslot.
p_{Γ_j}	The index of the last timeslot assigned to the jth node in the superframe.
$I_i(t)$	The interference experienced by the ith flow due to high-priority flows at the time t.
P_i	The period of a PriMulA message of the ith flow.
X_i	The largest number of consecutive timeslots needed to transmit a message of the ith flow.
$w(X)$	The longest time a node has to wait to see X consecutive timeslots.
C	The number of sub-coordinators.
E	The number of nodes belonging to a sub-network.
p_m	The index of the timeslot assigned to the mth sub-coordinator.
LM	The number of late messages.
RxM	The number of received messages by the PAN-Coordinator.
GM	The overall number of generated messages.

To calculate T_{q1} and T_{q2} both the position within the superframe of the timeslots assigned to the sender end-node and the interference of high-priority messages have to be considered.

Calculating T_{q1}

To determine T_{q_1}, the worst case response-time (WCRT) analysis is made calculating both the resource availability and the worst case for the resource request. In PriMulA the resource availability for the jth node is given by the number of timeslots in a super-frame in which the node can transmit, while the resource request is the number of timeslots needed to transmit the messages generated by the node.

The analysis here presented, which applies to fixed priority non-preemptive scheduling, is based on the worst case response-time (WCRT) analysis that was proposed by Joseph and Pandya [36] and extended by Lehoczky with the "busy period" approach [90], and on the findings in Davis et al. [91]. Consequently, the response times of all messages of a flow within a busy period have to be examined. The busy period is defined as the maximum interval during which any message of priority lower than the priority of the ith flow is unable to start transmission. The busy period starts at the time t^s in which a message with a priority higher than or equal to the one of the ith flow is enqueued and there are no messages with priority higher than or equal to the ith flow waiting to be transmitted that were queued strictly before t^s. The busy period ends at the earliest time t^e in which there are no messages of priority equal or greater than the ith flow waiting to be transmitted that were queued strictly before time t^e. With this definition all the messages with the same or higher priority than the ith flow that were enqueued before the end of the busy period are transmitted during the busy period [91].

In PriMulA the resource availability in the interval [0,t) is calculated as the maximum number of messages ($s_j(t)$) that the jth node can transmit in the interval $[0, t)$. In Fig. 4.4 an example of the $s(t)$ function for a generic Node A with $\Omega = 2$ is shown. In the Figure, at

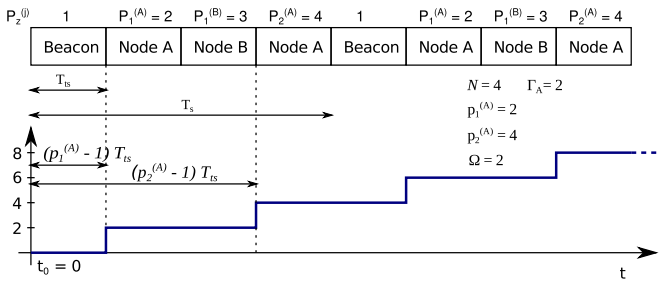

Figure 4.4: Example of $s_j(t)$ function: the $s_1(t)$.

$t_0 = 0$ a superframe starts. From this instant, the time that the Node A has to wait for the arrival of its first timeslot ($z = 1$), i.e., the time interval between the start of the superframe and the time at which the first timeslot assigned to node A starts, is $(p_1^A - 1)T_{ts}$, where p_1^A is the position of the first timeslot assigned to the Node A (i.e., $p_1^A = 2$ in Fig. 4.4). At this time the Node A can start transmitting up to Ω messages. Then Node A waits for its second timeslot ($z = 2$) that, in the case of Fig. 4.4, from the beginning of the same superframe starts at $(p_2^A - 1)T_{ts} = (4 - 1)T_{ts}$. Hence, Node A can transmit other Ω messages. This pattern periodically repeats with the superframe period T_s. Hence, $s_j(t)$ is given by

$$s_j(t) = \sum_{z=1}^{\Gamma_j} \left\lfloor \frac{t + T_s - (p_z^{(j)} - 1)T_{ts}}{T_s} \right\rfloor \Omega, \tag{4.9}$$

where $(p_z^{(j)} - 1)T_{ts}$ is the offset of the zth timeslot from the beginning of the superframe. Therefore $[t + T_s - (p_z^{(j)} - 1)T_{ts}]/T_s$ is the number of timeslots assigned to the jth node available in the time interval $[0, t)$. The floor operator is intended to make the function stepwise, as any transmission can only start at the beginning of the corresponding timeslot. In fact, $s_j(t)$ is increased by Ω (in Fig. 4.4, $\Omega = 2$) only at the beginning of each timeslot. Finally, the summation is required to consider all the timeslots assigned to the jth node within a superframe. As node transmissions do not collide with each

other, thanks to the TDMA mechanism, for the T_{q1} calculation only the interference of the messages generated within the same node has to be analyzed. The worst case arrival time for a message of the jth node occurs when the message arrives shortly after the beginning of the timeslot ($p^{(j)}_{z_{worst}}$) that, among those assigned to node j, is the furthest one from the next timeslot for the same node.

To calculate T_{q_1}, the longest time ($w_j(X)$) the jth node has to wait to see X consecutive timeslots is calculated as

$$w_j(X) = QT_s + \left(p^{(j)}_{\lceil \frac{(R+1)}{\Omega} \rceil} - p^{(j)}_{z_{worst}} \right) T_{ts}, \tag{4.10}$$

where Q is the quotient of the Euclidean division of $X-1+(z_{worst}\cdot\Omega)$ by $\Gamma_j\Omega$ and R is the reminder, i.e., $[X-1+(z_{worst}\cdot\Omega)] = Q\Gamma_j\Omega + R$.

For instance, in the case of Fig. 4.4, Node A has $\Gamma_A = 2$, $\Omega = 2$ and $p^{(A)}_{z_{worst}} = 2$ (i.e., $z^{(A)}_{worst} = 1$), so for $X = 5$, we have $Q = 1$ and $R = 3$. Therefore, Node A has to wait for $w(5) = T_s + (p^{(A)}_2 - p^{(A)}_1)T_{ts} = T_s + (4-2)T_{ts} = T_s + 2T_{ts}$. In fact, looking at the example in Fig. 4.4, if the first message arrives at the beginning of timeslot 2, it will be transmitted in timeslot 4, thus the time at which Node A can start transmitting its 5th message is $t = T_s + 2T_{ts}$.

To apply the busy period approach. First, we calculate the number of messages of the ith flow generated in any interval of length t, i.e.,

$$I_i(t) = \left\lceil \frac{t}{P_i} \right\rceil. \tag{4.11}$$

Then, we calculate the largest number (X_i) of timeslots that are needed to transmit the ith message, which is given by the smallest value of X_i that satisfies

$$X_i = 1 + \sum_{\mathsf{pri}(h)>\mathsf{pri}(i)} I_h(w_j(X_i)) \quad + \sum_{\substack{\mathsf{pri}(h)=\mathsf{pri}(i) \\ h\neq i}} 1, \tag{4.12}$$

where 1 is the timeslot required to transmit the message, while the parameter (X_i) encompasses the following sources of interference:

- The hth higher priority messages (i.e., $\mathsf{pri}(h) > \mathsf{pri}(i)$).

- The messages with the same priority as the ith message that compete with it in the end-node queue (in this case a FIFO policy is adopted).

Equation (4.12) has not simple solution, as the X_i term appears both in the Left-hand-side (LHS) and in the Right-hand-side (RHS) of the equation under the ceiling operator. Thus, the calculation of X_i is performed by the following iterations

$$\begin{cases} X_i^{(0)} & = 1 \\ X_i^{(k+1)} & = 1 + \sum_{\mathsf{pri}(h)>\mathsf{pri}(i)} I_h(w_j(X_i^{(k)})) \\ & \quad + \sum_{\substack{\mathsf{pri}(h)=\mathsf{pri}(i) \\ h \neq i}} 1 \end{cases} . \tag{4.13}$$

Iteration starts with $X_i^{(k)} = 1$, with $k = 0$, as the message to be transmitted requires one timeslot. Then $X_i^{(k+1)}$ is iteratively calculated until the LHS is equal to the RHS, i.e., until the interference stops growing. Equation (4.13) is proved to converge if the timeslots required to transmit every message of the jth node is lower than or equal to the number of timeslots available for the jth node [36]. This way, T_{q1_i} for the ith message is calculated as $T_{q1_i} = w_i(X_i^{(k)})$.

Calculating T_{q2}

To compute T_{q2}, all the messages that a sub-coordinator receives from the nodes belonging to its sub-network plus the messages generated by the sub-coordinator itself are considered. The $s_m(t)$, $I_i(t)$ and $w_m(X)$ functions can be calculated using Equations (4.9), (4.11), and (4.10), but taking into account, instead of p_j, the timeslot index (p_m) of the mth sub-coordinator in the HLN superframe, as the messages in the sub-coordinator queue are generated by all the sub-network nodes.

Applying the busy period approach, the largest number (X_i) of timeslots that are needed to transmit a message of the ith flow is calculated as in

$$\begin{cases} X_i^{(0)} & = 1 \\ X_i^{(k+1)} & = 1 + \sum_{\mathsf{pri}(h)>\mathsf{pri}(i)} I_h(w_j(X_i^{(k)})) \\ & \quad + \sum_{\substack{\mathsf{pri}(h)=\mathsf{pri}(i) \\ t_{start}(h)<t_{start}(i)}} I_h(w_j(X_i^{(k)})) \, . \\ & \quad + \sum_{\substack{\mathsf{pri}(h)=\mathsf{pri}(i) \\ h\neq i}} 1 \end{cases} \tag{4.14}$$

The ($X_i^{(k+1)}$) parameter encompasses the following sources of interference:

- The messages with a higher priority than the ith message (i.e., $\mathsf{pri}(h) > \mathsf{pri}(i)$).
- The messages with the same priority as the ith message that are either transmitted to the sub-coordinator in the timeslots preceding the one in which the ith message is transmitted or that are transmitted in the same timeslot as the ith message, but before it (ties are broken in a FIFO way) within.
- The messages with the same priority as the ith message generated by the sub-coordinator.

This way, T_{q2_i} for the ith message is calculated as $T_{q2_i} = w_m(X_i^{(k)})$.

According to [36], Equation (4.14) is proved to converge if the number of timeslots needed to transmit all the sub-network messages is lower than or equal to the number of timeslots available for the relevant sub-coordinator.

4.5 Simulation Scenario

This section presents a comparative assessment, obtained through simulations, of the LLDN, the MC-LLDN proposed in [59] and PriM-

ulA. A simulation model was developed using the OMNeT++ framework and the inetmanet-2.0 library. The aim is to evaluate the network scalability and reliability, while taking into account the message deadlines. The simulated PHY layer uses the DSSS O-QPSK modulation and operates at 250 kbps in the 2.4 GHz frequency band.

To assess scalability two metrics are defined. The first one is the network saturation point, which represents the maximum number of nodes for which condition (4.15) is met, under the assumption that all the nodes generate the same traffic

$$\sum_{i=1}^{NumFlows} \frac{1}{P_i} < \sum_{j=1}^{E} \frac{\Gamma_j \Omega}{T_s}, \forall \text{ sub-network.} \tag{4.15}$$

In condition (4.15) the LHS is the maximum achievable workload generated in a sub-network and the RHS is the overall sub-network throughput, which takes into account that each end-node or sub-coordinator can transmit up to $\Gamma_j \Omega$ messages every cycle time.

The second metric is the Deadline Miss Ratio, i.e., the ratio of the number of late messages (LM) to the overall number of messages received (RxM) by the PAN-Coordinator, as in

$$DMR = \frac{LM}{RxM}. \tag{4.16}$$

Two other metrics are used for the reliability assessment. The first one is the Packet Loss Ratio (PLR), defined as in

$$PLR = 1 - \frac{RxM}{GM}, \tag{4.17}$$

where RxM is the number of messages correctly received from the PAN-Coordinator and GM is the number of messages transmitted to the PAN-Coordinator. The second is the reconfiguration time, which is the time the network takes to reconfigure itself after a DCC run. A realistic networked control system scenario was set up, in which each node generates and periodically transmits 18-byte data

messages. The message periods are shown in Table 4.2. In all the simulations the relative deadline of a message is considered equal to the message period. For the sake of generality, the message periods

Table 4.2: Data messages configuration

Message ID	**Period P (ms)**
1	100ms
2	250ms
3	450ms

are not multiple of each other and belong to the range [100,500]ms, to comply with the data sampling periods that are typically found in realistic networked control system applications [92].

4.5.1 Scalability assessment

In this scenario, each node periodically transmits to the PAN-Coordinator three kinds of messages. In each run the number of nodes is increased to approach the situation in which the network workload is higher than the maximum achievable throughput. In this assessment no retransmissions and errors are considered, as the aim here is to compare the scalability of the three different approaches, while their reliability is dealt with in Sect. 4.5.2. In the LLDN network configuration, the superframe provides one timeslot for each node and messages are transmitted in a FIFO order. In both the MC-LLDN and PriMulA networks, instead, there are multiple superframes. The aim of the simulations is to provide some insight on the number of nodes that can be supported without reaching saturation even in the case of deadline miss. For this reason and for the sake of scalability, in all the assessed protocols the network configuration is not based on the worst case analysis, but is heuristically chosen so as to maximize the network throughput while maintaining the lowest possible cycle times and, consequently, to obtain the lowest end-to-end delays.

In the simulated scenario PriMulA is schedulable with up to 40 nodes. When the number of nodes is 50, the lowest priority flows

miss their deadline, as their WCRT is equal to 0.5208s, while their relative deadline is 0.45s.

Table 4.3 shows the LLDN configuration parameters and results. As the messages transmitted by each node in every configuration are

Table 4.3: LLDN configuration

# of Nodes	T_{ts} (ms)	N	T_s (ms)	Ω	**Workload (kb/s)**	DMR
20	2.656	21	55.776	3	46.72	0
30	2.656	31	82.336	3	70.08	0
40	2.080	41	85.280	2	93.44	0.35%
(NSP) 45	2.080	51	95.680	2	105.12	0.67%

the same, in terms of period and length, the timeslot duration depends on the Ω parameter only. We refer to the notation in Table 4.1. Table 4.4 shows the MC-LLDN and PriMulA network configuration parameters and results. For the configurations in which the remain-

Table 4.4: MC-LLDN and PriMulA configurations and results

MC-LLDN configuration							
# of Nodes	C	E	N	T_s (ms)	Ω	**Workload (kb/s)**	**DMR**
20	5	4	7	23.520	4	46.72	0
30	6	5	8	31.744	5	70.08	0.002%
40	8	5	10	39.680	5	93.44	0.025%
50	10	5	12	47.616	5	116.80	1.61%
60	10	6	12	54.912	6	140.16	7.43%
(NSP) 67	11	6	13	59.488	6	156.51	16.89%
PriMulA configuration							
# of Nodes	C	E	N	T_s (ms)	Ω	**Workload (kb/s)**	**DMR**
20	5	4	7	10.752	1	46.72	0
30	5	6	7	15.008	2	70.08	0
40	7	6	9	24.768	3	93.44	0
50	7	7	9	30.240	4	116.80	0
57	8	7	10	45.760	6	133.15	0
64	9	7	11	50.336	6	149.50	0.03%
(NSP) 70	14	5	16	73.216	6	163.52	1.79%

der of the division of *Number of nodes* by C is equal to 1, the spare node is assigned directly to the HLN (e.g., this occurs in the PriMulA configuration with 50 nodes). Moreover, in the PriMulA configuration the unused timeslots in a sub-network are assigned as a second

timeslot to the nodes belonging to the sub-network (starting from the end-node which is assigned the first uplink timeslot) to minimize the message waiting time in the node queue. On the contrary, in the MC-LLDN approach the unused timeslots are not assigned, as the sub-coordinators within a superframe have to transmit one message for each timeslot in the sub-network. Note that the $max(\Omega) = 6$, as higher values would exceed the maximum payload that can be transmitted by the physical layer (i.e., 127 bytes). The configurations in Tables 4.3 and 4.4 already provide some insights on the scalability of the three networks. The network saturation point (NSP) of the addressed approaches significantly varies. It goes from 45 nodes for the LLDN up to 67 and 70 nodes for the MC-LLDN and PriMulA, respectively. Tables 4.3 and 4.4 show that PriMulA outperforms both the LLDN and the MC-LLDN in terms of cycle times (T_s), as shorter sampling periods and end-to-end delays (i.e., from the sensor to the PAN-Coordinator) are achieved.

As far as the DMR is concerned, results in Tables 4.3 and 4.4 show that PriMulA outperforms the LLDN in terms of scalability. In fact, PriMulA allows up to 57 nodes in the overall network without deadline miss and 70 nodes with a deadline miss of 1.79%, while the LLDN allows up to 30 nodes without deadline miss and up to 45 nodes with a deadline miss of 0.67%. The MC-LLDN approach supports a high number of nodes before reaching saturation, but only 20 nodes without deadline miss (Table 4.4). The reason for this is that the MC-LLDN does not provide any prioritization mechanism, hence there is no way to prioritize the messages with shorter relative deadlines. Moreover, Table 4.4 shows that when the number of nodes is less than 60 the MC-LLDN cycle times are higher than those of PriMulA. This is because in the MC-LLDN, in each cycle, the sub-coordinators have to transmit at least one message for each sub-network node (i.e., $\Omega = E$), hence timeslots are larger.

4.5.2 Reliability assessment

Here the standard LLDN protocol and PriMulA are comparatively assessed in a scenario characterized by packet loss and interference. MC-LLDN is not evaluated, as in [59] retransmissions were not addressed.

Packet Loss and Deadline Miss Ratio

Simulations refer to a scenario with 20 nodes randomly placed in a sensing area of 100m x 100m, with the PAN-Coordinator in the center. Simulations were performed with and without retransmissions. The PriMulA sub-coordinators are chosen among the nodes that are placed halfway between their sub-network and the PAN-Coordinator. The result of this choice is that the distances between the PAN-Coordinator and the sub-coordinators are always lower than 20m. Retransmissions are configured using the *separated group ack* defined by the standard [48]. The retransmission timeslots are placed after the uplink ones, so the corrupted messages can be retransmitted within the same superframe. The LLDN protocol retransmissions are scheduled according to the algorithm defined for this purpose in the standard [48]. In this scenario, each node runs an application that generates two messages, m_1 and m_2, with a 16-byte payload and periods $P_1 = 100$ms and $P_2 = 250$ms, respectively. The superframe configurations for both approaches are shown in Table 4.5. In

Table 4.5: Superframe configurations

Retransmissions not enabled								
Network	Mngt.	Uplink	Retr.	N	C	$max(E)$	Ω	T_s (ms)
PriMulA	2	5	0	7	4	5	1	10.30
LLDN	0	20	0	21	-	-	1	30.24
Retransmissions enabled								
Network	Mngt.	Uplink	Retr.	N	C	$max(E)$	Ω	T_s (ms)
PriMulA	2	15	7	18	7	3	3	46.08
LLDN	0	41	20	42	-	-	2	81.98

the case of retransmissions enabled, the PriMulA network consists of

seven sub-networks, each one composed of a maximum of three nodes. With $\Omega = 3$, all the highest priority messages will be transmitted by the sub-coordinator within one superframe. For this configuration of PriMulA, the schedulability analysis presented in Sect. 4.4 gives a worst case response time of 94.72ms for the messages with 100ms period and of 186.88ms for the messages with 250ms period. Packet loss was obtained applying the Log-normal shadowing channel model with $n = 2.04$ and $\sigma = 6.7$, as these values are realistic for the industrial context [68]. This model was chosen as it is adopted in large scale indoor industrial environments [68]. Simulations were repeated six times varying only the seed used for random numbers generation. The simulated time was set equal to 300s to collect statistics on more than 10^5 messages, while the other parameters were unchanged. At the end of simulations, the queues in all the nodes were empty, so all the generated messages were transmitted. Fig. 4.5 shows the simulation results in terms of PLR with and without retransmissions. Results show that with retransmissions both the LLDN and PriMulA obtain PLR values lower than 1%, with negligible differences between them. Without retransmissions the PLR for both approaches increases and PriMulA provides a lower PLR than the LLDN al-

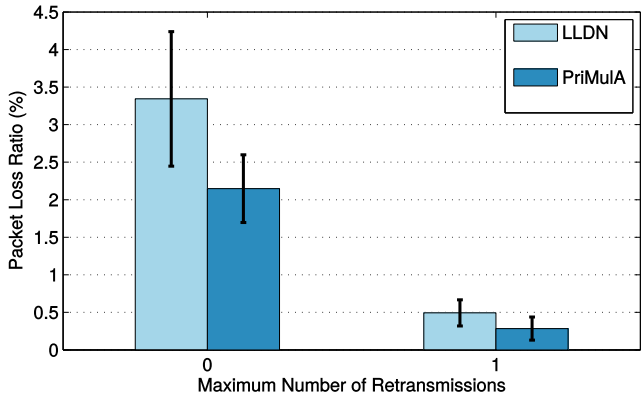

Figure 4.5: Packet Loss Ratio.

though the messages are transmitted twice. In PriMulA the PLR measured on the HLN (i.e., on the links from the sub-coordinators to the PAN-Coordinator) in the case of no retransmissions is very close to zero (i.e., 0.05%), as the sub-coordinators are placed nearby the PAN-Coordinator. Conversely, the PLR of the links between the end-nodes and the sub-coordinators is higher than 2%, as the end-nodes are placed at a longer distance from the sub-coordinators. The distances between transmitters and receiver in the PriMulA scenario, on average, are lower than in the LLDN scenario, as in PriMulA the sub-coordinators are chosen among the nodes that are placed halfway between the sub-network nodes and the PAN-Coordinator, while in the LLDN nodes directly transmit to the PAN-Coordinator. Hence it is the distance between the nodes the parameter that has the major influence on the PLR in the simulation. Different channel models than the Log-normal shadowing might lead to different results, however, the comparison between different channel models is out of the scope of this work.

As far as the DMR is concerned, the results in Table 4.6 show that when retransmissions are not enabled no deadline miss was experienced by PriMulA and the DMR for LLDN is negligible. However, when retransmissions are enabled, PriMulA outperforms the standard LLDN. In fact the DMR for the PriMulA network is always

Table 4.6: Deadline Miss Ratio

Retransmissions not enabled				
Network	**Avg. DMR**	**Min. DMR**	**Max. DMR**	**Std. Dev.**
PriMulA	0%	0%	0%	0%
LLDN	0.015%	0.003%	0.028%	0.010%
Retransmissions enabled				
Network	**Avg. DMR**	**Min. DMR**	**Max. DMR**	**Std. Dev.**
PriMulA	0.42%	0.24%	0.65%	0.17%
LLDN	3.38%	2.22%	4.77%	0.94%

lower than 1%, while in the LLDN it reaches 4.77% in the worst case and 2.22% in the best one. Such a result is due to the higher cycle time of the LLDN. In fact, when a beacon is lost, all the messages

have to wait in the node queue for the next available timeslot in the next cycle, thus the waiting time for the standard LLDN is increased by a cycle time (i.e., 81.98ms) and the messages arrive after their deadline. Conversely, in PriMulA the network cycle time is 46.08ms and consequently a message has to wait a shorter time than in the LLDN.

Network reconfiguration time

For the sake of comparison PriMulA is compared with an improved LLDN that implements the DCC mechanism here proposed. To assess the DCC mechanism, an interferer node was added and the simulations were repeated in the same scenario. The interferer node, after 90s of simulation time, starts to transmit data with an exponentially distributed interarrival time between consecutive transmissions. A simple online algorithm based on the packet loss observed is used to activate the DCC mechanism, but any approach in the literature able to detect an unreliable channel can be adopted. Both the PAN-Coordinator and the sub-coordinators collect the number (RX) of LL-Data frames that arrived within a given number of superframes (DCC_SfN), chosen according to the channel quality variability. The DCC procedure is triggered if this number is below a given fraction (η) of the average RX_i, as in

$$RX_i < \eta * avg(RX_{n=1..(i-1)}), \tag{4.18}$$

where i indicates the ith superframe, while $\eta \in (0, 1]$. In the PriMulA scenario, the DCC was assessed in two cases, i.e., with the interferer on the HLN channel and on a sub-network channel, respectively. In the standard LLDN scenario the interferer affects the channel on which the network operates. Simulation parameters are summarized in Table 4.7. The assessed metric is the reconfiguration time, which is given by the sum of the durations of the discovery and of the configuration phase. Such a value depends on the discovery time defined as a parameter in the IEEE 802.15.4e amendment [48]. In the simulations, the discovery time is set to the minimum value that allows the

Table 4.7: DCC assessment parameters

Network	Interf. Ch.	η	*DCC_SfN*	Discovery Time
PriMulA	11	0.905	11 (HLN)	PanCoord: 16s, SubCoord: 32s
PriMulA	17	0.667	11	PanCoord: 16s, SubCoord: 32s
LLDN	11	0.667	6	32s

discovery of all the subnetwork nodes. Such a value indicates the duration of the discovery state. Table 4.8 shows that the reconfiguration

Table 4.8: DCC Reconfiguration Times

Network	Avg. Reconf. Time	Conf. Interval 95%
PriMulA (HLN)	48.52s	±0.99s
PriMulA (SN)	33.76s	±2.21s
LLDN	34.50s	±0.002s

time is dominated by the discovery phase (whose duration is shown in Table 4.7), while the time for the configuration phase is negligible. PriMulA results show that if the interferer affects a sub-network (SN in Table 4.8), the reconfiguration time is close to the discovery time of the sub-network, while if the interferer is on the HLN channel, the entire network is reconfigured, so the discovery time is close to the sum of the discovery time in the HLN plus the discovery time in the sub-network. In the LLDN with DCC, the reconfiguration time is comparable with that of a PriMulA sub-network.

4.6 Implementation of PriMulA on real devices

A proof-of-concept PriMulA implementation was realized to prove the PriMulA feasibility on COTS devices. The devices are the TelosB motes equipped with a CC2420 radio transceiver, which provides an IEEE 802.15.4-compliant physical layer that adopts the 2.4 GHz DSSS O-QPSK PHY with a datarate of 250 kbps. The implementation is realized using TinyOS. The considered scenario comprises

one PAN-Coordinator, three end-devices and two sub-coordinators, in charge of two sub-networks. The traffic patterns are taken from a realistic industrial scenario [93], N is equal to 6 and the timeslot (T_{ts}) is equal to 1.3 ms, a value very close to the theoretical one (1.2ms). All the statistics were collected every 30s. Under a workload of 3.8 kbps, an average throughput of 3.6 kbps was achieved in a noisy and error-prone environment. The time for the sub-coordinator to commute between the HLN and the sub-network channel was measured equal to 213.6μs and is comparable with the one in the standard (192μs). In the HLN, end-devices and sub-coordinators co-exist and the multichannel approach allows for multiple sub-networks operating in parallel without interferences.

4.7 Conclusions

PriMulA improves the LLDN in several respects. Comparative simulations in realistic scenarios show that the number of nodes that the PriMulA network can support without reaching saturation is increased by 56% compared to the LLDN. The introduction of message priorities not only reduces the cycle time compared to the LLDN and to the MC-LLDN, but it also provides a lower deadline miss ratio. The proof-of-concept implementation on the TelosB motes proves the implementation feasibility without any hardware modification. Future work will address the PriMulA implementation on more complex testbeds, the stochastic analysis [94] and the support for event-driven transmissions.

Chapter 5

A novel MAC protocol for low datarate cooperative mobile robot teams

Current advances in robotic applications and in wireless communication systems are increasing the interest in cooperative mobile robot applications both in academia and in industry. In this applications, the communication system plays a major role, as robots need to share their status information with other robots in order to accomplish a task. Nowadays the commonly used communication technologies and protocols for cooperative robotics are based on the IEEE 802.11 standard (here, called WiFi) in various flavors [95,96].

Recent works and recent applications [97–99] propose the view of cooperating robots as the mobile sensors of a Wireless Sensor Network (WSN) based on low datarate communication technologies (e.g., the IEEE 802.15.4 standard). WiFi protocols are not suitable to be implemented in low datarate devices, as they are specifically devised to operate with high datarate. To cope with the multiple requirements of the considered applications, cooperating robots are generally equipped with multiple different communication devices. For instance, in [98] WiFi and Bluetooth devices support the com-

munication between robots, while IEEE 802.15.4 devices enable the communication with the sensors in the WSN. However, such a design choice entails high costs and complexity. For this reason, a new approach is needed for cooperating robots, i.e., a protocol for low-datarate low-energy communications that can be integrated in WSNs.

The main requirements of networks for cooperative mobile robots (NCMRs) are the following.

Mobility. Mobility support is of outmost importance, as robots have to transmit and receive messages while on the move and regardless of their position.

Bounded delays. In cooperative mobile robot applications, the messages exchanged enable robots to share their status (e.g., position, sensor values, etc.) with the other robots in the network. As a result, these messages have to be consumed within a given time interval, that depends on both the robot speed and the required localization accuracy. Therefore, messages have deadlines (e.g., from tens of milliseconds up to one second), which communication protocols must cope with.

Multi-hop transmissions. Cooperative mobile robot applications generally require that data is transmitted both to all the robots of a network (or sub-network) and to a single robot (e.g., for jointly carrying a load, but also for joint sensory tasks, such as distributed exploration and map building). In some cases, a network managed by coordinators or access points is a suitable choice, while in other cases, due to the limited coverage of wireless devices, multi-hop transmissions are required. To comply with the requirement of bounded network delays, the routing protocols have to be deadline-aware and have to enable nodes to forward data messages in a predictable way.

High reliability. Mobility may affect the quality of the communication links between robots. This fact, in addition to possible interference, noise and fading on the wireless channel, entails a high potential for a large number of message losses. NCMRs cannot properly operate when a high number of messages is lost. For this reason,

reliability is an important requirement that has to be met taking also into account that delay grows with the number of retransmissions. NCMRs have also to provide the capability to tolerate the interference due to external traffic.

This chapter presents RoboMAC, a new MAC protocol for mobile cooperating robots. RoboMAC enables the integration of robots with WSNs, supports real-time communications and mobility, and provides high scalability. RoboMAC was implemented on the STMicroelectronics SPIRIT1 Sub-GHz devices, which operate on less crowded frequencies than the other Industrial, Scientific and Medical (ISM) ones and provide a higher radio coverage.

The chapter is organized as follows. Sect. 5.1 presents related works. Sect. 5.2 describes the main concepts and approaches underpinning the protocol here envisaged and presents possible solutions, while Sect. 5.3 presents the RoboMAC protocol, discussing the implementation choices and the adopted solution. Sect. 5.4 presents the experimental assessment and results. Finally, Sect. 5.5 concludes the chapter and discusses ongoing work.

5.1 Related Work

Research on protocols for mobile low datarate networks mainly focuses on monitoring and WSNs applications. For instance, in [100] a TDMA protocol (called M_TDMA) that supports mobility was proposed. In M_TDMA the network is partitioned into clusters, each one coordinated by a cluster-head node. As a node that joins a cluster requests a slot to the cluster head, a number of slots have to be reserved for this purpose. Moreover, slots have to be reserved to allow inter-cluster communication. However, this reservation entails high message latencies.

In [101] the MobiSense protocol is presented. It provides a hybrid network architecture made up of fixed and mobile nodes organized in a cluster-tree topology. In each cluster, nodes communicate on

different channels. A superframe-based TDMA transmission scheme is adopted. Each superframe provides uplink and downlink slots for data transmission, discovery slots for beacon transmission, and admission mini-slots used for the nodes that want to join a cluster. Such a protocol provides high throughput thanks to multi-channel communication and the adoption of a fast cluster selection mechanism. However, MobiSense is not suitable for cooperative robot communications, as it requires fixed nodes, which may not be always available in these application scenarios.

Other WSNs protocols addressing mobility are presented in [102]. Some of them provide bounded delays, but none of them is able to cope with all the requirements addressed in this paper.

In [103] a protocol for real-time communications in mobile ad-hoc networks is proposed. This protocol provides a TDMA scheduling mechanism in which nodes periodically transmit messages according the Earliest Deadline First (EDF) algorithm. Each node maintains a Communication Requirements Table (CRT), which contains the properties of all the messages to be scheduled in the network. The CRT is updated with a consensus mechanism that enables the network to cope with changes in the message streams and allows nodes to dynamically join or leave the network. To support topology self-checking (i.e., online monitoring of topology changes), synchronization, and admission control, each node periodically broadcasts a message with its CRT, the Neighbor Node Matrix (NNM), the local clock value and other information relevant to the consensus procedure. The protocol in [103] requires one dedicated timeslot per round for the transmission of the CRT and NNM. This choice is suitable in the case of high datarate networks (at least 1-2 Mb/s), while in low datarate networks (up to 256 kb/s) with a high number of nodes (e.g., tens of nodes). This choice increases the NNM update time, the round period and the end-to-end message delay, as all the timeslots have to be sized so as to accommodate the transmission of large frames.

Several other works addressed protocols to support communica-

tions between mobile robots, however, most of them addressed the IEEE 802.11 technology. For instance, in [95], an adaptive TDMA protocol is proposed. Such a protocol partitions the time into cyclic temporal windows, in which robots are allowed to transmit following a TDMA mechanism. Synchronization is based on the reception time instants of the messages transmitted by the other nodes. This protocol performs better when communications go through an access point than in ad-hoc networks. However, the protocol requires a high bandwidth, so it is not suitable for low datarate communications. In [96], the latest version of a distributed protocol called RT-WMP is presented. The protocol operates on IEEE 802.11 networks and consists of three phases. In the first phase, a token circulates between nodes. Such a token contains information about the network topology (through a Link Quality Matrix) and information about the node that holds the highest priority message. The last node that receives the token starts the second phase transmitting an "authorization to transmit" to the node that has the highest priority message, so that it starts data transmission (third phase). The RT-WMP protocol [96] fulfils most of the requirements for cooperative robots applications, as it provides real-time transmissions, mobility support and routing based on the nodes position. However, the the token-passing mechanism, due to its overhead, does not make the RT-WMP protocol the best choice for low datarate networks.

In [104] and [105] an isochronous medium access (IsoMAC) for real-time wireless communications in industrial automation systems was presented as one of the outcomes of the ^{flex}WARE European project [106]. The protocol operates on top of the IEEE 802.11 physical layer and provides a scheduled phase for process data, in which nodes transmit according to a TDMA mechanism, and a contention phase for best-effort and management traffic. The protocol requires access points that transmit the beacon frames containing the schedule information for the nodes. In the ^{flex}WARE architecture mobility is enabled by suitable hand-off mechanisms. The IsoMAC protocol meets the real-time, mobility and reliability requirements of coop-

erative robots networks. However, IsoMAC needs access points to manage communications and this is a drawback in cooperative robot networks, as nodes may lose the network connectivity if they move away from the coverage area of the access points.

This thesis investigates an innovative protocol for cooperative mobile robot applications suitable for low datarate communications and able to cope with all the requirements of the considered applications.

5.2 Design choices

In this section the foundations of a low datarate protocol for cooperative mobile robot applications are addressed.

A typical network topology for these applications consists of nodes that are not fully-linked (i.e., the radio coverage of a node does not reach all the network nodes), as shown in Fig. 5.1. For instance, node 6 is able to receive or transmit messages only from node 5, so if node

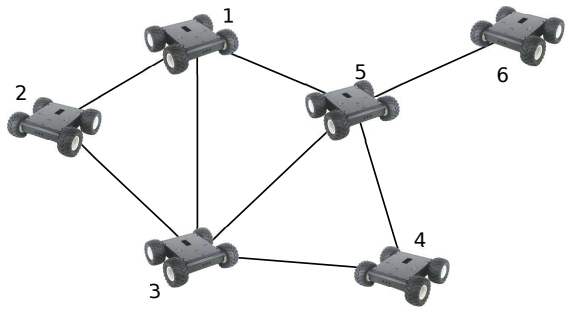

Figure 5.1: Example of a network topology of cooperating robots.

6 moves away from node 5, the latter will lose the connection with the entire network.

5.2.1 Bounded delays

To ensure connectivity in a scenario like the one in Fig. 5.1 a distributed network management approach represents the most suitable

choice. In order to achieve predicable delays, a TDMA-based access mechanism is recommended, as other approaches (like the one proposed in [96]) require high datarate due to the overhead introduced by the network management messages. The approach here proposed is therefore distributed and TDMA-based. The MAC protocol foresees that the time is divided into superframes, which are cyclically repeated. Each superframe is, in turn, divided into slots. A node has assigned one or more slots in which it is allowed to transmit a single frame. In RoboMAC, each node schedules its messages according to their priority. Here static priorities are assumed, which derive from and depend on the application.

Unlike other TDMA approaches (e.g., those in [100, 103]) which provide slots dedicated to the network management messages, here a node is allowed to transmit control information together with data, in the same slot assigned for data transmission. Moreover, to check the neighbor availability and to allow for node synchronization, control data is always transmitted regardless of whether a node has data to transmit or not. In this way, the number of slots in the superframe is significantly reduced, thus allowing for shorter cycle times and, hence, lower message latencies.

5.2.2 Scalability

To efficiently support large networks, in RoboMAC nodes are organized in clusters, depending on their position in the network (i.e., the nodes that are close to each other belong to the same cluster). During the network initialization, the position of network nodes is estimated by exchanging a matrix containing the Received Signal Strength Indicators (RSSI), which holds the link relations between the nodes. In this way the nodes are aware of the network topology. Two transmission channels are used, one for intracluster communications, the other for intercluster ones. Intercluster communications are allowed between the nodes of two clusters when the relevant clusters do not have on-going intracluster communications.

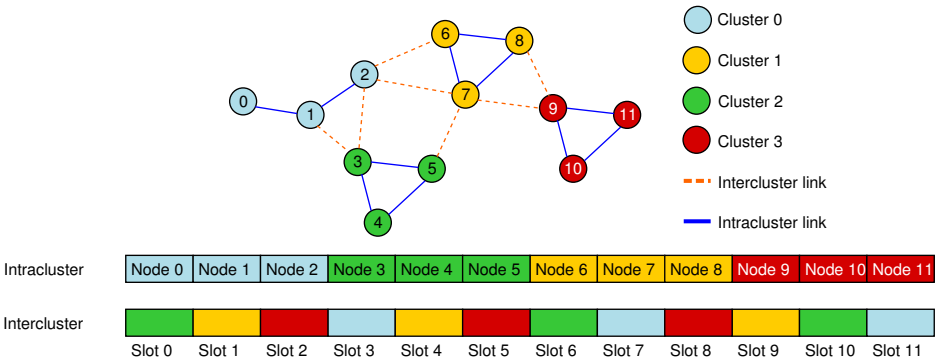

Figure 5.2: RoboMAC network example.

Looking at the example in Fig. 5.2, during the slots assigned to the Cluster 2 nodes (i.e., Nodes 3, 4, and 5), the nodes of Clusters 0, 1 and 3 can communicate on a different channel provided that the message destination does not belong to Cluster 2. Intercluster slots are assigned to the nodes in turn.

5.2.3 Node Mobility

Cooperative mobile robots typically provide relative or absolute localization mechanisms, based, for instance, on the Received Signal Strength Indicator (RSSI), Time-of-Flight, Global Positioning System, etc. The robot position is shared with the other robots and such an information is exploited by several network mechanisms and procedures. For instance, a routing approach based on the node position avoids the need for a network topology discovery mechanism, thus reducing the network overhead. In the case robots are not provided with a localization mechanism, the envisaged approach foresees that they share a bitmap, here called the Network Link Matrix (NLM), that holds the link relations between the nodes. In this way the nodes are aware of the network topology.

RoboMAC provides dynamic clusters, so their composition varies over time. In fact, due to the mobility, the distances change, and so

the RSSI. For this reason, RoboMAC provides a distributed topology management that is based on the RSSI regularly acquired during the communications. Mobility issues, like the unpredictability of the network topology, are solved transmitting either intercluster or intracluster topology information within the header of each frame.

As far as the message routing is concerned, the nodes position can be exploited to adopt position-based routing. For instance, in the case nodes share their position the geographic routing [107], a modified version of the Greedy Perimeter Stateless Routing (GPSR) [108] is an option. Otherwise, in the case nodes shares the NLM, a routing approach based on the lowest number of hops can be adopted. In both cases, the network allows message transmissions in a predictable time and without the need for additional routing messages.

RoboMAC adopts a flooding-based protocol in both the intracluster and intercluster communications. Every time a node receives a message, it tests if the message is for a node of its cluster and, in this case, it sends the message on the intracluster channel. Otherwise the node transmits the message on the intercluster channel. Mobility is supported thanks to the flooding algorithm. In fact, the flooding allows messages to reach all the nodes of the network or cluster. To avoid message retransmission loops due to the flooding approach, the routing algorithm takes into account the message sequence number and the maximum number of hops. Each message received by a node on the intracluster channel or on the intercluster one is retransmitted once to each neighbor until it reaches the destination. Moreover, every node belonging to a cluster periodically shares a matrix containing the RSSI for the messages transmitted in the cluster, so nodes are aware of the physical topology of the cluster. This also allows a message to be routed directly to the destination, thus avoiding the flooding. The intracluster and intercluster communications allow for a reduction of the number of nodes involved in the flooding, thus reducing the number of retransmissions and consequently the transmission overhead generated in the network.

5.2.4 Advantages of the proposed approach

The addressed approach offers several advantages to cooperative mobile robot applications.

Predictable and low delays. The TDMA mechanism not only provides deterministic medium access time, but also allows for the adoption of real-time algorithms to prioritize the messages using either static or dynamic priorities.

Seamless mobility. Thanks to the distributed topology management, which enables the knowledge of the link relations between nodes, messages can be transmitted by mobile robots regardless of whether they are moving or not.

Improved scalability. TDMA mechanisms generally suffer from scalability problems (as the number of slots grows with the number of nodes and so also the message delays grow). To solve this issue, the TDMA-based mechanism combined with multichannel transmissions and clustering allows for shorter superframe and simultaneous transmissions on multiple channels, thus supporting tens of nodes while maintaining bounded message delays in the order of hundreds of milliseconds.

5.3 The RoboMAC protocol

Fig. 5.3 shows the RoboMAC node architecture. The MAC layer provides two sublayers, i.e., the Medium Access and Synchronization sublayer and the Clustering and Routing one. The upper sublayer communicates with the lower one through two prioritized queues (i.e., the IntraCluster and the InterCluster PrioQueue, respectively) for the frame transmissions, and one FIFO queue for the incoming frames.

Each data frame arrived from the Application layer is sent to the Clustering and Routing layer, which processes the data frame, encapsulates it in a routing frame and inserts the frame in the intracluster queue, in the case the message is for the intracluster, or in the intercluster queue, otherwise. Both the intracluster and interclus-

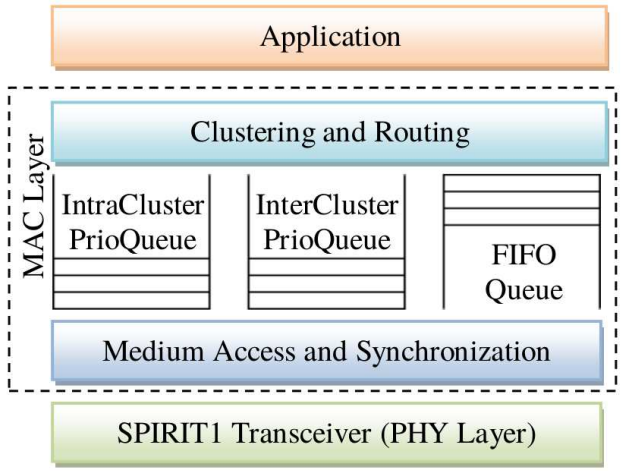

Figure 5.3: RoboMAC node architecture.

ter queues are prioritized. The Medium Access and Synchronization layer extracts the frame from the relative queue and transmits it to the transceiver. In each superframe, the Clustering and Routing layer modifies the superframe configuration by setting for each slot the slot owner, the channel and the relevant associated queue. In this way the Medium Access and Synchronization layer at the beginning of each slot knows the actions to undertake.

5.3.1 Physical Layer

The RoboMAC protocol was specifically devised to work on low datarate devices. The STMicroelectronics SPIRIT1 [109] Sub-GHz device was chosen as it works on frequencies that are less crowded than the 2.4GHz ones, and allows for a higher radio coverage. The adopted development board is the STMicroelectronics STEVAL-IKR002V5 [109] (Fig. 5.4) that is commercially available. The board is composed of a motherboard equipped with a STM32L1 family microcontroller (MCU) and the SPIRIT1 transceiver that operates at 915MHz and

provides a datarate of 250kbps.

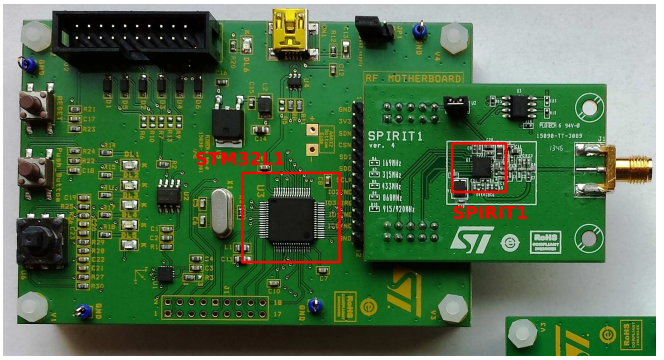

Figure 5.4: STMicroelectronics STEVAL-IKR002V5.

The SPIRIT1 transceiver maintains two FIFO queues, called TX-FIFO and RX-FIFO, 96 bytes long, that contain, respectively, the frame to be transmitted and the incoming one. Moreover, multi-channel communications are allowed as the transceiver provides a function to switch the channel, but the channel switching time is not negligible, as the oscillator should be recalibrated every time[1]. Some measurements proved that the channel switch time plus the recalibration time is no higher than 1.92ms.

The development kit provides an API set to manage the communication between the MCU and the transceiver. Due to the SPI communication, the time delay for each operation between the transceiver and the MCU is non-negligible. For this reason these communication delays were measured. Results are reported in Table 5.1.

In Table 5.1 the T_{ptx} represents the time to transmit a frame from the microcontroller to the TX-FIFO of the transceiver, T_{tx} is the transmission time and T_{rx} is the time required for receiving a frame from the RX-FIFO to the microcontroller.

[1]Recalibration is not mandatory as the calibration values can be measured and set offline. However, the oscillator recalibration leads to a better configuration reducing the communication errors.

Table 5.1: STEVAL-IKR002V5 measured timings.

Payload (Bytes)	$T_{ptx}(\mu s)$	$T_{tx}(\mu s)$	$T_{rx}(\mu s)$	Tot. (μs)
87	900	3400	800	5100
75	800	3100	700	4600
63	700	2700	700	4100
51	600	2300	600	3500
39	500	1900	500	2900
27	400	1500	400	2300
15	300	1100	300	1700
3	200	800	200	1200

5.3.2 Medium access and synchronization

RoboMAC provides a TDMA mechanism in which the time is divided in superframes that, in turn, are divided into slots. Each node can transmit in its assigned slots. A triplet {*slot owner, transmission channel, transmission queue*} is assigned to each slot of the superframe. This way the transmission algorithm knows when it has to transmit, the channel to switch to and the relevant queue from which the frame has to be extracted.

To enable node synchronization, each node transmits the frame at a fixed time (T_g) after the start of the slot and inserts the slot start time in the header of the MAC frame. When a node receives the sync word of the physical frame (i.e., after the first 6 bytes) an interrupt is raised. In the Interrupt Service Routine (ISR) the receiver node calculates the offset of its clock using Eq. (5.1)

$$Offset = CurrentTime - (SlotStartTime + T_g + \frac{6 \times 8}{datarate}). \quad (5.1)$$

where *SlotStartTime* is the timestamp written in the header of the MAC frame by the transmitter node, $(6 \times 8)/datarate$ is the time required to transmit the first 6 bytes and *CurrentTime* is the current clock value of the receiver node. To reduce errors caused by babbling idiot nodes, all the received offsets are collected within a time interval and the clock is periodically adjusted according to the average of the collected offsets.

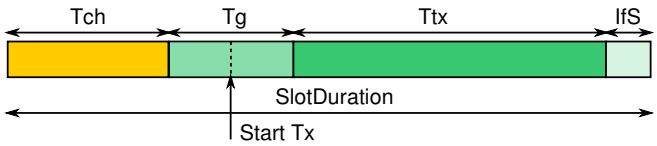

Figure 5.5: RoboMAC slot timings

As shown in Fig. 5.5, in RoboMAC, slots are sized taking into account:

- The maximum time required for channel switching (T_{ch}), which is equal to 1.92ms;
- The maximum between the time required to cope with the synchronization accuracy (T_{sync}) and the maximum time required to transmit the frame from the microcontroller to the transceiver (i.e., $T_g = max(T_{ptx}, T_{sync})$);
- The time needed for data transmission T_{tx};
- The Interframe Space (IfS).

Hence, slots are sized according to the Eq. (5.2)

$$SlotDuration = T_{ch} + T_g + T_{tx} + IfS. \tag{5.2}$$

This way the transmission completion within a slot is guaranteed.

5.3.3 Clustering and routing layer

The Clustering routing layer provides the functionalities of network and topology management. Five functional states are foreseen in this layer, as depicted in Fig. 5.6.

- **Discovery state.** This is the initial state in which each node discovers its neighbor nodes. This state has a fixed duration that is configured offline. The discovery state duration varies according the number of network nodes.

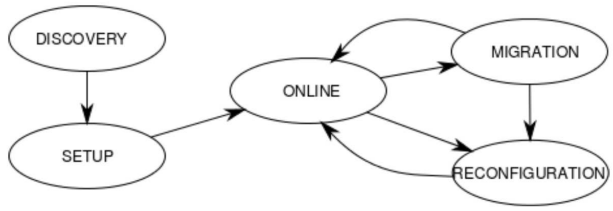

Figure 5.6: Clustering routing layer functional states.

- **Setup state.** When the discovery period is elapsed, nodes enters in the setup state. In this phase all the nodes exchange and update the Network Link Matrix (NLM). At the end of the setup state all the network nodes are aware of the network topology and execute the clustering algorithm, which assigns each node to a cluster. The clustering algorithm runs in each node without message transmissions and takes into account the connectivity relations of nodes provided in the NLM. The result of the clustering algorithm is a vector containing the cluster identifier associated to each node (here called the Cluster Member Vector).

- **Online state.** The Setup state is followed by the online state in which the clustering and routing layer enables the operations of cluster maintenance and routing. The cluster maintenance operation is needed to update the information of the cluster member vector and the cluster topology. In the online state, nodes transmit in the header of the intracluster messages the NLM of the nodes belonging to a cluster. Such an information is used to discover changes in the topology and to start the migration phase or the reconfiguration phase.

- **Migration state.** A node enters in this state when it moves far away from the nodes of its cluster. In this case the node starts the procedure to join another cluster.

- **Reconfiguration state.** This state is reached when there are not the minimum conditions to maintain a cluster and a new clusterization is required, for instance, when a cluster is composed of a single node. The reconfiguration state involves only two clusters. The first cluster is the one to which a node requested the reconfiguration, the second cluster is chosen by this node according to its position and to the number on nodes in the cluster.

Both the Migration and the Reconfiguration states involve a restricted set of network nodes and operate within a bounded time.

5.4 Experimental assessment

Experimental assessments were performed with a twofold aim. First, the packet loss ratio of the RoboMAC protocol was assessed to provide some insight on the reliability of the proposed approach. Second, the RoboMAC protocol was tested in a real multi-robot application specifically devised to prove that the constraints in terms of maximum message delay are meet.

The Packet Loss Ratio (PLR) is calculated as the number of lost messages over the number of transmitted messages, i.e.,

$$PLR = 1 - \frac{NumRxMessages}{NumTxMessages}. \tag{5.3}$$

where *NumRxMessages* is the number of messages correctly received at the application of the destination node, and the *NumTxMessages* is the number of messages transmitted from the application of the source node.

5.4.1 Packet Loss Ratio

The testbed for the measurement was composed of 5 STEVAL-IKR002V5 and one PC. In each experiment the network nodes were placed in a

daisy-chain topology, as shown in Fig. 5.7, with no obstacles between them. This topology is the worst configuration for the RoboMAC protocol, as each node has only two neighbors and the delay of a message from the source to the destination is the highest, due to the fact that a message has to cross all the network nodes to reach the destination.

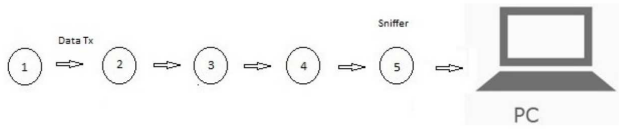

Figure 5.7: Testbed topology for PLR measurements.

In this assessment, the sender node generates N=1000 messages 86-byte long. The payload of each message contains the overall number of generated messages. Each time the destination node receives a message, it calculates the PLR according to Eq. (5.3) and waits for another message. Measurements were repeated varying the number of nodes and thus the number of hops (i.e., from 1 to 3 hops).

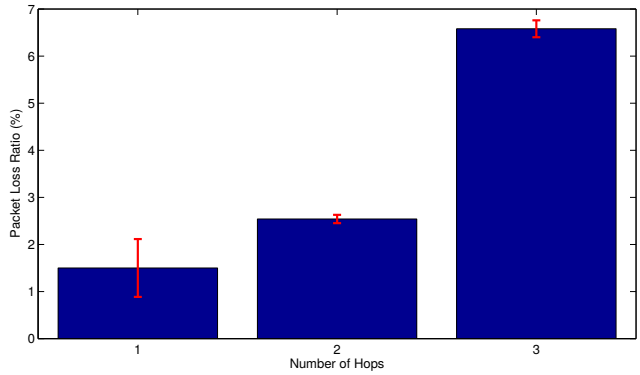

Figure 5.8: PLR results.

Fig. 5.8 shows the results of the PLR measurements. The graph

shows that, in the case of 1 and 2 hops, the PLR is comparable and is lower than 3%. Instead, in the case of 3-hop communication the PLR grows and is close to 7%. However, no consecutive packet losses where experienced.

In cooperating robots applications, typically, nodes share their state adopting the "last-is-best" approach (i.e., no retransmission are needed as the last transmitted state is the best). Due to the high PLR experienced, to share an updated state guaranteeing its arrival with no error at least once within a bounded interval, the status of the nodes has to be transmitted more times it is really required.

5.4.2 RoboMAC test on a real appliction

This experiment was performed with the aim to prove that RoboMAC is able to cope with the real-time constraints in a real multi-robot application. RoboMAC was tested in a simple localization application in which two robots (controlled by a centralized PC) cooperate to search and reach a radio target randomly placed in the environment. The assessed scenario is presented in Fig. 5.9.

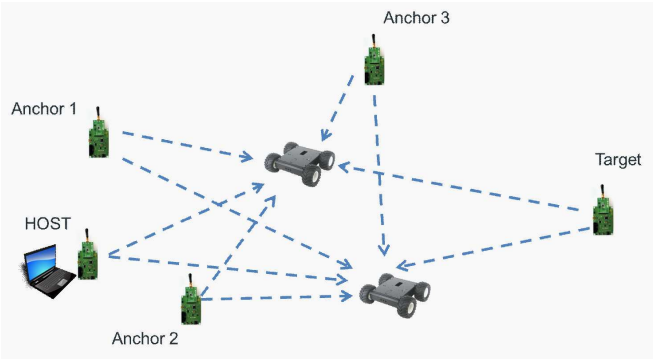

Figure 5.9: Assessed scenario.

The scenario is composed of 7 nodes:

- 2 mobile robots, equipped with a STEVAL-IKR002V5 communication device, one IMU ST iNemo-M1 (Accelerometer, Gyroscope, Magnetometer), 3 ultrasonic sensors and an STM32F4 microcontroller.

- 3 STEVAL-IKR002V5 nodes, which act as anchors and are placed in defined and fixed positions. Each anchor periodically transmits a frame to the robots, to allow them to retrieve the RSSI and thus estimate the distance to the anchor.

- 1 Notebook connected to one STEVAL-IKR002V5 that acts as a host controller for the robots.

- 1 STEVAL-IKR002V5 device randomly placed into the test area. Such a device acts as a target for the robots. It periodically transmits a beacon frame to the robots so as to allow them to estimate their distance to the target.

The robots collects all the RSSI values received from the anchors and from the target. Then, each robot transmits such an information plus its status (i.e., motors status and sensors sampled values) to the host that, in turn, estimates the position of both robots, of the target and transmits the relevant motors command to the robots.

As the host works only with the data collected from the two robots, the classical trilateration method to estimate the position of the target does not provide a single point, but two points, as shown in Fig. 5.10

As it is possible to see in Fig. 5.10, one of the two estimated positions is the one in which the real target is placed, so the host sends one robot to one estimated position and the other one to the second estimated position.

All of these operations are required within 200 ms, that in this case is the minimum cycle time, i.e., the time within which all nodes in the network have to exchange their information/status at least once.

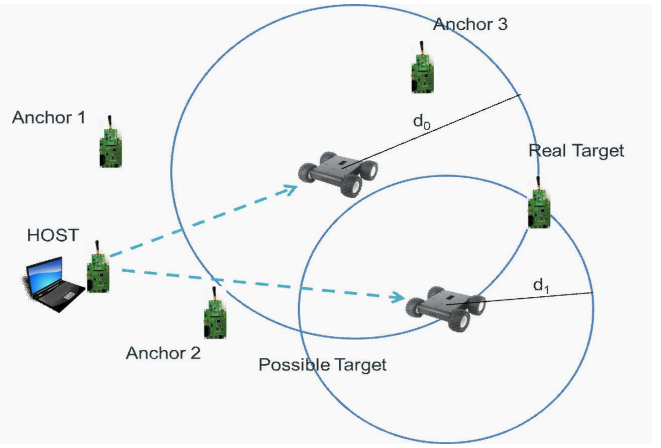

Figure 5.10: Estimated target positions.

In this scenario the superframe is made up of 8 slots. Each node is assigned one slot, with the exception of the host that has assigned two slots for communicating with both the robots. The slot length was configured to be 10ms, so the entire superframe length is of 80ms. This means that the application cannot tolerate 2 consecutive packet losses. In order to test that the real-time constraints are met, if no commands arrive to a robot for more than 200ms, the entire application terminates.

Note that the commands from the controller to the robots are very simple commands (e.g., "move forward with speed Y" or "turn right with speed Z", etc.).

The experiment was repeated more than 10 times and robots never stop moving until they reached the target. Such a result confirms that all messages always arrived to the destination within the deadline (i.e., within 200ms) regardless of the high PLR experienced by the devices. This to confirm that the RoboMAC protocol is able to cope with the real-time constraints imposed by multi-robot applications while supporting the mobility of the nodes.

5.5 Conclusions

This chapter focused on the investigation of an innovative MAC protocol, called the RoboMAC, specifically devised for low datarate networks of cooperating mobile robots. The RoboMAC protocol supports the requirements of mobility, bounded delays and provides high scalability. RoboMAC supports mobility thanks to the flooding mechanism and the distributed topology discovery mechanism, which enable each node to join to a cluster without the need for any centralized coordinator. In particular, mobility issues, like the unpredictability of links between nodes, were solved by combining clustering with flooding transmissions. Bounded delays are achieved through the TDMA mechanism. Another important feature of the RoboMAC protocol is its high scalability. Thanks to the clustering and to the multichannel approach, the network overhead is split between the clusters. Moreover, multichannel communications enable different nodes to communicate at the same time. A proof-of-concept version of the protocol was implemented on the STMicroelectronics Spirit1 Sub-GHz devices and experimental evaluations proved the feasibility of the proposed protocol. Results showed that the protocol provides bounded delays and this makes it suitable for cooperating mobile robot networks. Future work will deal with the investigation of mechanisms to improve reliability through innovative retransmission policies in order to reduce the packet loss ratio. Moreover, energy savings policies will be investigated, as robots are typically battery-powered. Finally, an improvement of the routing algorithm which operates on the RoboMAC network layer will be investigated so as to further reduce the network overhead.

Chapter 6

An Innovative Approach to support Scheduled Traffic in Ad-hoc Industrial IEEE 802.11 networks

Wireless industrial networks have to cope with the requirements that industrial applications impose on communication [110]. Real-time capabilities, i.e., bounded delays for the time-sensitive traffic classes, are one of the properties that these networks have to provide. One of the most commonly adopted wireless technologies for industrial automation applications is the IEEE 802.11 standard [111], that specifies mechanisms to achieve QoS requirements. Among them, the Enhanced Distributed Channel Access (EDCA), which introduces QoS support in the IEEE 802.11 standard providing traffic prioritization, the Hybrid Coordination Function Controlled Channel Access (HCCA), which offers parameterized QoS support, and the Mesh Coordination Function Controlled Channel Access (MCCA), that allows nodes to access the channel at selected times with low contention. However, these mechanisms are not suitable for real-time traffic support, as shown in [112], [113], [114], [115].

For this reason, several approaches were proposed in the literature to support real-time communications over wireless industrial networks. Most of them either operate in managed mode or are based on TDMA mechanisms. However, access points limit the node mobility as, in order to maintain connectivity, there must be an access point in the area a node is moving to, while TDMA-based approaches offer a limited flexibility for the transmission of dynamically added traffic flows (e.g., aperiodic flows).

This work considers a particular class of real-time traffic, called Scheduled Traffic (ST), which is a high priority traffic class that is transmitted according to a fixed schedule. A novel approach is proposed, here called SchedWiFi, that provides a flexible support to the Scheduled Traffic over IEEE 802.11 ad-hoc networks. The aim of SchedWiFi is to provide ST flows with low and bounded end-to-end latency, very low jitter and low packet loss. SchedWiFi modifies the EDCA mechanism introducing the possibility to transmit ST flows in a way that prevents any interference from non-ST traffic thanks to both the introduction of the concept of ST Window and the use of a Time-Aware Shaper (TAS). The ST Window of an ST flow represents the time window in which the transmission of the ST flow is foreseen. The TAS blocks all the transmissions which could interfere with the ST Windows of any ST flow. Thanks to the TAS, the ST Windows of ST flows are temporally isolated, i.e., the medium access is granted to each ST flow in an exclusive way, i.e., no other ST or non-ST flows can transmit within the ST Window of the considered flow.

One main feature of SchedWiFi is that it does not require any predefined superframe structure (as the time is not partitioned in superframes) or timeslots, thus allowing a more flexible transmission of the non-ST flows, which can be transmitted whenever they will not interfere with the ST Window of any ST flow. Moreover, ST Windows have different lengths for different ST flows, based on the flow payload, while when building a superframe the same timeslot size is chosen for all the flows, regardless of the different payloads of their frames. Moreover, SchedWiFi works in ad-hoc mode and does

not need an access point or network coordinator, that represents a single point of failure for the network.

This chapter is organized as follows. Sect. 6.1 outlines related works, while Sect. 6.2 recall the basics of the EDCA mechanism in the IEEE 802.11 standard. Sect. 6.3 introduces the SchedWiFi approach. Sect.6.4 addresses the simulation scenario and the traffic model, while Sect. 6.5 presents the simulation results obtained using the OMNeT++ simulation tool. Finally, Sect. 6.6 concludes the chapter and outlines directions for future work.

6.1 Related Works

Many protocols in the literature propose methods to handle real-time communications over IEEE 802.11 networks. Most of them operate in managed mode, i.e., they require access points that manage the transmissions. For instance, in [105] and [104] a MAC protocol enabling for real-time communications in IEEE 802.11 networks, called IsoMAC, was proposed. The protocol provides a scheduled phase for process data, in which nodes transmit according to a TDMA mechanism, and a contention phase for best-effort and management traffic. In [116], a TDMA-based approach based on the EDCA was proposed. The protocol operates modifying the Contention Window of the EDCA for real-time traffic, that is transmitted according to a TDMA mechanism. In [117] a TDMA mechanism, called RT-WiFi, that adopts a centralized channel and time management to access the channels according to strict timing schedule, is described. RT-WiFi supports predictable, high-speed wireless control systems. However it cannot operate on ad-hoc networks. The paper in [118] presented a mechanism, called Group Sequential Communication (GSC) that uses a Publish/Subscribe paradigm and improves the HCCA. The mechanisms in [104,105,116–118] are able to support real-time transmissions, but require access points and are not suitable for ad-hoc networks. In [119] a wireless adaptation of the dominance proto-

col used in the CAN bus is proposed. However, it does not provide support to scheduled traffic, as the temporal isolation of the highest priority messages is not guaranteed. In [120] and in [121] two approaches for real-time communications based on polling mechanism were presented. Both approaches use an Earliest Deadline First (EDF) scheduler, work on top of the MAC IEEE 802.11 and allow for real-time communications without modifying the standard. However, they introduce a high overhead, due to the polling mechanism. Some approaches to handle real-time traffic on IEEE 802.11 networks in ad-hoc mode (i.e., without the need for access points) were also proposed. In particular, in [122] a Stream Reservation protocol is described. The network nodes use RTS/CTS messages to grant the medium access to a stream. In order to prevent collisions, each node maintains a reservation table that contains the schedule. However, such a protocol is specifically designed for voice traffic and does not address reliability issues. In [123] a distributed mechanism, in which the time is split into two phases, i.e., the scheduled phase and the contention phase, of a so-called virtual frame, is presented. Slots are assigned to the nodes in a distributed way, thus providing collision-free transmissions. The length of the virtual frame is adapted to the workload condition so as to have low delays. The protocol in [123] does not provide any real-time mechanism and may suffer from jitter. Another distributed protocol, called RT-WMP, for real-time communication was investigated in [96]. RT-WMP is specifically designed for mobile robot communications and provides a token passing mechanism. The results proved that RT-WMP works well with a low number of nodes, but it suffers from scalability problems. In [95], an adaptive TDMA protocol is proposed. Such a protocol partitions the time into cyclic temporal windows, in which nodes are allowed to transmit following a TDMA mechanism, and can be used in ad-hoc networks. In [124] a wireless version of the TTEthernet [125] standard that provides support to scheduled traffic is presented. The proposed approach is centralized and needs an access point node. In [126] the real-time behavior is achieved in a topology management

protocol which provides bounded delays, while in [127] a load balancing mechanism was proposed to reduce the load in access points and achieve a better QoS. In [128] an overview of QoS protocols for ad-hoc networks is proposed and the importance of the QoS support in IEEE 802.11 network is proved. Finally, in [129] a wireless traffic smoother was proposed to enable soft real-time traffic over IEEE 802.11 wireless networks. The SchedWiFi approach proposed in this work aims to provide ST flows with low and bounded end-to-end latency, very low jitter and low packet loss and has several nice features. One is the temporal isolation property for the ST flows, which is enforced by the TAS. Second, the enhanced flexibility compared to a fixed superframe structure or timeslots, as the ST Windows have different lengths for different ST flows (based on the flow payload) while when building a superframe the same timeslot size is chosen for all the flows, regardless of the different frame payloads. Moreover, in SchedWiFi the ST flow transmissions follow the application-assigned schedule and therefore non-ST flows are transmitted whenever they do not interfere with the ST traffic transmissions. Conversely, in TDMA-based mechanisms transmissions are bound to their assigned slots and typically there is an over-provisioning of the number of slots (e.g., to enable the superframe to also accommodate aperiodic traffic and periodic flows with periods longer than the superframe length). This way, in TDMA-based mechanisms the network cycle time may be longer than needed. The third relevant property is that, unlike most of the existing approaches (e.g., IsoMAC, RT-WiFi), SchedWiFi works in ad-hoc mode without any access point or network coordinator. This results in higher flexibility, as access points typically are to be connected to a wired network, and higher reliability, as the single-point of failure of the centralized approaches is avoided.

6.2 Background of the EDCA mechanism

The SchedWiFi approach presented in this chapter adopts the Enhanced Distributed Channel Access (EDCA) mechanism defined in the IEEE 802.11e amendment [130], which is part of the IEEE 802.11-2012 standard [111] and offers prioritized QoS support.

The EDCA mechanism provides the following priority classes, called Access Categories (ACs):

- AC_VO: Voice traffic (highest priority).
- AC_VI: Video traffic.
- AC_BE: Best Effort traffic.
- AC_BK: Background traffic (lowest priority).

Each AC has its own transmission queue and the highest priority queue (i.e., the AC_VO queue) is served first. When the AC_VO queue is empty, the AC_VI queue is served, and so on. A node that has data to transmit listens to the channel. If the channel is busy, the transmission will be deferred, otherwise the node listens to the channel for a time interval, called an Arbitration InterFrame Space (AIFS), as shown in Fig. 6.1.

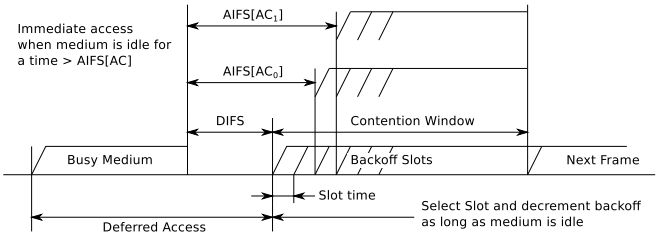

Figure 6.1: EDCA medium access timings [111].

Each AC has a different AIFS, that is calculated as in Formula (6.1)

$$AIFS[AC] = AIFSN[AC] \times SlotTime + SIFS \qquad (6.1)$$

where AIFSN[AC] > 2 is defined according to the AC, while the SlotTime and SIFS are parameters defined in the standard [111] which depend on the Physical Layer adopted.

To avoid collisions of messages belonging to the same AC, a random backoff time is added to AIFSN. The backoff time is within a Contention Window (CW), that is bounded for each AC by the parameters CW_{min} and CW_{max}. In case of retransmissions, the CW is iteratively incremented until it reaches the CW_{max} value for a given AC. To allow the transmission of the highest priority messages before the lower priority ones, the CWs are set so as to avoid overlap between CWs (e.g., $CW_{max}[AC\_VO] \leq CW_{min}[AC\_VI]$). Finally, TXOP is the maximum time duration for a node to transmit after winning access to the channel.

The EDCA mechanism does not provide any guarantee that messages priorities are respected. When the workload increases, the performance of the protocol deteriorates due to the narrow range of backoff values [112]. Moreover, traffic prioritization is not sufficient to handle specific kinds of traffic, such as Scheduled Traffic (e.g., control traffic), which require real-time capabilities and collision-free transmissions. Unlike EDCA, the SchedWiFi approach is specifically designed to offer support to Scheduled Traffic, as it will be discussed in the following Section.

6.3 The SchedWiFi approach

6.3.1 Scheduled Traffic

The Scheduled Traffic in our model is a high priority periodic traffic that is transmitted according to a time schedule so as to ensure no interference from other traffic types. The characteristics of ST flows (period P, frame size L) are fixed and a priori known. The transmission sequence of ST flows is handled off-line by a scheduler which adopts offset-scheduling techniques [131] to avoid collisions between different ST flows. The end-to-end transmission of an ST flow along

its path from source to the final destination, that goes through one or more relay nodes when direct transmission is not possible, occurs within a temporal window, called an ST-Window (STW). During the STW only the messages of the ST flow assigned to the ST-Window can be transmitted from the source node to the destination node along the defined route. Moreover, the nodes that are not involved in the transmission of the ST-messages are not allowed to transmit during the STW, and so collisions are avoided. In Fig. 6.2 a scheduling example with two flows, f_0 and f_1, is shown. There are two STWs, one for each flow, i.e., the STW_0 (dark gray) and the STW_1 (light gray). The two considered flows have two different periods, P_0 and P_1. In our schedule two different offsets, O_0 and O_1, to avoid any possible overlap between the two STWs, are used.

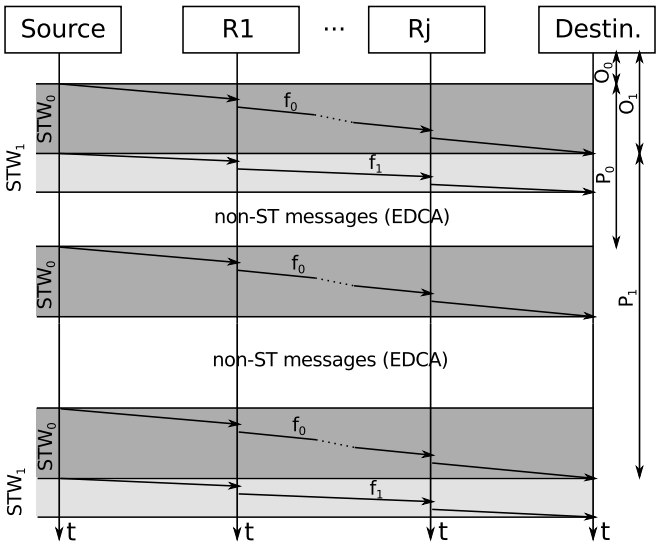

Figure 6.2: Example of ST-Windows.

All the network nodes are aware of the start time and the end time of each ST-Window, through an off-line configuration. This way, no collision may occur even in case of hidden nodes. In order to establish a common notion of time in the proposed approach, a clock synchro-

nization mechanism is required. Several synchronization algorithms are proposed in literature, for example [132–135]. Here nodes are assumed to be synchronized according to the algorithm in [135], which provides a synchronization accuracy of 25 μs in a network with up to 100 nodes.

Each ST-Window has a fixed length and is sized to accommodate the end-to-end message transmissions from the source to the destination. The ST-Window is sized taking into account the message length, the number of hops, the maximum number of retransmissions allowed for each hop, and the synchronization accuracy. The ST-Window sizing will be discussed in Sect. 6.3.2.

As far as the routing policy is concerned, in this approach nodes maintain a static routing table in which the ST flows have fixed routes defined during the deployment phase. The main advantage of such a choice is that in this way the routing delays for the ST traffic are constant and predictable, and the ST-Windows can be easily sized.

To improve fault-tolerance, in SchedWiFi each node has to maintain a routing table with a backup path (with the same number of hops) for each flow. When a sender node does not receive the acknowledgement from the receiver one, the sender transmits the message on the backup path.

6.3.2 Sizing the ST-Window

In the SchedWiFi approach, the use of ST-Window isolates scheduled traffic from the interference of other traffic categories. During this time interval, nodes can only transmit scheduled traffic, hence the ST-Window for an ST flow has to be sized so as to accommodate the transmission of a message of the flow from the source to the final destination, including acknowledgments and retransmissions. The ST-Window size therefore depends on:

- The maximum clock difference between nodes (Δ), that is equal to $\Delta = max\_sync\_error + max\_skew$, where max_skew is the clock difference just before a synchronization round.

- The number of hops (H) from the source to the destination.
- The message size (L).
- The maximum number of retransmissions for each hop (R). Note that, in the SchedWiFi approach a message is acknowledged on each hop, hence the time for the ACK transmissions has to be considered too.

Hence, the ST-Window size is calculated as in Formula (6.2)

$$STW = 2\Delta + (TX_{data} + 2SIFS + TX_{ack})\,(1 + R)H \qquad (6.2)$$

where TX_{data} is the time required for the transmission of L bytes (in industrial contexts L, i.e., the size of ST messages, is small) and TX_{ack} is the time required for the transmission of an ACK message.

6.3.3 Non-ST traffic

The SchedWiFi approach foresees that the non-ST traffic is transmitted out of the ST-Windows and, according to the EDCA rules, after the contention phase, as explained in Sect. 6.2. As a result, delays for non-ST traffic are not predictable. However, in order to provide a different QoS to different flows, messages are prioritized according to their access category, as defined by EDCA protocol specified in the standard [111]. The SchedWiFi approach provides three non-ST traffic Categories:

- **Periodic High Priority (PHP):** The periodic traffic flows with the highest priority among the non-ST flows.
- **Periodic Low Priority (PLP):** The periodic traffic flows with the second highest priority among the non-ST flows.
- **Best Effort (BE):** The aperiodic traffic, with the lowest priority.

The mapping between the SchedWiFi Categories (SC) and the Access Categories (AC) defined by the EDCA protocol in the IEEE 802.11 standard is shown in Table 6.1.

Table 6.1: Mapping between SCs and ACs

SchedWiFi	ST	PHP	PLP	BE
802.11	-	AC_VO	AC_VI	AC_BE

The shorter the periods of the ST flows, the smaller the space between consecutive ST windows, and this could limit the probability to accommodate the transmission of non-ST flows between them. In order to avoid the resulting potential for starvation of the non-ST flows, the non-ST messages can be fragmented according to the fragmentation technique described in [111], thus enabling them to fit the space between two consecutive ST Windows.

6.3.4 Time-Aware Shaper (TAS)

In order to isolate the ST traffic from the non-ST traffic, SchedWiFi provides each node with a Time-Aware Shaper (TAS) that prevents the transmission of the traffic that could interfere with the transmission of ST messages. As shown in Fig. 6.3, the TAS function is implemented just before the frame transmission in the physical layer. The TAS checks whether the transmission duration and acknowledgement reception of a non-ST frame exceeds the start time of a ST-Window. If so, the TAS blocks the transmission and restores the MAC state to the one before the start of the CSMA/CA operations. The TAS also blocks the transmission of ST messages that would exceed their own STW and interfere with other ST transmissions. Note that it is recommendable that the ST windows are consecutive, in order to minimize the bandwidth waste due to the TAS mechanism.

In this way, the ST messages can be transmitted without any interference from both other ST messages and non-ST messages.

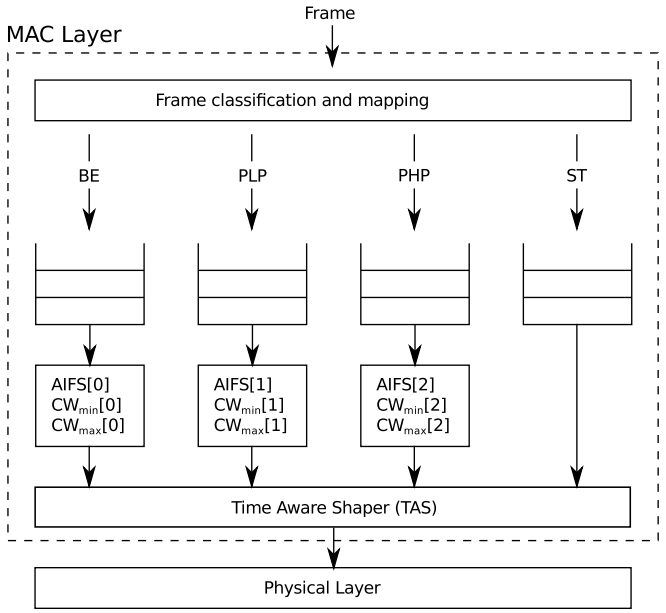

Figure 6.3: Frame flow diagram of SchedWiFi.

6.4 Performance Evaluation

This section describes the scenario and the traffic model used in the OMNet++ simulations.

6.4.1 Simulation Scenario

In Fig. 6.4 the considered set of nodes is shown. The sensing area is 100x100 m^2 large and the SchedWiFI nodes within the area are pseudo-randomly located.

There are three kinds of SchedWiFI nodes in the network:

- The source nodes ($S_{1...6}$, in grey in the picture). They generate and send messages.
- The receiver nodes ($R_{1...3}$, in black in Fig. 6.4). They are the sinks that receive messages.

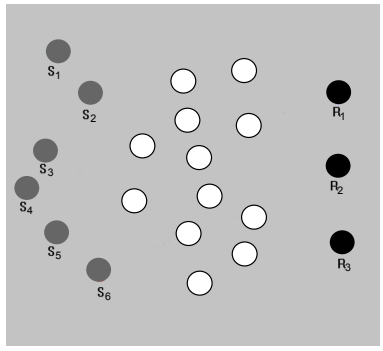

Figure 6.4: Network topology

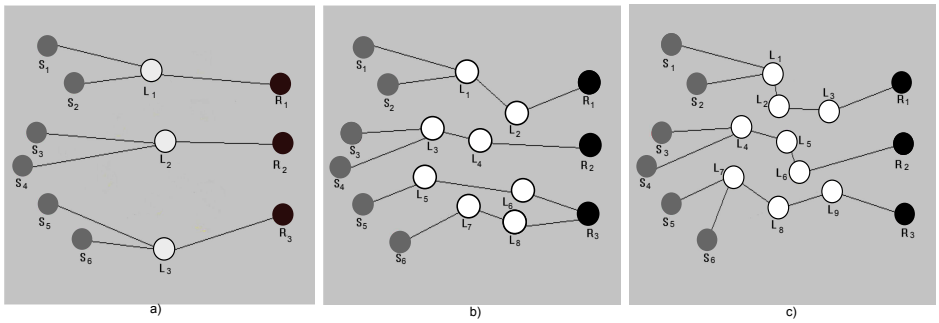

Figure 6.5: a) 2-hop configuration b) 3-hop configuration c) 4-hop configuration

- The relay nodes (in white in Fig. 6.4). They do not generate messages at the application layer, but forward the messages received from other nodes (either source or relay nodes) according to their routing tables.

In the simulation, the start time of each flow is obtained through an exponential distribution with mean equal to 1 s.

The SchedWiFi approach, due to the ST-Windows, requires high bandwidth, as bandwidth is reserved in each node in the entire network for each transmission of ST messages and for the possible retransmissions. For this reason, in this scenario the IEEE 802.11n [136]

physical layer was chosen, which provides a theoretical datarate up to 600 Mb/s. SchedWiFi is also suitable for working on higher datarate physical layers, e.g., the IEEE 802.11ac-2013, which provides a theoretical datarate up to 6.93 Gb/s, while lower datarates PHY types (e.g., the IEEE 802.11b/g) are not recommended for efficiency reasons.

The adopted propagation model is the Log Normal Shadowing [137], with alpha equal to 2.4 and standard deviation sigma equal to 6.7. These values were chosen so as to reflect the signal propagation conditions in realistic industrial environments. A static routing algorithm was chosen, as the network is configured so that the coverage does not allow the source nodes to directly reach the relevant receiver nodes and all nodes in the network do not move, but keep a fixed position assigned during the configuration phase.

Using the same scenario, three different configurations were created and named 2-hop, 3-hop and 4-hop depending on the number of hops that the messages must traverse to reach the receiver nodes. The given scenario includes 12 available relay nodes. The propagation delay due to the distance between the nodes is negligible and does not affect the transmission delay. For each configuration, the relay nodes that are actually used were randomly chosen among all those available in the network. For each configuration, five runs were performed by varying the random parameters, in order to prove that the accuracy of the results is not affected by the use of a probabilistic distribution.

In Fig. 6.5 all the topologies for the 2-hop, 3-hop and 4-hop configuration are shown. Fig. 6.5.a describes the topology for the 2-hop configuration. Three relay nodes were chosen as follows. L_1 forwards the messages from S_1 and S_2, L_2 forwards the messages from S_3 and S_4, while L_3 forwards the messages from S_5 and S_6.

Fig. 6.5.b shows the 3-hop configuration. Eight relay nodes were chosen as follows. L_1 and L_2 forward the messages sent by S_1 and S_2. L_3 and L_4 forward the messages sent by S_3 and S_4. L_5 and L_6 forward the messages sent by S_5. Finally, L_7 and L_8 forward the

Table 6.2: Traffic characterization

Source node	Priority	Payload (byte)	Period (ms)	Destination
S1 - S2	ST	46	5	R1
S1 - S2	Periodic-HP	1200	3	R1
S1 - S2	Periodic-LP	1200	3.5	R1
S3 - S4	Periodic-HP	1200	3	R2
S3 - S4	Periodic-LP	1200	3.5	R2
S5 - S6	Best Effort	1200	-	R3

messages sent by S_6.

Fig. 6.5.c shows the 4-hop configuration. Nine relay nodes were chosen as follows. L_1, L_2 and L_3 forward the messages sent by S_1 and S_2. L_4, L_5 and L_6 forward the messages sent by S_3 and S_4. Finally, L_7, L_8 and L_9 forward the messages sent by S_5 and S_6.

6.4.2 Traffic Model and Evaluation Metrics

Tab. 6.2 shows the traffic characterization and each row describes a traffic flow. The first column shows the source nodes, the second column the traffic class, the third and fourth column the payload and the period, respectively, and the last column the receiver node for the relevant flow. The source nodes S_1 and S_2 every 5 ms generate 46 bytes long ST messages and send them to R_1. In parallel, S_1 and S_2, every 3 ms, send to R_1 periodic high priority (PHP) messages and, every 3.5 ms, periodic low priority (PLP) messages. Both the high and the low priority messages are 1200 bytes long. With the same timings, S_3 and S_4 send to R_2 PHP and PLP messages. Finally, S_5 and S_6 send 1200 bytes long aperiodic best effort messages to R_3.

Table 6.3 shows the workload value for each traffic class.

The considered performance metrics are:

- The end-to-end delay, defined as the time interval between the message creation time and the message delivery time at the destination node, measured at the application layer.

Table 6.3: Workload

Traffic Class	Workload
ST	147.2 Kbps
PHP	6.40 Mbps
PLP	5.48 Mbps

Table 6.4: PHY parameters

Parameter	Layer	Value
Datarate	PHY	300Mbps
Transmitter Power	PHY	2mW
Thermal Noise	PHY	-110dbm
Carrier Frequency	PHY	2.4 GHz
Modulation	PHY	QAM

- The jitter, defined as the difference between two adjacent end-to-end delay values.
- The Packet Loss Ratio (PLR), defined as the ratio between the number of packets lost (due to collisions or bit errors) and the number of packets sent.

Tables 6.4 and 6.5 show the MAC and PHY parameters used in the simulations. In Table 6.5, the AIFS parameter represents the time a node has to wait, listening an idle channel, before making a transmission attempt. CWmin and CWmax are the minimum and maximum length, expressed in slots, of the Contention Windows (CW), i.e., the time window within which the channel contention takes place, for a given access category. If a retransmission occurs, the CW for the relevant access category is increased up to the CWmax value.

6.5 Results

In this section the results obtained for each configuration are presented and discussed. The results are shown with the relevant confidence interval.

Table 6.5: MAC parameters

Access Category	CWmin	CWmax	AIFSN
PHP	7	15	2
PLP	15	31	2
BE	31	1023	3

6.5.1 2-hop configuration

ST Flows delay. In the 2-hop configuration, the maximum end-to-end delay obtained by SchedWiFi for the ST traffic is equal to 48 μs, that is twice the transmission time of a 46 bytes long ST message (i.e., 24 μs). The end-to-end delay of ST messages is bounded by design and in this simulation was found fixed, as there was no need for re-transmissions. This result confirms that the ST Windows mechanism works well for the 2-hop configuration, so predictable delays for the ST traffic can be guaranteed.

Non-ST Flows delay. Fig. 6.6 shows the distribution of the end-to-end delay for the PHP traffic class. Most the end-to-end delay results are in the range [0.2 ms, 0.4 ms]. The results show that most of the PHP messages arrive at destination with an end-to-end delay lower than 1 ms. However, there is also a non-negligible number of messages that experience end-to-end delay values higher than 1 ms. This higher delay is mainly due both to the delay that PHP messages experience in the queues and to the action performed by the TASs. Fig. 6.7 shows the distribution of the end-to-end delay for the PLP traffic class. All the messages are received with an end-to-end delay lower than 6 ms and the average end-to-end delay value is in the range [0.2 ms, 1 ms]. The PHP traffic class presents mean end-to-end delay values lower than those obtained for the PLP traffic class thanks to the traffic prioritization mechanism implemented in the queues, that allows to serve PHP messages with the highest priority among the non-ST flows.

Packet loss ratio and jitter. Table 6.6 shows the packet loss ratio for each traffic class and each configuration, while Table 6.7 shows

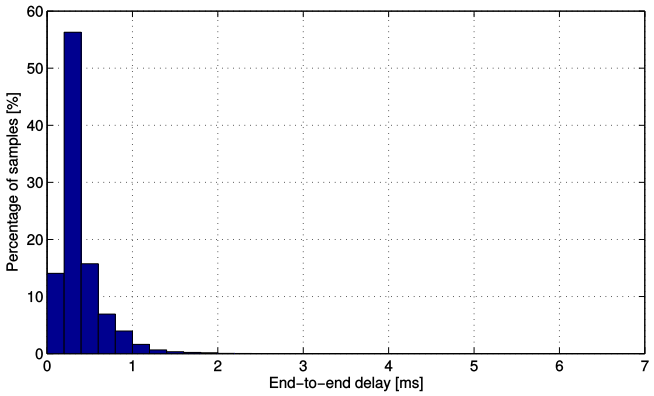

Figure 6.6: End-to-end delay for PHP traffic in the 2-hop configuration.

the mean jitter values obtained for each traffic class and for each configuration, with the relevant confidence interval.

The number of collisions experienced by non-ST flows is very low, especially for the PHP and PLP traffic classes. The packet loss ratio for the ST class is low and is not related to the number of collisions, as it depends entirely on the channel bit error rate. The throughput of the ST traffic, equal to 147 Kbps, is very close to the ST workload and this confirms that the introduced channel noise does not significantly affect the transmissions of ST flows. The jitter value is negligible for the ST class, while for both the PHP and PLP traffic classes the jitter value is higher than 100 μs. This value is still acceptable, as most process automation applications can tolerate jitter values up to 1 ms [138].

Table 6.6: Packet loss ratio for all the traffic classes

Traffic Class	**2-hop**	**3-hop**	**4-hop**
ST	0.19%	0.22%	0.31%
Periodic-HP	0.36%	0.98%	21.64%
Periodic-LP	0.57%	1.49%	25.77%
Best-Effort	1.34%	21.83%	97%

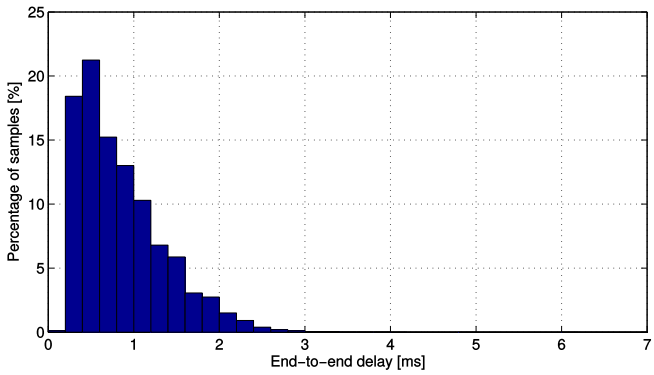

Figure 6.7: End-to-end delay for PLP traffic in the 2-hop configuration.

Table 6.7: Jitter values for all the traffic classes

Traffic Class	**2-hop** (μs)	**3-hop** (μs)	**4-hop** (μs)
ST	0.33 ± 0.003	0.47 ± 0.005	0.5 ± 0.006
Periodic-HP	111.74 ± 1.33	183.73 ± 1.79	1539 ± 44
Periodic-LP	159.85 ± 2.24	342.36 ± 2.95	1844 ± 67

6.5.2 3-hop configuration

ST Flows delay. The end-to-end delay value for the ST class is equal to 72.51 μs, i.e., three times the transmission delay of an ST message, which is 24 μs in the scenario here considered. This confirms that the SchedWifi mechanism works properly even in the 3-hop configuration, guaranteeing an upper bounded end-to-end delay for the ST class.

Non-ST Flows delay. In Figure 6.8 the distribution of the end-to-end delay values for the PHP traffic class in the 3-hop configuration is shown.

The end-to-end delay values for the PHP traffic that most frequently occur are in the range [1.5 ms, 10 ms]. The difference between these results and those obtained with the 2-hop configuration

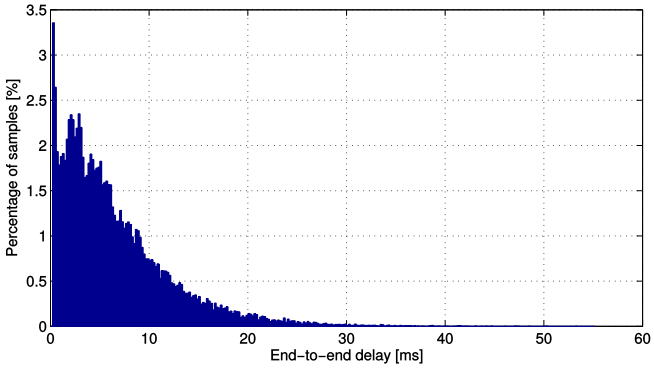

Figure 6.8: End-to-end delay for PHP traffic in the 3-hop configuration.

is mainly due to the increase of the queuing delay for PHP messages, which undergo an additional hop to reach the destination node.

Fig. 6.9 shows the end-to-end delay distribution for the PLP traffic class. More than the 80% of the PLP class messages obtain a value below 50 ms. Comparing these results with those of the 2-hop configuration, in the 3-hop configuration the delay values are in general higher. Moreover, the delay increases more significantly for the PLP traffic class than for the PHP traffic class. This result depends on the queuing time in the relay nodes, where the PLP messages have to wait for the transmission of the PHP messages before being transmitted.

Packet loss ratio and jitter. Table 6.7 shows the mean jitter results and the confidence intervals calculated at the 95%. Similarly to the 2-hop results, the jitter values obtained for the ST class are negligible, while those of both the PHP and PLP messages are lower than 1 ms, i.e., they are higher in this case than in the 2-hop configuration. As shown in Table 6.6, the packet loss ratio for the ST class is low and, as discussed in the 2-hop configuration, is not related to the number of collisions, as it depends on the channel bit error rate. The packet loss ratio for both PLP and PHP traffic classes is lower than

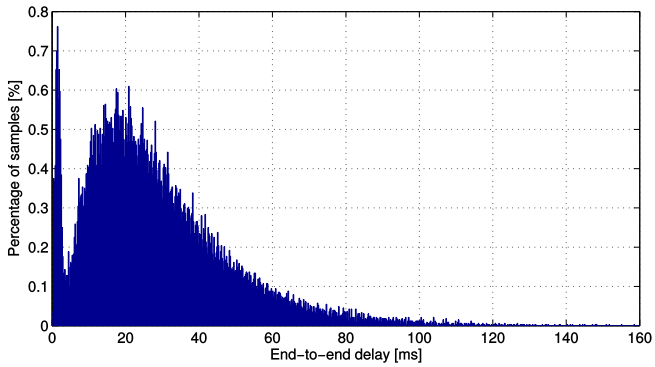

Figure 6.9: End-to-end delay for PLP traffic in the 3-hop configuration.

1.50%, while for the best-effort traffic is equal to 21.83%. This result, which exceeds the 3% threshold required in industrial automation applications [139], is due to both the queue overflow in the relay nodes and the high number of collisions experienced by the best-effort messages.

6.5.3 4-hop configuration

ST Flows. In this configuration the end-to-end delay for the ST class is bounded and equal to 96 ms, i.e., four times the transmission time of a single ST message, and the jitter value is negligible, as shown in Table 6.7. The fourth column of Table 6.6 shows the packet loss ratio for the 4-hop configuration. The result for the ST class is comparable to those obtained for the 2-hop and 3-hop configurations. This proves that the ST messages, thanks to the ST Window mechanism, do not experience collisions when the hop number increases.

Non-ST Flows. Fig. 6.11 and 6.10 show the end-to-end delay distribution for the PHP and PLP traffic class, respectively. Comparing with the results obtained with the previous configurations, here the end-to-end delay values for the PHP traffic class are higher

than in the previous configurations, up to 160 ms, while the 80% of the messages experienced an end-to-end delay lower than 50 ms. The end-to-end delay of the PLP traffic class is significantly increased and reaches quite high values, up to 650 ms, while the 80% of end-to-end delay results for the PLP class are lower than 210 ms. Jitter for non-ST flows also increases in this configuration.

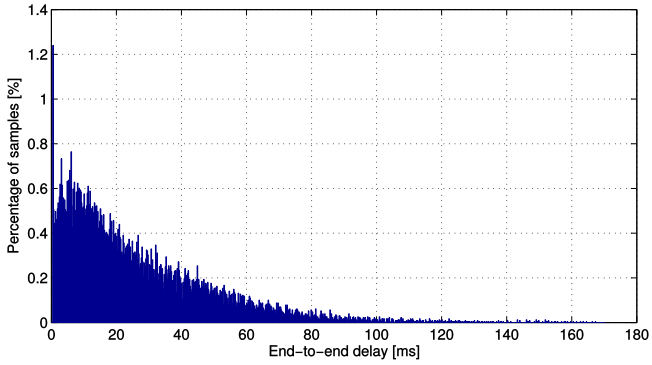

Figure 6.10: End-to-end delay for PHP traffic in the 4-hop configuration.

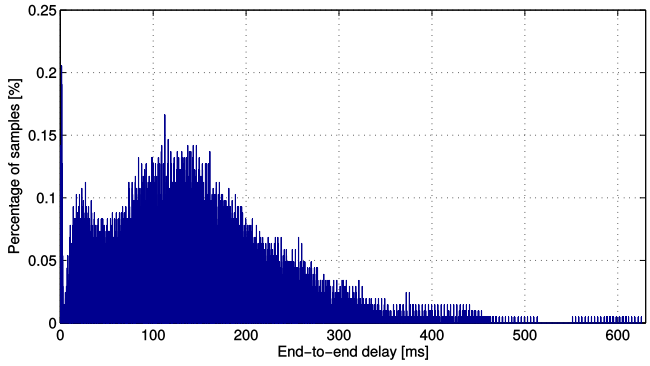

Figure 6.11: End-to-end delay for PLP traffic in the 4-hop configuration.

Table 6.6 shows that the packet loss ratio for the PHP and PLP

traffic class are higher, i.e., equal to 21.64% and 25.77%, respectively. This result is mainly due to the increased number of hops, which increases the probability of collisions for non-ST flows. The packet loss ratio obtained for the best effort traffic class is very high, i.e., close to 97%. This is because the best-effort traffic has the lowest priority, so best-effort flows in such an unfavourable configuration perform worse than in the previous configurations.

Table 6.8 summarizes the end-to-end results of the ST messages for each configuration.

Table 6.8: End-to-end Delay for ST Messages

2-hop	**3-hop**	**4-hop**
$48\mu s$	$72\mu s$	$96\mu s$

In Table 6.9 the average end-to-end delay and the confidence interval calculated at the 95% of confidence level for the simulations are presented. The results show that the confidence intervals are negligible if compared to the order of magnitude of values.

Table 6.9: Average end-to-end delay with confidence interval

Traffic Class	**2-hop** (ms)	**3-hop** (ms)	**4-hop** (ms)
PHP	0.365 ± 0.003	6.600 ± 0.057	26.804 ± 0.303
PLP	0.855 ± 0.005	28.394 ± 0.215	142.492 ± 1.182

Since the entire SchedWiFi approach is distributed and no polling mechanisms are foreseen, scalability depends on the number of ST flows in the network and the lengths of the ST Windows. In fact, the longer the ST Windows, the shorter the time available for non-ST transmissions. The results shown in this Section prove that with up to 17 nodes, 2 ST-flows, 10 non-ST flows and adopting paths long up to three hops, the packet loss ratio is always lower that 22%, even for the traffic with the lowest priority.

Nevertheless, the higher is the number of hops the larger are the ST Windows, which means less time for non-ST transmissions. In this

case, the results proved that the traffic flows with lowest priority, i.e. the best effort traffic, could experience higher packet loss ratio and, eventually, starvation, as shown in Table 6.6. Anyway, it is likely that a large area will be covered by multiple clusters, each one with a small number of hops.

6.6 Conclusions and future works

This chapter presented the SchedWiFi approach that provides support for scheduled traffic over IEEE 802.11 ad-hoc networks.

The SchedWiFi approach was presented and evaluated through simulations in three different configurations, in order to assess both the support provided to ST class and the scalability of the approach with an increasing number of relay nodes between the source and the destination.

Results proved that, as far as the ST class is concerned, in all the considered configurations SchedWiFi allows for low and predictable end-to-end delay values, negligible jitter and very low packet loss ratio. Compared to the MCCA mechanism defined in the IEEE 802.11 standard [111] and assessed in [115], in the SchedWiFi approach non real-time traffic does not affect the end-to-end delay of ST flows. On the other hand, the end-to-end delay values of all the non-ST classes is sensitive to the increase of the number of hops.

Future work will deal with mechanisms of bandwidth reservation for non-ST periodic traffic classes, and in particular for the PHP and PLP flows. In addition, the adoption of a dynamic routing protocol (such as controlled flooding) will be investigated. Comparative evaluations between SchedWiFi and other relevant approaches proposed in literature, such as [123] [96] [95] will be performed. Moreover, using the quality criteria and trade-offs described in [140], the security provided in SchedWiFi will be evaluated and discussed. Finally, as SchedWiFi requires only software modifications to the IEEE 802.11 MAC layer, the feasibility of implementing SchedWiFi on real devices

will be investigated.

Chapter 7

A Bluetooth Low Energy real-time protocol for industrial wireless mesh networks

Bluetooth Low Energy (BLE) is a short-range wireless transmission technology intended for low-power low-cost and low-complexity communications [47] that offers interesting properties for Industrial Wireless Sensor Networks (IWSNs). BLE significantly reduces the power consumption of the communication stack, thus guaranteeing a longer lifetime than other wireless protocols. However, BLE is not suited for supporting real-time traffic, since message delays cannot be precisely bounded. Furthermore, BLE also suffers from other limitations (e.g., on the distance between nodes and on the network topology) that make it difficult to build a BLE mesh network featuring multiple hops. These limitations reduce the extent to which the coverage of a BLE network can be expanded. To overcome these limitations while maintaining the advantages of low-energy and low-cost communications this work proposes the Real-Time BLE (RT-BLE), a real-time protocol for industrial wireless mesh networks developed on top of

BLE. The proposed protocol exploits a Time Division Multiple Access (TDMA) approach with an optimized transmission allocation that provides data packets with real-time support, while maintaining a good trade-off between the maximum guaranteed delay and the throughput.

This work provides a twofold contribution. a) A configuration method for the BLE standard that guarantees bounded message latencies with star topologies; b) a protocol working on top of BLE that allows for meshed topologies while maintaining bounded latencies.

The chapter is organized as follows. Section 7.1 overviews related work, while Section 7.2 describes BLE and the proposed configuration method to achieve bounded latencies on star topologies. Section 7.3 presents the RT-BLE protocol design, while Section 7.4 provides the protocol analysis. Section 7.5 presents some experimental results. Finally, Section 7.6 gives our conclusions and hints for future work.

7.1 Related Work

Many papers investigated the case for Bluetooth-based industrial communications, also proposing improvements and new features [141]. Recently, BLE raised some interest in the scientific community. The paper [142] analytically addresses the maximum peer-to-peer throughput and the minimum turnaround time of a BLE packet, pointing out some limitations of the results achieved by off-the-shelf transceivers compared with the analytical results. Another analytical model for calculating the maximum throughput of BLE was presented in [143], while the work in [144] analyzed and used a simulation tool to estimate the speed and latency of BLE communications. The BLE energy consumption was addressed in [145] and [146], and the results in [145] show that the energy consumption of a BLE transceiver is 2.5 times lower than that of a transceiver based on the IEEE 802.15.4 protocol. The results obtained in all the above mentioned works show that it is worthwhile to investigate novel extensions of BLE. In

particular, it is interesting to find a way to improve the BLE performance on real devices through configurations specifically optimized for industrial environments.

New transceivers allowing for the co-existence of multiple communication stacks make it possible to develop dual protocols. In [147] an approach to use BLE in a dual-protocol environment to support real-time traffic is proposed. However, the approach in [147] provides real-time behavior only for communications between nodes that implement the specific dual protocol addressed. Conversely, the RT-BLE protocol proposed in this chapter works on standard BLE devices.

The BLE standard is limited to a star topology. In order to increase the communication coverage, it is possible to use a network with a tree topology, as it provides for multi-hop relaying. The work in [148] proposes an approach to implement a scalable tree-based network using BLE network that aims at expanding the BLE network coverage, thus increasing its scalability. However, the approach in [148] does not provide data packets with real-time support, as it is intended for Internet of Things (IoT) applications. Conversely, the RT-BLE proposed in this work increases the BLE network scalability while supporting real-time traffic.

7.2 Achieving Bounded Latencies on BLE Networks

7.2.1 Overview on BLE

In BLE once a connection is established, two Link Layer (LL) roles for the devices are defined, i.e., master and slave. The master coordinates the medium access adopting a TDMA-based polling mechanism, in which it periodically polls the slaves. In BLE [47] the time is divided into units called Connection Events (CE). Frequency hopping is realized on each CE, i.e., in each CE a different channel is

used. The connection event starts with the transmission of a data packet from the master to a slave. The start time of a CE is called an Anchor Point (AP). The start times of connection events are regularly spaced, with a configurable interval called a Connection Interval (CI). During a connection event, the master and one slave transmit and receive packets alternately. The connection event is considered to be open as long as the devices continue to transmit packets. If none of the devices has data to transmit, the slave switches to the sleep mode until the next AP. At this point, the master starts the communication with another slave and the process repeats until all the slaves have transmitted. At that point, the master also goes to sleep until the next connection event starts. This way, the master basically splits the connection interval into as many connection events as the number of connections. Consecutive packet transmissions are separated by an Interframe Space (IFS) whose duration is 150 μs. Fig. 7.1 shows an example in which there are two slaves (S1 and S2) and one master (M).

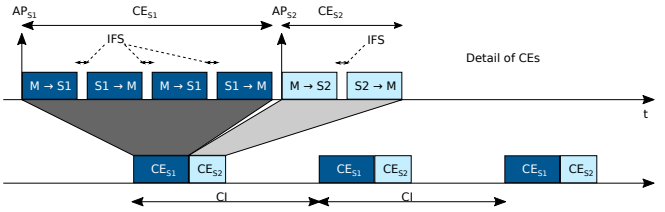

Figure 7.1: Example of the BLE medium access mechanism.

The bottom line of Fig. 7.1 indicates the connection events for the slave S1 and S2, which are periodically repeated with a period equal to CI. The top line of Fig. 7.1 presents the details of each connection event. At the anchor point of S1 (AP_{S1}) the master transmits a packet to S1, thus starting the polling and the alternating transmission sequence (M->S1, S1->M, etc.). At the end of the CE_{S1}, the master starts the connection event for the S2 (CE_{S2}), which is periodically repeated with a period equal to CI. While the CI is the

same for all the slaves connected to the same master, the relevant anchor points are shifted, as the master polls one slave at a time. The BLE specification also defines a parameter, called Connection Slave Latency, which specifies the number of consecutive CEs that a slave has to skip while remaining in sleep mode. The slave shall always send a reply to the master for any packet received, valid or not [47]. The ack for the previous packet is transmitted within the header of the reply to the master.

Several BLE implementations [149] require a guard band (GB) between the connection events to cope with synchronization accuracy. Moreover they provide configurable parameters to set the maximum size of each connection event.

7.2.2 Configuring BLE to obtain bounded latencies

This section explains how to set the main configuration parameters in order to enforce bounded message latencies over BLE networks with a star topology. A mathematical formulation is provided and discussed.

The maximum packet length (L) that can be transmitted at the physical layer is 47 bytes [47], thus the time needed for transmitting it at 1 Mbps, here called T_L, is equal to $376\mu s$. Hence, the duration of the connection event j (T_S^j) that can embed the transmission of M_j maximum-length data packets can be calculated as

$$T_S^j = 2M_jT_L + (2M_j - 1)\mathbf{IFS}. \tag{7.1}$$

Equation (7.1) takes into account the time for Mj master/slave – slave/master transmissions within the j-th CE plus the IFS intervals between the transmissions. Taking into account Eq. (7.1), the maximum time that the master takes for handling S connection events

(T_{trx}) (S is the number of connections) can be calculated as

$$T_{trx} = (S-1)\mathbf{GB}\sum_{j=0}^{S} T_S^j, \tag{7.2}$$

where (S-1) indicates the number of GB intervals between the connection events. The standard specifies that the connection interval can be configured as a multiple of $1.25ms$ in the range from $7.5ms$ to $4s$ [47]. Hence, we have to calculate the minimum multiple of $1.25ms$ that is higher than or equal to T_{trx} so that the minimum connection interval (CI_{min}) for S connection events can be calculated as

$$CI_{min} = max\left(\left\lceil \frac{T_{trx} + \mathbf{IFS}}{1.25ms} \right\rceil 1.25ms, 7.5ms\right). \tag{7.3}$$

The BLE protocol with a suitable configuration (i.e., with bounded connection events) allows for bounded delays when the network has a star topology, so a master can rule the timing and synchronization of all the network nodes. Some issues raise in the networks in which the slaves are connected to multiple masters (i.e., a slave holds connections with multiple BLE networks). In fact, as there are multiple masters and each of them provides synchronization to the associated slaves, it may happen that the CEs assigned to the same slave by different masters overlap. This can happen, for instance, in the topology that is shown in Fig. 7.2. In fact, the node D3 may experience a CE overlap problem, as such a node is connected to two different masters, i.e., the nodes D2 and D4. In the worst case, the two masters transmit a packet to the node D3 at the same time, thus determining a CE total or partial overlap. In particular, one of the four cases depicted in Fig. 7.3 may occur.

In Fig. 7.3A and 7.3B the CE overlap is partial, while in Fig. 7.3C the overlap is total. In Fig. 7.3D no overlap occurs. Section 7.3 proposes a solution to this problem that enables real-time communications over BLE and also offers support for mesh network topologies.

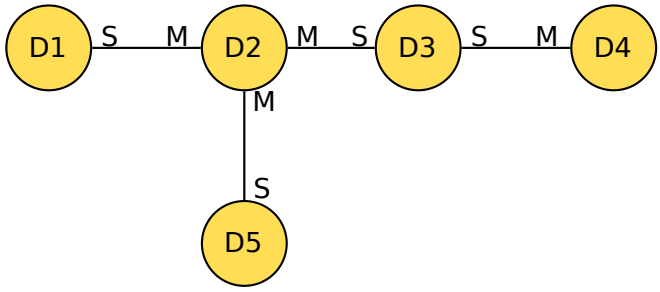

Figure 7.2: Example of topology with a potential for CE overlap.

7.3 Protocol Design

The RT-BLE protocol proposed in this chapter enables BLE to achieve bounded message delays, thus providing support for real-time communications, and also introduces multi-hop data transmissions, thus allowing for the creation of meshed networks. The main idea to achieve this result is to create multiple networks (here called sub-networks), each one coordinated by a master. The sub-networks in turn share one or multiple slaves that act as "bridges" between the sub-networks. Fig. 7.4 shows an example of network topology in which there are three sub-networks, coordinated respectively by nodes 2, 4, and 6. In Fig. 7.4, the node 3 is a slave of both the node 2 (i.e., the master of the sub-network 0) and the node 4 (i.e., the master of the sub-network 1). The node 6 is the master for the nodes 5 and 7, while it is a slave of the node 4. In the example of Fig. 7.4, the node 3 has to communicate with two masters that are not synchronized with each other. Consequently, due to the potential for CE overlap, the transmissions to node 3 cannot be guaranteed.

To deploy a meshed network that avoids the CE overlap, the nodes have to be configured according to the following rules:

1. A node not acting as a master can establish a connection with up to two masters.

2. A node (A) acting as a master can establish a connection with

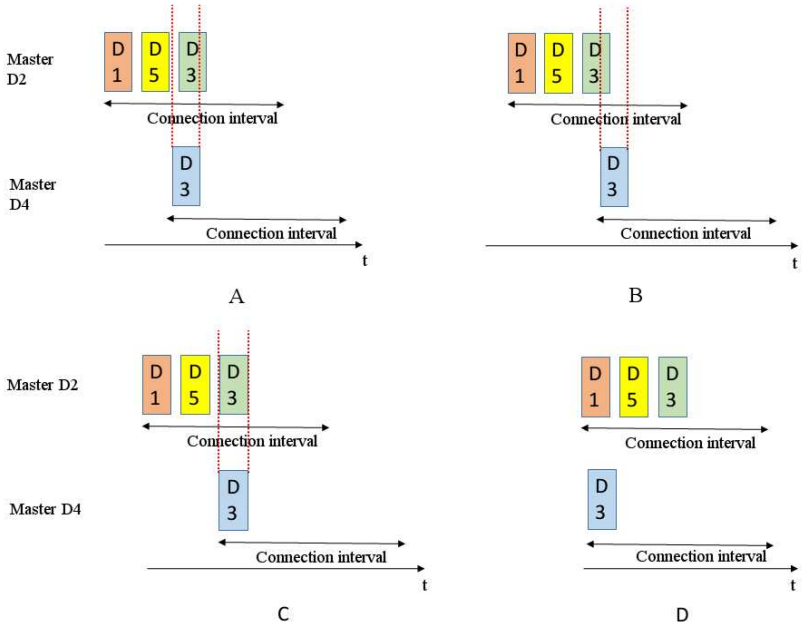

Figure 7.3: Possible overlap of the connection events.

at most another master (B). In this connection, the node A shall play the slave role.

Note that, according to the BLE specifications [47], a device can act both as a master and a slave for different connections and can also establish connections with several masters. Several devices, such as [149], support different roles at the same time.

In order to enable a slave node to be on the right sub-network at the right time, the connection intervals of the different sub-networks have to be the same. To this aim and to avoid CE overlap, the connection interval for each sub-network has to be calculated, using Eq. (7.3). Then the CI for any sub-network, henceforward called CI* is chosen equal to the maximum of the CI_{min} values among all the sub-networks.

The solution proposed in this chapter to solve the problem of

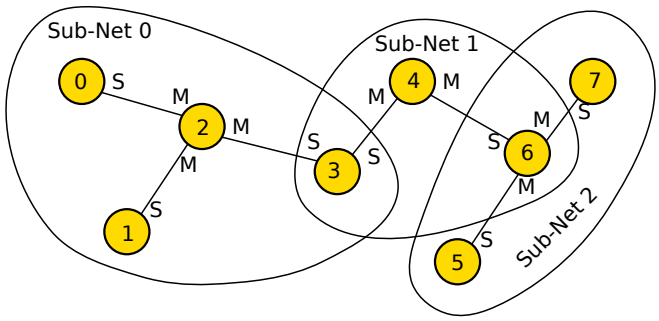

Figure 7.4: Example of RT-BLE network topology.

CE overlap and to enable real-time communications on BLE-based meshed networks is explained through the case-study shown in Fig. 7.5, in which two master nodes are connected with four slave nodes each.

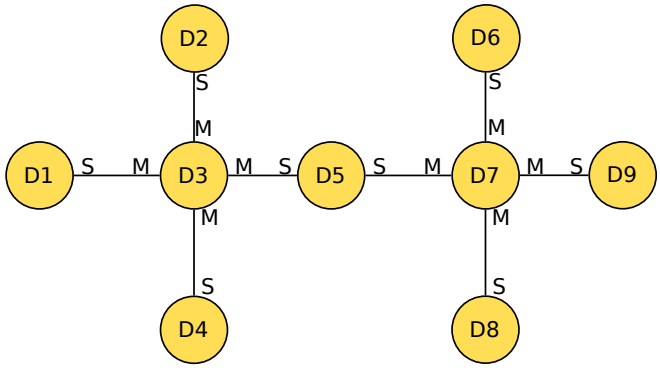

Figure 7.5: Case-study topology for the RT-BLE approach.

The node D5 is a slave connected with the nodes D3 and D7, therefore the connection events of the node D5 may overlap. The solution here proposed is to alternate the connection of the node D5 with a master at a time. This means that in a given connection interval the node D5 will disable the connection with the node D3, while for the next one it will enable the connection with the node

D3 and disable the connection with the node D7. This can be easily realized, as at the higher levels of the BLE stack a mechanism to enable/disable a connection setting the Client Characteristic Configuration Descriptor (CCCD) is provided. This is a two-bit field that works like a real switch. It allows one of the two nodes to enable or disable a connection, notifying both the nodes about the new status of the connection. The slave D5 in Fig. 7.5 can modify these bits at any time and each master, before sending data, has to check that the connection is enabled. If this is not the case, the master inserts the packet in a queue and transmits it in the next connection event. This way the CE overlap is avoided.

Fig. 7.6 shows the timing of the masters D3 and D7. The semi-transparent bigger rectangles represent the intervals during which the node D5 disables the connection with one of the two masters.

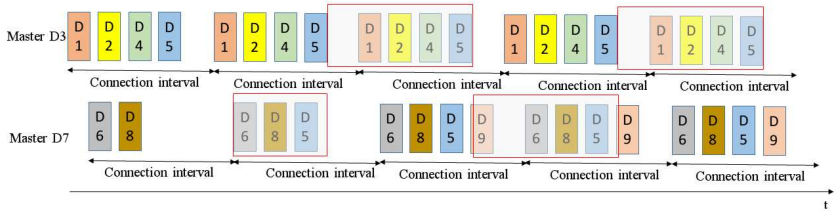

Figure 7.6: The solution to the overlapping CEs problem.

Fig. 7.6 shows that the node D5 is initially connected as a slave to the node D3. Subsequently the master D7, which in the initial phase has only two connections (with the nodes D6 and D8, respectively), establishes a connection with the node D5 (and later with the node D9). The master D7 allocates one CE for the connection with the node D5, which is a shared slave with two connections. Once the connection with the node D7 is established, it is immediately disabled (semi-transparent rectangle). As soon as the node D5 transmissions with the node D3 finish, the node D5 disables the connection with the node D3 for a time equal to the connection interval and enables

the connection with the node D7, for a connection interval. This mechanism repeats over time.

In the proposed approach the network is configured offline. Dynamic configuration will be investigated in future works.

7.4 Timing analysis and analytical assessment

This section presents a timing analysis to determine the end-to-end worst case latencies. We consider a network made up of N nodes. Each node generates multiple messages belonging to different flows (f). Messages are periodically transmitted with a period P_f. The routing path (R_f) for each flow is fixed and configured offline. To improve fault-tolerance, each node has to maintain a routing table with a backup path for each flow (this way the analysis can be repeated for each path). Each flow is thus characterized by the pair (P_f, R_f), where $R_f = (n_0^f, ..., n_h^f)$ is the vector of the nodes (n) that a message of the flow f has to traverse to reach the destination (n_h), while h is the number of hops.

For the sake of simplicity, here it is assumed that each node has assigned one CE within the connection interval. CEs are sized so as to provide room in every connection interval for all the messages that are transmitted and/or forwarded by the node. This way, all the messages in the transmission queue of the node are transmitted within one CE.

According to the approach presented in Sect. 7.3, a node has the opportunity to transmit its messages every two connection intervals and connection intervals have the same length by design o by configuration, as it was explained in Sect. 7.3. Therefore, the maximum time a node has to wait to transmit its messages (WT) is equal to

$$WT = 2CI^* \tag{7.4}$$

where CI^* is the maximum of the CI_{min} values among all the sub-

networks. To calculate CI_{min}, according to Eq. (7.1), (7.2) and (7.3), the maximum number of messages (M_j) that the node j has to transmit is calculated as

$$M_j = \sum_{f:j \in R_f} 1, \tag{7.5}$$

i.e., the number of flows that are generated by the node j plus those that have node j in their routing path (R). Finally, the end-to-end worst case latency for the flow f is calculated as the product of WT and the number of hops (h) the flow f has to traverse to reach the destination, i.e.,

$$RT_f = WT \cdot h^{(f)}. \tag{7.6}$$

7.4.1 Analytical results

This subsection presents an analytical evaluation of the RT-BLE protocol with aim of providing some quantitative performance indicators. The first computed metrics is the minimum connection interval, calculated using Eq. (7.3). The evaluation was performed varying the number of nodes (N) and the number of messages transmitted from each node (M). For the analysis, it was assumed that one CE is assigned to each node (i.e., $S = N$). Results are shown in Fig. 7.7.

Figure 7.7 shows that the minimum connection interval grows linearly with the number of nodes. In particular, in all the evaluated cases, the CI_{min} value is lower than 50ms. Such a value is close to and comparable with the network duty cycle of other IWSN protocols (e.g., those based on the IEEE 802.15.4 with its industrial flavors [78], [80]).

The second computed metrics is the worst case end-to-end latency, calculated as in Eq. (7.6). In the considered scenario, each node periodically transmits one packet to one sink node. Nodes are connected in a daisy-chain topology (Fig. 7.8). Such a topology represents the worst case for our protocol, as increasing the number of nodes, the number of flows for all the nodes in the chain also increases.

The results, shown in Fig. 7.9, are obtained varying the number of hops (h) and the number of chains connected to the sink node (each chain is a network segment, with a daisy-chain topology, connected to the sink node as shown in Fig. 7.8).

Figure 7.9 refers to the case of one subnet and five nodes, where the Node 1 transmits one message to the Node 2, the Node 2 transmits two messages to the Node 3, and so on. Fig. 7.9 shows that the maximum latency is lower than 100ms. In the case of 5 chains (i.e., 5 network segments with 5 nodes each) the maximum latency is lower than 350ms. Such a result proves that the RT-BLE approach makes BLE suitable for industrial WSN, as the maximum latencies achieved are comparable with those required for industrial communications [150].

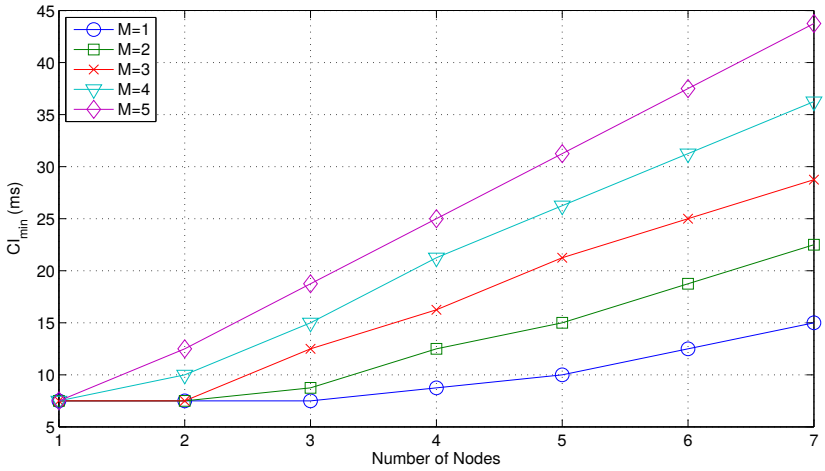

Figure 7.7: Minimum connection intervals (CI_{min}) varying the number of nodes (N) and the number of messages transmitted from each node (M).

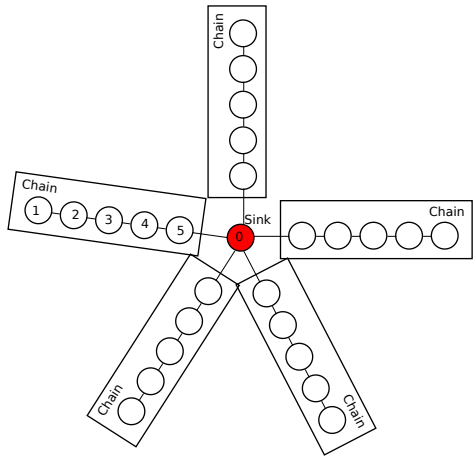

Figure 7.8: Assessed network made up of 5 chains of 5 nodes each.

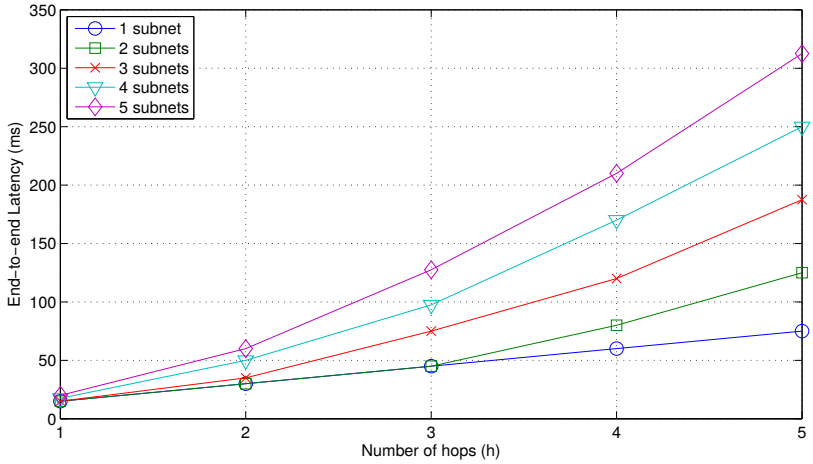

Figure 7.9: Worst case end-to-end latency results.

7.5 Experimental results

This section presents some experiments carried out to characterize the feasibility of the RT-BLE approach on real devices and to measure

the delay experienced by real-time data exchanges.

The testbed was devised so as to obtain the latency bound on a single hop, which represents the most important condition for the feasibility of the proposed approach.

Measurements were performed using the X-NUCLEO-IDB05A1 devices produced by STMicroelectronics. These devices are equipped with SPBTLE-RF BlueNRG-MS, i.e., a communication module compliant with the Bluetooth Specification v4.2 [47].

The testbed was composed of 4 devices, as shown in Fig. 7.10. Device A was the master for the devices B, C and D (slaves). All the slave nodes periodically generated one packet with a period equal to the CI. Only the mechanisms of RT-BLE for single-hop real-time communications were implemented in this phase, to prove that, with a suitable configuration of the CE lengths and of the connection interval, the network provides bounded latencies.

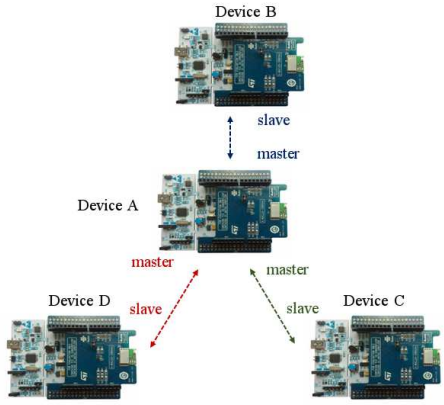

Figure 7.10: Implemented topology.

In the implemented topology, which is shown in Fig. 7.10, the connection interval was set to 20ms, as the Blue-NRG transceiver communicates with the MCU through the SPI port and this entails higher latencies.

The end-to-end delay is the time interval between the sending time of the application packet at the transmitter and the receiving time of the packet at the receiver. The results relevant to the packet end-to-end delay for a connection are shown in Fig. 7.11.

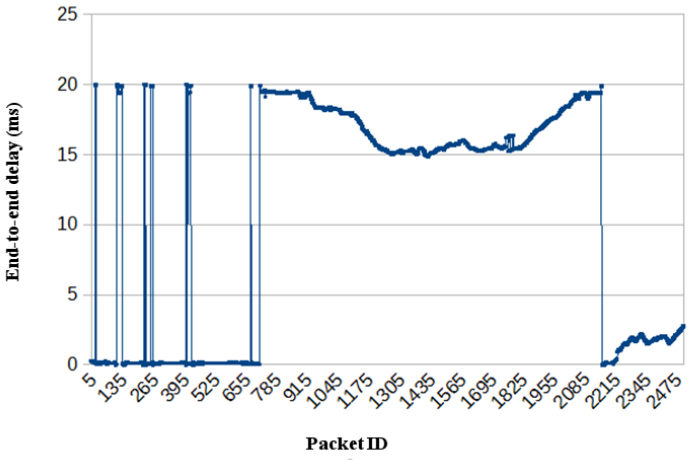

Figure 7.11: End-to-end delay.

As it was expected, the maximum end-to-end delay that was measured is 20ms, so it is equal to the connection interval. Thanks to the configuration methodology presented in Sect. 7.2.2, the end-to-end delay for a packet is always lower than the connection interval configured, as all the messages generated from a node are transmitted in the same connection event and all the connection events are scheduled within a single CI. This result provides an evidence of the latency bound on a single hop, which represents the most important condition for the feasibility of the proposed approach.

7.6 Conclusions

This work proposed RT-BLE, a protocol for industrial wireless mesh networks developed on top of BLE that provides real-time support for data packets and extends the network scalability and coverage. The chapter also presented an analytical evaluation of the connection intervals and of the worst case end-to-end latencies. The results obtained show that RT-BLE allows for latencies that are comparable to those of other IWSN protocols and prove that RT-BLE is suitable for industrial applications. Moreover, a proof-of-concept implementation of the same mechanisms as RT-BLE showed its feasibility on COTS devices.

Further work will investigate a dynamic configuration mechanism combined with load balancing techniques, such as those investigated in [127], to provide more flexibility and to increase reliability and fault-tolerance. Moreover, a full-fledged version of RT-BLE will be developed on COTS devices to assess the protocol in a realistic multi-hop scenario.

Chapter 8

Conclusions and future works

Although the many advances in automation network technologies and protocols, no wired and wireless solutions are able to cope with the different requirements imposed by the whole set of industrial applications. This makes unavoidable the adoption of multiple communication technologies, each one specifically devised for an application field. This mainly originates from the mismatches that the mechanisms adopted to meet specific requirements have with each other. For instance, the mechanisms to improve reliability, as retransmissions or relaying, affect the determinism required for real-time communications.

The work done in this thesis goes right in the opposite direction, i.e., to investigate innovative solutions to meet the requirements of both industrial automation and cooperative robot applications limiting the mismatches between the various mechanisms.

In particular, the problem of scheduling aperiodic real-time messages in an efficient way over EtherCAT networks was solved with the introduction of a novel telegram that is contented among the network nodes and a swapping mechanisms that allows for "preemption" with no message loss. The proposed solution allows for the transmission of both periodic and aperiodic real-time traffic guaranteeing both a cycle time of hundreds of microseconds and the compatibility with

the EtherCAT standard.

Two low datarate wireless solutions to support real-time applications for industrial automation and cooperative robot teams were also proposed. Starting from an assessment of the protocols defined in the IEEE 802.15.4e standard, the first solution, called PriMulA, is specifically devised to improve the scalability, the fault-tolerance and the support to real-time transmission of LLDN networks. Thanks to the introduction of a multichannel mechanism combined with message prioritization, the PriMulA approach, compared to the standard LLDN, provides an improvement in network scalability up to 56% and allows for cycle times up to three times lower. Moreover, the introduction of the dynamic channel configuration and blacklisting mechanism improves the fault-tolerance under radio interference. All of these features were designed so as to maintain the interoperability with LLDN standard devices. However, the LLDN does not support mobility. To cope with this requirement a novel protocol, called RoboMAC was proposed. RoboMAC is devised for low datarate devices and operates on mesh topology. Thanks to the multichannel communication, clustering and a TDMA-based medium access mechanism the RoboMAC protocol proved to be suitable for cooperating mobile robot applications supporting mobility, real-time communications and low latency. A proof-of-concept implementation of the RoboMAC protocol was realized on STMicroelectronics SPIRIT1 devices and the protocol was tested on a real application to validate and demonstrate its effectiveness.

Last, but not least, the challenge of scheduling real-time messages on mesh topologies and using consumer technologies (i.e., IEEE 802.11 (WiFi) and Bluetooth Low Energy) was addressed. The proposed solutions enable these technologies to be integrated in the industrial and robotics scenarios. Two solutions were proposed: Sched-WiFi, which introduces the support for scheduled traffic over IEEE 802.11 networks, and RT-BLE, which enables the support for real-time traffic and allows for mesh topologies over the Bluetooth Low Energy protocol. Assessment results demonstrated the applicability

of the proposed approaches on the industrial automation scenario.

Future works will deal with innovative auto-configuration mechanisms (such as scheduling algorithms, admission control systems and topology management protocols) improve the network flexibility avoiding the need of offline reconfiguration the networks or the applications changes. Self-adaptivity solutions for wireless networks to allow nodes to self-adapt their configuration and capabilities the environmental changes, while considering the the real-time constraints imposed from the applications.

References

[1] Z. Yan, N. Jouandeau1, and A. A. Cherif. A Survey and Analysis of Multi-Robot Coordination. *International Journal of Advanced Robotic Systems*, 10(339):1–18, 2013.

[2] Germany Federal Ministry for Economic Affairs and Energy (BMWi). *Industrie 4.0 - Aspects of the Research Roadmap in Application Scenarios*, April 2016. online available: http://www.plattform-i40.de/I40/Redaktion/EN/Downloads/Publikation/aspects-of-the-research-roadmap.html.

[3] M. Fazio and A. Puliafito. Cloud4sens: a cloud-based architecture for sensor controlling and monitoring. *IEEE Communications Magazine*, 53(3):41–47, Mar. 2015.

[4] T. Sauter, J. Jasperneite, and L. Lo Bello. Towards new hybrid networks for industrial automation. In *IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, Mallorca, Sept. 2009.

[5] Y. Al-Nidawi, H. Yahya, and A.H. Kemp. Tackling Mobility in Low Latency Deterministic Multihop IEEE 802.15.4e Sensor Network. *IEEE Sensors Journal*, 16(5):1412–1427, Mar. 2016.

[6] A. Cuenca, J. Salt, A. Salaand, and R. Piza. A Delay-Dependent Dual-Rate PID Controller Over an Ethernet Network. *IEEE Transactions on Industrial Informatics*, 7(1, 18-29), Feb. 2011.

[7] K. W. Schmidt and E. G. Schmidt. Distributed Real-Time Protocols for Industrial Control Systems: Framework and Examples. *IEEE Transactions on Parallel and Distributed Systems*, 23(10, 1856-1866), Oct. 2012.

[8] A. Pawlowski, A. Cervin, J. L. Guzman, and M. Berenguel. Generalized Predictive Control With Actuator Deadband for Event-Based Approaches. *IEEE Transactions on Industrial Informatics*, 10(1, 523-537), Feb. 2014.

[9] E. Moradi-Pari *et Al.* Design, Modeling, and Simulation of On-Demand Communication Mechanisms for Cyber-Physical Energy Systems. *IEEE Transactions on Industrial Informatics*, PP(99), 2014.

[10] S. Bose, B. Natarajan, C. M. Scoglio, N. N. Schulz, D. M. Gruenbacher, and S. Das. Shipboard Power Systems Reconfiguration—A Cyber-Physical Framework For Response Time Analysis. *IEEE Transactions on Industrial Informatics*, 10(1, 439-449), Feb 2014.

[11] D. M. E. Ingram, P. Schaub, R. R. Taylor, and D. A. Campbell. Performance Analysis of IEC 61850 Sampled Value Process Bus Networks. *IEEE Transactions on Industrial Informatics*, 9(3), Aug. 2013.

[12] W. Kang, K. Kapitanova, and S. H. Son. RDDS: A Real-Time Data Distribution Service for Cyber-Physical Systems. *IEEE Transactions on Industrial Informatics*, 8(2, 393-405), May 2012.

[13] *IEC 61784-2 Ed. 2, ¨Industrial communication networks - Profiles - Part 2: Additional fieldbus profiles for real-time networks based on ISO/IEC 8802-3¨*, 2010.

[14] T. Sauter. The Three Generations of Field-Level Networks—Evolution and Compatibility Issues. *IEEE Transactions on Industrial Electronics*, 57(11, 3585-3595), Nov. 2010.

[15] P. Gaj, J. Jasperneite, and M. Felser. Computer Communication Within Industrial Distributed Environment—a Survey. *IEEE Transactions on Industrial Informatics*, 9(1, 182-189), Feb 2013.

[16] L. Zhang, H. Gao, and O. Kaynak. Network-Induced Constraints in Networked Control Systems—A Survey. *IEEE Transactions on Industrial Informatics*, 9(1, 403-416), Feb 2013.

[17] M. May, D. Ganz, and H.D. Doran. HSR and PROFINET IRT bandwidth management in generic embedded systems. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Krakow, Poland, Sept. 2012.

[18] Z. Hanzalek, P. Burget, and P. Sucha. Profinet IO IRT Message Scheduling. In *Proc. IEEE Euromicro Conference on Real-Time Systems (ECRTS)*, Dublin, Ireland, July 2009.

[19] M. Elshuber and R. Obermaisser. Dependable and predictable time-triggered Ethernet networks with COTS components. *Journal of Systems Architecture*, 59, 2013.

[20] *IEC 61158-3-12 Ed. 2, ¨Industrial communication networks - Fieldbus specifications - Part 3-12: Data-link layer service definition - Type 12 elements¨*, 2010.

[21] J. Jasperneite, M. Schumacher, and K. Weber. Limits of Increasing the Performance of Industrial Ethernet Protocols. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Patras, Greece, Sept. 2007, 17-24.

[22] G. Prytz. A performance analysis of EtherCAT and PROFINET IRT. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Hamburg, Germany, Sept. 2008, 408-415.

[23] D. Orfanus, R. Indergaard, G. Prytz, and T. Wien. EtherCAT-based Platform for Distributed Control in High-Performance Industrial Applications. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Cagliari. Italy, Sept. 2013.

[24] G. Cena, S. Scanzio, A. Valenzano, and C. Zunino. A Distribute-Merge Switch for EtherCAT Networks. In *Proc. IEEE International Workshop on Factory Communication Systems (WFCS)*, Nancy, France, May 2010, 121-130.

[25] G. Cena, A. Valenzano, and C. Zunino. An arbitration-based access scheme for EtherCAT networks. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Hamburg, Germany, Sept. 2008, 416-423.

[26] G. Cena, I. C. Bertolotti, A. Valenzano, and C. Zunino. A high-performance CAN-Like arbitration scheme for EtherCAT. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Mallorca, Spain, Sept. 2009.

[27] A. Di Stefano, A. Gangemi, L. Lo Bello, and O. Mirabella. Slot swapping mechanisms for Process Control Networks. In *Proc. IEEE International Symposium on Industrial Electronics (ISIE)*, Guimaraes. Portugal, July 1997, 143-148.

[28] L. Lo Bello and A. Gangemi. A slot swapping protocol for time-critical internetworking. *Journal of Systems Architecture*, 51, 2005.

[29] A. Di Stefano, A. Gangemi, L. Lo Bello, and O. Mirabella. A slot swapping based fieldbus. In *Proc. Annual Conference of the IEEE Industrial Electronics Society (IECON)*, Aachen. Germany, Aug.-Sept. 1998, 214-219.

[30] G. Patti, L. Lo Bello, G. Alderisi, and O. Mirabella. An EDF-based Swapping Approach to Enhance Support for Asynchronous Real-Time Traffic over EtherCAT networks. In *Proc. IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, Cagliari. Italy, Sept. 2013.

[31] L. Liu and J. Layland. Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the ACM (JACM)*, 20(1, 46-61), 1973.

[32] G. Cena, I. C. Bertolotti, S. Scanzio, A. Valenzano, and C. Zunino. Evaluation of EtherCAT Distributed Clock Performance. *IEEE Transactions on Industrial Informatics*, 8(1, 20-29), Feb. 2012.

[33] Maneesh Soni. *White paper: EtherCAT on Sitara TM AM335x ARM Cortex TM -A8 Microprocessors.* Texas Instruments, November 2011.

[34] Sung-Whan Moon, Jennifer Rexford, and Kang G. Shin. Scalable Hardware Priority Queue Architectures for High-Speed Packet Switches. *IEEE Transaction on Computers*, 49(11), Nov. 2000.

[35] A. K. Mok and X. A. Feng. Towards compositionality in real-time resource partitioning based on regularity bounds. In *Proc. IEEE Real-Time Systems Symposium (RTSS)*, London, U.K., Dec. 2001, 129-138.

[36] M. Joseph and P. K. Pandya. Finding Response Times in a Real-Time System. *The Computer Journal*, 29(5):390–395, Oct. 1986.

[37] K. Gresser. An event model for deadline verification of hard real-time systems. In *Proc. Euromicro Workshop on Real-Time Systems*, Oulu, Finland, June 1993, 118-123.

[38] K. Richter, R. Rocu, and R. Ernst. Scheduling analysis integration for heterogeneous multiprocessor SoC. In *Proc. IEEE Real-Time Systems Symposium*, Cancun, Mexico, Dec. 2003, 236-245.

[39] S. K. Baruah, R. R. Howell, and L. E. Rosier. Algorithms and Complexity Concerning the Preemptive Scheduling of Periodic, Real-Time Tasks on One Processor. *Journal of Real-Time Systems*, 2, 301-324, 1990.

[40] S. K. Baruah, A. K. Mok., and L. E. Rosier. Preemptively scheduling hard-real-time sporadic tasks on one processor. In *Proc. IEEE Real-Time Systems Symposium (RTSS)*, Lake Buena Vista, FL, Dec. 1990.

[41] Sun Lim, Il-Kyun Jung, and Jung-Hoon Kim. A performance evaluation and task scheduling of EtherCAT networked soft motion control system. In *Proc. International Symposium on Robotics (ISR)*, Seoul, Oct. 2013.

[42] S. Vitturi, L. Peretti, L. Seno, M. Zigliotto, and C. Zunino. Real-time Ethernet networks for motion control. *Computer Standards & Interfaces*, 33, 465-476, 2011.

[43] A. Berger, A. Entinger, A. Potsch, and A. Springer. Improving IEEE 802.15.4e LLDN performance by relaying and extension of combinatorial testing). In *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*, pages 1–4, Sept 2014.

[44] V.C. Gungor and G.P. Hancke. *Industrial Wireless Sensor Networks: Applications, Protocols,and Standards*. Boca Raton: CRC Press Taylor Francis Group, 2013.

[45] *IEC 62591 - Wireless communication network and communication profiles - WirelessHART^{TM}*, 2010.

[46] *IEC 62591 - Wireless communication network and communication profiles - ISA100.11a*, 2014.

[47] *Bluetooth Core Specification 4.2*, 2014.

[48] *IEEE Std. 802.15.4e-2012, ¨Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), Amendment 1: MAC sublayer¨*, 2012.

[49] P. Ferrari, A. Flammini, M. Rizzi, and E. Sisinni. Improving simulation of wireless networked control systems based on WirelessHART. *Computer Standards and Interfaces*, 35(6):605–615, 2013.

[50] Z. Pouria, E. Mathews, P. Havinga, S. Stojanovski, E. Sisinni, and P. Ferrari. Implementation of wirelessHART in the NS-2 simulator and validation of its correctness. *Sensors*, 14(5):8633–8668, 2014.

[51] J.M. Winter, I. Muller, C.E. Pereira, S. Savazzi, L. Buss Becker, and J.C. Netto. Coexistence issues in wireless networks for factory automation. In *Industrial Informatics (INDIN), 2014 12th IEEE International Conference on*, pages 370–375, July 2014.

[52] A. Benlghazi, E. Chadli, and D. Moussaid. Bluetooth technologie for Industrial Application. In *International Conference on Information and Communication Systems (ICICS)*, pages 1–5, April 2014.

[53] Y.H. Yitbarek, Kan Yu, J. Akerberg, M. Gidlund, and M. Bjorkman. Implementation and evaluation of error control schemes in Industrial Wireless Sensor Networks. In *Industrial Technology (ICIT), 2014 IEEE International Conference on*, pages 730–735, Feb 2014.

[54] G. Gamba, L. Seno, and S. Vitturi. Performance indicators for wireless industrial communication networks. In *Factory Communication Systems (WFCS), 2010 8th IEEE International Workshop on*, pages 3–12, May 2010.

[55] G. Anastasi, M. Conti, and M. Di Francesco. A comprehensive analysis of the mac unreliability problem in ieee 802.15.4 wireless sensor networks. *Industrial Informatics, IEEE Transactions on*, 7(1):52–65, Feb 2011.

[56] *IEEE Std. 802.15.4-2011 - IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)*, 2011.

[57] Feng Chen, R. German, and F. Dressler. Towards IEEE 802.15.4e: A study of performance aspects. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, pages 68–73, Mannheim, Germany, March 2010.

[58] Bui Xuan Yen, Do Trong Hop, and Myungsik Yoo. Redundant transmission in wireless networked control system over ieee 802.15.4e. In *Information Networking (ICOIN), 2013 International Conference on*, pages 628–631, Bali, Indonesia, Jan 2013.

[59] G. Patti, G. Alderisi, and L. Lo Bello. Introducing multi-level communication in the IEEE 802.15.4e protocol: the MultiChannel-LLDN. In *IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, Barcelona, Spain, Sept. 2014.

[60] T. Paso, J. Haapola, and J Iinatti. Feasibility Study of IEEE 802.15.4e DSME Utilizing IR-UWB and S-Aloha. In *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1863–1867, London, UK, Sept. 2013.

[61] Junhee Lee and Wun-Cheol Jeong. Performance analysis of IEEE 802.15.4e DSME MAC protocol under WLAN interference. In *ICT Convergence (ICTC), 2012 International Conference on*, pages 741–746, Jeju Island, South Korea, Oct 2012.

[62] S. Capone, R. Brama, F. Ricciato, G. Boggia, and A. Malvasi. Modeling and simulation of energy efficient enhancements for IEEE 802.15.4e DSME. In *Wireless Telecommunications Symposium (WTS), 2014*, pages 1–6, Washington, USA, April 2014.

[63] Wun-Cheol Jeong and Junhee Lee. Performance evaluation of IEEE 802.15.4e DSME MAC protocol for wireless sensor networks. In *Enabling Technologies for Smartphone and Internet of Things (ETSIoT), 2012 First IEEE Workshop on*, pages 7–12, Seoul, South Korea, June 2012.

[64] D. De Guglielmo, A. Seghetti, G. Anastasi, and M. Conti. A Performance Analysis of the Network Formation Process in IEEE 802.15.4e TSCH Wireless Sensor/Actuator Networks. In *Computers and Communication (ISCC), 2014 IEEE Symposium on*, pages 1–6, Funchal, Portugal, June 2014.

[65] Deji Chen, Mark Nixon, Song Han, Aloysius K. Mok, and Xiuming Zhu. WirelessHART and IEEE 802.15.4e. In *Industrial Technology (ICIT), 2014 IEEE International Conference on*, Feb 2014, Busan, Korea.

[66] Jianwei Zhou, Ariton E. Xhafa, Ramanuja Vedantham, Ryan Nuzzaci, Arvind Kandhalu, and Xiaolin Lu. Comparison of IEEE 802.15.4e MAC Features. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Mar Seoul, South Korea.

[67] J. Akerberg, Gidlund, M., J. Neander, T. Lennvall, and M. Bjorkman. Deterministic Downlink Transmission in WirelessHART Networks enabling Wireless Control Applications. In *Proc. of 36th Annu. Conf. on IEEE Industrial Electronics Society (IECON 2010)*, pages 2120–2125, Glendale, AZ, USA, 7-10 Nov. 2010.

[68] E. Tanghe, W. Joseph, L. Verloock, L. Martens, H. Capoen, K. Van Herwegen, and W. Vantomme. The Industrial Indoor Channel: Large-Scale and Temporal Fading at 900, 2400, and 5200 MHz. *IEEE Transactions on Wireless Communications*, 7(7):2740–2751, July 2008.

[69] H. Le, J. Van Eck, and M. Takizawa. An Efficient Hybrid Medium Access Control Technique for Digital Ecosystems. *IEEE Transactions on industrial Electronics*, 60(3):1070–1076, Mar. 2013.

[70] H. Yan, Y. Zhang, Z. Pang, and L. D. Xu. Superframe Planning and Access Latency of Slotted MAC for Industrial WSN in IoT Environment. *IEEE Transactions on Industrial Informatics*, 10(2):1242–1251, May 2014.

[71] Wei Shen, Tingting Zhang, F. Barac, and M. Gidlund. PriorityMAC: A Priority-Enhanced MAC Protocol for Critical Traffic in Industrial Wireless Sensor and Actuator Networks. *IEEE Transactions on Industrial Informatics*, 10(1):824–835, Feb. 2014.

[72] V. C. Gungor, B. Lu, and G. P. Hancke. Opportunities and Challenges of Wireless Sensor Networks in Smart Grid. *IEEE Transactions on Industrial Electronics*, 57(10):3557–3564, Oct. 2010.

[73] O. Kreibich, J. Neuzil, and R. Smid. Quality-Based Multiple-Sensor Fusion in an Industrial Wireless Sensor Network for MCM. *IEEE Transactions on industrial Electronics*, 61(9):4903–4911, Sept. 2014.

[74] F. Dobslaw, T. Zhang, and M. Gidlund. End-to-End Reliability-Aware Scheduling for Wireless Sensor Networks. *IEEE Transactions on Industrial Informatics*, PP(99), Dec. 2014.

[75] V. C. Gungor and G. P. Hancke. Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches. *IEEE Transactions on industrial Electronics*, 56(10):4258–4265, 2009.

[76] E. Toscano and L. Lo Bello. Multichannel Superframe Scheduling for IEEE 802.15.4 Industrial Wireless Sensor Networks. *IEEE Transactions on Industrial Informatics*, 8(2):337–350, May 2012.

[77] H. Y. Tung, K. F. Tsang, K. T. Chui, H. C. Tung, H. R. Chi, G. P. Hancke, and K. F. Man. The generic design of a high-traffic advanced metering infrastructure using ZigBee. *IEEE Transactions on Industrial Informatics*, 10(1):836–844, 2014.

[78] G. Alderisi, G. Patti, O. Mirabella, and L. Lo Bello. Simulative assessments of the IEEE 802.15.4e DSME and TSCH in realistic process automation scenarios. In *IEEE Int. Conf. on Industrial Informatics (INDIN)*, Cambridge, UK, July 2015.

[79] M.R. Palattella, N. Accettura, L.A. Grieco, G. Boggia, M. Dohler, and T. Engel. On Optimal Scheduling in Duty-Cycled Industrial IoT Applications Using IEEE802.15.4e TSCH. *IEEE Sensors Journal*, 13(10):3655–3666, Oct. 2013.

[80] E. Toscano and L. Lo Bello. A multichannel approach to avoid beacon collisions in IEEE 802.15.4 cluster-tree industrial networks. In *IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, Mallorca, Spain, Sept. 2009.

[81] T. Semprebom, R. Moraes, C. Montez, P. Portugal, and F. Vasques. Quality of Service Provision Assessment for DDBP Approach in IEEE 802.15.4 Networks. In *IEEE Int. Conf. on Industrial Informatics (INDIN)*, pages 118–123, Porto Alegre, Brazil, July 2014.

[82] G. A. Shah, V. C. Gungor, and O. B. Akan. A Cross-Layer QoS-Aware Communication Framework in Cognitive Radio Sensor Networks for Smart Grid Applications. *IEEE Transactions on Industrial Informatics*, 9(3):1477–1485, Aug. 2013.

[83] M. Collotta, A. Lo Cascio, G. Pau, and G. Scata. A fuzzy controller to improve CSMA/CA performance in IEEE 802.15.4 industrial wireless sensor networks. In *IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, Cagliari, Italy, Sept. 2013.

[84] A. Willig and E. Uhlemann. Deadline-aware scheduling of cooperative relayers in TDMA-based wireless industrial networks. *Wireless Networks*, 20(1):73–88, Jan. 2014.

[85] D. Yang, Y. Xu, H. Wang, T. Zheng, Hao Zhang, Hongke Zhang, and M. Gidlund. Assignment of Segmented Slots Enabling Reliable Real-Time Transmission in Industrial Wireless Sensor Networks. *IEEE Transactions on industrial Electronics*, 62(6):3966–3977, June 2015.

[86] L. Dariz and M. Malaguti, G. Ruggeri. Performance analysis of IEEE 802.15.4 real-time enhancement. In *IEEE Int. Symp. on Industrial Electronics (ISIE)*, pages 1475–1480, Istanbul, June 2014.

[87] *IEEE Std. 802.15.4k-2013, ¨Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs), Amendment 5: Physical Layer Specifications for Low Energy, Critical Infrastructure Monitoring Networks¨*, 2013.

[88] L. Lo Bello and E. Toscano. Coexistence Issues of Multiple Co-Located IEEE 802.15.4/ZigBee Networks Running on Adjacent Radio Channels in Industrial Environments. *IEEE Transactions on Industrial Informatics*, 5(2):157–167, May 2009.

[89] Dong Yang, Youzhi Xu, and M. Gidlund. Coexistence of IEEE802.15.4 based networks: A survey. In *Conf. of IEEE Industrial Electronics Society (IECON)*, Glendale, USA, Nov. 2010, 2107-2113.

[90] J.P. Lehoczky. Fixed Priority Scheduling of Periodic Task Sets With Arbitrary Deadlines. In *IEEE Real-Time Systems Symposium (RTSS)*, pages 201–209, Lake Buena Vista, FL, Dec. 1990.

[91] R. I. Davis, A. Burns, R. J. Bril, and J. J. Lukkien. Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. *Real-Time Systems*, 35:239–272, Jan. 2007.

[92] M. García-Rivera and A. Barreiro. Analysis of networked control systems with drops and variable delays. *Automatica*, 43(12):2054–2059, 2007.

[93] J. Akerberg, M. Gidlund, and M. Bjorkman. Future research challenges in wireless sensor and actuator networks targeting industrial automation. In *IEEE Int. Conf. on Ind. Informat. (INDIN)*, July 2011.

[94] G.A. Kaczynski, L. Lo Bello, and T. Nolte. Deriving exact stochastic response times of periodic tasks in hybrid priority-driven soft real-time systems. In *IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, pages 101–110, Patras, Greece, Sept. 2007.

[95] F. Santos, L. Almeida, and L. S. Lopes. Self-configuration of an adaptive TDMA wireless communication protocol for teams of mobile robots. In *Proc. of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1197–1204, Hamburg, Germany, 15-18 Sept. 2008.

[96] Tardioli, D. et al. A wireless multi-hop protocol for real-time applications. *Computer Communications*, 55, 2015.

[97] A. Koubaa and K. Abdelmajid, editors. *Cooperative Robots and Sensor Networks 2014*. 554. Springer, 2014.

[98] A. M. Derbas, K. M. Al-Aubidy, M. M. Ali, and A. W. Al-Mutairi. Multi-Robot System for Real-Time Sensing and Monitoring. In *Proc. of the International Workshop on Research and Education in Mechatronics (REM)*, Elgouna, Egypt, 9-11 Sept. 2014.

[99] A. Wichmann, B. D. Okkalioglu, and T. Korkmaz. The integration of mobile (tele) robotics and wireless sensor networks: A survey. *Computer Communications*, 51:21–35, 2014.

[100] A. Jhumka and S. Kulkarni. On the Design of Mobility-tolerant TDMA-based Media Access Control (MAC) Protocol for Mobile Sensor Networks. In *Proc. of the 4th International Conference on Distributed Computing and Internet Technology (ICDCIT)*, pages 42–53, Bangalore, India, 2007. Springer-Verlag.

[101] A. Gonga, O. Landsiedel, and M. Johansson. MobiSense: Power-efficient micro-mobility in wireless sensor networks. In *Proc. of the International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, Barcelona, Spain, 27-29 June 2011.

[102] Q. Dong and W Dargie. A Survey on Mobility and Mobility-Aware MAC Protocols in Wireless Sensor Networks. *IEEE Communications Surveys and Tutorials*, 15(1):88–100, 2013.

[103] T. Facchinetti, L. Almeida, G. C. Buttazzo, and C. Marchini. Real-Time Resource Reservation Protocol for Wireless Mobile Ad Hoc Networks. In *Proc. of the IEEE International Real-Time Systems Symposium (RTSS)*, pages 382–391, Lisbon, Portugal, 5-8 Dec. 2004.

[104] H. Trsek and J. Jasperneite. An isochronous medium access for real-time wireless communications in industrial automation systems - A use case for wireless clock synchronization. In *Proc. of International IEEE Symposium on Precision Clock Synchronization for Measurement Control and Communication (ISPCS)*, pages 81–86, Munich, Sept. 2011.

[105] H. Trsek, L. Wisniewski, E. Toscano, and L. Lo Bello. A flexible approach for real-time wireless communications in adaptable industrial automation systems. In *Proc. of IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Toulouse, France, 5-9 Sept. 2011.

[106] flexWARE European project. available on: http://www.flexware.at/.

[107] F. Cadger, K. Curran, J. Santos, and S. Moffet. A Survey of Geographical Routing in Wireless Ad-Hoc Networks. *IEEE Communications Surveys and Tutorials*, 15(10):621–653, 2013.

[108] B. Karp and H. T. Kung. Gpsr: greedy perimeter stateless routing for wireless networks. In *Proc. Annual International Conference on Mobile Computing and Networking (MobiCom)*, page 243–254, New York, NY, USA, 2000. ACM.

[109] STEVAL-IKR002V5: SPIRIT1 - low data rate transceiver - 915 MHz - full kit, Apr. 2014.

[110] A. Willig, K. Matheus, and A. Wolisz. Wireless technology in industrial networks. *Proc. of the IEEE*, 93(6):1130–1151, June 2005.

[111] IEEE. *IEEE std. 802.11-2012 - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, March 2012.

[112] S. Vittorio, G. A. Kaczynski, and L. Lo Bello. Improving the real-time capabilities of IEEE 802.11e through a Contention Window Adapter. In *Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 64–67, 2007.

[113] G. Cena, L. Seno, A. Valenzano, and C. Zunino. On the Performance of IEEE 802.11e Wireless Infrastructures for Soft-Real-Time Industrial Applications. *IEEE Trans. on Industrial Informatics*, 6(3), 2010.

[114] A. Krasilov, A. Lyakhov, and A. Safonov. Interference, Even with MCCA Channel Access Method in IEEE 802.11s Mesh Networks. In *Proc. of IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, pages 752–757, Valencia, Spain, 17-22 Oct. 2011.

[115] C. M. D. Viegas, F. Vasques, P. Portugal, and R. Moraes. Real-time communication in IEEE 802.11s mesh networks: simulation assessment considering the interference of non-real-time traffic sources. *EURASIP Journal on Wireless Comm. and Networking*, 219:1–15, 2014.

[116] R. Costa, P. Portugal, F. Vasques, and R. Moraes. A TDMA-based Mechanism for Real-Time Communication in IEEE 802.11e Networks. In *Proc. of IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Bilbao, 13-16 Sept. 2010.

[117] Y. H. Wei, Q. Leng, S. Han, A. K. Mok, W. Zhang, and M. Tomizuka. RT-WiFi: Real-Time High-Speed Communication Protocol for Wireless Cyber-Physical Control Applications. In *Proc. of IEEE Real-Time Systems Symposium (RTSS)*, pages 140–149, Vancouver, 3-6 Dec. 2013.

[118] Viegas, R. et al. A new MAC scheme specifically suited for real-time industrial communication based on IEEE 802.11e. *Computers and Electrical Engineering*, 39, 2012.

[119] N. Pereira, B. Andersson, and E. Tovar. WiDom: a dominance protocol for wireless medium access. *Industrial Informatics, IEEE Transactions on*, 3:120–130, 2007.

[120] E. Toscano and L. Lo Bello. A Middleware for Reliable Soft Real Time Communication over IEEE 802.11 WLANs. In *Proc. of IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 115–122, Vasteras, 15-17 June 2011.

[121] L. Seno, S. Vitturi, and F. Tramarin. Tuning of IEEE 802.11 MAC for improving real-time in industrial wireless networks. In *Proc. of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Krakow, Poland, 17-21 Sept. 2012.

[122] Xuejun Tian, T. Ideguchi, T. Okuda, and N. Okazaki. Improving throughput of WLANs by scheduling random access. In *Proc. of International Symposium on Wireless and Pervasive Computing (ISWPC)*, Hong Kong, 23-25 Feb. 2011.

[123] A. Arkoub, U. A. Khan, and J. Seitz. Stream reservation MAC protocol for wireless ad-hoc networks. In *Proc. of IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, Amman, Jordan, 3-5 Dec. 2013.

[124] P. Gutierrez Peon, H. Kopetz, and W. Steiner. Towards a reliable and high-speed wireless complement to ttethernet. In *IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–4, Sept 2014.

[125] Wilfried Steiner, Gunther Bauer, Brendan Hall, and Michael Paulitsch. Ttethernet: Time-triggered ethernet. In R. Obermaisser, editor, *Itime Triggered Communication*. CRC Press, 2011.

[126] E. Toscano and L. Lo Bello. A topology management protocol with bounded delay for Wireless Sensor Networks. In *Proc. of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 942–951, Hamburg, Germany, 15-18 Sept. 2008.

[127] M. Collotta, L. Lo Bello, E. Toscano, and O. Mirabella. Dynamic load balancing techniques for flexible wireless industrial networks. In *Proc. of the Annual Conference on IEEE Industrial Electronics Society (IECON)*, pages 1329–1334, Glendale, AZ, USA, 7-10 Nov. 2010.

[128] M. Natkaniec, K. Kosek-Szott, S. Szott, and G. Bianchi. A survey of medium access mechanisms for providing qos in ad-hoc networks. *IEEE Communications Surveys and Tutorials*, 15(2), 2013.

[129] L. Lo Bello, G. A. Kaczyński, F. Sgró, and O. Mirabella. A wireless traffic smoother for soft real-time communications over IEEE 802.11 industrial networks. In *Proc. of the IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1073–1079, Prague, Czech Republic, 20-22 Sept. 2006.

[130] IEEE. *IEEE Std. 802.11e-2005 - Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements*, Nov. 2005.

[131] K. Tindell. Adding Time-Offsets to Schedulability Analysis. Technical Report YCS221, Dept. of Computer Science, York University, 1994.

[132] D Zhou, L. Huang, and T. H. Lai. On the scalability of IEEE 802.11 ad-hoc-mode timing synchronization function. *Wireless Networks*, 14(4):479–499, Aug. 2008.

[133] A. Mahmood, R. Exel, and T. Sauter. Delay and Jitter Characterization for Software-Based Clock Synchronization Over WLAN Using PTP. *IEEE Transactions on Industrial Informatics*, 10(2), 2014.

[134] J. P. Sheu, C. M. Chao, and C. W. Sun. A clock synchronization algorithm for multi-hop wireless ad hoc networks. In *Proc. of the 24th International Conference on Distributed Computing Systems*, 2004.

[135] D. Zhou and T. H. Lai. An Accurate and Scalable Clock Synchronization Protocol for IEEE 802.11-Based Multihop Ad Hoc Networks. *IEEE Transactions on Parallel and Distributed Systems*, 18(12):1797–1808, Dec. 2007.

[136] *IEEE 802.11n-2009 - Amend. 5: Enhancements for Higher Throughput.*

[137] Stuedi P. and Alonso G. Log-normal shadowing meets SINR: A numerical study of Capacity in Wireless Networks. In *4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, San Diego, CA, 2007.

[138] Richard Zurawski, editor. *Industrial Communication Technology Handbook, Second Edition.* CRC Press, 2014.

[139] H. Zha and X. Chen and Y. Fang, . A Call Admission and Rate Control Scheme for Multimedia Support over IEEE 802.11 Wireless LANs. In *Proceedings of the 1st International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QSHINE)*, Dallas, TX, 2004.

[140] Elena Lisova, Elisabeth Uhlemann, Johan Åkerberg, and Mats Björkman. Towards secure wireless ttethernet for industrial process automation applications. In *IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–4, Sept 2014.

[141] M. Collotta, L. Lo Bello, and O. Mirabella. An innovative frequency hopping management mechanism for Bluetooth-based industrial networks. In *IEEE International Symposium on Industrial Embedded Systems (SIES)*, pages 45–50, Trento, Italy, July 2010.

[142] K. Mikhaylov, N. Plevritakis, and J. Tervonen. Performance Analysis and Comparison of Bluetooth Low Energy with IEEE 802.15.4 and SimpliciTI. *Journal of Sensor and Actuator Networks*, pages 589–613, 2013.

[143] C. Gomez, I. Demirkol, and J. Paradells. Modeling the Maximum Throughput of Bluetooth Low Energy in an Error-Prone Link. *IEEE Commun. Lett.*, 15:1187–1189, 2011.

[144] C. Gomez, J. Oller, and J. Paradells. Overview and Evaluation of Bluetooth Low Energy: An Emerging Low-power Wireless Technology. *Sensors*, 12:11734–11753, 2012.

[145] S. Kamath and J. Lindh. Measuring bluetooth low energy power consumption. Technical report, Texas Instruments, inc., Dallas, TX, USA, 2012.

[146] M. Siekkinen, M. Hiienkari, J.K. Nurminen, and J. Nieminen. How Low Energy is Bluetooth Low Energy? Comparative Measurements with ZigBee / 802.15.4. In *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, page 232–237, Paris, France, Apr. 2012.

[147] M. Marinoni, G. Franchino, D. Cesarini, A. Biondi, P. Buonocunto, and G. Buttazzo. Dual-protocol support for Bluetooth LE device. In *International Conference on Industrial Informatics (INDIN)*, pages 919–922, Cambridge, UK, July 2015.

[148] J. Kim, S. Kang, and J. Park. Bluetooth-based Tree Topology Network for Wireless Industrial Application. In *International Conference on Control, Automation and System*, pages 1305–1308, Busan, South Korea, Oct. 2015.

[149] BlueNRG, BlueNRG-MS stacks programming guidelines. online available.

[150] A. Flammini, P. Ferrari, D. Marioli, E. Sisinni, and A. Taroni. Wired and wireless sensor networks for industrial applications. *Microelectronics Journal*, 40(9):1322–1336, 2009.

Biography

Gaetano Patti received the M.S. degree (*summa cum laude*) in computer engineering at the University of Catania, Italy, in July 2013. At the end of 2013, he awarded a PhD scholarship at the University of Catania funded by STMicroelectronics for the research project entitled "Systems, Technologies and Protocols for the development and the real-time coordination of cooperating robot networks in industrial automation applications".

Since the end of 2012, Gaetano Patti is working with Prof. Lucia Lo Bello in the field of real-time industrial networks, automotive networks, Wireless Sensor Actuators Networks (WSANs), powerline communications and networks for mobile robots applications. He co-authored about 20 papers published in the proceedings of international conferences and in important international journals.

Gaetano Patti collaborated in several research projects funded by industry (STMicroelectronics, ZVEI, etc.) and public authorities (INRIA, PON, EU FP7, etc.). He has been a Visiting Researcher at ABB AB Corporate Research, Sweden, in May 2016.

Gaetano Patti is member of the IEEE IES Technical Committee on Factory Automation (Subcommittee on In-Vehicle Embedded Systems). He is serving in the Track Technical Program Committee of the international conferences: IEEE INDIN 2017 and IEEE ICIT 2017, and as a reviewer in several international conferences, and journals (IEEE Transaction on Industrial Informatics, IEEE Transaction on Industrial Electronics, Journal of Computer and System Sciences, Future Generation Computer Systems, etc.).