



TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

GERARDO SANTILLÁN MARTÍNEZ

A REAL-TIME PACKET SCHEDULING SYSTEM FOR A
6LOWPAN INDUSTRIAL APPLICATION

Master of Science Thesis

Thesis topic approved in the Engineering Sciences Faculty Council meeting on 8th of May 2013.

*“Convince not by word or explanation,
but by action. Simply do.” – Sean Plott*

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY

Master's Degree Programme in Machine Automation

SANTILLÁN MARTÍNEZ, GERARDO: A Real-Time Packet Scheduling System for a 6LoWPAN Industrial Application

Master of Science Thesis, 108 pages

September, 2013

Major: Factory Automation

Examiner: Prof. José Luis Martínez Lastra

Supervisor: Prof. José Luis Martínez Lastra, Dr. Ivan Delamer

Keywords: 6LoWPAN, IPv6, IEEE 802.15.4, WSN, Wireless, Sensors, Wireless Packet Scheduler, Beacon-enabled mode, Superframe, Guaranteed Time Slots, HSDPA, Internet of Things, Contiki.

Nowadays, the industrial Wireless Sensor Networks (WSN) are crucial for the monitoring and control of the modern smart factory floor that is relying on them for critical applications and tasks that were performed by wired systems in the past. For this reason, it is required that the transmission mechanisms of wireless sensor networks are efficient and robust and that they guarantee real-time responses with low data losses. Furthermore, it is required that they utilize common networking standards, such as the Internet Protocol (IP), that provides interoperability with already existing infrastructures and offers widely tested security and transmission control protocols.

The theoretical part of this document focuses on the description of the current panorama of the industrial WSN, its applications, design challenges and standardizations. It describes the 6LoWPAN standard and the wireless transmission technology that it uses for its lower layers, the IEEE 802.15.4 protocol. Later, it describes the principles behind the wireless scheduling, a state-of-the-art in the IEEE 802.15.4 scheduled channel access and the features of the most used operating systems for WSN.

The practical part presents the real-time packet scheduling system for a 6LoWPAN industrial application proposed by this thesis work that adapts the HSDPA scheduling mechanisms to the IEEE 802.15.4 beacon-enabled mode. The system implemented manages the channel access by allocating Guaranteed Time Slots to sensor nodes according to the priority given by three scheduling algorithms that can be selected according to the traffic condition of the network. The system proposed was programmed using Contiki OS. It is based on the eSONIA 6LoWPAN firmware developed for the European Research Project and it was deployed on the FAST WSN for testing. The results, discussion and conclusions are documented at the final sections of this part.

PREFACE

So this is it, the end of another stage of my academic life, the end of my MSc. Degree studies. It was a challenging and nice experience and now it is time to honour those who supported me during it.

I would like to start by expressing my endless gratitude to Professor José L. Martínez Lastra for giving me the opportunity of being part of his working group and for his confidence and advice during the development of this thesis. I want to thank the eSONIA European research project for funding the research of this work.

Special thanks to Dr. Ivan Delamer for his guidance as most of the work needed for this thesis would not be possible without his constant comments and contributions.

I would like to express my gratitude to the Consejo de Ciencia y Tecnología del Estado de Hidalgo (COCYTEH) and to the Mexican Consejo Nacional de Ciencia y Tecnología (CONACYT) for their financial support given for my master's degree studies. Special thanks to Ing. José Calderón Hernández.

I would like to thank Oscar and Borja for their help during these years. Moreover, I would like to thank all the members of the FAST laboratory, especially Luis, Jorge, Angélica, Sergii, Hector and Johannes.

I want to thank Despoina for her love and support during the realization of this work. *Είσαι τα πάντα για μενα, ευχαριστώ Δεσποινα, σ'αγαπάω πάρα πολύ ζωή μου.*

Por último, pero no menos importante, quisiera agradecer a mis padres por el apoyo que me han dado siempre. Soy lo que soy gracias a ustedes.

Tampere, Finland, September 25th 2013.

Gerardo Santillán Martínez.

CONTENTS

1. Introduction.....	1
1.1. Background	1
1.2. Problem Definition	3
1.2.1. Problem statement.....	3
1.2.2. Justification for the work.....	4
1.3. Work description	5
1.3.1. Objectives	5
1.3.2. Methodology.....	5
1.3.3. Assumptions and limitations	6
1.4. Thesis Outline.....	6
2. Theoretical background.....	7
2.1. Wireless technology in industrial networks.....	7
2.1.1. Design goals and Technical challenges in industrial Wireless Sensor Networks.....	12
2.1.2. Industrial Wireless Sensor Networks standardization	16
2.2. IEEE 802.15.4 Standard.....	20
2.2.1. IEEE 802.15.4 Physical layer	22
2.2.2. IEEE 802.15.4 MAC layer	23
2.2.3. IEEE Beacon-enabled mode.....	26
2.3. 6LoWPAN.....	27
2.3.1. Architecture	28
2.3.2. Protocol stack	29
2.3.3. Adaptation layer.....	29
2.3.4. Link layers	30
2.3.5. Fragmentation and reassembly	31
2.4. Wireless Scheduling	32
2.4.1. Best-effort schedulers.....	35
2.5. State-of-the-art in IEEE 802.15.4 Scheduling.....	38
2.6. Operating Systems for Wireless Sensor Networks.....	40
3. Methodology selection.....	46
3.1. Packet scheduling in HSDPA.....	47
3.1.1. Fast scheduling methods	48
3.1.2. Slow scheduling methods.....	50
3.2. Contiki OS.....	50
3.3. FAST Wireless Sensors Network.....	53
3.3.1. Wireless Devices.....	56
3.3.2. eSONIA 6LoWPAN firmware	60
3.3.3. FASTory line	60
3.3.4. Wireless-based applications at FAST	61

4. Implementation	66
4.1. Development environment	67
4.2. Packet scheduler	68
4.2.1. Architecture	69
4.3. Algorithms implementation	70
4.3.1. Network monitoring module	71
4.3.2. Scheduling module.....	73
4.3.3. Guaranteed Time Slots assignment module	77
4.3.4. Sensor nodes' algorithm.....	79
4.4. Experimental implementation	80
4.4.1. Test bed	80
4.4.2. Network monitoring application.....	81
4.4.3. Energy consumption network implementation.....	82
4.4.4. Network throughput and Packet loss network implementation.....	83
5. Results.....	85
5.1. Energy consumption	85
5.2. Networks' performance limits.....	87
5.3. Scheduling algorithms' performance.....	88
5.4. Networks' performance comparison.....	92
6. Conclusions.....	99
6.1. Conclusions of the implementation	99
6.1.1. Battery consumption results	99
6.1.2. Scheduling algorithms' performance results	100
6.1.3. Networks' performance comparison results	100
6.2. General conclusions.....	101
6.3. Future work	102
7. References.....	103

LIST OF FIGURES

Figure 2.1: Sensors systems on industrial applications [22]	8
Figure 2.2: Expected global installed industrial WSN by 2016 [28].....	11
Figure 2.3: Industrial WSN applications distribution on 2012 [28]	12
Figure 2.4: Energy consumed by different tasks [32].....	14
Figure 2.5: Wireless Protocol stack [33].....	16
Figure 2.6: Network topologies used in wireless communications	17
Figure 2.7: WSN standards map comparing range vs data rate [35]	17
Figure 2.8: IEEE 802.15.4 protocol stack	21
Figure 2.9: ZigBee stack [5].....	21
Figure 2.10: Spectrum usage for 802.15.4 [36].....	23
Figure 2.11: Star topology [36]	24
Figure 2.12: Peer-to-peer topology [36]	24
Figure 2.13: Cluster-tree topology [36]	25
Figure 2.14: CSMA/CA algorithm for the non beacon-enabled mode [36]	25
Figure 2.15: 802.15.4 superframe structure	26
Figure 2.16: CSMA/CA algorithm for the beacon-enabled mode [36]	27
Figure 2.17: 6LoWPAN protocol stack [5].....	27
Figure 2.18: 6LoWPAN architecture [5]	28
Figure 2.19: IP and 6LoWPAN protocol stacks [4]	29
Figure 2.20: Edge router with 6LoWPAN support [5]	29
Figure 2.21: IPv4 fragmentation header	31
Figure 2.22: IPv6 fragmentation header	31
Figure 2.23: 6LoWPAN fragmentation header	32
Figure 2.24: Max C/I Scheduler scheme [43]	33
Figure 2.25: QoS scheduling scheme [43]	34
Figure 2.26: CDF plots for different schedulers mechanisms [43]	36
Figure 2.27: Global throughput comparison [43].....	37
Figure 2.28: Generic PF scheduler CDF [43].....	37
Figure 2.29: WSN nodes components [68].....	40
Figure 2.30: WSN software layers [68]	40
Figure 2.31: Tiny OS architecture [68].....	42
Figure 2.32: Contiki architecture [68]	43
Figure 2.33: Mantis architecture [68]	44
Figure 2.34: Nano-RK architecture [68].....	44
Figure 2.35 LiteOS architecture [68].....	45
Figure 3.1: Contiki's partitioning into core and loaded programs [71]	51
Figure 3.2: Contiki's communication stack [71].....	52
Figure 3.3: 6LoWPAN architecture installed at FAST laboratory	54
Figure 3.4: FAST WSN layout at the FASTory production line.....	55
Figure 3.5: Inico S1000 RTU and its wireless expansion.....	57

Figure 3.6: 6LoWPAN expansion module for the S1000 RTU	57
Figure 3.7: S1000 configuration interface	58
Figure 3.8: Wireless Accelerometer/Gyroscope.....	59
Figure 3.9: THL sensor node.....	59
Figure 3.10: FASTory production line.....	60
Figure 3.11: Workstation with main and bypass conveyor [20]	61
Figure 3.12: Cell phone drew by FASTory line [20].....	61
Figure 3.13: FASTory production line pallet	62
Figure 3.14: NFC reader connected to the Inico S1000 RTU	62
Figure 3.15: APIS Graphical User Interface	63
Figure 3.16: Door switch sensor connected to a wireless I/O sensor	64
Figure 3.17: Retroreflective sensor.....	64
Figure 3.18: Production rescheduling diagram	64
Figure 3.19: Safety monitoring system GUI	65
Figure 4.1: Inico development environment using Eclipse.....	67
Figure 4.2: JTAGICEMKII debugging tool.....	68
Figure 4.3: Packet scheduling mechanism	68
Figure 4.4: Packet scheduling system architecture.....	69
Figure 4.5: Network monitoring module	71
Figure 4.6: Packets counter algorithm flow chart.....	71
Figure 4.7: Data rate calculation algorithm flow chart.....	72
Figure 4.8: Active nodes counter algorithm flow diagram	73
Figure 4.9: Scheduling module.....	73
Figure 4.10: Scheduling manager flow chart	74
Figure 4.11: Max C/I and Proportional Fair scheduling algorithms flow charts.....	75
Figure 4.12: Score Based Scheduler algorithm flow chart	76
Figure 4.13: Guaranteed Time Slots assignation module	77
Figure 4.14: Superframe manager algorithm flow chart.....	77
Figure 4.15: Superframe assignation distribution example for 20 TTIs.....	78
Figure 4.16: Beacon builder algorithm flow chart.....	78
Figure 4.17: Sensor nodes' algorithm flow chart	79
Figure 4.18: FAST Lab Wireless Network Monitoring tool.....	81
Figure 4.19: WSN deployment for energy consumption tests	82
Figure 4.20: WSN deployment for throughput tests.....	83
Figure 5.1: Energy consumption comparison plot.....	86
Figure 5.2: Global limit throughput.....	87
Figure 5.3: Global throughput at 210[bytes/s]/node forcing scheduling algorithms.....	89
Figure 5.4: Packet loss ratio vs. Throughput at 210[bytes/s]/node forcing scheduling algorithms.....	89
Figure 5.5: Global throughput at 128[bytes/s]/node forcing scheduling algorithms.....	90
Figure 5.6: Packet loss ratio vs. Throughput at 128[bytes/s]/node forcing scheduling algorithms.....	90

Figure 5.7: Global throughput at 64[bytes/s]/node forcing scheduling algorithms.....	91
Figure 5.8: Packet loss ratio vs. Throughput at 64[bytes/s]/node forcing scheduling algorithms.....	91
Figure 5.9: Global throughput comparison at 210 [bytes/s] per node	93
Figure 5.10: Performance comparison at 210 [bytes/s] per node.....	93
Figure 5.11: Scheduling algorithm selection at 210 [bytes/s] per node.....	94
Figure 5.12: Global throughput comparison at 128 [bytes/s] per node	95
Figure 5.13: Performance comparison at 128 [bytes/s] per node.....	95
Figure 5.14: Scheduling algorithm selection at 128 [bytes/s] per node.....	96
Figure 5.15: Global throughput comparison at 64 [bytes/s] per node	97
Figure 5.16: Performance comparison at 64 [bytes/s] per node.....	97
Figure 5.17: Scheduling algorithm selection at 64 [bytes/s] per node.....	98

LIST OF TABLES

Table 2.1: Classes of sensors and control applications [5]	8
Table 2.2: Advantages of wireless networks [26]	10
Table 2.3: Technical challenges vs. design goals of industrial WSN.....	15
Table 2.4: Bluetooth approximate range by class.....	18
Table 2.5: WSN Standardizations [36]	19
Table 2.6: Wireless technologies usage in industrial automation [36]	19
Table 2.7: IEEE 802.15.4 standard versions [37].....	20
Table 2.8: Radios defined in IEEE 802.15.3-2003 [36]	22
Table 2.9: Best-effort schedulers for WSN [43].....	35
Table 2.10: Wireless Sensor Networks Operative Systems [68].....	41
Table 2.11: Features summary of the WSN Operative Systems	45
Table 3.1: Size of compiled Contiki code, in bytes [71].....	52
Table 3.2: FAST WSN Sensor nodes	55
Table 5.1: Energy consumption tests transmission parameters.....	85
Table 5.2: Network's message loss percentage for energy consumption tests	86
Table 5.3: Performance limits' tests parameters	87
Table 5.4: Scheduling algorithm performance tests parameters	88
Table 5.5: Network throughput and packet loss tests parameters	92

ACRONYMS

6LoWPAN	IPv6 over Low Power Wireless Area Networks
ACK	Acknowledgment
APIS	Advanced Pallet Information System
APL	Application Layer
BO	Beacon Order
CAP	Contention Access Period
CDF	Cumulative Distribution Function
CFP	Contention Free Period
CoAP	Constrained Application Protocol
CPU	Central Process Unit
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
ED	Event detection
EEPROM	Electrically Erasable Programmable Read-Only Memory
eSONIA	Diagnostics and Control: Towards the Asset-Aware and Self-Recovery Factory
FAST	Factory Automation Systems and Technologies
FFD	Full Function Device
FFT	Fast Fair Throughput
FHSS	Frequency Hopping Spread Spectrum
FT	Fair Throughput
GPF	Generic Proportionally Fair
GTSS	Guaranteed Time Slots
GUI	Graphical User Interface
HART	Highway Addressable Remote Transducer Protocol
HSDPA	High-Speed Downlink Packet Access
I/O	Input/output
ICMPv6	Internet Control Message Protocol version 6
ID	Identification
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IoT	Internet of Things
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISA	International Society of Automation
ISM	Industrial, Scientific and Medical radio band
LoWPAN	Low Power Wireless Area Network
M2M	Machine to Machine
MAC	Media Access Control

MANTIS	Multi-modal System for Networks of In-situ Wireless Sensors
Max C/I	Maximum Carrier to Interference Ratio
MCU	Micro Controller Unit
MEMS	Micro Electro Mechanic System
MTU	Media Transmission Unit
ND	Neighbor discovery
NFC	Near Field Communication
NWK	Network Layer
OS	Operative System
OSI	Open System Interconnection
PAN	Personal Area Network
PF	Proportional Fair
PHY	Physical
QoS	Quality of Service
RAM	Random-Access Memory
RF	Radio Frequency
RFD	Reduced Function Device
RFID	Radio Frequency Identification
ROM	Read-Only Memory
RPL	Routing protocol for Low-power and lossy Networks
RR	Round Robin
RTU	Real-Time Unit
SB	Score Based
SO	Superframe Order
SOC	System on Chip
SPE	Spatial Process Estimation
SUN	Smart Utility Networks
TCP	Transmission Control Protocol
THL	Temperature/Humidity/Light
TTI	Transmission Time Interval
TUT	Tampere University of Technology
UDP	User Datagram Protocol
UE	User Equipment
UWB	Ultra-Wide Band
Wi-Fi	Wireless Fidelity, Trade mark of Wi-Fi Alliance
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

TABLE OF SYMBOLS AND UNITS

i	Current user or node under evaluation by a scheduling algorithm.
$\max_j\{R_j\}$	Constant value that indicates the maximum average supportable data rate for all users or nodes given in bytes/s.
P_i	Proportional Fair scheduling policy expressed by a constant.
$R_i(t)$	The average throughput experienced by the user i given in bytes/s.
$r_i(t)$	The instantaneous data rate that can be supported by the user in the current TTI.
R_j	The average supportable data rate of user i .
S_i	$S_i=1$ when the user i is served in the current TTI, and $S_i=0$ otherwise.
α	The constant term that weights the Proportional Fair algorithm to equalize the user throughput.
β	Central Process Unit.
Δt	Length of the TTI in seconds.
$\lambda_i(t)$	Proportional Fair Scheduler Score.

1. INTRODUCTION

This Section exposes a background on the thesis work domain for citing the concepts that any reader of this document should know to understand the objectives, tasks, and results of this Master of Science thesis.

1.1. Background

The Internet of Things (IoT) has been a big promise during the last few years; the idea of having everything connected to a network so that the information from “things” can be combined with intelligent systems in order to create knowledge is now a reality. Helped by the installation of IP-enabled networks of devices, controlling the illumination of a room with intelligent light bulbs or having access to information such as the location of a package sent from the other side of the world now is possible thanks to a global network infrastructure that links physical and virtual objects through the information capture and communication capabilities. Things are being transformed into Smart Objects’ networks capable of “speaking” to the world. The field of applications of the IoT is immense and its full potential is yet to be seen.

A big part of the groundwork that gives foundations to the potential of the IoT is being laid to the manufacturing industries. The IoT has enabled the innovation in many fields of the manufacturing levels. The implementation of “Smart Factories”, capable of self-recovery, based on a continuous monitoring and control of assets, goes along with the deployment of IP-enabled Wireless Sensors Networks (WSN) over these intelligent manufacturing plants, as many well-stabilised fields are now being based on wireless technologies. WSN consist of low power embedded devices with a limited code and RAM space, which extremely low cost makes possible to have significantly high numbers of nodes.

The interoperability and scalability of devices and the open, reusable and evolvable nature of the industry needs to be addressed by using devices developed over open standards and protocols, and here is where the IP-enabled wireless networks play an important role, as the Internet Protocol assures the integration of wireless embedded devices with the Internet, a network that provides unique capabilities such as extensive interoperability, established naming, addressing, translation, lookup, discovery and security schemas connected to it. IP-based technologies have existed for decades, they are well known and widely tested, open and free.

The IPv6 over Low Power Wireless Area Networks (6LoWPAN) [1] is the transmission of IPv6 packets over the IEEE 802.15.4 [2] wireless protocol. It is a standard that brings IP directly down to small wireless embedded devices. Knowing that there are

not enough of the IPv4-format addresses to extend to the IoT, 6LoWPAN brings the IP version 6 to give an address to every device. The wireless communication technology that 6LoWPAN utilizes is the IEEE 802.15.4 [3], a promising standard for the lower (physical and link) layers. This standard allows Low-Power Wireless Embedded devices to be connected using a familiar networking technology such as Ethernet or WiFi.

Easy integration has been proven to be one of the most important factors for the success of a technology [4]. 6LoWPAN brings interoperability and easy integration between applications and traditional computing infrastructure by enabling the use of web-services and utilizing modern security techniques. 6LoWPAN has become an extension of IPv6 into the wireless embedded devices, with a wide range of applications in different domains: home and building automation, healthcare, logistics, smart metering, real-time environmental monitoring, security systems, asset management and industrial automation, being the last example one of the most important application areas as it requires a holistic system approach due to the special requirements for QoS, safety and security [5].

Before the 6LoWPAN WSN capabilities can be exploited, significant problems need to be solved, including simple adaptation and integration issues as well as the creation of mechanisms that reduce the energy consumption and enhance the transmission management at the lower layers of the 6LoWPAN protocol stack.

The 6LoWPAN-based applications need to be aware of the limitations of the physical links that carry their packets. IP packets can be large compared to the IEEE 802.15.4 maximum frame size. Considering that the IPv6 minimum Media Transmission Unit (MTU) is 1280 bytes (required at all links supporting it) and that an IEEE 802.15.4 frame can have a payload limited to 74 bytes in the worst case, a packet might end up fragmented into as many as 18 fragments at the 6LoWPAN shim layer. If a single one of those fragments is lost in transmission, all fragments must be resent, further packet loss. Past experience with fragmentation has shown that miss-associated or lost fragments can lead to poor network behaviour and, eventually, trouble at application layer [6], for this reason, data fragmentation and re-assembly is an important adaptation factor to be considered.

The IEEE 802.15.4 MAC layer provides access control to a shared channel and reliable data delivery. It is based on a CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) algorithm [7] that requires listening to the channel before transmitting to reduce the probability of collisions with other ongoing transmissions. The IEEE 802.15.4 MAC layer defines two different operational modes, namely beacon-enabled (mandatory when tree topologies are used) and non beacon-enabled, which correspond to two different channel access mechanisms, every WPAN coordinator must choose one of these modes and cannot support both at the same time.

In the beacon-enabled mode, the communication is managed through a superframe, starting with a packet, called beacon, transmitted periodically by the coordinator to the devices associated with the Wireless Personal Area Network (WPAN). After the beacon, the superframe is divided into 16 equally sized slots distributed in two parts: the

Contention Access Period (CAP) and the Contention Free Period (CFP) composed of Guaranteed Time Slots (GTSs) which can be assigned by the WPAN coordinator to specific nodes in order to give them channel access. The superframe is followed by an inactive period that allows to both, the coordinator and the nodes, to go in sleeping mode. The beacon-enabled mode is much more energy efficient than non beacon-enabled mode for low rate applications [8] as a PAN coordinator using the Beacon-enabled mode needs to transmit only during the active period and can sleep during the inactive period to save power. In order to communicate with the coordinator, nodes must be synchronized with it. The synchronization between nodes and coordinator is achieved by the assignation of GTSs to specific nodes. The GTSs must contain the packet transmitted by the node to which the GTS is allocated to and a minimum interval between two subsequent packets received called inter-frame space. The WPAN coordinator may allocate up to seven GTSs but leaving a part of the CAP for a contention-based channel access is important [9].

In the Information Technology context, scheduling is deciding how to manage resources between different possible tasks [10]. On the other hand, in WSN, scheduling is allocating transmission rights to subsets of network users at each time under different channel qualities. In the IEEE 802.15.4 context, scheduling methods decide how the GTSs, part of the superframe, are allocated between elements associated to a given WPAN. The scheduler, embedded in the coordinator, is the main manager of the channel access in the downlink as it decides which user will be scheduled in each Transmission Time Interval (TTI). Scheduled GTSs allocation maximizes the sum throughput across the network, reduces the packet collisions, it is used to achieve a larger throughput region and it is important to build a more stable network. The scheduling scoring mechanism takes into account the channel conditions, fairness between users, cell throughput and QoS parameters. Implementing an optimal scheduler involves trade-offs between these factors, this decision is the selection of different scoring and traffic classification methods and algorithms and must depend entirely on the network characteristics and more important, on the application for which the wireless packet scheduler will be used.

1.2. Problem Definition

1.2.1. Problem statement

Thanks to the success and constant development of the wireless technologies, more applications for wireless networks appear every day in the industrial domain, giving shape to the smart factory concept where sensors and actuators can intelligently interact with each other in order to create smart manufacturing floors with a more efficient resources management, minimum energy losses and self-recovery to failures capabilities.

The modern manufacturing industry has started to rely on WSN for critical tasks. Nowadays, WSN are not being used only for sensing and monitoring but they have been started to be used for control applications. This is an important factor that has helped to potentiate the growth of the industrial WSNs. On the other hand, it is crucial to understand that industrial end-users must feel confident with the solutions to these technologies critical issues before they will completely entrust control functions to a wireless system.

For 6LoWPAN, one of the most popular wireless technologies at the industrial domain, this means that addressing the problems that its data loss causes due to the message fragmentation at its MAC and adaptation layers, is vital in order to keep providing IPv6 connectivity to the modern production and manufacturing industry that requires robust and reliable wireless sensor and actuator networks.

Knowing that it is possible to improve the data transmission at the lower layers of the 6LoWPAN protocol stack, this thesis work focuses on the adaptation of scheduling techniques used by the mobile industry to the MAC layer of the IEEE 802.15.4, the standard used by 6LoWPAN for its MAC and PHY layer, in order to reduce the information losses and improve the bandwidth management, giving as a result a more predictable, stable and scalable network.

1.2.2. Justification for the work

The research work done for this Master of Science thesis was developed at the Factory Automation Systems and Technologies (FAST) laboratory, part of the Production Engineering Department of the Tampere University of Technology (TUT) as part of the eSONIA European research project [11] under the umbrella of ARTEMIS Joint Undertaking [12]. eSONIA stands for Embedded Service-Oriented Monitoring, Diagnostics and Control: Towards the Asset-Aware and Self-Recovery Factory. This project successfully addressed the need of tools for realising the asset-aware and self-recovery production plants through pervasive heterogeneous IPv6-based embedded devices that support continuous monitoring, diagnostics, prognostics and control of assets, regardless of their physical location [11].

The result of this research is software for a better and more proficient plant monitoring using 6LoWPAN-enabled hardware and IPv6-based applications, platforms that assure the interoperability of technologies and that support direct IP network connection for versatile data transportation without the need of gateways and using web services.

This Master's thesis describes the work directed towards the design, programming and implementation of firmware for low-power embedded devices. The firmware developed is designed to achieve a more robust and reliable industrial 6LoWPAN WSN network with higher data transmission efficiency and a network with less package loss.

1.3. Work description

1.3.1. Objectives

- 1) Implement firmware for the FAST Laboratory 6LoWPAN-enabled WSN capable of schedule the allocation of the IEEE 802.15.4 beacon-enabled mode' GTSs according to the industrial application running at the laboratory.
- 2) Develop wireless traffic monitoring tools, embedded in the firmware of the devices, capable of detecting the number of nodes and the amount of packets transmitted so that this it can be used by the scheduler manager.
- 3) Develop an adaptable scheduler manager, embedded in the devices' firmware, capable of selecting the optimal scheduling mechanism according to the traffic condition.
- 4) Deploy the resulting firmware on the FAST Lab WSN.

1.3.2. Methodology

Research of existing Wireless Technologies in Industrial Manufacturing Networks

Review of the wireless technologies and its implementations and applications in current industrial manufacturing environments focussing on the wireless sensors networks, its benefits, drawbacks and the design objectives that this industrial networks have.

Research of 6LoWPAN protocol for WSN

Review of the current implementations of the 6LoWPAN protocol for industrial wireless sensors networks. An extensive review of the IEEE 802.15.4 protocol and its beacon-enabled mode, used by the lower layers of the 6LoWPAN standard, will be conducted, making emphasis on the MAC layer of the stack as it is here where the majority of the implementation of this research work was held at.

Research of existing Wireless Schedulers implementations

Review of the current wireless scheduling techniques, common scoring mechanisms and algorithms and its implementation in different domains such as the mobile phone industry and the industrial manufacturing. A short review of the most common wireless schedulers implemented in the industry will be done.

Evaluation of suitable technologies and tools for implementing a 6LoWPAN scheduler for an industrial application

A study of the existing scheduling techniques and its implementations using the 6LoWPAN protocol over industrial wireless sensor networks will be presented. A comparison of scheduling technologies will be done in order to decide the best implementation to be used. Finally, a selection of theoretical principles and practical techniques and methods that fulfil the research work requirements and objectives will be conducted.

Implementation and testing of the principles supported by the previous evaluation

Once the optimal scheduling technique is chosen based on the previous technologies evaluation, the scheduling mechanism will be designed and programmed. This mechanism will be supported by other tools, embedded in the firmware, that will be monitoring the wireless traffic of the wireless network as well as managing the scheduling score algorithm.

Deployment of the resulting implementation

The resulting firmware will be installed on the FAST Laboratory WSN used by different applications for factory automation research purposes and then tested in order to evaluate the performance of the packet scheduling system developed during this thesis.

1.3.3. Assumptions and limitations

Assumption 1: The firmware developed was programmed on Contiki OS using Eclipse IDE as the programming platform running on Microsoft Windows ®

Assumption 2: The firmware was developed taking the eSONIA 6LoWPAN firmware as a base.

Assumption 3: All the traffic using the same channel for transmission goes to a single sink node.

1.4. Thesis Outline

This thesis is structured as follows: Chapter 2 gives the theoretical review of technologies and concepts that give support to this research work. Chapter 3 contains a detailed description of the technologies and platforms chosen for the development of the packet scheduling system along with the description of the FAST wireless sensor network and the applications' platform using the wireless network at FAST laboratory. Chapter 4 describes the implementation of the real-time packet scheduling system for a 6LoWPAN industrial application. Chapter 5 presents the discussion of the thesis' results. Finally, Chapter 6 contains the conclusions of the research held for this work.

2. THEORETICAL BACKGROUND

This chapter is a literature and technology review which defines and describes the concepts, technologies and tools used to this thesis work. Beyond the previous introduction, this chapter explains some of the cited concepts that are in relation with different performed tasks for this thesis work. Note that this information is focused on manufacturing systems, since this is the environment in which the applications of the thesis implementation are planned to be applied.

2.1. Wireless technology in industrial networks

During the last decade, advantages such as mobility, deployment and expandability of the wireless networks, along with the convenience of having connected devices without the need of wires have been directing the success of the wireless technology industries. Thanks to this success, numerous applications started to appear in different fields. The manufacturing industry and its factory floor have been benefiting from the use of these technologies [13] by reducing the wired-related maintenance costs, increasing the environment sustainability and easing the plant set up and reconfigurability. The manufacturing industry's constant needs and demands for more efficient control and monitoring applications requires the use of reliable, intelligent and low-cost data collecting methods or sensors, capable of communication with devices that digitalize, transfer and process the information gathered.

There are different approaches for the classifications for the Industrial automation: According to [5] industrial automation can be split into two application areas: process control and factory automation. Process control includes applications such as petrochemical-related processes. Factory automation covers the manufacturing of appliances and consumer products. For [14], industrial automation systems can be divided into two categories: closed loop and open loop systems where open loop systems are applied in discrete operations such as actuator control, while open loop systems are applied in process monitoring.

Regardless of the classification criteria, the industrial automation relies on the data acquired from the physical world to perform specific procedures. Control of manufacturing processes [15] [16], production monitoring [17] [18] [19], or assets tracking [20] [21], are examples of tasks that require the information from sensors whether to carry on actions or to be aware of production events. Therefore, industrial automation needs the implementation of complex sensing systems and networks that integrate a wide range of field devices in order to fulfil the quality and production standards. The International Society of Automation (ISA) has defined six different classes for sensor and control applications within the market domain.

Table 2.1 shows the categorization done by ISA.

Table 2.1: Classes of sensors and control applications [5]

<i>Class</i>	<i>Application</i>	<i>Description</i>
Class 0	Safety	Emergency action required: <ul style="list-style-type: none"> • Emergency shutdown • Automatic fire control • Leak detection
Class 1	Control	Closed loop control – critical <ul style="list-style-type: none"> • Direct control of actuators, pumps and valves • Automated shut-down
Class 2	Control	Closed loop control – non- critical <ul style="list-style-type: none"> • Optimizing control loops • Flow diversion
Class 3	Control	Open loop control – human intervention <ul style="list-style-type: none"> • Operator perform manual adjustment
Class 4	Monitoring	Alerting – necessary maintenance <ul style="list-style-type: none"> • Event based maintenance • Low battery • Vibration monitoring • Motor temperature monitoring
Class 5	Monitoring	Logging – preventive maintenance <ul style="list-style-type: none"> • Preventive maintenance records • History collection

In industrial applications, sensors are used to transform physical measurable characteristics from the process into signal values that can be transferred to an information processing entity to be analysed and then fed into a process control system for further controlling tasks either directly by means of automatic devices or robots. This process is shown by Figure 2.1.

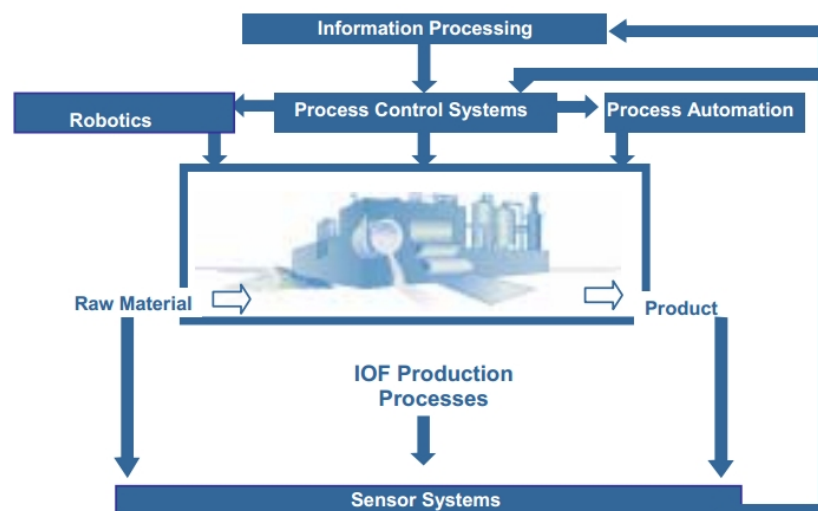


Figure 2.1: Sensors systems on industrial applications [22]

The industrial controls are algorithms that respond to the information from sensors, generating a signal that drives an actuating mechanism. Industrial controls, connected to information processing mechanisms, can be seen as means that respond to a change state of a given process knowledge. This knowledge is generated from the information given by a process data using sensors or networks of sensors [22]. The robustness of sensors is a key factor for a reliable data acquisition and transmission, because strong electromagnetic power sources can reduce the measurement and transference quality causing errors. The data acquisition speed is critical for the performance of a system. Sensors capable of fast data transfer are crucial for a high performance industrial communication system. The “real-time” is the ability of a system to provide a required result in a bounded time [22]. A real time communication system is able to transfer data in real-time. The best way to respect data transfer time deadlines is using centralized architectures, where sensors are read when needed by a central managing system using point-to-point connections. In this case, the event delay and reaction is minimal.

On the other hand, the microcontrollers’ technology is driving the new generation of sensors as it is offering functions beyond those necessary for generating a correct representation of a sensed or controlled quantity [23]. Key advantages of this new microcontroller-based “smart sensors” are: the networking capability; real-time data sharing capability throughout a facility to increase industrial efficiency and productivity; ease of replacement and upgrading; reduced connector failure; greater mobility. Using smart sensors in a centralized architecture is uneconomic and reductive, since just a few of their qualities can be exploited. The distributed architecture is more suitable for smart sensors, thanks to their networking capabilities and interfaces. Using distributed architectures brings other advantages such as high flexibility, improved performances, predictive maintenance, simple installation and cabling cost reduction. Unfortunately, using this architecture also implies transmission delays that can heavily affect their performance.

An industrial network is a network that communicates some sort of automated system, its control and monitoring systems [24]. Traditional wired industrial networking offers many advantages but it requires cables to connect devices, meaning high installation and maintenance costs caused by the low scalability and high failure rate of connectors and cables. For this reason, the level of adoption that wireless solutions at the sensors network has gained popularity due to the multiple advantages that this networks offer: continuous, high-resolution, ubiquitous sensing; support to mobility; redundancy and compactness [25]. The communication and data transfer in a WSN is with the transmission of electromagnetic waves through the air. The concepts behind the use of WSN are shown by Table 2.2.

Table 2.2: Advantages of wireless networks [26]

Advantage	Description
Lower costs	Costs associated with installing, maintaining, troubleshooting and upgrading wiring have scaled while costs for wireless technology have continued to drop, particularly in the areas of installation and maintenance
Reduced connector failure	Most failures in any network occur at the connectors; wireless sensors eliminate this problem.
Improved flexibility	Without the constraint of wires, plant managers can better track materials and more easily reconfigure assembly lines to meet changing customer demands.
Exploitation of MEMS	Micro-electromechanical systems (MEMS) offer a rapidly expanding wealth of sensing capabilities.
Rapid commissioning	Simple wireless sensor systems can rapidly organize and configure themselves into an effective communications network.
Reliability	Wireless sensor systems can offer built-in redundancy and capabilities for anticipatory system maintenance and failure recovery
Ease of use	Integrated wireless sensor systems with distributed intelligence can enable operator-independent control of industrial process and can dynamically adapt to and compensate device failure or degradation.
Security	Encrypting and even hiding wireless data transmissions promise a level of security that equals or surpasses that of wired systems.
Robust design	Integrated sensor nodes encased in advanced materials should be able to endure repeated exposure to caustic gases and high temperatures.
Open architecture	A generic development architecture should allow specialized applications from a wide spectrum of devices without requiring cumbersome interfaces

Wireless networks can be considered as an extension of wired networks and even if the wireless networks seem to be the logic “next step” of the wired systems, it is not possible to conceive a modern factory floor without wired connected devices as there are a range of industrial applications that require the use of wires as their networking method. In the modern manufacturing business a complete industrial networking is the result of the integration of wired and wireless networks. For this reason, the industry needs to be aware of the importance of the compatibility and interoperability between these solutions as the lack of agreement about common architectures has been limiting and slowing its growth and development.

Even if past research projects such as RUNES project [27] showed that the adoption of wirelessly networked embedded systems is slower in the industrial sector than in others, wireless sensor technology is moving rapidly into applications in production plants and other industrial environments. According to ON World [28], the industrial WSN market has doubled over the past two years (2011 and 2012). Within the next five years, installed industrial wireless networks will be increased by 553% meaning that there will be nearly 24 million wireless-enabled sensors and actuators. By 2016, 39% of deployed nodes will be used for new applications that are uniquely enabled by WSN technology. WSN is impacting industrial automation by disrupting wired automation, extending wired sensor networks and driving new sensing and control solutions. Figure 2.2 shows the globally installed industrial wireless sensing points expected growth by 2016.

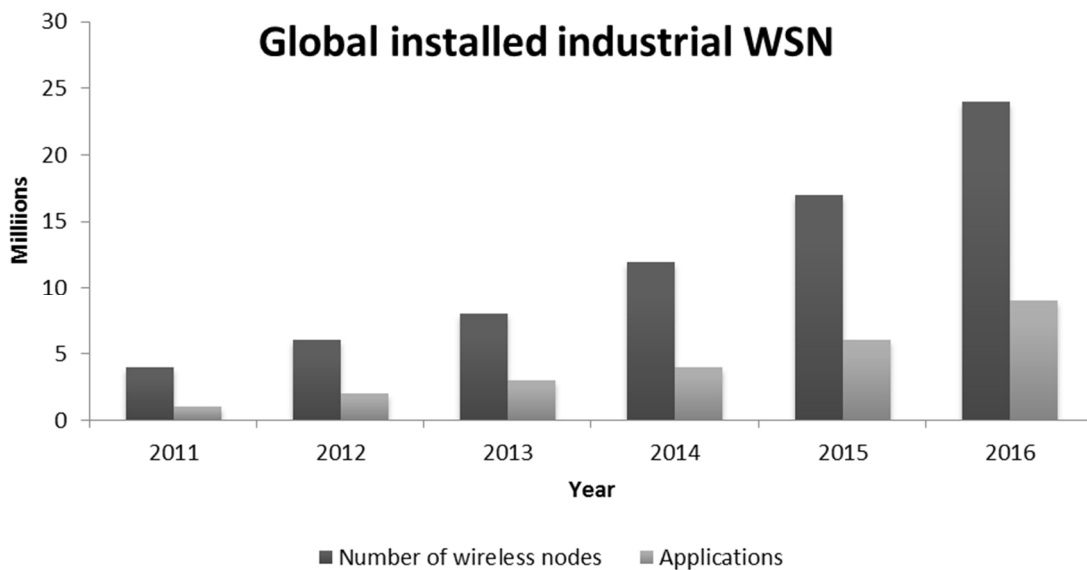


Figure 2.2: Expected global installed industrial WSN by 2016 [28]

The relation between installed WSN nodes and applications is very important since, as it is shown by Figure 2.2, they will have a proportional expansion and impact in the industrial application market. The current variety of possible applications of WSN to the industrial automation is limited to control (ISA’s application classification classes 0 to 3) and monitoring (ISA’s application classification classes 4 to 5). Almost any application can be classified into one out of two categories: Event Detection (ED) and Spatial Process Estimation (SPE) [28]. In Event Detection applications, sensors are used to detect an event, for example a change on

the temperature of a room or the detection of a vibration in a machine. In SPE, the WSN aims at estimating a given physical phenomenon which can be modelled as a bi-dimensional random process, for example, the toxic emission levels of a chimney. There are also applications that belong to both categories like the tracking of assets [20].

According to an ON World survey done in 2012 to industrial automation related companies, the process monitoring, has become the most used application for which WSN are being used for in the industry automation, with 80% of the users surveyed using a wireless solution for it, followed by tank level monitoring, machine diagnosis and valve monitoring. Figure 2.3 shows the complete chart of WSN applications at the industrial automation [29]. Most of the industrial applications currently in use perform monitoring rather than control due to security and performance issues. Industrial end-users must feel confident in the solutions to these issues before they will entrust control functions to a wireless system. It has been stated before [27] that wireless should not be used for critical control applications as some lessons must be learnt from successful telemetry deployments before moving in the control applications.

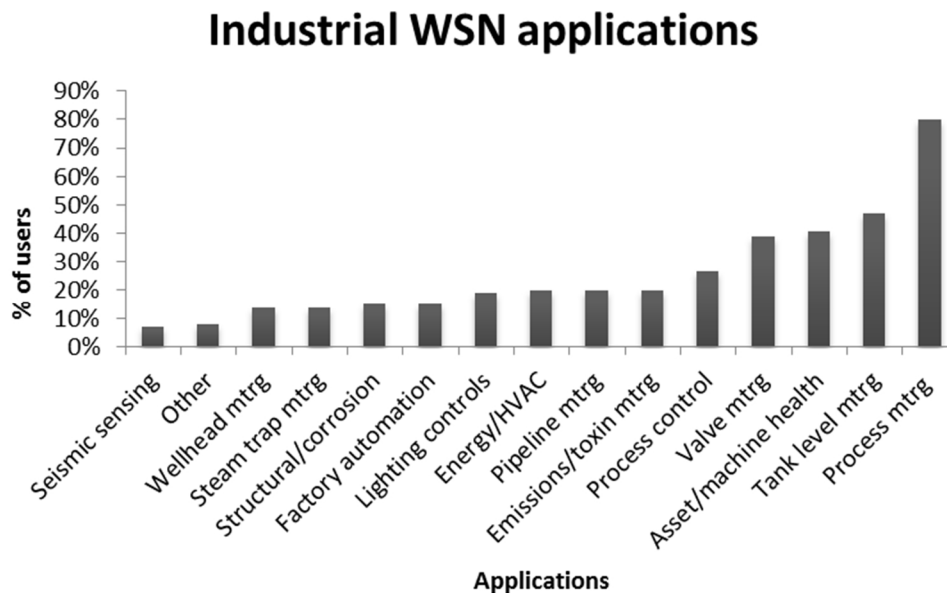


Figure 2.3: Industrial WSN applications distribution on 2012 [28]

2.1.1. Design goals and Technical challenges in industrial Wireless Sensor Networks

Several technical challenges need to be overcome in order to meet the design goals of an industrial Wireless Sensor Network implementation as it requires the integration of different expertise areas. There is not a single solution for designing and deploying industrial WSN since most of them perform sensing functions for a specific application, in the same way, most of their design goals are design specific. Another factor to be considered is that sensing devices are becoming increasingly complex, reliable and cheaper helping them to evolve from

different implementations with functionalities specific to a single application into sophisticated heterogeneous WSN deployments with multi-application capabilities and different sensing tasks.

Moreover, industrial WSN design goals have rapidly changed during the last years due to the increasing number of nodes installed per implementation and the technology used for the sensing devices. In the recent past, wireless sensor networks implementation consisted of a relatively small amount of nodes connected through centralized network architectures to a data processing unit. Nowadays, as explained in the previous Section, the use of smart sensors in modern industrial WSN is becoming a requirement due to the complex applications for which sensing and data acquisition systems are required. This new implementations come along with new challenges for the design and deployment of wireless networks as wireless smart sensors networks, with increased networking and processing capabilities, use distributed network architectures with a significantly higher number of nodes connected in order to exploit all their capabilities. WSN technology has emerged as a premier research topic and as a dynamic industry, for this reason, many of the WSN design goals and implementation needs will keep evolving during the near future.

Even if WSN design goals are application specific and due to different factors they are constantly changing, Wireless Sensor Networks for industrial applications design goals are limited to the constraints of the different levels of the factory and production systems and they share their design objectives and technical challenges. Some of the most important design goals of the modern industrial WSN applications are: power efficiency; low-cost sensor nodes; fault tolerance and reliability; secure operability; need for self-organizing and self-healing; efficient traffic distribution; reduced data redundancy; computation over communication; need of multi-hop support; integration and interoperability; scalability; time synchronization. Not all the applications have the same requirements, properties or objectives but high efficiency is always looked to be accomplished. In WSN, as in other technologies, for every design goal to fulfil there is a technical challenge associated to it and designing an industrial Wireless Sensor Network that accomplishes all the design goals is very challenging for it is necessary to consider many technical factors in order to satisfy all the needs. Previously presented goals and some of their technical challenges can be explained as follows [21] [30] [31]:

Power efficiency: as sensor nodes are low power devices not connected to a fixed power supply system, they depend on the use of batteries as their main power source. Power efficiency should be optimized at all levels of the protocol selected for the WSN implementation, including low levels (Physical and Media Access Control layers), routing and application layers. Energy efficiency is enhanced by anticipating minimum energy usage and allowing tuning and trade-off between requirements of the available resources.

One of the most important factors that need to be considered in order to achieve power efficiency is the amount of energy consumed by different tasks of the device. A comparison between the energy consumption of different device operations is shown in Figure 2.4 [32] where a 100 byte message under perfect radio conditions is transmitted, received, written on the flash memory and read from the flash memory of a commercial MCU with a wireless transceiver.

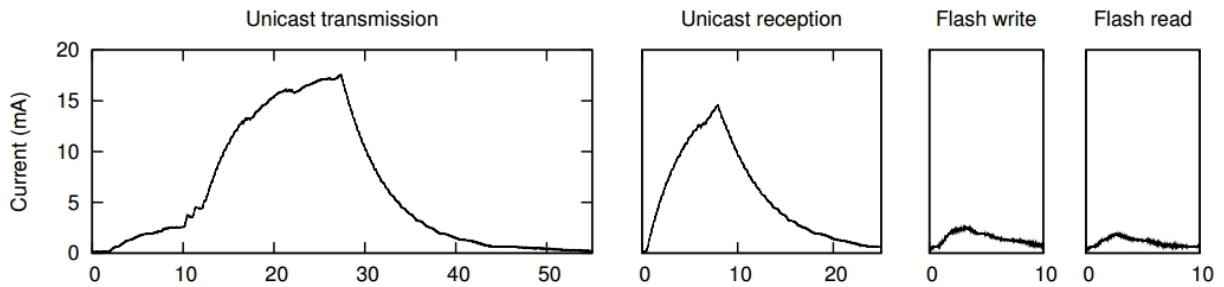


Figure 2.4: Energy consumed by different tasks [32]

The transmission and reception of data is clearly the most power consuming task of a wireless device, for this reason, an efficient transmission mechanism is extremely important for achieving power efficiency of a wireless device.

Low-cost sensor nodes: the unit cost per sensor is highly relevant for the overall cost-benefit relation of a WSN deployment. The unit cost must remain as low as possible in order to contribute to the feasibility of the network.

Fault tolerance and reliability: the communication in an industrial WSN is susceptible to errors because of harsh environments, noise, obstructions among others. The reliable delivery of data from sensor nodes is crucial for the network. The network must be able to tolerate communication errors in order to improve its reliability.

Secure operability: complete protection of anything against anything is obviously impossible. It is easier to target a single sensor that aim to attack a very large of very small sensors as these kinds of deployments enjoy relative protection of numerosity. Secure operability must be based on a design that addresses all weak links of the WSN operability by helping them to continue functioning as long as possible in spite of power restriction or security issues, maximizing its availability and minimizing its vulnerabilities.

Need for self-organizing and self-healing: WSN need to be able to adapt to network topology or connectivity changes autonomously. The routing protocol is the one responsible for moving information across network and it must be able to tolerate networks faults and adapt to network changes. Most of the times, mobility adds complexity to this design goal as it causes node's connectivity changes.

Efficient traffic distribution: in order to extend network lifetime, the traffic should be spread to the entire network. Using compression and aggregation techniques can help to improve energy imbalance when the source sink ratio is low.

Reduced data redundancy: at some applications, several sensors measure the same physical phenomenon, their readings can be aggregated to a single node in order to reduce the variance in reading that many sensors making the same measurement can have. Another way to reduce the energy spent on a WSN is combining as many messages as possible in order to reduce the overhead and contentions of the messages for the communication.

Computation over communication: due to large number of nodes, network-scale interaction is energy intensive. It is desirable to have a localized computation at a node level since communication cost is expensive since transmitting is the task that needs more energy. In-network data processing will save energy for many applications and it will help to reduce bottleneck tunnelling problems.

Need of multi-hop support: direct routing works well for applications where the nodes can be close to the sink but most of the time this is not possible or suitable for the application. Multi-hop communication enables reuse of the radio spectrum frequencies and allows large scale sensor deployments with less energy resources needed.

Scalability: WSN must have a flexible and scalable architecture, able to ingrate a wide variety of applications in the same infrastructure.

Integration and interoperability: the easy integration of WSN with existing technologies and even with other similar networks needs be exploited by using standardized and well known protocols. The high number of proprietary systems and the lack of agreement in the wireless industry is slowing and making difficult this integration and interoperability support for the WSN.

Time synchronization: the QoS in an industrial WSN strongly depends on the latency of the delivered data; therefore, the time synchronization is a key design goal in order to achieve needs of a certain application. Scheduled transmissions help in the reduction of collisions and retransmissions resulting in energy conservation in the network. Table 2.3 presents the previously described design goals along with its technical challenges associated to them:

Table 2.3: Technical challenges vs. design goals of industrial WSN

Design goals	Technical challenge
Power efficiency	Energy Resources constrains
Low-cost sensor nodes	Low-cost built in embedded devices
Fault tolerance and reliability	Packet error and variable link capacity
Adaptive Network Operation	Dynamic topologies and harsh environmental conditions
Secure operability	Security
Self-organization	Nodes' firmware capabilities
Self-configuration & self-healing	Nodes' hardware and firmware processing and storing capabilities
Efficient traffic distribution	Efficient routing protocols
Reduced data redundancy	Data management protocols
Computation over communication	In-network processing over network communication
Need of multi-hop support	Distributed architectures, RPL and mesh protocols implementations
Integration and interoperability	Interoperable protocols and scalable architectures
Scalability	Scalable devices, architectures and protocols
Time synchronization	Quality of service requirements

2.1.2. Industrial Wireless Sensor Networks standardization

In recent years, the constant search for a robust, reliable, efficient and interoperable Wireless Sensor Network has brought as result the development of standards that address many of the previously mentioned design goals. WSN protocols are based on the OSI model [33]. The protocol stack is shown by Figure 2.5 and it is used by all the nodes of the network.

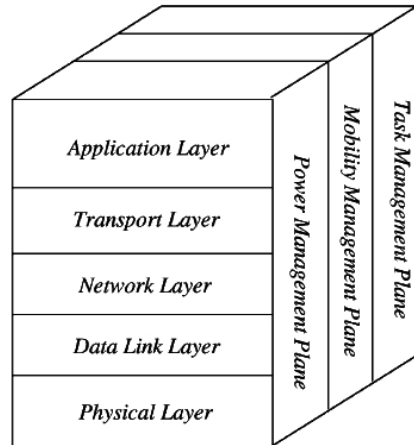


Figure 2.5: Wireless Protocol stack [33]

The protocol stack shown in Figure 2.5 is comprised by five layers to handle the communication protocol. One of the differences with the OSI model is the lack of a session and presentation layers (6th and 7th layer) in the OSI model on the first plane of the protocol stack in charge of the communication protocol. WSN protocol stack contains three plane layers in charge of the management protocols that address the power management, the connection and the task management problems associated to the wireless communication. A summary of the capabilities of each communication protocol layer is the following [34]:

Application layer: contains a variety of application-layer protocols to generate various sensor network applications as well as application processing, data aggregation, external querying, query processing and external data base capabilities.

Transport layer: is responsible for reliable data delivery required by the application layer. It is capable of transport, including data dissemination and accumulation, caching and storage. Due to energy, computation, and storage constrains of sensor nodes, traditional transport protocols such as Transport Control Protocol (TCP) cannot be applied directly to sensor networks without modification.

Network layer: is responsible for routing the data sensed by source sensor nodes from the transport layer to the data sink. It has adaptive topology management and topological capabilities.

Data link layer: is the layer responsible for data stream multiplexing, data frame transmission and reception and error control. One of the most important functions of the data link layer is medium access control (MAC). The primary objective of MAC is to fairly and efficiently share the shared communication resources among multiple sensor nodes in order to achieve a good network performance in terms of energy consumption, network throughput and delivery latency.

Physical layer: is responsible for signal transmission and reception over a physical communication medium, including frequency generation, signal modulation, transmission and reception and data encryption. It is also in charge of the sensing, actuation and signal processing. This layer must deal with various issues, for example, transmission medium and frequency selection, carrier frequency generation, interferences and data encryption. In addition it must deal with the design of the hardware.

The purpose of this communication protocol stack is to enable the communication over wireless medium with the highest power efficiency and to deal with the self-organization and cooperation capabilities of wireless networks. The network topology depends on the stack implementation and in the application; a complete network can be comprised by several sub-nets with different topologies. Different network topologies in wireless communications are shown by Figure 2.6.

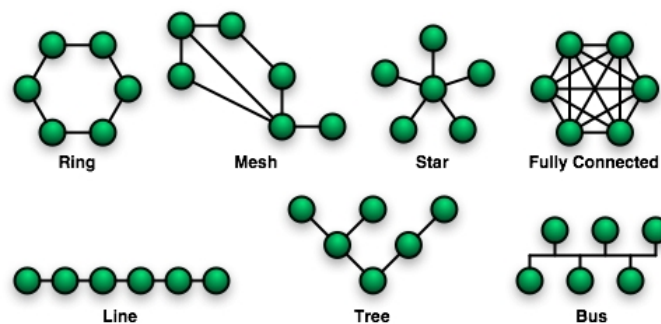


Figure 2.6: Network topologies used in wireless communications

Existing protocols are the result of the industry need for standardization. As standardization brings many advantages: communication improvements, easier application of expert knowledge, increases safety, allows interoperability of devices and exchangeability of components, it improves specifications, reduces inspection and testing times, and it provides a better utilization of resources. Existing industrial wireless standards are, for example: WirelessHART, ISA100.11a, Wibree, IEEE 802.15.3, IEEE 802.15.4, ZigBee, ZigBeePro, UWB, Bluetooth, Wi-Fi, 6LoWPAN among others. The functions and communication protocols need to interact with the nodes of its own and other network nodes defined by this standards. Figure 2.7 shows the existing wireless standards and some of their features.

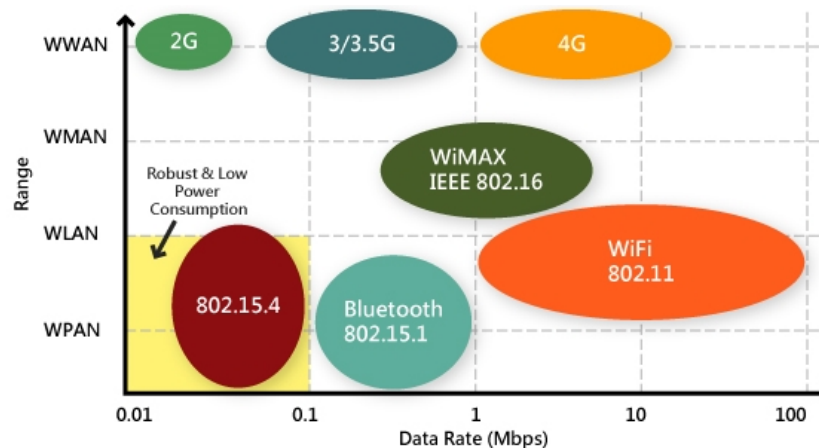


Figure 2.7: WSN standards map comparing range vs data rate [35]

A brief description of the standards shown by Figure 2.7 is presented [36]:

IEEE 802.11-based standard: this standard is focused on physical and data link layers for a LAN where at least two nodes are connected. The modulation techniques that this protocol uses is a comprise version of the one used by WiFi standard operating in the band of 2.4 GHz. It has a transmission frequency of 5 GHz with a data rate of 54 Mbps and it covers a range up to 35 meters in indoor environments.

IEEE 802.15.1-based standard: also known as Bluetooth®. Bluetooth is a low-range and low-power wireless technology. It is designed to offer high levels of security and robustness. This protocol operates in the ISM band of 2.4 GHz using a modulation scheme based on Frequency Hopping Spread Spectrum (FHSS) and it has throughputs of one Mbps. The number of connections of a single node is limited up to seven. The range of the Bluetooth technology is summarized in Table 2.4.

Table 2.4: Bluetooth approximate range by class

Bluetooth class	Max. transmission power	Approximate range
Class 1	100 mW (20 dBm)	100 meters
Class 2	2.5mW (4 dBm)	10 meters
Class 3	1 mW (0 dBm)	1 meter

Wibree specification can also be included in the 802.15.1 specification as it is a derivation of Bluetooth designed for ultra-low power devices. Like the Bluetooth wireless technology, its bit rate is of one Mbps but the range is from 5 to 10 meters and it uses a frequency of 2.45 GHz. Wibree do not uses FHSS but an unspecified modulation scheme and it has a variable length packet structure.

Radio Frequency Identification (RFID): the International Organization for Standardization (ISO) 18000 series standardizes the automatic identification and device management in RFID technology. The RFID network is comprised by at least one radio frequency transponder known as tag that contains exclusive information that can be read remotely by a node commonly known as RFID reader. There are two types of RFID tags: active and passive tags. The active tag uses batteries as power source, its reachable range goes from 20 up to 100 meters, and it handles higher frequencies than the passive tag, typically of 455 MHz, 2.45 GHz or 5.8 GHz. The passive tags do not use an internal power supply for the RFID reader transmits the required power; its range is 10 meters; and the work in lower frequencies such as 128 kHz, 13.6 MHz 915 MHz or 2.45 GHz.

The Near Field Communication (NFC) is an additional technology related to the RFID. With a maximum range of 20 cm by using point-to-point connectivity, this technology combines the RFID identification and the interconnection technologies.

IEEE 802.15.3-based standard: it specifies the physical layer for the UWB radio technology, a short range wireless technology that offers good capabilities for positioning systems. The range of the UWB is about ten meters with a typical throughput of 100 Mbps and a maximum 480 Mbps. Its frequency band ranges from 1.60 GHz to 77 GHz.

IEEE 802.15.4-based standard: it focuses on wireless sensor networks that require short range communications with an extended life battery. This standard specifies the physical layer for radio communication operating in 868 MHz, 915 MHz and 2.4 GHz frequency bands and it uses the DSSS modulation technique which limits its data rate to 250 Kbps on a single channel in 2.4 GHz frequency band.

Table 2.5 presents a description of the previously explained standards and their characteristics.

Table 2.5: WSN Standardizations [36]

Standard	IEEE 802.11	IEEE 802.15.1	IEEE 802.15.3	IEEE 802.15.4
Frequency band	2.4 GHz, 5 GHz	2.4 GHz	3.1-10 GHz	868/916Mhz,2.4GHz
Max. throughput	54 Mbps	1 Mbps	110 Mbps	250 Mbps
Nominal range	100 m	10 m	10 m	10-100 m
Transmission power	15-20 dBm	0-10 dBm	-41.3 dBm/Mhz	-25 – 0 dBm
RF Channels	14	79	1-15	1/10, 16

The basic requirements of an Industrial WSN include low data rate, wireless interconnection of ultra-low cost sensor/actuator/processing devices using a relatively short range. The wireless technologies available have benefits and drawbacks that make them suitable for a specific field of application. For Industrial WSN use, certain technologies present disadvantages: IEEE 802.11 standard offers a high throughput but its power consumption is high and the hardware requirements make the implementations cost to rise. Bluetooth's high robustness and security levels are outstanding but their design involves high complexity and its sensor nodes power consumption is high. The RFID standard range comprises its application for industrial wireless networks as active tags require high power consumption and the scalability of the network is strongly limited because of the cost of the tags. IEEE 802.15.3 has been under development and it is still addressing problems such as the multiple access node and multipath interference. Table 2.6 shows the usage of WSN radio technologies for industrial automation by 2008.

Table 2.6: Wireless technologies usage in industrial automation [36]

Radio Technology	Usage
802.15.4	49.10%
Proprietary	23.60%
IEEE 802.11x	11.00%
ZigBee and ZigBee Pro	7.80%
IEE 802.15.1	4.30%
Others	4.30%

The flexible control design, suitable for many applications, the low-data-rate, low cost and the low QoS requirements along with the low Micro Controller Unit (MCU) requirements

and the support of large network orders ($\leq 65K$ nodes) make IEEE 802.15.4 standard the most suitable for industrial WSN deployments. This protocol offers selectable levels of security, encryption, sender authentication and message integrity. This standard is explained with more detail in the next section.

2.2. IEEE 802.15.4 Standard

The IEEE 802.15.4 standard defines the protocol and interconnection of devices via radio communication in a personal area network (PAN) by specifying the physical (PHY) and media access control (MAC) layers, which could be used by a wide range of higher-layer protocols.

Table 2.7: IEEE 802.15.4 standard versions [37]

Version	Description
802.15.4-2003	Initial release: PHY at 868 MHz, 915 MHz and 2.4 GHz.
802.15.4-2006	Updated release: data rate increase and new modulation schemes
802.15.4a	Further PHYs defined: UWB PHY and 2.4 GHz using chirp spread spectrum
802.15.4b	Most recent update
802.15.4c*	Standard specifications for China
802.15.4d*	Standard specifications for Japan
802.15.4e*	Standard specifications for industrial applications
802.15.4f*	Standard specifications for Radio-Frequency Identification (RFID)
802.15.4g*	Standard specifications for smart utility networks (SUNs) for monitoring the Smart Grid

* Special version using the same base radio technology and protocol as defined in 802.15.4a/b

IEEE 802.15.4 is probably the largest standard for low-data-rate WPANS, and it was developed for low-data-rate monitor and control applications. As other established standards, 802.15.4 has been evolving through different versions, shown by Table 2.7.

The objective of IEEE 802.15.4 standard is to provide a base format to other protocols and features for upper layers (3 through 7). This standard does not intend to specify higher-layer protocols, leaving this task to different standard working groups so that they can define it according to their market or application. The best know is ZigBee, but the IEEE 802.15.4 radio transmission standard is used by at least a dozen of other standards. The stack of this protocol is presented in Figure 2.8.

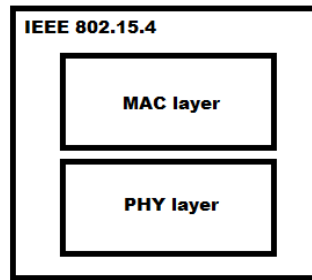


Figure 2.8: IEEE 802.15.4 protocol stack

IEEE 802.15.4 uses CSMA/CA and supports star as well as peer-to-peer topologies. The media access is contention based; however, using the optional superframe structure, time slots can be allocated by the PAN coordinator to devices with time critical data. The PAN coordinator provides connectivity to higher performance networks.

Industrial applications use mainly four protocols based on the 802.15.4 RF standard: WirelessHART, ZigBeePRO/ZigBee RF4CE, ISA 100.11a and 6LoWPAN.

WirelessHART: It is the wireless version of the HART protocol, a specialized standard used in factory and process control. It is a fully featured mesh network containing field devices, gateways and network manager. This protocol takes the approach of using synchronized devices allowing the implementation of, automatic and coordinated channel hopping to reduce the effects of interference.

ZigBee: it is by far the best known of the networking standards that sits on top of the IEEE 802.15.4 MAC/PHY. It has mesh network capability, bringing redundancy and extended range to applications. The ZigBee stack is presented in Figure 2.9.

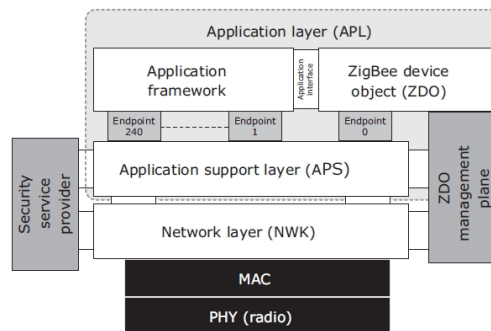


Figure 2.9: ZigBee stack [5]

The key components of it include the network layer (NWK), the application support layer (APS) and the application layer (APL) with its profiles. The NWK layer is responsible to manage the network formation, the addressing of the nodes and routing tasks. The application layer proposes a framework for distributed application development and communication. The application framework can be comprised by a maximum of 240 application objects that consist of software units handling dedicated hardware distributed all over the network. ZigBee technology can be used in industrial monitoring and control applications as well as in building and home automation, embedded sensing applications and energy system automation by deploying low cost and low power consumption devices.

ISA 100.11a: it is an open standard supported by ISA and targeted to the development of low data rate monitoring and process automation applications. It addresses the problem of

non-critical monitoring, alerting and supervisory control as well as to overcome the needs of process control. It supports mesh and star topologies and enables the implementation of simple, flexible and scalable security functionalities.

6LoWPAN: the IPv6 over Wireless Low Power Area Networks protocol was developed by the Internet Engineering Task Force (IETF) as an effort to enable the end-to-end communications by using a common protocol working on top of different radio technologies such as WiFi, Ethernet, and 802.15.4. The usage of an open standard such as IP as the common networking layer provides greater robustness and flexibility to the system.

The 6LoWPAN adaptation layer solves the problem of the time-varying link relationship among the nodes of a WSN. It has support to the implementation of routing protocols at either link or network layer, for example, mesh under routing or route over routing. Using this standard makes possible the deployment of an IPv6 low-power wireless personal area network, providing internet connectivity to low-power devices.

One of the reasons that have made 802.15.4 so well-known and used is that there are no licence fees or restrictions around its use. Its simplicity and the availability of development kits and chips have helped to keep it as a popular choice for the development of new solutions, applications and products. However, these advantages have also associated intellectual risks, as there is no guarantee that using this protocol is not transgressing existing patents.

2.2.1. IEEE 802.15.4 Physical layer

The original version of IEEE 802.15.4 was released in 2003; it establishes the physical layer for radio communications operation in 868 MHz, 915MHz and 2.4 GHz frequency, as shown in Table 2.8. Later releases increased the data rates in the 868 MHz and 915 MHz and the most recent (802.15.4a) adds modulation schemes for accurate location and an UWB PHY [38]. The lower frequencies 868 MHz and 915 MHz frequencies offer greater range and less power consumption but since they are not global and they have channel limitations in the European region they are unattractive for manufacturers. As a result, the 2.4 GHz option is the most used by commercial chips and products. The radios defined by the IEEE 802.15.4 are shown in Table 2.8.

Table 2.8: Radios defined in IEEE 802.15.3-2003 [36]

Frequency	Channels	Throughput (kbps)	Region
868 MHz	1	20	Europe
915 MHz	10	31	USA
2.4 GHz	16	250	Global

The 2.4 GHz Radio Frequency (RF) specification uses Direct Sequence Spread Spectrum (DSSS) for the radio transmission. Since the low-power radios for which 802.15.4 is designed is typically optimized to transfer occasional bits of data with low latency, the data rate is irrelevant. The raw bit rate for the 802.15.4 RF transmission is 2 Mbps. Instead of using this data rate for a high transfer rate, the radio uses a chipping scheme, whereby 32 chips

are used to represent every four bits of data. This reduces the real throughput by a factor of eight, to 250 kbps but it improves the receive the receiving sensitivity as there is an effective gain of eight in the resistance to interference, making it easier to pull data out of noise during the receive correlation performance. The result of this is a range increase.

Since IEEE 802.15.4 networks are comprised of low-power devices, the RF transmission power is limited to between -3 dBm and 0 dBm. The chipping scheme previously described helps to compensate this low transmission power by improving the receive sensitivity. The basic receive sensitivity called for in 802.15.4 is -85 dBm for 2.4 GHz radios [29].

In the 2.4 GHz band, 802.15.4 specifies up to 16 channels, each 2MHz wide, spaced 5 MHz apart. Channel numbering starts at 11 for 2.405 GHz, up to 26 at 2.480 GHz. Lower channel numbers are allocated to the 868 MHz and 915 MHz. Figure 2.10 shows the spectrum usage of the standard.

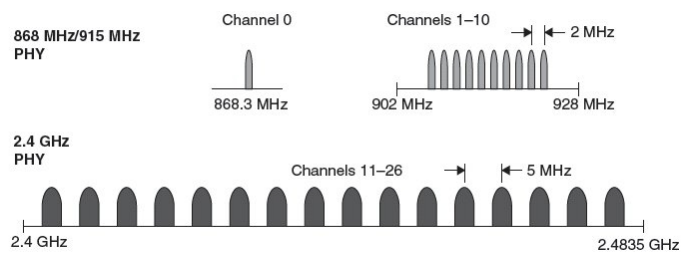


Figure 2.10: Spectrum usage for 802.15.4 [36]

Power consumption is a primary concern for 802.15.4, in order to guarantee a long battery lifetime the energy must be taken at an extremely low rate or in small amounts at a low power duty cycle meaning that devices are active only during a short time. The standard allows devices to be inactive over 99% of the time, for this reason the data rate supported are relatively high in order to minimize the device duty cycle.

IEEE 802.15.4 transmission is organized in four frames, used depending on the application. Each of these frames is designated as a Physical Protocol Data Unit (PPDU): a beacon frame, a data frame, an ACK frame and a MAC command frame. They are all structured with a Synchronization Header (SHR), a Physical Header (PHR), and a Physical Service Data (PSDU), which is composed of a MAC Payload Data Unit (MPDU), which, is constructed with a MAC Header (MHR), a MAC Footer (MFR), and a MAC Service Data Unit (MSDU), excepting the ACK frame which does not contain an MSDU. A more detailed description of the frames can be found at [29].

2.2.2. IEEE 802.15.4 MAC layer

The MAC layer controls the flow of frames that pass through the radio and travel over the air. 802.15.4 MAC is designed to serve as the base for many different network topologies and higher-layer stacks by offering security, guaranteed bandwidth, beaconing services and network formation by the node association. In order to overcome the limited transmission range, multi-hop self-organizing network topologies are required. 802.15.4 Standard describes two types of network nodes: full-function devices (FFD) and reduced-function devices (RFD).

RFD: it is a simple end node than can only talk to a FFD as they lack of a routing functionality. They are also called child devices for they need a parent to communicate. They are able to go through long sleeping periods as they are not needed for routing purposes. Because of their low functionality, these devices can be deployed at a very low cost.

FFD: they are capable of routing network data between nodes and also work as a RFD. A WPAN coordinator is a special form of a FFD that, in addition to the routing capabilities, it is responsible for setting up and maintaining the network.

Two basic topologies are allowed, but not completely described by the standard as the definition of higher layer is out of the scope of the IEEE 802.15.4 protocol:

Star: formed around a FFD acting as a WPAN coordinator being the only node that is allowed to form links between nodes. Although all of the nodes can talk to the coordinator, none of them can communicate with each other even if some of them are FFDs themselves. Figure 2.11 shows the 802.15.4 star topology.

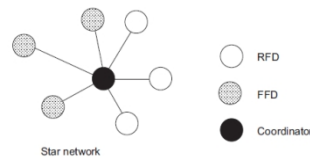


Figure 2.11: Star topology [36]

Star topology is preferred when the coverage area is small and a low latency is required by the application. In this topology, the coordinator controls the communication by sending packets, denoted as beacons, for synchronization and device association. A network device wishing to join a star network listens to a beacon message and after receiving it, it sends an association request back to the WPAN coordinator, which allows or denies the association.

Peer-to-peer: where each device is able to form multiple direct links to other devices so that redundant paths are available. There is not any difference in the way the nodes are physically constructed. The WPAN coordinator has allowed them to act as routers rather than just FFDs. Figure 2.12 shows the 802.15.4 peer-to-peer topology.

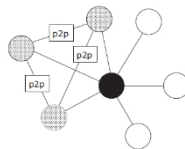


Figure 2.12: Peer-to-peer topology [36]

This topology is preferable when a large area has to be covered and latency is not a critical issue. It is a more complex network where the FFDs are communicated via multiple hops. Each device needs to proactively search for other network devices and once found, they can exchange parameters to recognize the services and features each supports.

Star networks can be strung together with a backbone of FFDs to produce a cluster-tree network. Each cluster-tree network requires at least one PAN coordinator. Mesh networks can be implemented in the same way but they require network formation and routing controlled by higher layers. Figure 2.13 shows the cluster-tree topology.

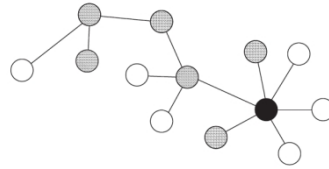


Figure 2.13: Cluster-tree topology [36]

IEEE 802.15.4 transmission is organized in four frames, used depending on the application. Each of these frames is designated as a Physical Protocol Data Unit (PPDU): a beacon frame, a data frame, an ACK frame and a MAC command frame. They are all structured with a Synchronization Header (SHR), a Physical Header (PHR), and a Physical Service Data (PSDU), which is composed of a MAC Payload Data Unit (MPDU), which, is constructed with a MAC Header (MHR), a MAC Footer (MFR), and a MAC Service Data Unit (MSDU), excepting the ACK frame which does not contain an MSDU.

The IEEE 802.15.4 security includes 128-bit AES. It provides protocols to use this at the baseband level, and also exposes the AES engine for use by higher layers.

The MAC layer provides access control to a shared channel and reliable data delivery. IEEE 802.15.4 uses a protocol based on CSMA/CA algorithm. Collision avoidance algorithms require listening to a channel before transmitting in order to reduce the probability of collisions with other ongoing transmissions. IEEE 802.15.4 defines two different operational modes for the channel access namely beacon-enabled mode and non beacon-enabled mode.

In the non beacon-enabled mode, nodes use an unslotted CSMA/CA protocol to access the channel and transmit their packets. Figure 2.14 illustrates the steps of the CSMA/CA algorithm for the non beacon-enabled mode starting when the data from the node needs to be transmitted.

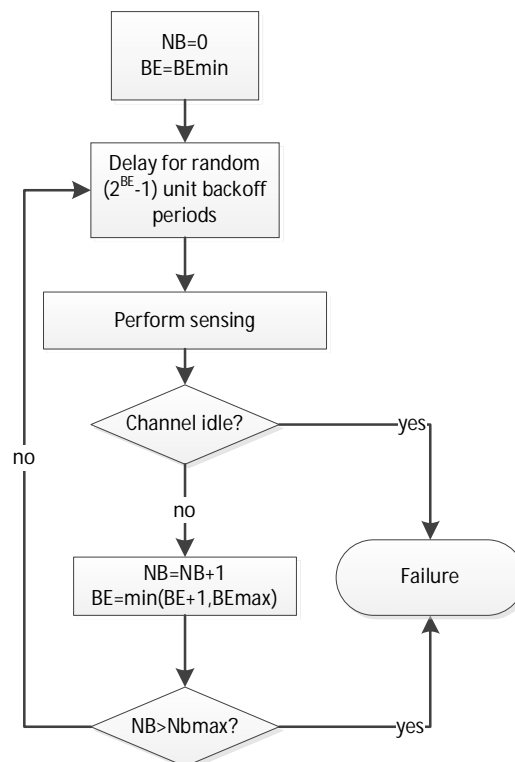


Figure 2.14: CSMA/CA algorithm for the non beacon-enabled mode [36]

2.2.3. IEEE Beacon-enabled mode

In the beacon-enabled mode, the access to the channel is managed through a superframe, starting with a packet, called beacon, transmitted by the WPAN coordinator. The superframe can contain an inactive part, allowing nodes to go in sleeping mode. The active part is divided into two parts: the Contention Access Period (CAP) and the Contention Free Period (CFP) composed of Guaranteed Time Slots (GTSs) that can be allocated by the coordinator to specific nodes. The use of these GTSs is optional. Figure 2.15 shows the superframe. There are 16 timeslots between the beacons, during which any node can transmit, using a standard CSMA/CA scheme. Up to seven of the slots can be configured to be used for contention-free access, where nodes are assigned guaranteed time slots.

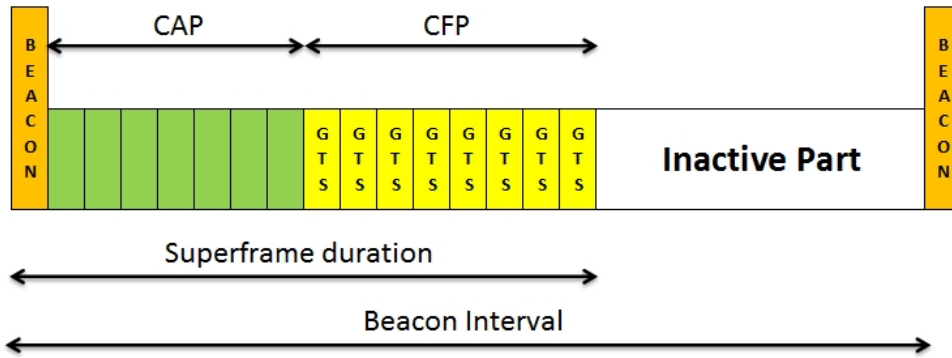


Figure 2.15: 802.15.4 superframe structure

The duration of the active part of the superframe is called superframe duration (SD) and can be expressed as

$$SD = 960 \cdot 2^{SO} \cdot T_s \quad 2.1$$

Where SO is the superframe order, an integer parameter ranging from 0 to 14, and T_s is the symbol time. For a 2.4 GHz band, the symbol rate is 62.5 K symbols/s, which bring $T_s = 16 \mu\text{s}$.

The duration of the superframe is the interval of time between two successive beacons, is called beacon interval (BI)

$$BI = 16 \cdot 60 \cdot 2^{BO} \cdot T_s \quad 2.2$$

where BO is the beacon order, an integer parameter ranging from 0 to 14. BO must be not smaller than SO. According to the standard, each GTS must have a duration multiple of $60 \cdot 2^{SO} \cdot T_s$ and must contain the packet transmitted by the node to which the GTS is allocated to and an inter-frame space, that is the minimum interval between two subsequent packets received. The WPAN coordinator must allocate up to seven GTSs, but a sufficient portion of the CAP must remain for contention-based channel access.

The CSMA/CA algorithm used in the CAP portion of the superframe is presented in Figure 2.16.

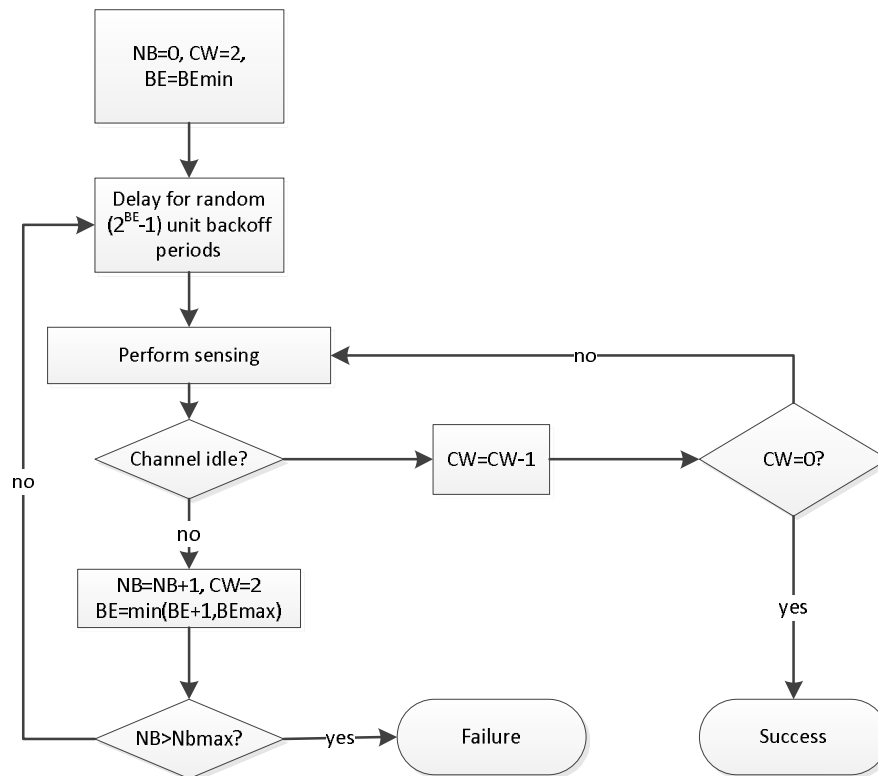


Figure 2.16: CSMA/CA algorithm for the beacon-enabled mode [36]

2.3. 6LoWPAN

The IPv6 over Low-Power Wireless Personal Networks, 6LoWPAN, is an 802.15.4 based standard that breaks down the barriers to using IPv6 in low-power, processing-limited embedded devices. This protocol brings the IP directly down to small, low-cost sensor devices. Knowing that there are not enough of the current IP-format addresses with the current IP version in order to extend the IoT, 6LoWPAN starts from the premise of IPv6, with the aim of giving an address to every device.

If a set of 40-byte IPv6 addresses were to be placed in a 127-byte 802.15.4 frame, there would be little room for any payload. In order to give a solution to this problem, 6LoWPAN uses stateless address compression to reduce the address of the devices to a handful of bytes.

6LoWPAN was developed by the Internet Engineering Task Force (IETF) as an open standard that promises to extend existing IP networks to individual sensor nodes. The stack, shown in Figure 2.17, specifies UDP and not TCP as a transport, limiting any unnecessary clutter in the packets.

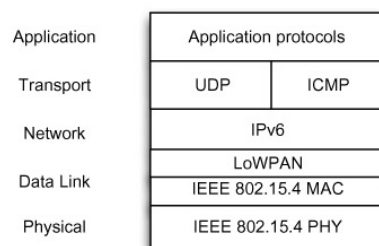


Figure 2.17: 6LoWPAN protocol stack [5]

Using the wireless embedded internet approach brings multiple benefits to a great range of applications by integrating them to the IoT, for example: easy connection to other IP networks without the need of translation gateways; integration with existing network infrastructure; high robustness and reliability given by IP protocol; use of open, free standard with documentation available to everyone; tools for managing, commissioning and diagnosing IP-based networks.

IPv6 features such as a simple header structure and its hierarchical addressing model made it ideal for use in wireless embedded networks with 6LoWPAN. The possibilities, that IPv6 offers of creating lightweight implementations for low computing capabilities and the design of a Neighbor Discover (ND) specifically for 6LoWPAN have helped this standard to become an efficient extension of IPv6 into the wireless embedded domain that has been able to address the integration problems [5].

2.3.1. Architecture

The wireless embedded internet is comprised of islands of connected wireless embedded devices, creating a stub network on the Internet which IP packets are sent from or destined to and it never works at a transit media for other networks. 6LoWPAN architecture, presented in Figure 2.18 is built of IPv6 stub networks of low-power wireless area networks (LoWPANs) which share a common IPv6 address prefix (the first 64 bits of an IPv6 address).

LoWPANs consist of nodes, which may play the role of hosts or routers, along with one or more edge routers. Nodes need to register with an edge router in order to enable network operation such as the ND, which defines how hosts and routers interact with each other on the same link. The edge routers are in charge of incoming and outgoing traffic routing while handling 6LoWPAN compression and ND. In case the network is connected to an IPv4 network, the edge router will handle IPv4 interconnectivity. Each LoWPAN node is identified by a unique IPv6 address and is capable of sending and receiving IPv6 packets, the support user datagram protocol (UDP) as transport mechanism [5].

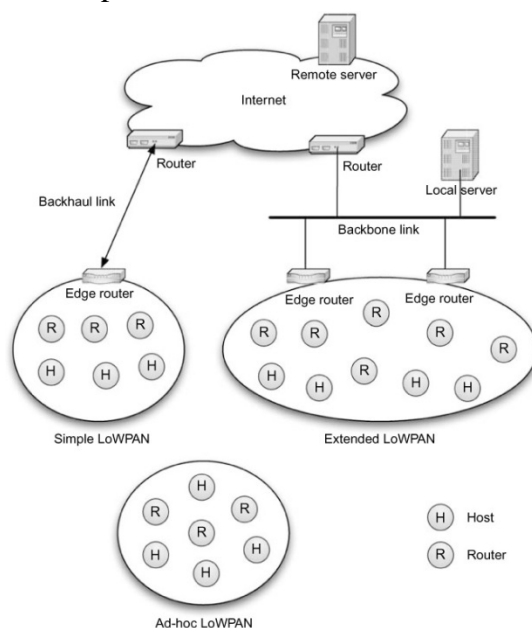


Figure 2.18: 6LoWPAN architecture [5]

Three different LoWPANs have been defined: Simple, Extended and Ad hoc:

Simple LoWPAN: it is a LoWPAN connected through one edge router to another network.

Extended LoWPAN: it is a LoWPAN connected through multiple edge routers along with a backbone link interconnecting them.

Ad hoc LoWPAN: it is a LoWPAN that is not connected to the internet, it operates without an infrastructure.

2.3.2. Protocol stack

The 6LoWPAN protocol stack only supports IPv6, for this reason, a small adaptation layer called the LoWPAN adaptation layer, is defined to handle the IPv6 packets transmission over 802.15.4. The transport protocol used with 6LoWPAN is a compressed version of UDP. The transmission control protocol (TCP) is not used because of its complexity. The internet control message protocol version 6 (ICMPv6) is used for control messaging. Application protocols are application specific and in binary format. Figure 2.19 shows the protocol stack and its comparison with the IP protocol stack.

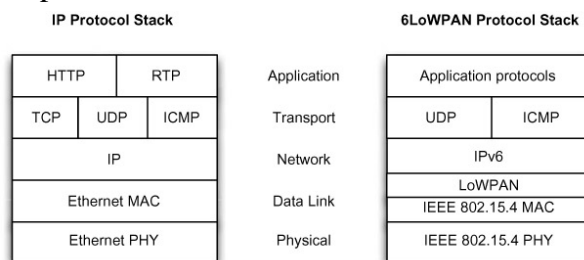


Figure 2.19: IP and 6LoWPAN protocol stacks [4]

2.3.3. Adaptation layer

Adaptation between full IPv6 and the LoWPAN format is performed by the edge routers of the stub networks. This adaptation must be a transparent, efficient and stateless transformation in both directions. Figure 2.20 shows an IPv6 edge router with 6LoWPAN support. Note that, inside the LoWPAN, hosts and routers do not need to work with full IPv6 or UDP header formats.

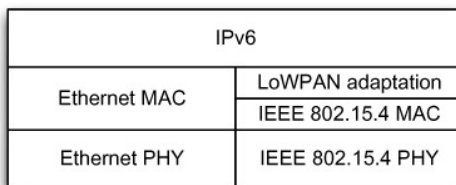


Figure 2.20: Edge router with 6LoWPAN support [5]

The functions of the adaptation layer are to map the IP datagrams to the services provided by the sub network, usually considered to be at layer 2 (L2) of a layered reference model.

Once the IP layer has decided on the IP address of the next hop for a packet, one of the main tasks of the adaptation layer is to find out to which link-layer address the packet needs to be addressed so that it advances to its intended IP-layer destination. The sub network may not immediately provide a path for packets to proceed to the next IP node; therefore, the adaptation layer must set up and then manage the connections.

The IP packet needs to be encapsulated in the sub network so that it can be transported through it and then extracted by the L2 again, the adaptation layer must address important problems that are caused by this encapsulation. Links may need to distinguish different kinds of encapsulation, meaning that they need to be able to carry packets of other types than just IP datagrams. Since IP was designed so that each packet can stand on its own [39], the header may contain a lot of information that can be inferred from its context. The typical IP/UDP header size of 48 bytes consumes a big part of the payload space available in an 802.15.4 packet; for this reason, the adaptation layer is also in charge of the header compression discussed in depth by [5].

Another important function of the adaptation layer is to deal with this fragmentation and reassembly of packets. IP packets may not fit into the data units that L2 can transport. IPv6 defines a minimum value for the MTU of 1280 bytes, on the other hand, IEEE 802.15.4 can only transport L2 packets of up to 127 bytes. In order to be able to transport larger IPv6 packets, L3 packets must be fragmented into multiple L2 packets and then reconstruct the segments of the packet again.

2.3.4. Link layers

6LoWPAN is designed to be used with a special type of link with its own requirements and specifications. This standard was designed with IEEE 802.15.4 protocol in mind, for this reason certain optional concepts in the 6LoWPAN format are similar to the 802.15.4 link layer. At the early stages of the 6LoWPAN standardization work, 802.15.4 specific features were assumed to be used for 6LoWPAN implementations, recent standardizations, have been generalizing and even avoiding the use of IEEE 802.15.4-exclusive features.

The basic service required of the link layer is the possibility of sending packets of a limited size to another node, this process is also called unicast packet. As explained in Section 2.2.1, IEEE 802.15.4 MAC layer defines four types of frames: data, acknowledgment, MAC layer command and beacon frames. The only frame that concerns the 6LoWPAN specification is the data frame as it used to carry protocol data units (PDU) defined by the adaptation layer, these PDU contain embedded IPv6 packets.

The link layer must have a concept of a globally unique addressing. The fact that an address uniquely identifies a node does not mean that it can be used to know the location of this node. Data frames carry both, destination and source address. IEEE 802.15.4 nodes are permanently identified by EUI-64 identifiers, which fit in at 8 bytes. Since a pair of 64-bit source and destination already utilizes one eighth of the usable space in a packet 802.15.4 also specifies a short address format that can be dynamically assigned. For further description about the addressing mechanism used by 6LoWPAN refer to [5].

Data confidentiality and integrity is required in order to have a robust and reliable mechanism for LoWPAN separation. 6LoWPAN relies on the IEEE 802.15.4 security mechanisms for encryption and message integrity check including key identification of its link layer.

6LoWPAN maintains a neutral position with respect to some of the IEEE 802.15.4 MAC layer functionalities; for instance, it allows the use of the beacon-enabled networks, although this is not usually being done in practice today. 6LoWPAN networks are much more likely to run in beaconless mode, performing wireless media access control using the IEEE 802.15.4 channel access method.

2.3.5. Fragmentation and reassembly

As explained in the Section 2.3.3, fragmentation and reassembly mechanisms are required in order to be able to transmit IPv6 packets wirelessly through 802.15.4.

IPv4 reassembly operates at the destination of the packet. As the Internet does not guarantee the sequential of packets, the fragments of a packet might arrive out of order. All the datagram fragments that share source address, destination address, IP protocol field value, and identification field value are positioned into the reassembly buffer based on their fragment offset in order to be grouped together at the final destination of the datagram. The datagram is complete when the last fragment is present and all the byte positions up to the last fragment have been filled by other fragments. This means that IPv4 reassembly process does not know how big the datagram will be until the last fragment has been received. It is possible that fragments of packets are lost, leaving the complete datagram with fragments missing, for this reason, the reassembly process needs to employ timers to give up on reassembling a datagram after some time. In this case, the fragments that did arrive have been transmitted in vain as the information cannot be processed or used for any purpose. Figure 2.21 shows IPv4 fragmentation fields in the header.

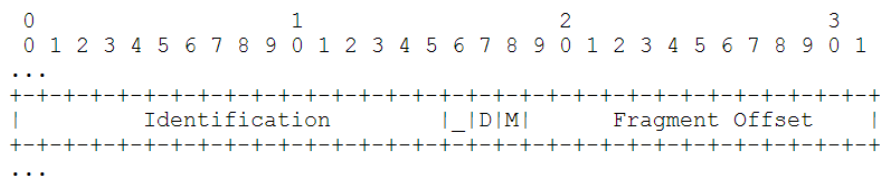


Figure 2.21: IPv4 fragmentation header

IPv6 simplified the base IP header by not using the IPv4's fragmentation fields. If a source of a datagram needs to use fragmentation, it will need to add an additional fragmentation header as an extension header shown by Figure 2.22. IPv6 minimum MTU is of 1280 bytes. In IPv6 packets may never be fragmented in transit, for this reason, the defragmentation flag field is not needed.

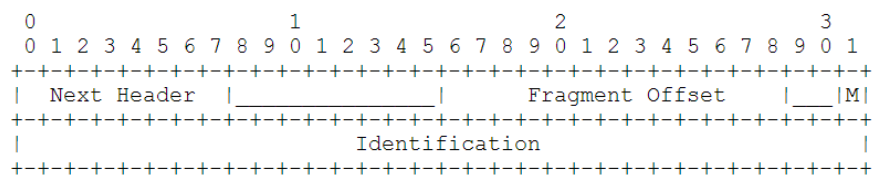


Figure 2.22: IPv6 fragmentation header

In summary, even if IPv6 provides its own fragmentation for datagrams bigger the minimum MTU of 1280 bytes, it relies on the sub network layer to be able to transport them; meaning that the adaptation layers are in charge of this fragmentation and reassembling of packets at the receiving node.

6LoWPAN fragmentation and reassembly mechanism copies the size of the packet to be reassembled (header + payload) into every fragment instead of just providing a “more-fragments” flag like IPv4 does. Doing this allows to allocate the fragments into a buffer for the whole reassembly unit upon reception of the first fragment, independent of which of the fragments actually arrives first. This datagram size field is 11 bits in length, allowing reassembling units of 2047 bytes. In order to distinguish the different packets to be reassembled, a 16-bit datagram combined with the sender’s link layer address, the destination’s link layer address and the datagram size is used. An 8-bit datagram offset indicates the position of the fragment in the reassembled IPv6 packet. The first byte of a fragment indicates the type of frame. Figure 2.23 shows the resulting frame of an initial fragment of a 6LoWPAN fragment sequence.

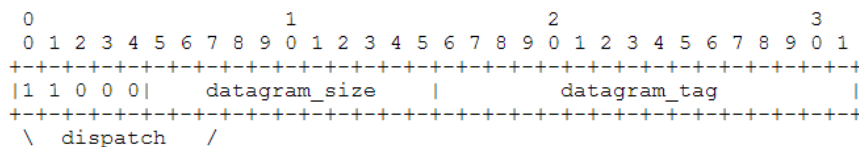


Figure 2.23: 6LoWPAN fragmentation header

In 6LoWPAN, the WPAN is able to reassembly fragments from only one source at a time; if fragments from a different source from the one being processed arrive the coordinator will discard the packet until the processed datagram is complete. For further explanation about the 6LoWPAN reassembly and fragmentation mechanism and the complete sequence that a node follows in order to send a 6LoWPAN PDU too big to fit into a link-layer refer to [5].

2.4. Wireless Scheduling

A network user is a networked device, messaging system or service that uses a computer network to transfer organized messages [40]. In the WSN context, the network users are all the sensor nodes connected to the sink node or router that are wirelessly receiving and transmitting messages. In a similar way, in wireless telecommunication, a network user, called User Equipment (UE), is a networked device or equipment wirelessly broadcasting and requesting organised data from or to a sink node. Knowing a network user has a similar concept regardless of their context, the concept of network user is equivalent to network node or sensor node and it will be used during the following sections.

Scheduling is the process of deciding how to commit resources between a variety of possible tasks [41]. Regardless of their application, in wireless networks, the problem of assigning transmission rights to subsets of network users at each time and under different channel conditions is known as scheduling problem. It concerns to wireless environments due to three main reasons that are related to the wireless medium characteristics: geographically sep-

arated users share communication resources; transmission interference; transmissions undergo impairments such as fading [9]. In other words, scheduling is identifying the users that are able to transmit.

The scheduler is the mechanism that identifies and evaluates user's properties in order to perform the decision of giving channel access to a certain user or subsets of users [42]. There are numerous of possible objectives to achieve when the scheduler manages the transmission in a wireless network. One of the most important objectives can be maximizing the total packet throughput (data rate) on the network. Another one is to achieve the largest throughput region for each source/destination pair. A further important objective can be the minimization of data losses and transmission power usage that might occur in networks with fragmentation and reassembly mechanisms where data from nodes can be transmitted in vain if the mechanism in charge of reassembling datagram packets is only able to process and reassemble datagrams from finite number of users. These performances objectives are considered to be rate criteria, also called best-effort criteria, where the capacity of the nodes is the most important characteristic for the scheduling decision.

The objectives of the fairness criteria are a variation of the rate criteria objectives. For the fairness criterion, in order to maximize the throughput of the network, the channel access distribution between source and destination is fair. This criterion seeks a balance between overall high throughput and individual throughput components employing a utility function in order to score network nodes. Since fairness criterion is not looking to guarantee a QoS, it is considered to be part of the best-effort criteria.

By using a best effort scheduled channel access mechanisms, the users with good channel quality can benefit from high data rate and, in presence of several users, the packet scheduling algorithms can benefit from user diversity and schedule users during constructive fades. In the case where there are several users, the scheduler can adapt to fluctuating radio conditions and choose the next user to be served based on the channel conditions of all users, maximizing the global throughput by choosing the users with the best channel conditions. Figure 2.24 shows how this mechanisms works: at every TTI, the User Equipment (UE) with the best channel quality will be scheduled.

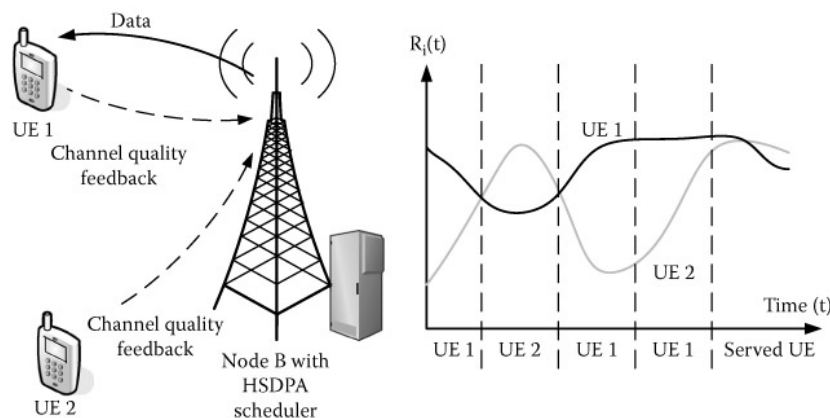


Figure 2.24: Max C/I Scheduler scheme [43]

Yet another scheduling objective can be having a stable throughput region where packets at the nodes' buffers experience minimum finite delays. There is another category for this criteria classification called QoS criterion. Most wireless networks can be affected by arbitrary changes over time caused by mobility, variable fading and addition or deletion of users to or from the network. In cases where this factors affect the throughput values, different set of objectives need to be considered, this objectives are part of the QoS scheduling criteria. One option is to find an ultimate capability by considering an initial traffic volume and then finding the scheduling policy that empties the network in minimum time [43]. QoS schedulers are required by users of services such as video streaming whose requirements have a minimum bit rate and maximum packet delay. User satisfaction is the main factor to consider for the scheduling decision. Figure 2.25 presents a QoS scheduling schema where the packet scheduling should aim to fulfil QoS requirements instead of just focusing on the network throughput or fairness.

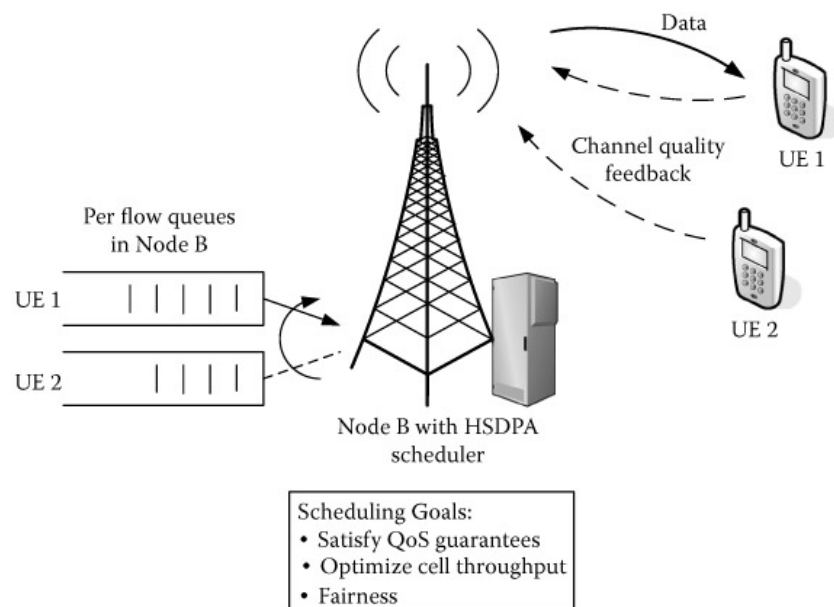


Figure 2.25: QoS scheduling scheme [43]

The scheduler is the key to resource management in the downlink because it decides which user is to be scheduled in each TTI. As it has been explained the scheduling decision not only takes into account the channel condition, but factors such as the fairness between users, the cell throughput or QoS parameters can be considered. The choice of a scheduler involves some trade-offs between these factors [43].

As it has been explained, there are two set of schedulers: best-effort schedulers and QoS schedulers. Best-efforts schedulers do not provide QoS guarantees, whereas QoS guarantees can provide it. Further studies about QoS-aware scheduling can be found at [43]. The following section will focus on the best-effort schedulers as they are important for the purposes of this thesis work.

2.4.1. Best-effort schedulers

The best-effort scheduler algorithm is designed to optimize the global throughput and ensure that resources are divided fairly among different users. Table 2.9 shows a summary of the scheduler using best-effort criteria for the scheduling decision.

The Round Robin (RR) scheduler is probably the simplest one in terms of implementation; it gives time slot to users in cyclic manner making it being fair in terms of resources or time slots but the capacity is not shared equally among the UEs as each user may experience different channel conditions. The RR policy is not optimal with respect to maximizing the global throughput because it does not utilize channel condition feedback. Figure 2.26 shows the cumulative distribution function (CFD) of user throughput when ten users are in the system.

Table 2.9: Best-effort schedulers for WSN [43]

Scheduler	Algorithm (Note that ties are broken randomly):	Scheduled User i^* Is Given by:
Round Robin (RR)	Schedule users in cyclic order	Cyclic manner
Fair Throughput (FT)	Equalize throughput of all users	$\arg \min_i \{\lambda_i(t)\}$
Max C/I	Choose user with best instantaneous supported data rate	$\arg \max_i \{R_i(t)\}$
Proportionally Fair (PF)	Choose user with best transmission rate relative to user's own average throughput	$\arg \max_i \left\{ \frac{R_i(t)}{\lambda_i(t)} \right\}$
Generic Proportionally Fair (GPF)	Similar to PF algorithm with additional exponents γ and η	$\arg \max_i \left\{ \frac{[R_i(t)]^\eta}{[\lambda_i(t)]^\gamma} \right\}$
Score Based (SB)	Choose user with best rank for channel quality based on history	$\arg \min_i \{S_i(t)\}$
Fast Fair Throughput (FFT)	Equalize throughput of all users and exploit channel conditions	$\arg \max_i \left\{ \frac{R_i(t)}{\lambda_i(t)} \frac{\max_j \{\overline{R_j(t)}\}}{\overline{R_i(t)}} \right\}$

Note: Here, for a given user i at time t , $\lambda_i(t)$ is the exponentially weighted moving average throughput, $R_i(t)$ is the instantaneous supportable data rate, $\overline{R_i(t)}$ is the exponential weighted moving average of $R_i(t)$, $\max_j \{\overline{R_j(t)}\}$ is the maximum among different users, $S_i(t)$ is the rank of the current channel condition with respect to past W values for the same user. In addition, W , γ , η are implementation parameters.

The most aggressive policy to utilize the channel feedback is called the Max C/I scheduling. Max C/I stands for maximum carrier to interference ratio and it denotes the signal-to-noise ratio that is used to estimate the optimal value of the instantaneous supportable data rate, $R_i(t)$, with a given block error rate target, $BLER_{target}$. This algorithm is illustrated by Figure 2.24. The Max C/I scheduler gives access to the channel to a user having the best channel conditions, in other words, to the user that can support the maximum instantaneous data rate in the current TTI. This scheduler provides the highest cell (global) throughput because it always serves users with the highest data rate. On the other hand, this scheme is very unfair because a user nearest to Node B can get all the resources, and the users farther will be starved. Figure 2.26 shows the CDF. It can be seen that some users get very good throughput as they are scheduled most of the time due to their good channel quality while others experience the opposite throughput due to their poor channel conditions.

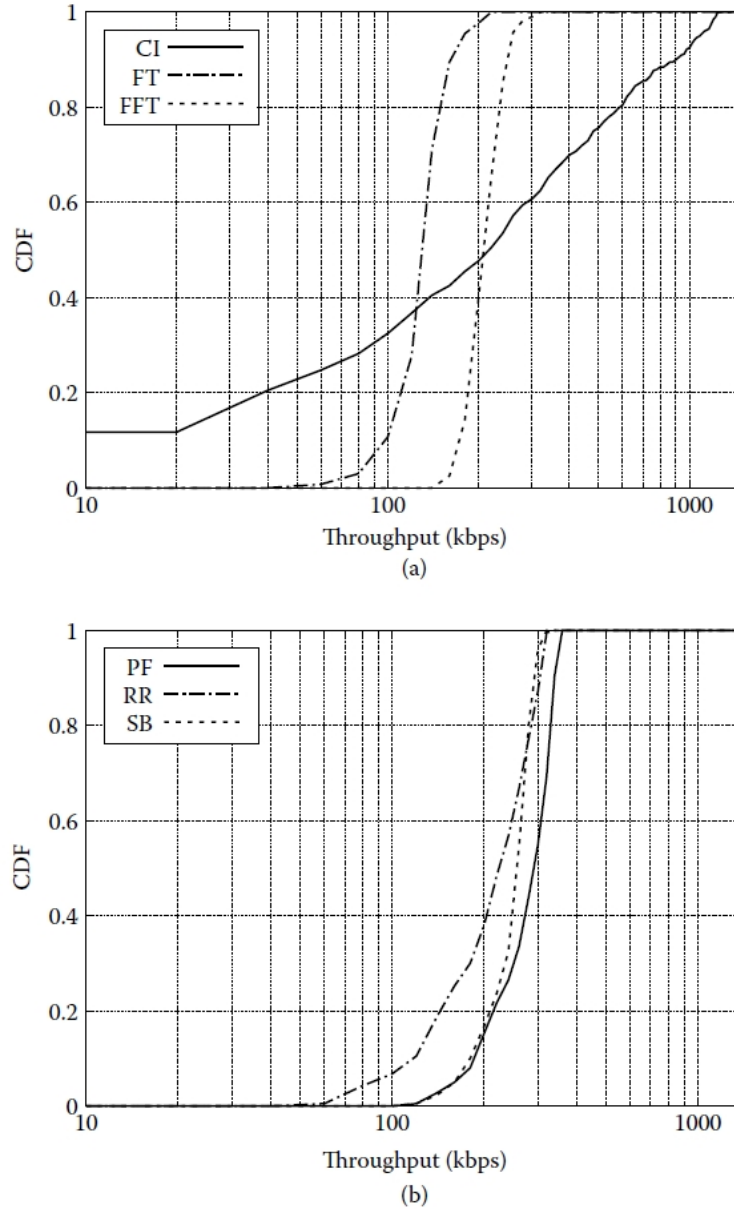


Figure 2.26: CDF plots for different schedulers mechanisms [43]

The Proportionally Fair (PF) scheduling policy looks at the relative channel conditions of a user with respect to its own past channel conditions. PF assigns the current TTI to a user i^* having maximum ratio of supportable instantaneous rate $R_i(t)$ and its own exponentially weighted moving average throughput $\lambda_i(t)$, which is calculated as follows:

$$\lambda_i(t) = \left(1 - \frac{1}{\tau}\right) \cdot \lambda_i(t - \Delta t) + \frac{s_i}{\tau} \cdot R_i(t) \quad 2.1$$

with $\tau > 1$, Δt is equal to the length of the TTI, $s_i = 1$ when user i is served in the current time slot, and $s_i = 0$ otherwise. It is important to note that different implementation specific variants are possible in order to deal with special cases. PF scheduler offers a good trade-off between cell throughput and fairness, as it gives the channel to the user having “relatively

good” channel conditions resulting in a fair resources distribution. Figure 2.27 shows the global throughput with different scheduling policies.

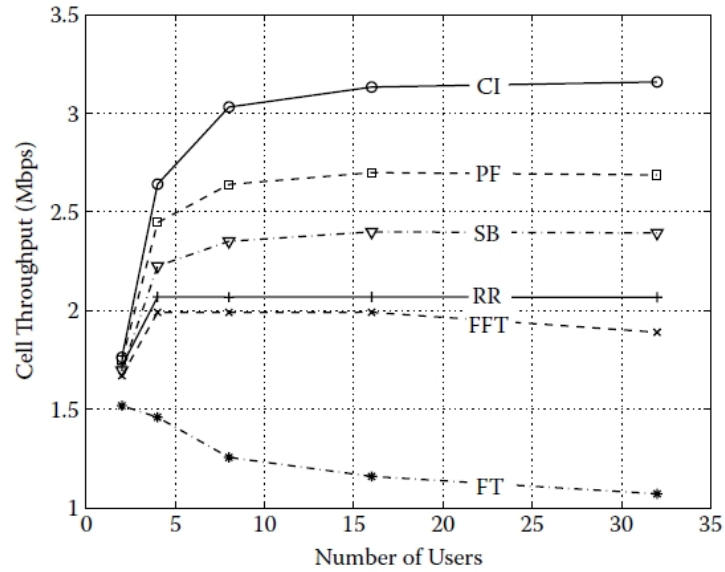


Figure 2.27: Global throughput comparison [43]

The Generic PF (GPF) scheduler has a similar behaviour to the PF but it uses the ratio of $[R_i(t)]^\eta$ and $[\lambda_i(t)]^\gamma$ for deciding the scheduling decision. When γ is increased to more than 1.0 or η is decreased less than 1.0, then the GPF scheduler starts giving priority to users with bad conditions. On the other case, when γ is decreased to more than 1.0 or η is increased less than 1.0, GPF scheduler will behave like a Max C/I. This behaviour is presented by Figure 2.28 where the CDF is shown given different η and γ values.

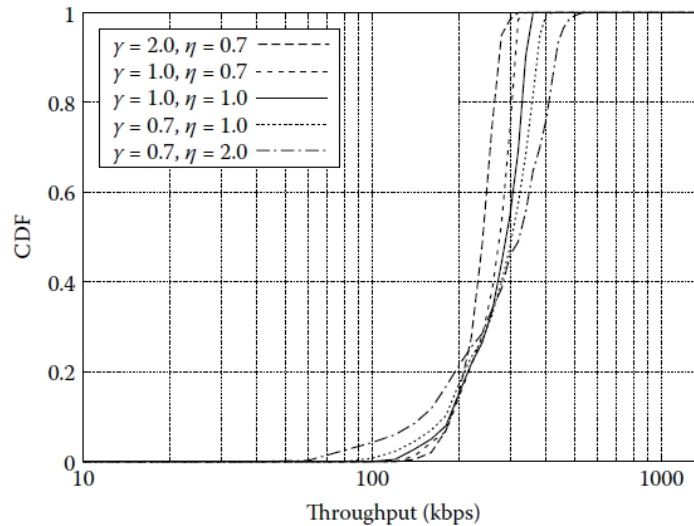


Figure 2.28: Generic PF scheduler CDF [43]

A Score-Based (SB) scheduler ranks the current channel condition of a user with respect to the past channel condition over a window of size T . Users are sorted according to the rank of their current channel condition and then the user having the best rank is scheduled in the current TTI. SB scheduling can distribute the channel access equally even under asymmetric,

this can be seen in Figure 2.26. Figure 2.25 presents the CDF of the user throughput obtained with a SB scheduler.

There is another scheduler that uses the fairness criteria for its scheduling decision: the Fair Throughput (FT). The FT scheduler equalizes the throughputs of all the UE by scheduling the user with minimum average throughput so that all the users in the system obtain equal data rate at a given time. Its CDF can be seen in Figure 2.26. FT scheduler may be affected by bad channel quality and the overall global throughput can be reduced. The reason behind this cell data rate deterioration is that the scheduler will start providing more resources to the users with bad quality meaning a lower total throughput. Therefore, FT must be used supported by a resource manager able to drop some users experiencing low average data rate. Another version of FT is the Fast Fair Scheduler FFT, proposed in [44], that provides fair throughput distribution by exploiting channel condition feedback; its scheduling decision is given by the equation shown in Table 2.9.

The Scheduling priority indicator (SPI) is a score given by the scheduler decision algorithm to schedule users with strict priority, depending on its SPI, such that lower priority users are not scheduled until there is data available in the queues of higher priority users.

2.5. State-of-the-art in IEEE 802.15.4 Scheduling

As it was stated in previous sections, one of the most well-known standards for WSN is the IEEE 802.15.4; it is generally considered one of the most suitable protocols for industrial applications [5]. This protocol defines the PHY and MAC layers specifications for low-power and low-data rate (up to 250Kbps at 2.4 GHz) wireless personal area networks connectivity.

IEEE 802.15.4 MAC layer features two operating modes: a non beacon-enabled mode, in which nodes access the channel using a classic (unslotted) CSMA/CA mechanism and a beacon-enabled mode, in which time is subdivided in superframe, with a slotted CSMA/CA mechanism. In the beacon-enabled mode, nodes subdivide their time into beacon intervals that are delimited by beacon frames periodically broadcast by each coordinator. The beacon interval is divided into an active section, called superframe, and an inactive section, where nodes may go in to energy-save mode (Figure 2.14). The beacon-enabled mode of the IEEE 802.15.4 supports collision-free time slots, called GTSs, which can be exploited for transmitting real-time traffic. The allocation of GTSs guarantees a defined bandwidth. Beacon-enabled mode only supports star or cluster-tree networks [45].

During the last years, many researchers have been working towards the evaluation of the 802.15.4 standard performance [46] [47] [48], this has resulted in the improvements [49] [50] [51]. The GTSs performance analysis was introduced in [108] and then an implicit GTS allocation mechanism intended to improve the bandwidth utilization of GTS was presented in [52] where introduced the scheduling concept was introduced to the 802.15.4 standard by proposing a round robin scheduler algorithm for GTS allocation. However, as it has been presented in previous section, in this kind of scheduling mechanism nodes share bandwidth equally regardless of their requirements which is not efficient.

The 2006 version of IEEE 802.15.4 introduced support for time division superframe scheduling that can be used along with an algorithm to schedule the GTS allocation of the beacon-enabled mode. However, the standard does not define actual mechanisms to schedule superframes to certain nodes in order to improve the network performance by avoiding collisions, distributing the bandwidth efficiently or reducing the power consumption of the WPAN [53]. On the other hand, packet collisions are a major issue. In [38] it is shown that even if multi-hop beacon-enabled networks are feasible when the beacon order is greater than one, the distribution of coordinators is not very dense and the traffic is very low, and due to the collisions, the probability of losing synchronization may not meet the reliability requirements of industrial wireless networks. Furthermore, the scheduling of multiple IEEE 802.15.4 superframes having different beacon intervals and superframe durations presents a problem similar to the Multi-cycle Scheduling Problem [54], studied for field buses devices.

The bandwidth scheduling in the IEEE 802.15.4 protocol has been widely studied during the last years, focusing on the adaptation of the beacon-enabled mode parameters in order to accommodate the timeliness of periodic messages.

Authors in [55] developed a traffic-based algorithm for GTSs scheduling that determines the proper value for beacon order (BO) and superframe order (SO) in order to improve the energy consumption of the network. An adaptive GTSs scheduler is presented in [56], this design proposes a scheduling based in two phases: a device classification phase, using the usage feedback, and a scheduling phase. This work results indicate a traffic latency and fairness improvements. In [57], the authors propose a GTS scheduling algorithm for time-sensitive transactions where QoS-based scheduled algorithms are used. A GTS allocation algorithm for periodic real-time messages is presented in [58]. The algorithm determines network and GTS parameters and guarantees real-time services resulting in an optimal schedulability for real-time applications but, as the GTS information have to be transmitted at the beginning of the beacon interval, the power consumption increases. In [59], a time division beacon frame scheduling (TDBS) mechanism is presented, where IEEE 802.15.4 superframes are scheduled according to a centralized algorithm run by the PAN coordinator called Superframe Duration Scheduling (SDS) based on the Pinwheel scheduling algorithm where the schedulability and the beacons intervals are analysed in order to take the scheduling decision. An optimization of the time division superframe scheduling problem is presented in [60], where an algorithm that finds a collision-free superframe schedule is presented; this approach minimizes the energy consumption of the nodes.

There are some works in the literature that propose mechanisms to avoid collisions of beacon frames on distributed protocols [60] [61]. However, these approaches, that are suitable for WSN applications in home and building automation, do not match the industrial demands such as sensing or control WSN. On the Industrial WSN field, several research focus on the design of data link layers scheduling in WirelessHART networks [62] [63] [64] and on how to satisfy the reliability requirements on the routing area, but the strict limitations of TDMA slots, which is not as flexible as the IEEE 802.15.4 in terms of superframe durations and beacon intervals, has pushed the research to the last one. The advantages of multichannel IEEE 802.15.4 communication in terms of reliability and throughput are assessed in [65] and [66].

For industrial applications, a more centralized approach might be more appropriate [53] due to two reasons: nodes' local knowledge may be not enough to avoid interference and the schedule of a given set of superframes is non-deterministic as it depends on the beacon requests arrival. While a pure time-division approach solves the beacon frame collision problem, it limits the network scalability in terms of the number of clusters for which a feasible superframe schedule is found, as no parallel communication is allowed unless coordinators are distant enough not to collide [53].

2.6. Operating Systems for Wireless Sensor Networks

Wireless sensor nodes are considered Systems on Chip (SoC) as they have communication, computation, sensing, and storage capabilities. They have stringent resources constraints regarding their processing power, battery life and program memory. Figure 2.29 shows the nodes components. A WSN consists of hundreds of thousands of sensor nodes capable of multi-hop communication between each other for monitoring, tracking and controlling applications, among others [67].

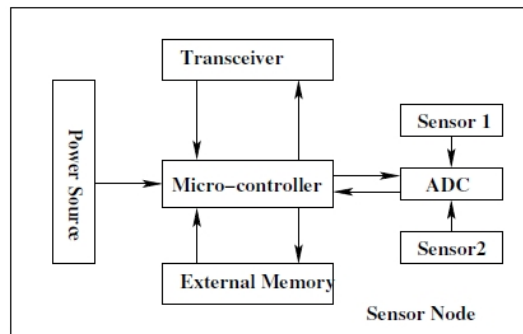


Figure 2.29: WSN nodes components [68]

The Operating Systems (OS) for WSN act as the main resource managers for the nodes. Its basic functionality is to hide low-level details of the node providing interfaces to the external world. OS provide processing, memory and device management, scheduling functionalities and multi-tasking services as well as providing concurrency mechanism for a dynamic use of modules and Application Programming Interfaces (APIs) in order to access underlying hardware. OS for WSN must provide the previous functionalities considering the resource limitations of the sensor nodes [67]. As shown in Figure 2.30, the OS lays over the WSN software layers whereas middleware and application layers are distributed across the nodes.

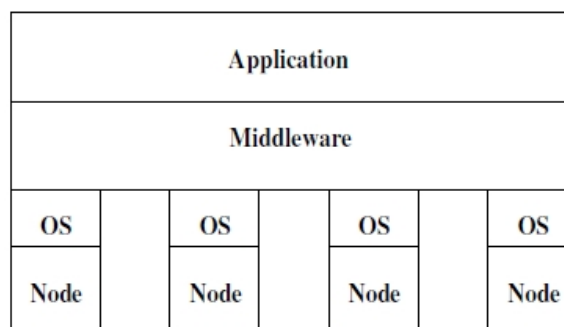


Figure 2.30: WSN software layers [68]

The major issues that have to be considered for the design of WSN OS are: the architecture of the OS; the programming model supported; the task scheduling; memory management and protection; the support of communication protocols; and the resource sharing.

WSN OS give support for real-time applications by providing implementations of communications protocols that support real-time multimedia streams and MAC, network and transport layer protocols that minimize the end to end stream delay. OS provide to the application programmers allowing the implementation of customized protocols. Furthermore, the OS needs to provide an implementation of routing protocol that constructs route according to the QoS requirements applications. Finally, it must support a MAC algorithm that schedules packets according to their priority.

Table 2.10 shows the most used open OS used for WSN development.

Table 2.10: Wireless Sensor Networks Operative Systems [68]

OS/ Feature	Architecture	Programming model	Scheduling	Memory Management and Protection	Communication Protocol Support	Resource Sharing
TinyOS	Monolithic	Primarily event Driven, support for TOS threads has been added	FIFO	Static Memory Management with memory protection	Active Message	Virtualization and Completion Events
Contiki	Modular	Protothreads and events	Events are fired as they occur. Interrupts execute w.r.t. priority	Dynamic memory management and linking. No process address space protection.	mIP and Rime	Serialized Access
MANTIS	Layered	Threads	Five priority classes and further priorities in each priority class.	Dynamic memory management supported but use is discouraged, no memory protection.	At Kernel Level COMM layer. Networking Layer is at user level. Application is free to use custom routing protocols.	Through Semaphores.
Nano-RK	Monolithic	Threads	Rate Monotonic and rate harmonized scheduling	Static Memory Management and No memory protection	Socket like abstraction for networking	Serialized access through mutexes and semaphores. Provide an implementation of Priority Ceiling Algorithm for priority inversion.
LiteOS	Modular	Threads and Events	Priority based Round Robin Scheduling	Dynamic memory management and it provides memory protection to processes.	File based communication	Through synchronization primitives

A brief description of the WSN OSs included on the table 2.10 is presented:

TinyOS: is an open source, flexible, component based and application-specific WSN OS. It supports concurrent programs with very low memory requirements. Its footprint fits in 400 bytes and its library includes network protocols, distributed services, sensor drivers, and data acquisition tools. TinyOS has a monolithic architecture as it is self-contained, and independent from other computing applications. This OS provides a single shared stack without separation between kernel space and user space. Its architecture is shown by Figure 2.31.

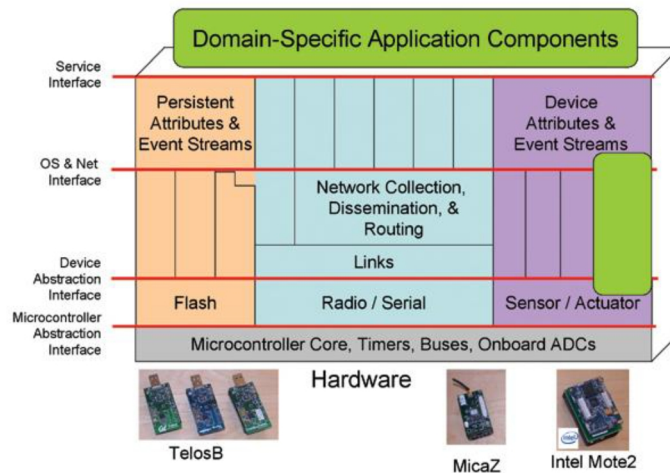


Figure 2.31: Tiny OS architecture [68]

At the MAC layer, TinyOS provides implementations of the following communication protocols: single hop TDMA protocol, TDMA/CSMA hybrid with Z-MAC's slot stealing optimization, B-MAC and optional implementation of IEEE 802.15.4 compliant MAC.

Due to memory scarce at the node, TinyOS provides a single level file system, meaning that only a single application can run on the node at any given point time. It provides database support in the form of a mechanism called TinyDB. The protocol in charge of the security is called TinySec. This OS supports application development in NESC programming language, which is a dialect of the C language. It is well documented and supported. More information about this OS can be found at [69].

Contiki: it is a lightweight open source WSN OS developed written in C for sensor nodes. It is highly portable and it is built around an event-driven kernel. It provides multitasking that can be used at the individual process level. Contiki configuration consumes 2 kilobytes of RAM and 40 Kilobytes of ROM and it includes multitasking kernel, multithreading, proto-threads, TCP/IP networking, IPv6, a GUI, web browser, personal web server, telnet client, screensaver and virtual networking computing. It has a modular architecture, shown at Figure 2.32, as it separates the functionality of the OS into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality [70]. It supports dynamic memory management and it does not provide any memory protection mechanism between different applications.

Contiki supports a wide variety of communication protocols. An application running Contiki can use both versions of IP: IPv4 and IPv6. It has an implementation of *uIP*, a memory efficient TCP/IP protocol stack for 8-bit micro-controllers that does not require its peers to have a complete stack and that is able to communicate with peers running similar lightweight stack. For network-based communication, Contiki supports Rime and it supports an implementation of RPL (IPv6 routing protocol for low power and lossy networks).

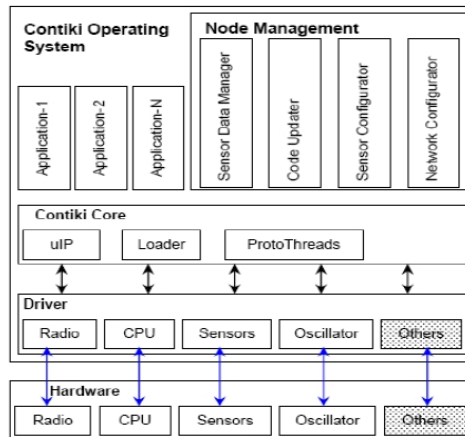


Figure 2.32: Contiki architecture [68]

This OS supports preemptive multithreading implemented as a library on top of the event-driven kernel and divided in two parts: a platform independent, for interfacing to the event kernel, and a platform specific part that implements stack switching and preemption primitives. For multithreading, Contiki uses protothreads [71] designed for memory limited devices. Protothreads are very small memory overhead and highly portable. Contiki is an event-driven OS; therefore it does not employ any sophisticated scheduling algorithm.

Contiki provides file system support for flash-based sensor devices in the form of Coffee file system [72] by providing a programming interface for building efficient and portable storage abstractions. It is a platform independent storage abstraction through a programming interface.

There is no support for secure communication in Contiki but there are proposals and implementations for a secure communication protocol under the name of ContikiSec [72]. Contiki provides simulation support through Cooja [72]. It is widely used for WSN applications, well documented and with a big support community.

MANTIS: the Multimodal system for NeTworks of In-situ wireless Sensors (*MANTIS*) is a lightweight, multithread and energy efficient operation system. Its footprint is of 500 bytes, which includes kernel, scheduler and network stack. The *MANTIS* OS (*MOS*) key feature it is its portability across multiple platforms, for example, its application can be tested on a PDA or a PC [73]. Afterwards the application can be ported to the sensor nodes. *MANTIS* is written in C and it supports applications development in C. It has a layered architecture, presented in Figure 2.33 where the services provided are implemented in layers. Each layer acts as a virtual machine to the layers above.

MANTIS' network stack is divided in two parts: the first part contains implementation of layer 3 protocols to give flexibility; the second part contains the implementation of the MAC and PHY layer operations and it implements synchronization. This OS does not provide support for multicast applications or real-time multimedia applications in its communication protocol.

This OS gives simulation support through *AVRORA* [74] and its documentation can be found at [73].

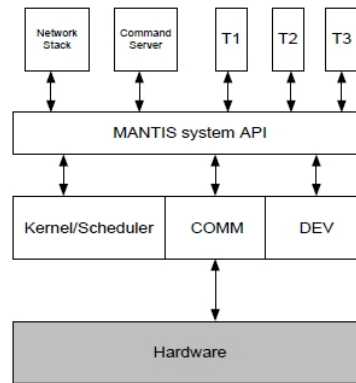


Figure 2.33: Mantis architecture [68]

Nano-RK: it is fixed, preemptive multitasking real-time WSN OS, its main goals are multitasking, multi-hop networking support, priority-based scheduling support, timeless and schedulability, extended WSN lifetime and small footprint. Nano-RK uses a 2 Kb of RAM and 18 Kb of ROM, it provides support for CPU, sensors, and network bandwidth reservations. Its architecture, shown in Figure 2.34, follows the monolithic kernel architecture model where the application can change different parameters (deadline, period, CPU reservation, and network bandwidth reservation) associated with the tasks to arrive at a configuration that meets the overall objectives.

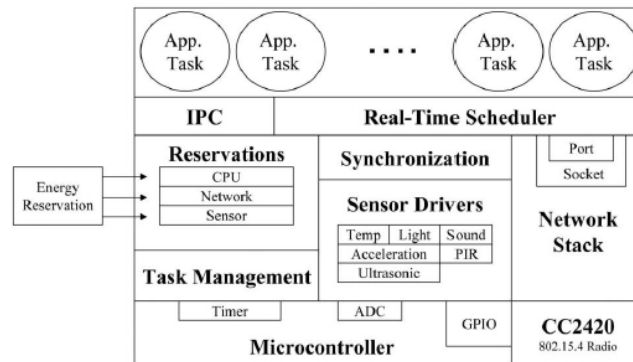


Figure 2.34: Nano-RK architecture [68]

Nano-RK is designed to facilitate application development by allowing developers to work in a familiar multitasking paradigm. This results in a short learning curve, rapid application development, and improved productivity.

This OS has a lightweight networking protocol stack that provides a communication abstraction similar to sockets. An application that wants to send data can create a socket and then it can start the communication via that socket. An application can bind and listen to a particular two bytes port number to receive data. A Time Synchronized Link Protocol for Energy Constraint Multi-hop Wireless Networks (RT-Link) has been implemented in Nano-RK, its goal is to prolong a sensors network's lifetime and to provide guarantees on end-to-end delay. Since Nano-RK is a reservation-based OS, it provides APIs to applications so that they can reserve network bandwidth according to the application requirements.

Nano-RK is developed in C language, it does not have any simulation support and its documentation can be found at [75].

LiteOS: it is a Unix-like OS for WSN designed at the University of Illinois. Its motivation was the design of a new OS Unix-like for WSN and its objectives are to provide system programmers with a familiar programming paradigm, a hierarchical file system, support for object-oriented programming in the form of LiteC++, and a Unix-like shell. Its footprint is 128 bytes of program flash and 4 bytes of RAM. It follows a modular architecture design. Its architecture is shown by Figure 2.35. LiteOS is primarily composed of three components: LiteShell, LiteFS and the Kernel.

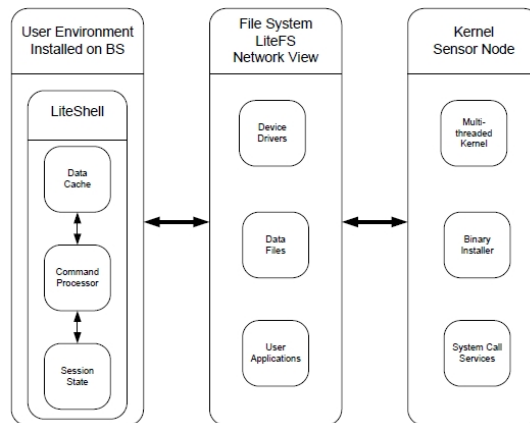


Figure 2.35 LiteOS architecture [68]

LiteShell is one of the three subsections of LiteOS and it resides on a base station or a PC, it supports more complex commands as the base station has higher resources. It can be used only when there is a user present on the base station. Some local processing is done on the user command and then transmitted wirelessly to the destination source node. LiteFs mounts all the neighboring sensor nodes as a file. The kernel that resides on the sensor node provides concurrency in the form of multithreading, provides support for dynamic loading, it uses RR scheduler, allows programmers to register event handlers through callback functions and provides synchronization support.

LiteOS provides communication support in the form of files. LiteOS creates a file corresponding to each device on the sensor node and another for the radio interface. Whenever there is data to send, it is placed in the radio file and is afterwards wirelessly transmitted. LiteOS supports simulation through AVRORA [74]; its documentation is available at [76].

Table 2.11 presents a summary of the features of the Wireless Sensor Networks Operative Systems previously described.

Table 2.11: Features summary of the WSN Operative Systems

OS/Feature	Communication Security	File System Support	Simulation Support	Programming Language	Shell
TinyOS	TinySec	Single level file system	TOSSIM	NesC	Not available
Contiki	ContikiSec	Coffee file system	Cooja	C	Unix-like shell runs on sensor mote
MANTIS	Not available	Not available	Through AVRORA	C	Unix-like shell runs on sensor mote
Nano-RK	Not available	Not available	Not available	C	Not available
LiteOS	Not available	LiteFS	Through AVRORA	LiteC++	Shell that runs on base station

3. METHODOLOGY SELECTION

The previous theoretical review showed the increasing importance of WSN for industrial applications and presented the reasons behind the huge success that wireless technologies have been experiencing during the last decades at the manufacturing automation market. Nowadays, driven by the constant evolution of different technologies, the industry relies on these devices for more critical applications. Even if WSN have started to be used for the control of processes instead of just monitoring applications, further effort is required to keep pushing their use in new tasks by maximizing their robustness and stability. In order to give support to the wireless technology growth, it is important to be aware of the need of wireless applications, as the WSN development is highly dependent on them.

There are many different wireless standards for industrial WSN; however, features such as the low data rate-optimized design, low costs and QoS requirements along with the low Micro Controller Unit (MCU) requirements and the support of large network orders, make IEEE 802.15.4 standard the most suitable for industrial WSN deployments. This protocol offers, among other features, selectable levels of security, encryption, sender authentication and message integrity. It is not surprising that this protocol has been the most used by the industry automation applications [36].

One of the most interesting and promising IEEE 802.15.4-based technologies is the IPv6 over Low Power Wireless Area Networks, 6LoWPAN. This technology is based on the deployment of IP stacks over low power wireless devices extending the IoT concept to different domains. 6LoWPAN has been highly important for the factory automation WSN applications. This standard has been adopted by ISA SP100 wireless industrial networks for their network layer [77]. Thanks to the IP, 6LoWPAN networks can be connected to Machine to Machine (M2M) services using simple network routers, as IP-based devices can be connected easily to other IP networks without the need for translation gateways or proxies. For this reason, it can also be considered an extension of M2M, an important driving force in the growth of the IoT at the manufacturing floor.

As the theoretical revision explained, the IEEE 802.15.4 standard, used by the lower layers of the 6LoWPAN technology, is designed for devices with low power consumption. The use of the IEEE 802.15.4 beacon-enabled mode, allows the use of GTSs scheduling methods for the bandwidth distribution in order to address the need of an efficient transmission management, an important function for low energy consumption as it is known that the packet transmission is the most energy consuming task of wireless devices

The state of the art of the current IEEE 802.15.4 scheduling techniques exposed a need for scheduling methods and algorithms designed specifically for 6LoWPAN and its industrial applications with low data rate; specific wireless traffic conditions and densities; and a link

layer that deals with fragmentation and reassembly processes. On the other hand, the evaluation of existing scheduling technologies shows that the WSN packet scheduling methods can be nourished with concepts from other fields. This chapter introduces the principles behind the High-Speed Downlink Packet Access (HSDPA) scheduling, also known as the 3.5G, an enhanced version of the current 3G technology used by the mobile telecommunication industry. HSDPA has a dynamic scheduling mechanism that adapts to the channel conditions and changes its scheduling decision according to the application requirements. This mechanism is adapted to the WSN packet scheduling described in this thesis work. The HSDPA description includes a brief introduction to the most common algorithms for the scheduling decision used by HSDPA.

Later during this chapter, the structure, functionalities and the main features of Contiki, the Operative System for WSN used for the implementation of the real-time packet scheduling system for a 6LoWPAN industrial application, are explained. This open source OS brings the possibility of dynamically loading individual programs with a very flexible architecture for constrained sensor devices. The kernel of Contiki is not protected from applications, meaning that any application can access global data, making it suitable for embedded systems with low memory management capabilities. In other words, the kernel and the application share the memory space for a more efficient use of it, a tight integration between the application and the kernel and a better management of system resources. Contiki has been supported on multiple platforms and deployed within a number of different industries for commercial and non-commercial usage. Its development team consists of a wide range of international companies with contributors from Atmel, Cisco, ETH, Redwire, LLC, SAP, Inico and many others.

The last section of this chapter focus on the FAST Wireless Sensor Network description for it serves as the implementation test bed that will be described during the following chapters. This description includes a brief introduction to the devices, their firmware and the FAST wireless sensor network as well as a small description of the wireless-based applications running at FAST.

3.1. Packet scheduling in HSDPA

High-Speed Downlink Packet Access (HSDPA) is an enhanced 3G (third generation) mobile telephony communications protocol in the High-Speed Packet Access (HSPA) family [78]. Also known as 3.5G or turbo 3G, HSDPA allows networks based on the Universal Mobile Telecommunications System (UMTS) to have a higher data transfer speeds and capacity. By adopting advanced techniques, such as adaptive modulation and coding (AMC), Hybrid Automatic Repeat request (Hybrid ARQ, HARQ) and fast packet scheduling, HSDPA can accommodate both real-time and non-real-time services with QoS guarantee. For a complete overview of the HSDPA technology refer to [43], [79].

When serving packet services, resource sharing among users is more efficient than dedication to a certain user [43]. Scheduling is an important function in determining when, at what resource and at what rate the packets will be transmitted from a user. The scheduling

problem is an implementation specific issue that is flexible and it is based on the system requirements. Certain methods can improve the network cell throughput. Others scheduling methods are designed to enhance the user fairness, traffic priority or the QoS.

In HSDPA, different scheduling priority indications or mechanisms can be used per user and per service. This is one of the main features of HSDPA as the scheduling mechanism and its scheduling decision can be managed in order to dynamically adapt to the application, density of users, traffic or channel conditions. For example, the Voice-over-IP (VoIP) service can have a higher priority than background-type data services. The packet buffer is controlled by a discard timer that indicates the maximum time packets that should be kept in the buffer and, once the timer expires, it determines the number of packets to be discarded.

The HSDPA MAC layer is in charge of the scheduling in the Base Transmission Station (BTS) and the handling of the HARQ process between the terminal and the BTS. However, HSDPA specifications do not contain details for scheduler implementations or deployment. Moreover, the specifications do contain parameters to control scheduler behaviour but its behaviour will depend on its application parameters and requirements.

As there are several possible solutions to such scheduling problem, HSDPA scheduling uses techniques that support multiple classes of flows per user session. With the purpose of enhancing the cell throughput, HSDPA scheduling algorithm can take advantage of instantaneous channel variations and raise the priority of a certain user. The scheduling must balance the conflicting goals of maximizing throughput, while at the same time ensuring some degree of fairness to all users requesting channel access.

Opportunistic scheduling methods or channel-aware schedulers are those that use the multi-user diversity in a way that the total throughput in a large network increases by giving priority to users with favourable channel conditions over those with poor channel conditions. Blind scheduling does not take into consideration the channel conditions for the scheduling decision. According to Gutierrez [80], the HSDPA scheduling algorithms can be classified as slow and fast methods.

3.1.1. Fast scheduling methods

Fast scheduling methods base the scheduling decision on the recent user channel quality so that any variation in the user's supportable data rate can be tracked. Fast scheduling methods execute the scheduling decision on a TTI basis and they are able to exploit the multi-user selection diversity to provide high capacity gain when there are a sufficient number of users connected.

Maximum C/I scheduler: The Maximum Carrier-to-Interference Ratio scheduler (Max C/I) is also known as the maximum throughput scheduler, it is designed to maximize the total throughput by always scheduling the user with the best instantaneous channel quality. Since Max C/I restrains the network resources so that only the users with the most favourable channel conditions are scheduled, there may be a number of users that may never be scheduled. On the other hand, the fast fading dynamics have a larger range than the average radio propagation conditions, for this reason, it is possible that users with poor average radio conditions

to still have channel access. This scheduler's main disadvantage draws in the fact that it is unfair in scheduling resources and suffers from coverage limitations.

Proportional fair scheduler: The Proportional Fair (PF) scheduler serves users on the top of their fades. It bases its scheduling decision on the following equation:

$$P_i = \frac{r_i(t)}{R_i} \quad i = 1, \dots, K \quad 3.1$$

where P_i denotes the user priority, $r_i(t)$ is the instantaneous data rate that can be supported by the user in the current TTI if served by the packet scheduler, and R_i is the average throughput experienced by the user i . The denominator of the equation 3.1 allows a user that is being allocated little scheduling resources to increase its priority over time. R_i is calculated before every TTI using the recursion:

$$R_i(n) = (1-\beta) R_i(n-1) + \beta r_i(n-1) \quad 3.2$$

where $r_i(n-1)$ is the requested rate of user i in the previous TTI ($n-1$) if the user i was selected for transmission in TTI ($n-1$), and 0 otherwise. B is the time constant of the exponential smoothing filter, β^{-1} equals the equivalent averaging period in TTIs. It is recommended that the averaging length is set long enough to ensure that the process averages out the fast fading variations, but not short enough to still reflect medium-term conditions such as shadow fading [81].

The PF scheduler provides a trade-off between fairness and achievable cell throughput, thus providing significant coverage extension over the Max C/I scheduler. It is designed to serve users under very favourable instantaneous channel conditions relative to their average ones.

Fast Fair Throughput Scheduler: Fast fair throughput scheduling is also known as Proportional Fair Throughput (PF-T) in [82]. It is designed to distribute the channel access fairly among the users while taking advantage of the short-term fading variations of the radio channel. PF-T algorithm is based on a modified version of the PF algorithm (Equation 3.1) and it is the following:

$$P_i = \alpha \frac{r_i(t)}{R_i} \quad i = 1, \dots, K \quad 3.3$$

where α is the term that weights the PF algorithm to equalize the user throughputs and is given by:

$$\alpha = \frac{\max_j \{\overline{R}_j\}}{R_j} \quad 3.4$$

where $\max_j \{\overline{R}_j\}$ is a constant that indicates the maximum average supportable data rate from all users, and \overline{R}_j is the average supportable data rate of user i .

Score Based Scheduler: this scheduler gives a score to the current channel condition of a user with respect to the past channel condition over a window of size T . Score based scheduling can distribute the channel access equally even under asymmetric networks.

3.1.2. Slow scheduling methods

Slow scheduling methods are less aggressive than the fast methods. Their scheduling decision can be based on the average user's signal quality. There are other slow schedulers that schedule the channel access without any consideration for user channel quality.

Round Robin Scheduler: it is a very simple scheduling mechanism that assures a fair resource distribution among the users in the network. This is considered to be a blind scheduling technique as serves users in a cyclic manner without taking their radio channel conditions into consideration. Round Robin scheduler employs a "fair time" method where the same power is allocated to all users such that a higher throughput is experienced by users with better channel conditions.

Average C/I Scheduler: this scheduler selects, in every TTI, the user with the largest average C/I that has buffered data ready for transmission. An averaging length is used so that only the user with maximum slow-averaged channel quality/throughput is scheduled.

Fair Throughput Scheduler: it is designed to ensure that all simultaneously queued users receive the same average throughput, meaning that users in bad channel conditions receive relatively more allocation of HS-DSCH resources. In every TTI, the user with the lowest average throughput is scheduled. This scheduler does not employ any instantaneous channel quality information.

3.2. Contiki OS

The real-time scheduling system implementation developed for this thesis work was programmed over the Inico devices firmware which at the same time is based on Contiki OS. The Inico wireless devices will be described during the Section 3.3.

Contiki, introduced at Section 2.6, is an operative system that connects tiny, low-cost, battery-operated and low-power systems to the Internet, enabling the Internet of Things, a collection of technologies that network things using sensors and actuators. This OS is open source software which full code can be freely used for commercial and non-commercial systems. Contiki provides a low-power internet communication that supports fully standard IPv6 and IPv4, along with the low-power wireless standards: 6LoWPAN, RPL and CoAP. The development of applications with Contiki is easy and fast as its applications are written in standard C language, it has simulation support using Cooja simulator and Instant Contiki provides an entire development environment. Contiki runs on a range of low-power devices which can be purchased online [72].

A running Contiki system consists of kernel, libraries, program loader and processes. A process can be either an application program or a service that implements functionality used by multiple application process. The communication between processes is always held through the kernel. A Contiki system is partitioned into two parts: the core and the loaded programs, this is shown by Figure 3.1.

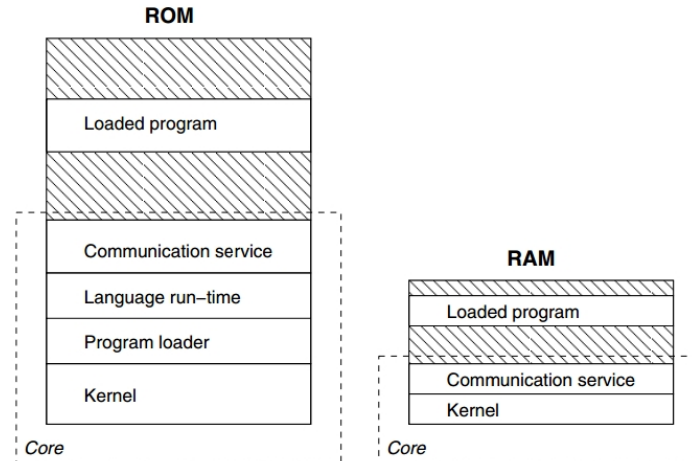


Figure 3.1: Contiki's partitioning into core and loaded programs [71]

The core is compiled into a single binary image that is stored in the devices prior to deployment and it is not modified after deployment but it is possible to overwrite or patch it. It consists of the Contiki kernel, the program loader, the language run-time and the communication services with the device drivers for the communication hardware. The program loader is in charge of loading the programs into the system. It may get the program binaries by using the communication stack. Typically, programs are first stored in EEPROM before they are programmed into the code memory [71].

Contiki has an event-driven kernel that consists of a lightweight event scheduler that dispatches events to running processes. The program execution is triggered by events or through a polling mechanism. The kernel supports asynchronous and synchronous events. Asynchronous events are enqueued by the kernel and are not immediately dispatched to the target process. On the other hand, Synchronous events immediately cause the target process to be scheduled. In addition to the events, Contiki's kernel has a polling mechanism that schedules high priority events in-between each asynchronous event. Events can only be preempted by hardware or real time executive interrupts.

Loadable programs are implemented using a run-time relocation function and a binary format that contains relocation information. When a program is loaded into the system, the loader first tries to allocate sufficient memory space based on information provided by the binary, if it fails, the program loading is aborted. Once the program is loaded, the initialization function is called.

Contiki's kernel does not contain an explicit power save abstraction; instead, it lets the application part of the system implement such mechanism by exposing the size of the event queue. This information can be used to power down the processor when there are no events scheduled.

In Contiki, a service is a process that implements a functionality that can be used by other processes; it can be seen as a shared library that can be dynamically replaced at run-time. Services are managed by a service layer that keeps track of running services. Examples of services can be communication protocol stacks, sensor device drivers or sensor data handling algorithms. Implementing the communication as a service provides for multiple communication stacks to be loaded simultaneously. The communication stack is shown in Figure 3.2. Communication services use the service mechanism to call each other and synchronous events to communicate with application programs.

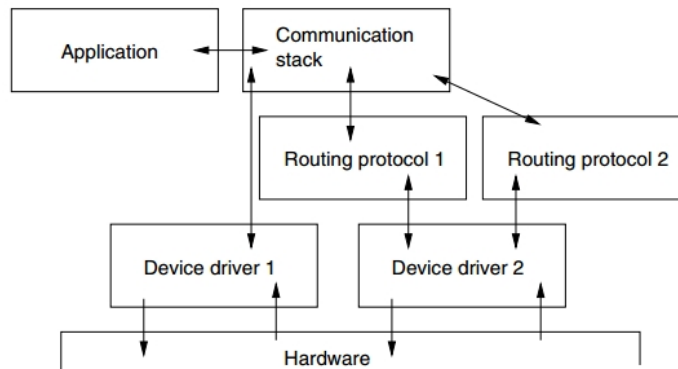


Figure 3.2: Contiki's communication stack [71]

The over-the-air programming is possible through an implemented protocol that transmits a single program binary to selected concentrator nodes using point-to-point communication. The binary is stored in EEPROM and when the entire program has been received, it is broadcasted to neighbor nodes. A comparison between the time for reprogramming a 40 sensors network using manual and over-the-air protocol can be seen in [71]. Using over-the-air programming reduces considerably the amount needed for a reprogramming process.

Contiki is an operation system for constrained devices and it is compact in terms of both code size and RAM usage in order to leave room for applications running on top of the system [71]. Table 3.1 shows the compiled code size and the RAM usage of the Contiki system compiled for the Texas Instruments MSP430 and the Atmel AVR.

Table 3.1: Size of compiled Contiki code, in bytes [71]

Module	Code size (AVR)	Code size (MSP430)	RAM usage
Kernel	1044	810	10 + + 4e + 2p
Service layer	128	110	0
Program loader	-	658	8
Multi-threading	678	582	8 + s
Timer library	90	60	0
Replicator stub	182	98	4
Replicator	1752	1558	200
Total	3874	3876	230 + 4e + + 2p + s

The kernel only gives the basic CPU multiplexing and event handling features. The rest of the system is implemented as system libraries placed in the Contiki Core that can be linked

with loadable programs. These libraries feature a high number of different functionalities. Some of the most important libraries are:

contiki/core/net/: libraries containing the functionalities for a full IP networking. Contiki provides a full IP stack with standard protocols such as UDP, TCP and HTTP as well as the low-power standards 6LoWPAN, RPL and CoAP. The Contiki IPv6 stack, certified under the IPv6 Ready Logo Program, is developed by Cisco.

contiki/apps/erbium/ & *contiki/core/net/rpl/*: libraries containing the functionalities for the recently standardized by the IETF protocols for low-power IPv6 networking: 6LoWPAN, RPL and CoAP standards.

contiki/core/sys/pt.h: Contiki uses a mechanism to save memory with a good control flow in the code called protothreads, a mixture of the event-driven and the multi-threading programming mechanism. The protothreads contained in this library are used by event-handlers to block or wait events to occur.

contiki/core/net/rime/: Contiki provides a wireless networking stack called Rime to be used in situations where the bandwidth is at a premium or where the full IPv6 networking is overkill. Rime supports simple operations such as sending a message to all neighbours and to a specific neighbour or more complex network mechanisms such as network flooding and address-free multi-hop semi-reliable scalable data collection.

contiki/core/net/mac/: in wireless network, nodes may need to relay messages from others to reach their destination. With Contiki, routers can be battery-operated thanks to the ContikiMAC radio duty cycling mechanism which allows them to sleep between relayed messages.

Some other important libraries' functionalities of this OS include: power awareness tools that can be found at *contiki/sys/energest*, used for estimating the system power consumption; simulation support using Cooja network simulator, found at *contiki/tools/cooja*; and a lightweight flash file system called Coffee. With Coffee, programs can open, close, read from and write to on the external flash. Coffee can be found at *contiki/core/cfs/cfs-coffee.[ch]*. For more information related to Contiki's libraries, refer to [72].

Along with all these functionalities, Contiki provides support through both, a community support and commercial support.

3.3. FAST Wireless Sensors Network

The wireless sensor network installed at FAST is a 6LoWPAN-based network, presented in Figure 3.3. It relies on an island of an IPv6 stub low power wireless area network (LoWPAN). A stub network is a network which is capable of receiving or sending IP packets but does not act as transit to other networks. Each LoWPAN network consists of router nodes and host nodes. These nodes differ to each other only in way whether a proximal node takes a responsibility of message redirection between distant nodes and edge router, which one would be otherwise for the distal node out of reach. The edge router is a device that routes traffic in and out of the LoWPAN, while handling 6LoWPAN compression and ND for the LoWPAN.

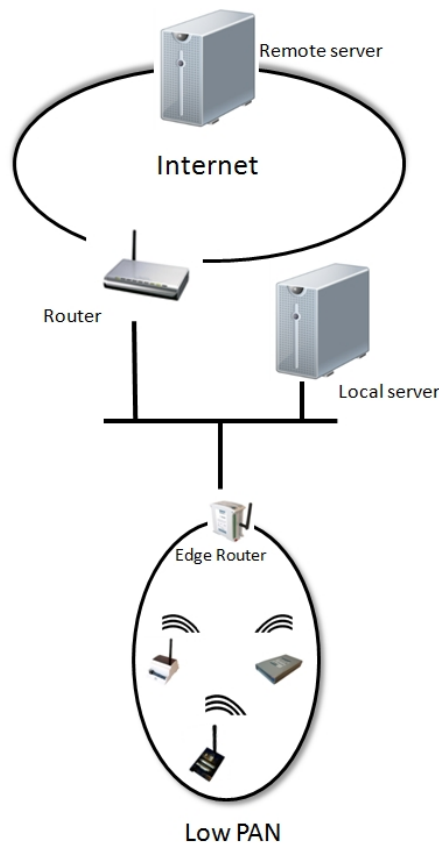


Figure 3.3: 6LoWPAN architecture installed at FAST laboratory

The 6LoWPAN-enabled WSN consists on Reduced Function Devices nodes connected to an IPv6 network for wireless embedded applications through Full Function Devices in order to enable the use of web-services end-to-end over low power wireless devices employing a passive gateway solution, using Atmel hardware and Contiki operating system ported to Atmel Radio Frequency platform as the core of the embedded firmware.





The wireless platform consists of three different battery powered RFD wireless sensor nodes:

- Temperature, humidity, light sensors
- Accelerometer/Gyroscope
- Digital & Analogue Inputs

The sensor nodes are directly reachable using their unique IPv6 address allowing easy access to their web-based configuration interface. Each sensor node consists of a list of subscribers, which will receive further the sensor data. As the 6LoWPAN coordinator, the WSN uses the Inico Technologies S1000 industrial controller with a wireless networking expansion specifically designed for 6LoWPAN applications.

Table 3.2 shows a more detailed description of the sensor nodes as well as the FAST wireless-based application (explained with detail later at this section) used for each device.

Table 3.2: FAST WSN Sensor nodes

Number	Device	Description	Function	Image
1	Edge Router	6LoWPAN coordinator	WPAN coordinator, routes traffic from and out of the LoWPAN	
2	Temperature, humidity and light sensor	Battery powered, wireless sensor node with temperature, humidity and light sensors	Temperature/Humidity/Light sensing. 6LoWPAN-enabled	
3	Digital & analogue I/O	Battery powered wireless sensor node with 4 digital outputs and 4 analogue inputs	Digital outputs & analogue inputs wireless node. 6LoWPAN-enabled	
4	Accelerometer/Gyroscope	Battery powered wireless sensor node with 4 axis accelerometer/gyroscope	Accelerometer & gyroscope wireless node. 6LoWPAN-enabled	

FAST WSN is deployed at the FASTory production line. This line is the testing scenario for most of the research work done at the FAST laboratory and it is fully described at the Section 3.2.2. The location of the sensor nodes at the FASTory line is shown by Figure 3.4. The figure consists of the production cells, where the red dots represent production pallets and the distribution of the sensors among the production floor. The numbers next to the devices are referred to Table 3.2.

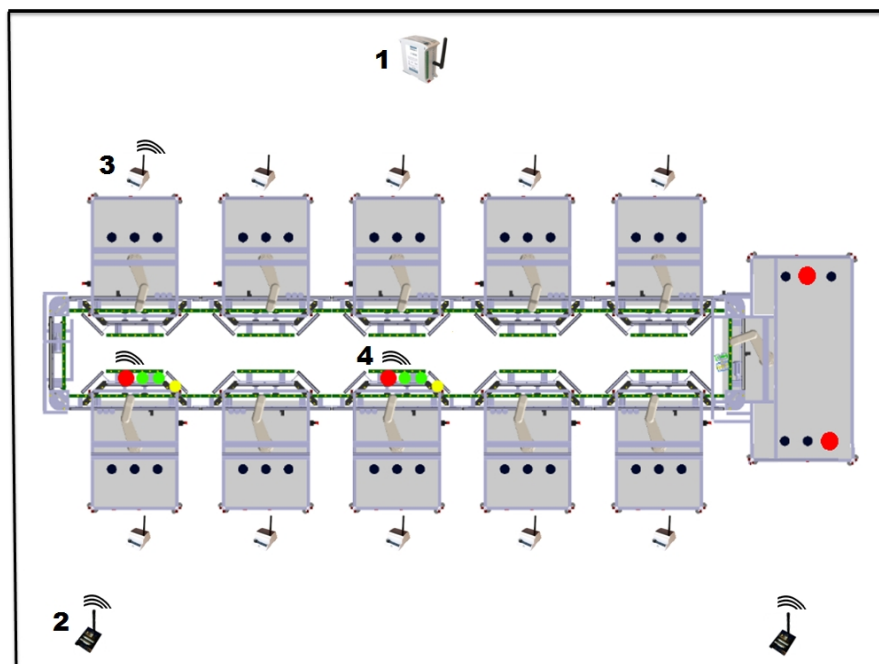


Figure 3.4: FAST WSN layout at the FASTory production line

FAST WSN is fully IPv6 enabled, meaning that interoperability and easy integration with the current version of the Internet network is assured. The Wireless sensor network installed at FAST has connectivity and communication with other networked devices and it is possible to access both, sensors (through their IPv6 addresses) and their interfaces (all the applications are accessible through internet) using computers, smart phones or tablets.

3.3.1. Wireless Devices

The devices of the FAST WSN are designed by Inico Technologies Ltd. Most of FAST laboratory solutions are based on Inico products but are modified in order to respect the open nature of research.

Inico Technologies Ltd. was founded in 2007 with the objective of offering innovative solutions to the industrial sector based on the latest information and communication technologies. Based on the findings of several R&D projects across Europe and North America, Inico introduced the S1000, a smart Remote Terminal Unit for monitoring and I/O control, and capable of integrating to a Service-Oriented Architecture (SOA). Based in Calgary, Alberta, Canada, Inico Technologies Ltd. has its origins at FAST as the current CEO of the company is the first PhD. student graduated from the laboratory. [83]

The FAST WSN devices, presented in Table 3.1, are using the AtmelAtMega1281 microcontroller as the central processing unit and the AT86RF230 analog radio with a built-in digital de-modulation transceiver, specially designed for the low cost IEEE 802.15.4 applications and used as the SPI-to-antenna radio transceiver interface between the antenna and microcontroller of the sensors.

The firmware of the devices is called eSONIA 6LoWPAN firmware and it is based on Contiki that as it was explained during previous sections it is an open source operating system, meaning that the source is always available, solving problems that current proprietary-based solutions have. It is designed to operate in extremely low-power systems and provides mechanisms to estimate the system power consumption. A description of the FAST WSN devices is presented:

Inico S1000 RTU: The Inico S1000 is a Smart Remote Terminal Unit capable of real-time control, field data processing, Web-based monitoring, and integration to SCADA/HMI or Enterprise IT systems. The device is presented in Figure 3.5.

The Central Processing Unit (CPU) of the S1000 is an Atmel AT91SAM7X512 32-bit processor. It operates at a maximum speed of 55MHz and features 512 KB and 128KB of SRAM. The peripheral set includes a USB 2.0 full speed device port, Ethernet MAC 10/100 port, CAN 2.0A and 2.0B compliant controller, two USART, UART, two SPI, SSC, TWI (I2C), three 16-bit timers, RTT and 8x10-bit ADC [84].

The Inico S1000 interfaces to virtually any industrial equipment and legacy PLC/RTU processing field data in real time. The S1000 is a programmable RTU device that offers process control capabilities as well as a Web-based Human-Machine Interface (HMI) and support for Web Services. It is designed to operate in typical industrial settings and it is compatible with most industrial signal types and levels. In addition to built-in I/O ports, the S1000 sup-

ports expansion modules, which offer additional I/O points as well as specialty functions such as wireless networking and energy monitoring.



Figure 3.5: Inico S1000 RTU and its wireless expansion

The wireless expansion module, shown in Figure 3.6, is the WPAN coordinator. It is responsible for all 6LoWPAN coordination duties such as network initialization, communication, maintenance or security and it is connected to the S1000 controller. On the other hand, the S1000 itself serves only as a gateway to IPv6 network without having any processing responsibilities. The combination of the S1000 and its expansion works as the edge router of the network and it is connected to the IPv6 backbone local area network and to Internet.

The wireless expansion of the S1000 uses a Synapse RF200PD1 Radio Frequency Engine for all the wireless tasks of the device. The RF Engine is an IEEE 802.15.4-based, low power, reliable solution to embedded wireless control and monitoring network needs. The RF200 has an embedded ATmega128RFA1 MCU with an integrated transceiver for the wireless duties [85].



Figure 3.6: 6LoWPAN expansion module for the S1000 RTU

The S1000 firmware is the eSONIA 6LoWPAN firmware. It includes IPv6 enabled DPWS porting, based on a modified version of the open-source WS4D JMEDS stack that improves the devices handling with dual IPv4 and IPv6 DPWS stacks by giving preference to IPv6 endpoints and supporting the discovery of web services exposed by the S1000 via IPv6. The S1000 includes a Low Weight IP (LwIP) implementation. LwIP is a widely used open source TCP/IP stack, developed by the creator of Contiki, It is used for embedded systems and its purpose is to reduce the resource usage while still having a full scale TCP [72].

The S1000 configuration interface, shown in Figure 3.7, is accessible through a web browser using either its IPv6 or IPv4 address. It is used to adjust different parameters of the device, to create programs, monitor variables of its I/O as well as adjust parameters of the expansion connected to the device. The S1000 is equipped with an HMI engine, which allows the development of customized web pages to display real-time process data. This engine is also accessible through its configuration interface. For more information related to the programming of the device, refer to its user's manual [86].

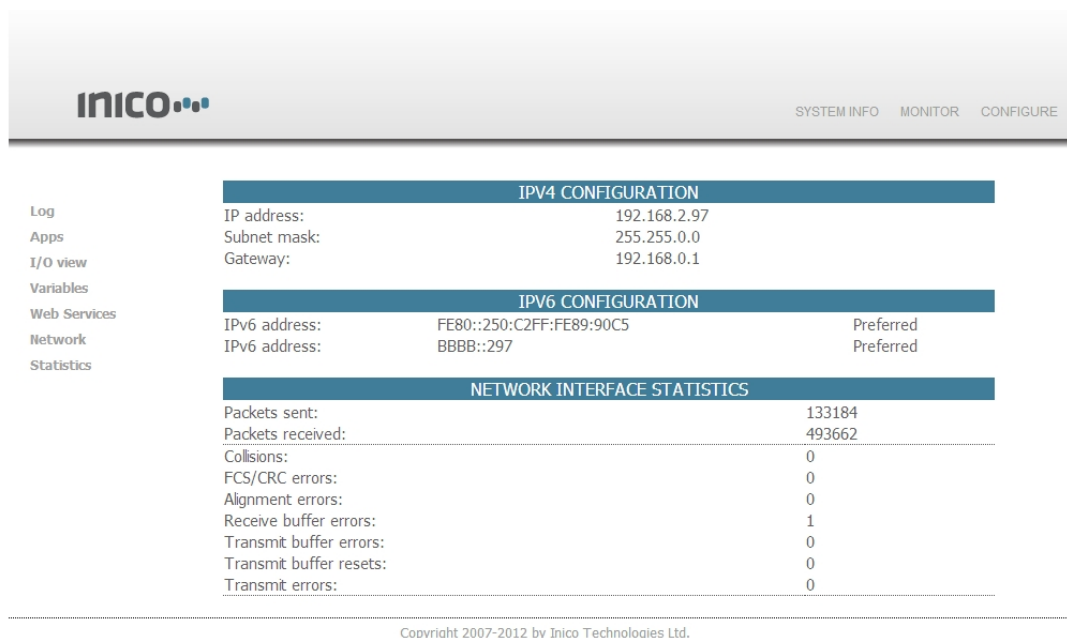


Figure 3.7: S1000 configuration interface

Accelerometer/Gyroscope: This Inico device, shown in Figure 3.8, is a 6LoWPAN-enabled 3-axis accelerometer and 3-axis gyroscope that provides complete movement on the space information of the device. The CPU of the sensor node is an Atmel AT1281 MCU with dual antenna and IEEE 802.15.4 OEM module capable of broadcasting in 2.4 GHz frequency band. It is powered by two AA batteries. The firmware of the device is based in Contiki OS. Its configuration interface is accessible through a web browser using its IPv6 address and it is similar to S1000 RTU, showed in Figure 3.7.



Figure 3.8: Wireless Accelerometer/Gyroscope

Temperature/humidity/light (THL) sensor: This device, shown in Figure 3.9, is a 6LoWPAN-enabled node equipped with a temperature, humidity and light sensors. Like the accelerometer, the CPU of the sensor node is an Atmel AT1281 MCU with dual antenna and IEEE 802.15.4 OEM module capable of broadcasting in 2.4 GHz frequency band. It is powered by two AA batteries. The firmware of the device is the eSONIA 6LoWPAN firmware, based in Contiki OS. Its configuration interface is accessible through its IPv6 address using a web browser.



Figure 3.9: THL sensor node

Analogue & digital inputs sensor: The analogue and digital inputs wireless sensor nodes is a 6LoWPAN-enabled node equipped with 4 analogue inputs and 4 digital outputs for control and monitoring. Like the other sensor nodes, the CPU of the sensor node is an Atmel AT1281 MCU with dual antenna and IEEE 802.15.4 OEM module capable of broadcasting in 2.4 GHz frequency band. It is powered by two AA batteries. The firmware of the device is the eSONIA 6LoWPAN firmware, based in Contiki OS. Its configuration interface is accessible through its IPv6 address using a web browser. Its physical case is similar to the THL sensor node, shown in Figure 3.9.

3.3.2. eSONIA 6LoWPAN firmware

The FAST WSN devices firmware is called eSONIA 6LoWPAN firmware. It is a modified version of the Inico technologies proprietary firmware with a number of different features added by the FAST laboratory researchers in cooperation with Fluid House OY [87] and Inico ltd. as part of the eSONIA European Research Project.

The eSONIA 6LoWPAN firmware represents one of the many TUT's contributions to the research project. It belongs to the Tampere University of Technology and to the companies that were involved with its development and debugging.

3.3.3. FASTory line

The FASTory line, showed in Figure 3.10, is the production line that serves as the testing scenario for most of the research held at the Factory Automation Systems and Technologies laboratory. This production line, currently installed at the laboratory facilities, belonged to a Finnish electronics company and it was used for product assembling. After going through an overhauling process it was adapted for research purposes and its production process was changed: now the line draws cell phones. The decision behind using a drawing process is simple: this process does not need specific raw materials and the finished pieces do not require being stored making it ideal as a testing environment for the research work developed at FAST.



Figure 3.10: FASTory production line

The production line consists of twelve work stations connected by a conveyor system. Ten of the work stations are used for drawing operations and there are two others responsible for the raw material loading (a sheet of paper) and final product (drew cell phone) unloading respectively. There is a pallet buffer used for storing and classifying production pallets.

All the drawing production stations are equipped with a SONY SRX-611 4-axis SCARA robot, a pen feeder and safety systems. The buffer has a pneumatic manipulator for pallet handling and storing. The conveyor system has RFID/NFC and safety systems installed. Each workstation has a main conveyor and a bypass conveyor shown in Figure 3.11. As soon as the pallet reaches the entry point of a particular work station, the control system must decide whether the pallet will be processed or bypassed to another work station.

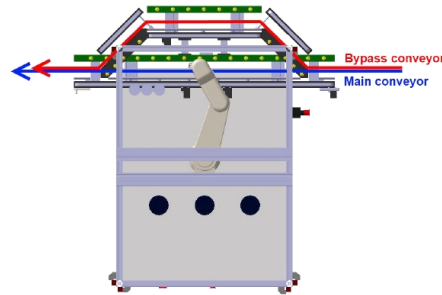


Figure 3.11: Workstation with main and bypass conveyor [20]

Once the pallet goes through all the required operations and the drawing is complete, it moves to the unloading work station where the robot takes the finished drawing and it places a new paper sheet on the pallet.

The cell phone drawing represents a final product and it must include a keyboard, a screen and a frame drawn on a sheet of paper, each of them drew by a different work station, the final drawing is shown in Figure 3.12. There are three different versions of keyboard, screen and frame and there are three different colours for each element, this means that there are over seven hundred possible combinations of cell phone drawings.

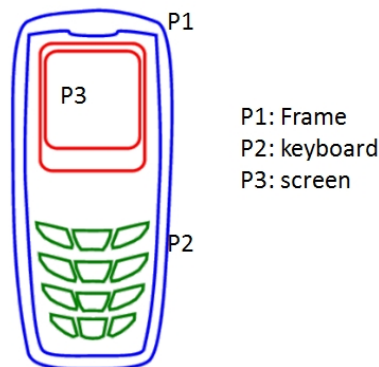


Figure 3.12: Cell phone draw by FASTory line [20]

The line is controlled by an Inico S1000 RTU network, installed at each work station and conveyor systems. FASTory line is a fully IPv6-enabled production line where both wired and wireless devices are capable of handling IPv6 as their communication protocol by using web services.

3.3.4. Wireless-based applications at FAST

The wireless based applications running at the FASTory line are grounded on the integration of smart objects, based on IP technologies, specifically 6LoWPAN and IPv6 protocols as the foundation of the communication function.

Production carriers' real-time positioning system [20]: the real-time positioning system aims to develop expedient tools for real time positioning and monitoring of production assets in form of manufacturing carriers transporting products by implementation of low-cost and low-powered wireless embedded devices. Wide interoperability of the final solution is ensured by using 6LoWPAN as a wireless communication technology that supports direct net-

work connection by IPv6 without a need for any gateways and versatile web services as a way of data transportation.

In order to detect pallet position while it moves throughout the conveyor system, certain sensors must be installed to provide desired information. The pallet positioning system estimates the position helped with the information of two sensors:

- 6LoWPAN-enabled Wireless Accelerometer/Gyroscope (described in Section 3.2.1): located inside the pallet, it sends information about the events that the pallet experiences such as the turns, the stop and moving events using 6LoWPAN standard.
- RFID readers: located through the conveyor of the production line at strategic points, this sensor reads an RFID tag located at each pallet to give the exact position of it and a pallet recognition mechanism. It is also used to know if the pallet has been introduced to or taken from the production line as well as to know when it enters or exit a working cell of the FASTory line.

The pallets used at FASTory are especially built [88] for transportation of a sheet of paper of size A6 along the conveyor system of production line described in the Section 3.2.2. The conveyor system uses for pallet movement a complex system of running belts which carry away the pallet. The pallet shown in Figure 3.13 consists of three parts. In order to place the Wireless accelerometer/gyroscope and the RFID tags, the pallets were re-designed.

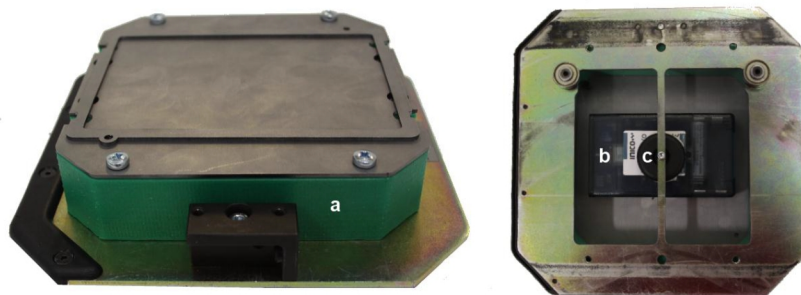


Figure 3.13: FASTory production line pallet: a) Pallet; b) Wireless accel/gyro; c) RFID tag

Each pallet is equipped with a passive NFC tag that has a unique identification number for the pallet. This tag is placed at the bottom part of the pallet so that the NFC readers, shown in Figure 3.14, can identify the tag. The NFC reader is connected by a serial link to the Inico S1000 RTU that transmits the data from the reader to the subscriber using web services.



Figure 3.14: NFC reader connected to the Inico S1000 RTU

The tool developed for this application is called APIS. It stands for Advanced Pallet Information System. The APIS tool gathers the information from the events of the pallet that are transmitted wirelessly by the accelerometers/gyroscopes using 6LoWPAN and the information from the NFC readers in order to estimate the exact position of a production pallet within the production line. APIS uses artificial intelligence to locate the pallet by using a multi agent system where each pallet is represented as an agent, Monte Carlo localization algorithm and a Simple Decision-Making algorithm. The Graphical User Interface (GUI) of APIS is presented in Figure 3.15. The APIS interface can show, among other production-related data, the current position of a pallet, historical position and the events transmitted in real-time by the sensors. For more information about this application developed at FAST, refer to [20].

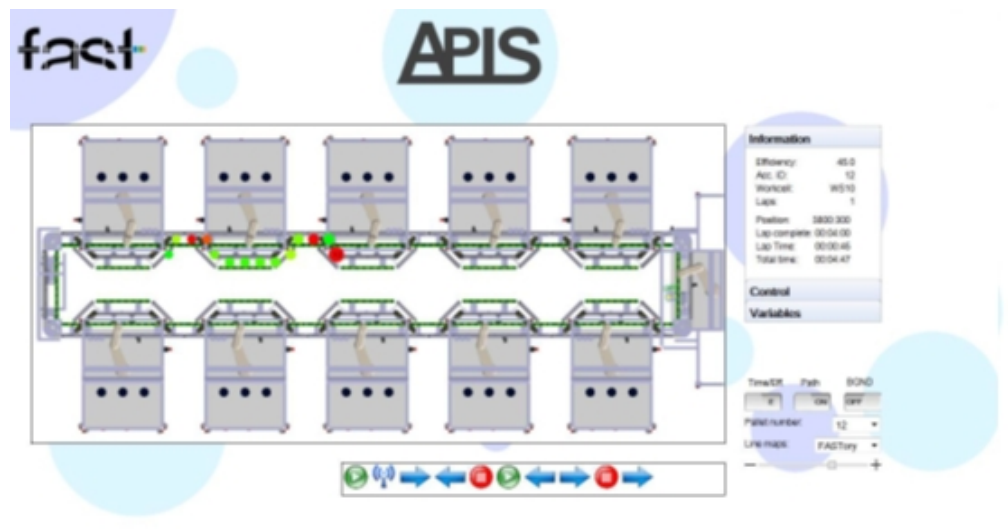


Figure 3.15: APIS Graphical User Interface

Production Re-scheduler: this is an application based on a non-disruptive monitoring system that intends to ensure a constant production regardless of any kind of work cell failure. The transmission of events, as it will be explained later, is done using the 6LoWPAN protocol.

The production re-scheduler is a functionality of FASTory line that allows the rescheduling of pieces production based on the availability of working cells. The availability of the cells is determined by sensors installed on them.

The functionality's principle is simple, as soon as any anomaly is detected and the working cell is not available, the production of the piece will be rescheduled and the task will be given to another cell with similar capabilities. The cell would not be available in case there is a disruption on the safety system or the emergency security button has been pressed

The safety system installed at the line consists of door switch sensors, shown in Figure 3.16, which detect if the maintenance door of the work cells is closed or opened; and retro-reflective sensors, shown in Figure 3.17, which detect a safety disruption inside the working cell. All these sensors are connected to a Inico wireless digital and analogue input sensor described at Section 3.2.1 that use 6LoWPAN as their communication protocol, they do not re-

quired any kind of configuration as they are “plug and play” sensors. There is a huge advantage on using these wireless devices as this represent a non-disruptive monitoring technique.



Figure 3.16: Door switch sensor connected to a wireless I/O sensor



Figure 3.17: Retroreflective sensor

In case of a disruption or if the safety system is activated, Figure 3.18 a, the production will be rescheduled as shown in Figure 3.18 b, In case the work cell is deactivated or not working, the piece will go through the bypass conveyor, shown in Figure 3.9, so that the flow of the production is not affected. In that case, there will be a re-assignment of tasks for the production line.

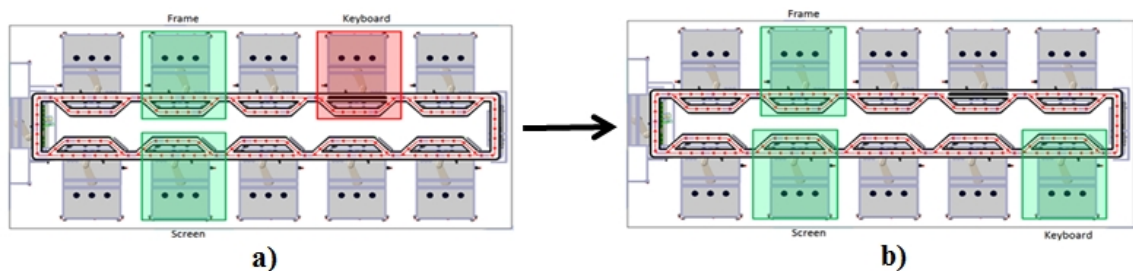


Figure 3.18: Production rescheduling diagram

The GUI of the application is presented in Figure 3.19. This GUI is used to monitor the safety status of the line. The station with a safety disruption will be shown and the sensor (shown at the right part of the interface) will be pulsing to show the security disruption location. For more information about this application, refer to [89].

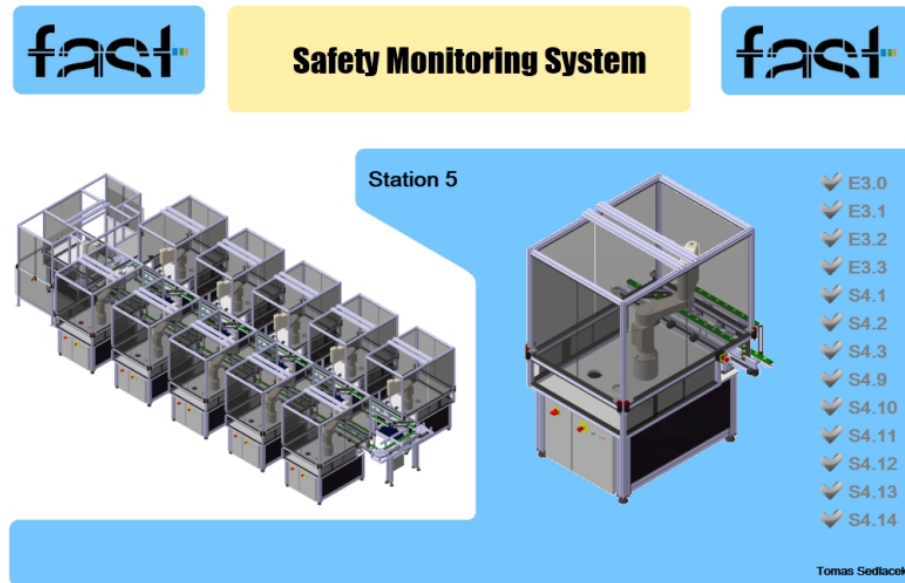


Figure 3.19: Safety monitoring system GUI

There are other applications currently under development that use the FASTory WSN. For more information about them refer to [90].

4. IMPLEMENTATION

6LoWPAN enables the IoT by connecting embedded devices to the Internet. For the manufacturing industry, the use of 6LoWPAN-based devices represents the possibility of providing to the manufacturing floor a wide number of functionalities for monitoring and controlling process using one of the greatest tools of our time: the Internet. The use of IP technologies guarantees the interoperability and scalability of devices; it provides unique capabilities such as established naming, addressing, translation, lookup, discovery and security schemas connected to the IP. Bringing the IP-based technologies to the manufacturing automation industry is bringing technologies that have existed for decades, that are well known and that have been widely tested and are open and free.

As stated in Section 1.2.1, this thesis work focuses on the implementation of a wireless packet scheduler embedded in wireless devices firmware for a more efficient and predictable bandwidth distribution with a lower packet loss transmission of an industrial 6LoWPAN-based Wireless Sensor Network, resulting in a more robust and reliable network with a higher number of nodes networked. The firmware implementation is deployed and tested in the WSN described in Section 3.3 and it based on Contiki as the operative system.

The current state-of-the art of the scheduled GTSs allocation part of the IEEE 802.15.4 beacon-enabled mode, reviewed at the Section 2.5, showed that is possible to bring concepts from other wireless technologies into the 802.15.4 standard, the most used wireless standard for industrial applications [36]. The wireless scheduler algorithm used by this implementation is based on the scheduling decision algorithms that the HSDPA technology employs for the mobile phone industry. In the HSDPA technology, the MAC layer is able to adapt and change its scheduler algorithm in order to satisfy the application based on the information of the channel condition and, in applications that require it, on the QoS requirements of the user equipment. This thesis work adapts this concept to the WSN for industrial applications domain, a context where even if there have been other scheduler implementations, as Section 2.5 shown, there is a lack of deployments that address the unique traffic conditions and network nodes orders requirements that the industrial applications present.

This chapter focuses on the description of the wireless scheduler implementation, starting with a brief introduction to the development environment required for the programming of the sensors' firmware. Later, the structure of the scheduler is presented along with its architecture, the algorithm implementation and its function modules. Finally, the experimental implementation, the test bed and the experimental set up are described and discussed.

4.1. Development environment

An Integrated Development Environment (IDE) is a software application that provides facilities to programmers for software development [91]; it consists of a code editor, build automation mechanisms and debugger tools. The wireless scheduler implementation described by this thesis work required the reprogramming of the Inico devices firmware that use the eSONIA 6LoWPAN firmware, based on the Contiki operative system. The Contiki OS is designed to run on many different platforms and it is possible to compile and build the system and the applications on many different development platforms; it is written and can be programmed using standard C programming language making it possible to use different IDEs such as Eclipse [92] or Instant Contiki [93].

Eclipse offers different useful features for browsing the Contiki C code, for example indexing all keywords, and a call hierarchy view. The hierarchy view shows for each function which other functions calls, and by whom it is called. Eclipse offers the possibility of debugging the code running on the hardware in order to monitor variable values using its debugging tool and breakpoints. The debugging tool has its own customizable interface view that displays information from the code and the hardware device. With Eclipse's plug-ins it is possible to load compiled programs to AVR MCUs using the AVRdude functionality or to access SVN and GIT repository systems using their plug-ins. All this features along with the possibility of running this IDE over Microsoft Windows®, reducing additional configuration options, made it the best choice for the development of the packet scheduling system.

The Eclipse development environment running over Microsoft Windows® used for the software programming requires a set-up and a preliminary configuration before it can be used for programming or manipulation of software. Instructions for this set-up are available at [94].

As it was previously explained, the packet scheduling system is programmed over the eSONIA 6LoWPAN firmware which requires Contiki for the program building. Both, Contiki OS and the firmware source codes are located in repositories but they use a different versions control system. For this reason, Eclipse requires repository plug-ins to access and import the source code to the IDE work place so that the code can be modified and reprogrammed. Contiki is available through a GIT repository system and it can be found at [95]. The eSONIA 6LoWPAN firmware is proprietary software stored at a SVN repository system. Figure 4.1 maps the source codes and their versions control systems.

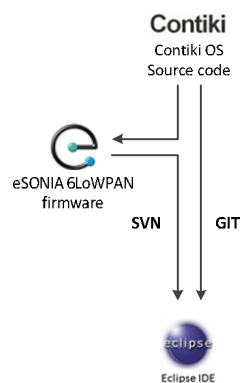


Figure 4.1: Inico development environment using Eclipse

The integration between the eSONIA firmware and the Contiki OS is direct as the Contiki build system is designed to make it easy to compile applications for either to a hardware platform or into a simulation platform by simply supplying different parameters to the make command, without having to edit *makefiles* or modify the application code [96]. Once both codes are imported to the Eclipse work place, the code can be manipulated.

Compiled programs can be loaded to the hardware using the Eclipse IDE thanks to the combination of the AVR Eclipse plug-in and the JTAGICEMKII debugging tool shown in Figure 4.2. For a more detailed description of the program loading and the Inico devices configuration refer to [94].



Figure 4.2: JTAGICEMKII debugging tool

4.2. Packet scheduler

The wireless packet scheduler proposed by this thesis work is an adaptation of the scheduling mechanisms employed by the HSDPA MAC layer, where the scheduling decision algorithm changes according to the application requirements, to a 6LoWPAN industrial network and its IEEE 802.15.4 MAC layer's beacon-enabled mode.

Knowing that due to the multiple wireless applications that can be running during different stages of production, the traffic of an industrial network can change from time to time, the scheduling mechanism developed selects the scheduling algorithm for the that best fits the application requirements in order to assure a balance between low power consumption and an efficient wireless network transmission with a minimum wireless packet loss. Figure 4.3 shows the packet scheduling mechanism schema.

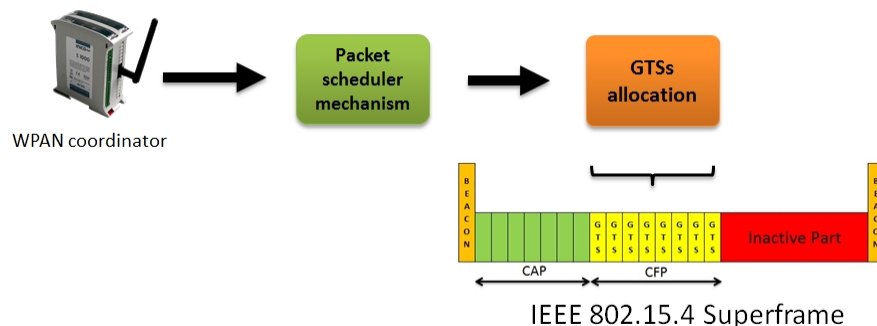


Figure 4.3: Packet scheduling mechanism

The packet scheduling mechanism embedded on the S1000 RTU firmware takes the decision of which scheduling algorithm to use based on the information coming from a traffic monitor and a throughput monitoring modules. These modules monitor the number of nodes

transmitting packets and the total throughput incoming to the WPAN coordinator respectively. Based on this data, the scheduling manager algorithm chooses from one of the three schedulers implemented.

As it will be later explained, each of the three schedulers is ideal for a traffic condition of the network. The scheduling algorithms give a score to the nodes transmitting so that the channel access is granted by the GTSs assignment module according to their position on the score table.

The score is given according to the decision algorithm formula of each of the schedulers:

- Score Based scheduler (SB): gives the higher score to the node with the higher historical data rate.
- Maximum Carrier/Interference scheduler (Max C/I): gives higher score to the node with higher instant data rate.
- Proportional Fair scheduler (PF): gives a score to the node based on the result of the division between the instant data rate and the historical data rate.

The GTSs assignment module is in charge of allocating the Guaranteed Time Slots of the IEEE 802.15.4 superframe to the nodes according to their score using as many superframes as required to give channel access to all the nodes transmitting.

4.2.1. Architecture

The scheduling mechanism architecture consists of three modules, shown in Figure 4.4 and explained below.

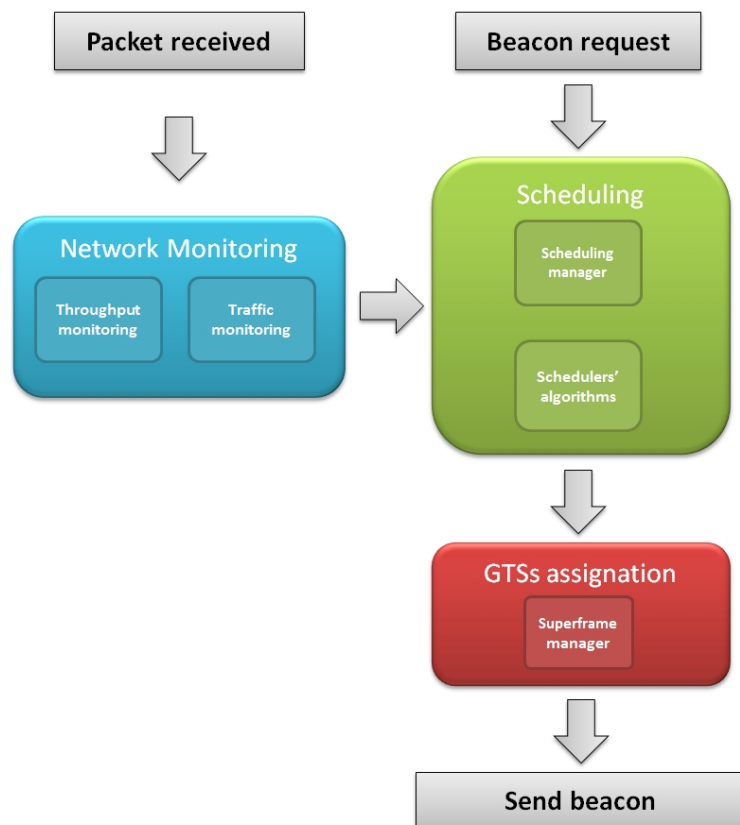


Figure 4.4: Packet scheduling system architecture

Network monitoring module: The network monitoring module encapsulates two functions:

- Throughput monitoring: consists of two main algorithms that calculate the network total throughput and the node throughput.
- Traffic monitoring: a series of algorithms that calculate the number of active nodes sending packets to the coordinator.

The data generated by the algorithms of this module is required for the scheduling process. Every time a packet is received by the coordinator, the network monitoring module will re-calculate values such as the number of active nodes and the throughput. The information is written on the nodes previously table created by the scheduling manager module.

Scheduling module: The scheduling module encapsulates two functions:

- Scheduling manager: a series of algorithms that choose the ideal scheduler to be used based on the information from the network monitoring module.
- Schedulers' algorithms: three scheduling decision algorithms: Proportional Fair, Maximum Carrier/Interference, Score based.

The scheduling module code starts executing every time there is a superframe beacon request. The scheduling manager reads the information generated by the Network monitoring module and chooses the scheduling algorithm to be used, later, the chosen scheduling algorithm creates a table of nodes and it sorts them according to the priority given by the scheduling equation.

Guaranteed Time Slots assignment module: This module's code is executed immediately after the scheduling module execution is completed. It encapsulates the IEEE 802.15.4 Superframe manager function:

- Superframe manager: in charge of allocating the GTSs of the superframe to the nodes transmitting packets. The algorithm of this function is designed to use as many superframes as necessary to guarantee channel access to all the nodes of the network. Another important task of this module is the beacon building, in charge of building the superframe structure based on the information generated by the network monitoring and scheduling module and it sends the information to the network embedded on the beacon. The beacon builder algorithm "constructs" the beacon; once the beacon information generated is complete it is sent to the function in charge of transmitting the beacon to the WPAN.

4.3. Algorithms implementation

The code implemented for the scheduling system was programmed over the eSONIA 6LoWPAN firmware using the development environment described in the Section 4.1 for the building and debugging of programs. The firmware is based on Contiki meaning that the code was written in standard C programming language and using different Contiki libraries.

As the Section 4.2 explained, the code is divided in three modules with specific functionalities. This document does not show the code deployed, instead, it presents and explains the functions and tasks of the algorithms programmed.

4.3.1. Network monitoring module

The network monitoring module encapsulates the Traffic and the Throughput monitoring functions. Figure 4.5 shows the module with its functions. Each function contains algorithms in charge of specific tasks. The algorithms are explained below.

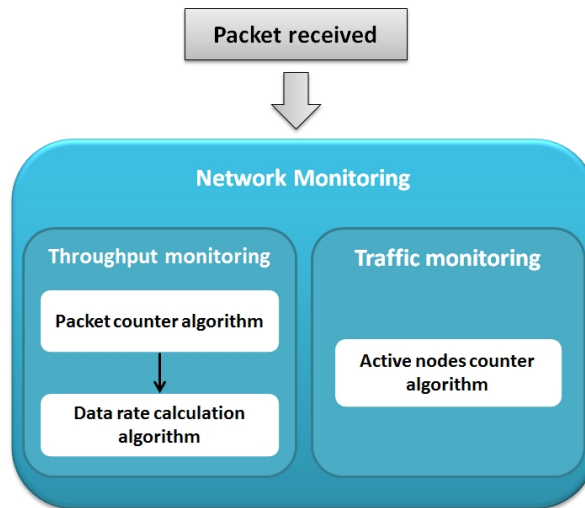


Figure 4.5: Network monitoring module

Throughput monitoring:

- Packet counter algorithm: this algorithm counts the number of packets received in a similar way as the Active nodes counter algorithm. After a reset timer is completed, the value will be set to 0. Its execution is triggered by the reception of a wireless packet from the nodes to the coordinator. Figure 4.6 shows the flow chart of this algorithm.

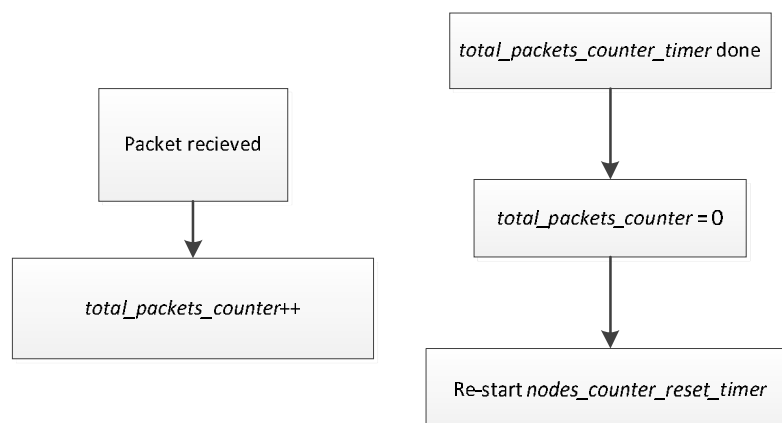


Figure 4.6: Packets counter algorithm flow chart

- Data rate calculation algorithm: calculates the data rate of each node in packets per second. It creates a table of nodes with its instant and historical data rate. The table of nodes is later sorted by the Scheduling module. Figure 4.7 shows the flow chart of this algorithm.

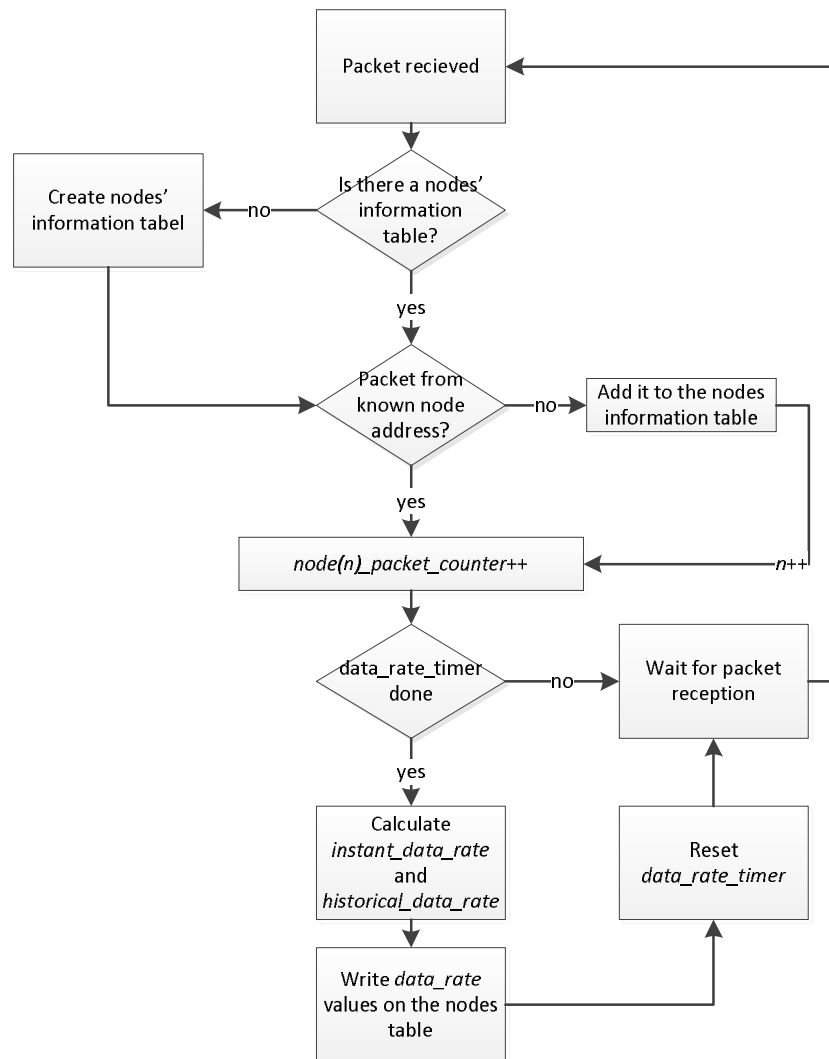


Figure 4.7: Data rate calculation algorithm flow chart

The Nodes' information table is created by the Data rate calculation algorithm but it is manipulated by other algorithms from the Monitoring, Scheduling and GTSS assignment modules at different stages of the scheduling process. Each node has a unique table associated to it.

The table is an *array of structs* with the following information from the node:

- Address
- Message size
- Instant data rate
- Historical data rate
- Score of the node given by each of the scheduling algorithms
- Different counters associated to the node used by different algorithms for other calculations, timers and counters resets

The information tables are reseted after a time T in order to avoid any disruption on the scheduling process.

Traffic monitoring:

- Active nodes counter algorithm: this algorithm is in charge of counting the nodes transmitting during a TTI, different from the scheduling TTI. This algorithm resets the number of nodes transmitting value after a given time so that nodes that stopped transmitting are not considered for scheduling by the scheduling module. Its execution is triggered when a packet is received by the coordinator. Figure 4.8 shows the flow diagram of the algorithm implemented.

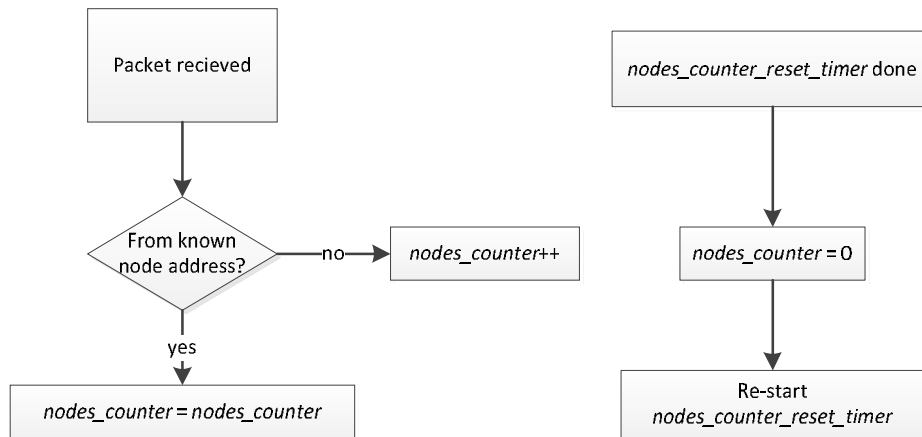


Figure 4.8: Active nodes counter algorithm flow diagram

4.3.2. Scheduling module

The Scheduling module encapsulates the Scheduling manager function and the Scheduler's algorithms. Figure 4.9 shows the module with its functions. Each function contains algorithms in charge of specific tasks. The algorithms are explained below.

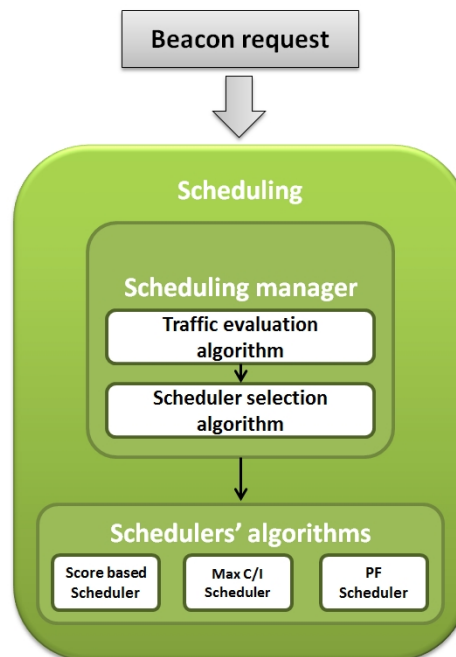


Figure 4.9: Scheduling module

Scheduling manager:

The scheduling manager structure is shown by Figure 4.10. Its execution is triggered by the beacon request from the coordinator firmware.

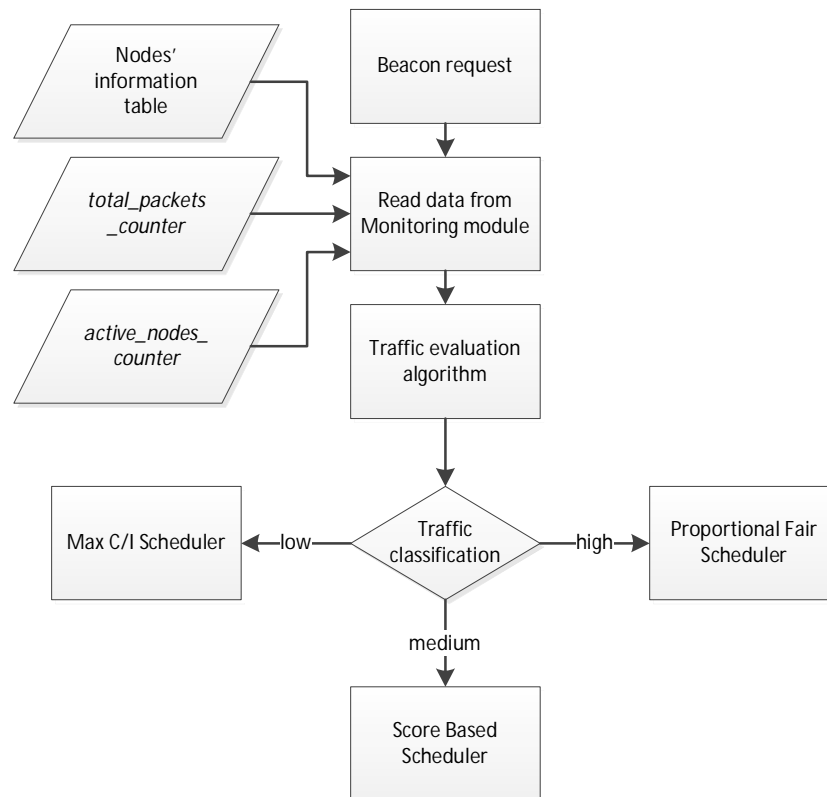


Figure 4.10: Scheduling manager flow chart

This function contains two algorithms that are mapped by Figure 4.10:

- **Traffic evaluation algorithm:** based on the information incoming from the monitoring module, this algorithm evaluates and classifies the wireless traffic condition of the network using the number of packets sent, the number of active nodes sending packets and the historical data contained in the Nodes' information table. There are three possible traffic conditions that depend on the previous data: low, medium and high. The result of this algorithm is sent to the scheduler selection algorithm.
- **Scheduler selection algorithm:** it is a simple switch-case structure algorithm that chooses the scheduler to be used for the GTSs allocation.

Schedulers' algorithms:

There are three scheduling decision algorithms implemented inside this functionality. Each of the scheduling algorithms is designed to optimize the GTSs scheduling according to different wireless traffic conditions.

Once the Scheduler selection has chosen the algorithm to be used based on the Traffic evaluation, the algorithm of the scheduler selected will start executing.

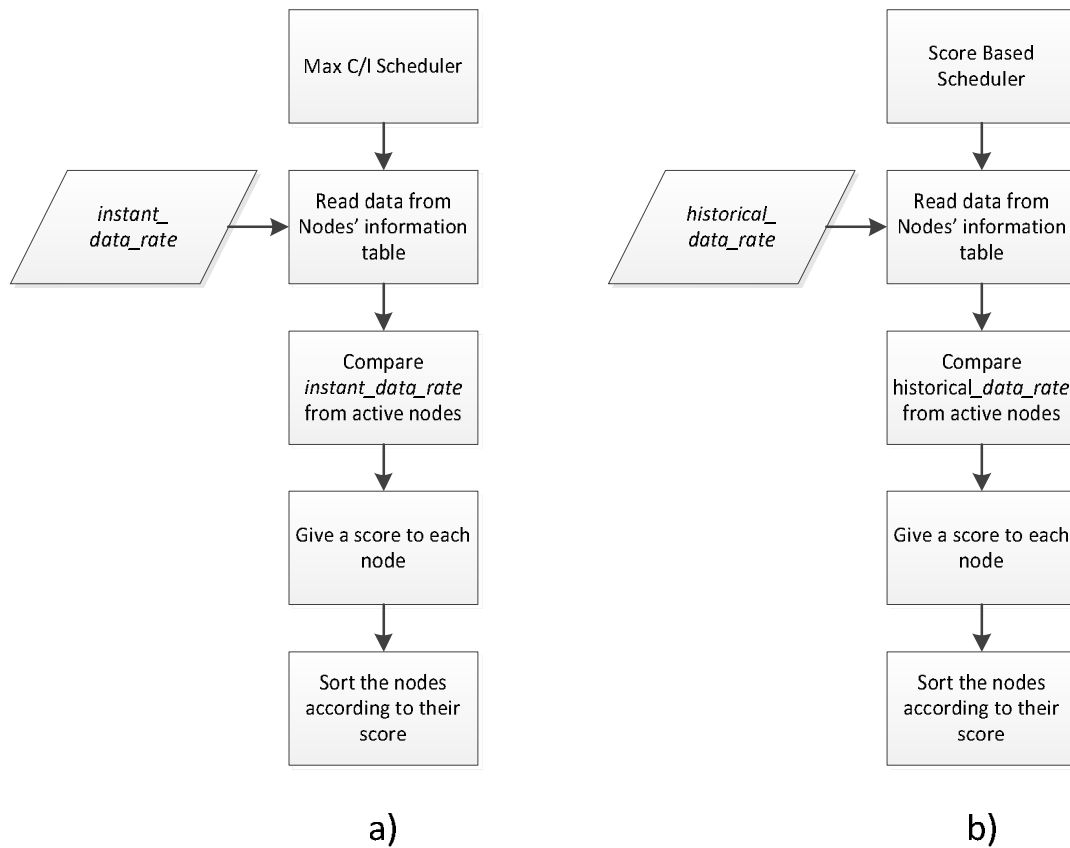


Figure 4.11: Max C/I and Proportional Fair scheduling algorithms flow charts

The three scheduling algorithms are:

- **Max C/I Scheduler algorithm:** the Maximum Carrier to Interference ratio Scheduler algorithm is shown in Figure 4.11a. The Max C/I scheduler gives priority for accessing the channel to nodes with the maximum instantaneous data rate in the current TTI. This scheduler provides the highest cell (global) throughput because it always serves users with the highest data rate. On the other hand, this scheme is very unfair because not all the nodes of a network can get all the resources and some nodes of the network will be starved, making it ideal for low traffic with a low number of nodes transmitting critical data to the coordinator. After giving a score, the algorithm sorts the nodes according to their score so that the GTSs allocation module builds the superframe based on this information.
- **Score Based Scheduler algorithm:** the Score Based Scheduler algorithm is shown in Figure 4.11b. It is designed to ensure that all simultaneously queued users receive the same average throughput. This scheduler is ideal for medium traffic conditions as, in every TTI, the priority to have channel access is given to the user with the highest average throughput or historical data rate. This algorithm sorts the nodes according to their score, in the same way as the Max C/I Scheduler.

- Proportional Fair Scheduler algorithm: shown in Figure 4.12, this scheduler algorithm gives a score to the instant data rate of a user with respect to the historical throughput over a window of size $T = 60$ seconds. The Proportional Fair scheduling can distribute the channel access equally even under asymmetric networks with a high number of nodes where there might be a limited number of nodes sending a high amount of critical data and others with a lower data rate, making it ideal for a network with a high wireless traffic.

Like the other scheduling algorithms, the Proportional Fair Scheduler algorithm sorts the nodes after giving them a score for the GTSs assignment module to build the beacon and the superframe.

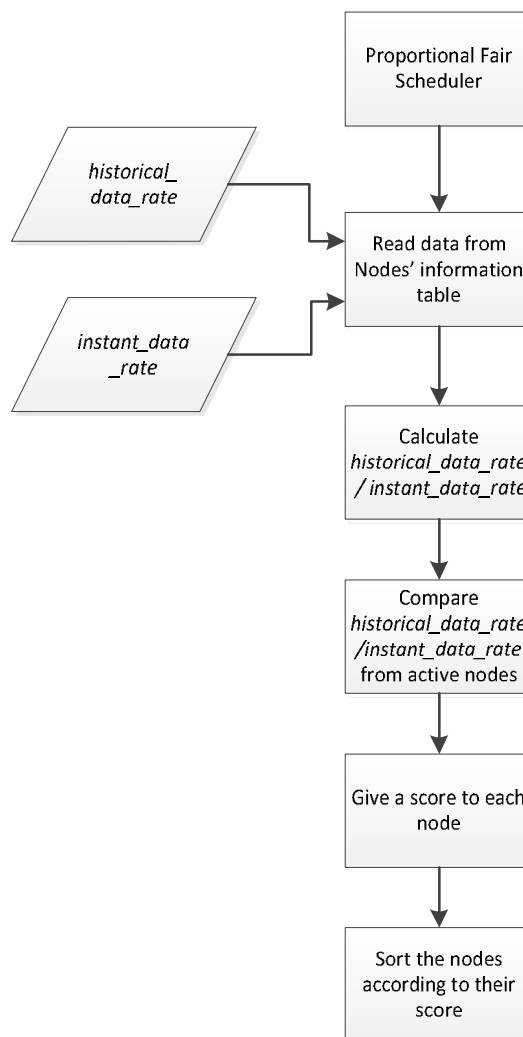


Figure 4.12: Score Based Scheduler algorithm flow chart

When Scheduling module code has completed its execution, the GTSs assignment module function is called.

4.3.3. Guaranteed Time Slots assignment module

The GTSs assignment module encapsulates the Superframe manager function that contains the Superframe manager and the Beacon Builder algorithm. Figure 4.13 shows the module with its functions. The algorithms' functions are explained below.

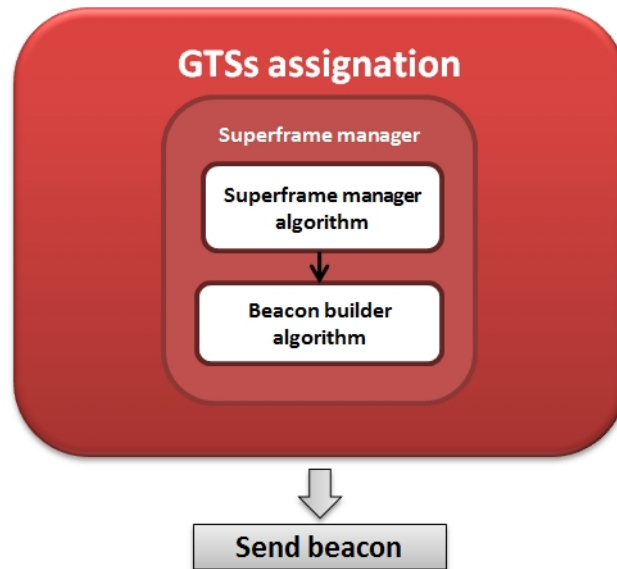


Figure 4.13: Guaranteed Time Slots assignment module

Superframe manager:

- Superframe manager algorithm: is in charge of allocating the GTSs to the nodes transmitting to the WPAN coordinator. This algorithm assigns as many Guaranteed Time Slots as the node needs according to their message size, taken from the Node's information table.

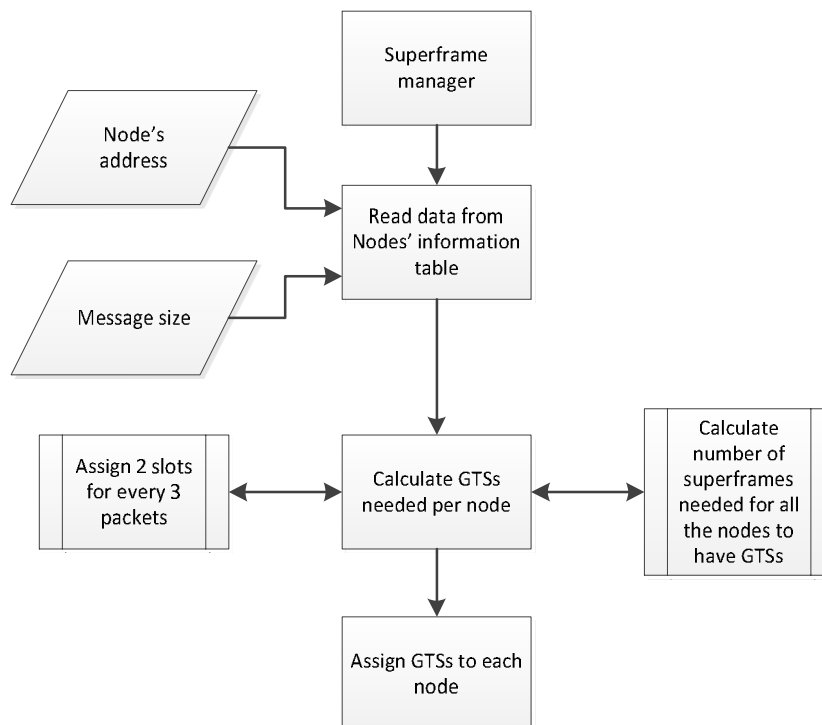


Figure 4.14: Superframe manager algorithm flow chart

The Superframe manager algorithm flow chart is shown in Figure 4.14. The GTSs calculation has two sub-processes that run simultaneously to give support to it.

To optimize the number of GTSs used per node, this algorithm assumes that three 6LoWPAN packets fit into two slots.

The superframe manager algorithm utilizes as many superframes as necessary in order to guarantee GTSs to all the nodes of the network by using the GTSs assignation distribution illustrated by Figure 4.15 where nodes with higher priority are scheduled with a higher frequency than nodes with a lower priority.

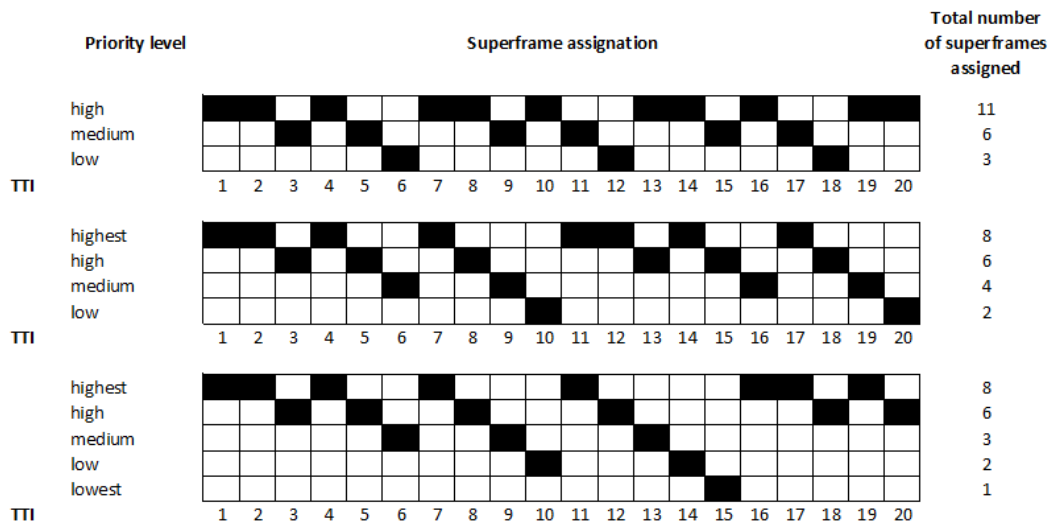


Figure 4.15: Superframe assignation distribution example for 20 TTIs

- Beacon builder algorithm: shown in Figure 4.16, this algorithm gathers and organizes the information required by the beacon so that it can be sent by the beacon transmission function and then transmitted to the nodes. It is a very simple algorithm that reads the information from the functions where it is allocated so that the beacon can be organized and sent.

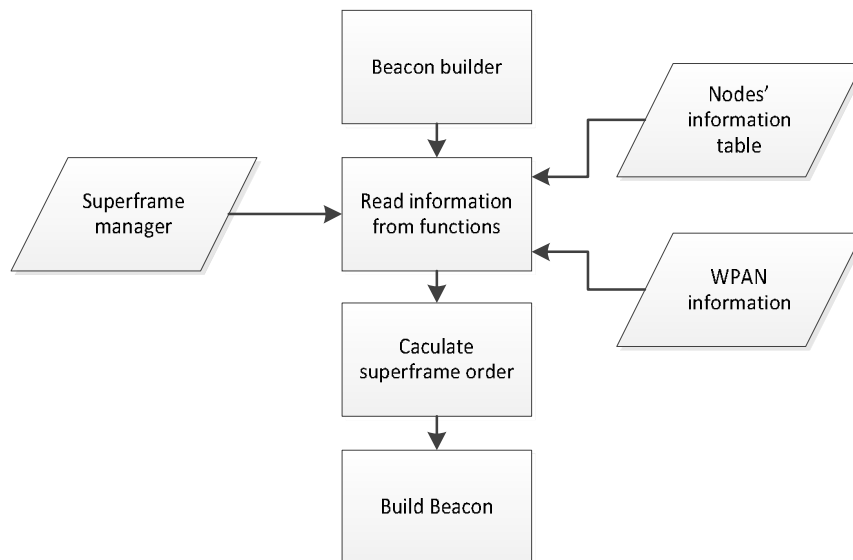


Figure 4.16: Beacon builder algorithm flow chart

4.3.4. Sensor nodes' algorithm

The nodes' firmware includes a function module for the scheduled GTSs and superframe handling. The nodes are programmed to send packets only during the CFP at their assigned Guaranteed Time Slots. If a node has not been scheduled before the *unscheduled_node_counter* expires, it means that the coordinator has lost the information from the node, this can happen, for example, after the Nodes' information table has been reset. In this case, the node will attempt to send packets during the CAP so that the coordinator takes the node into consideration for the scheduling process by recalculating the node's information on the Nodes' information table.

The *unscheduled_node_counter* expiration period depends on the type of the sensor as different sensors require fewer attempts to be considered by the coordinator for the scheduling. Figure 4.17 shows the sensor nodes' algorithm flow chart.

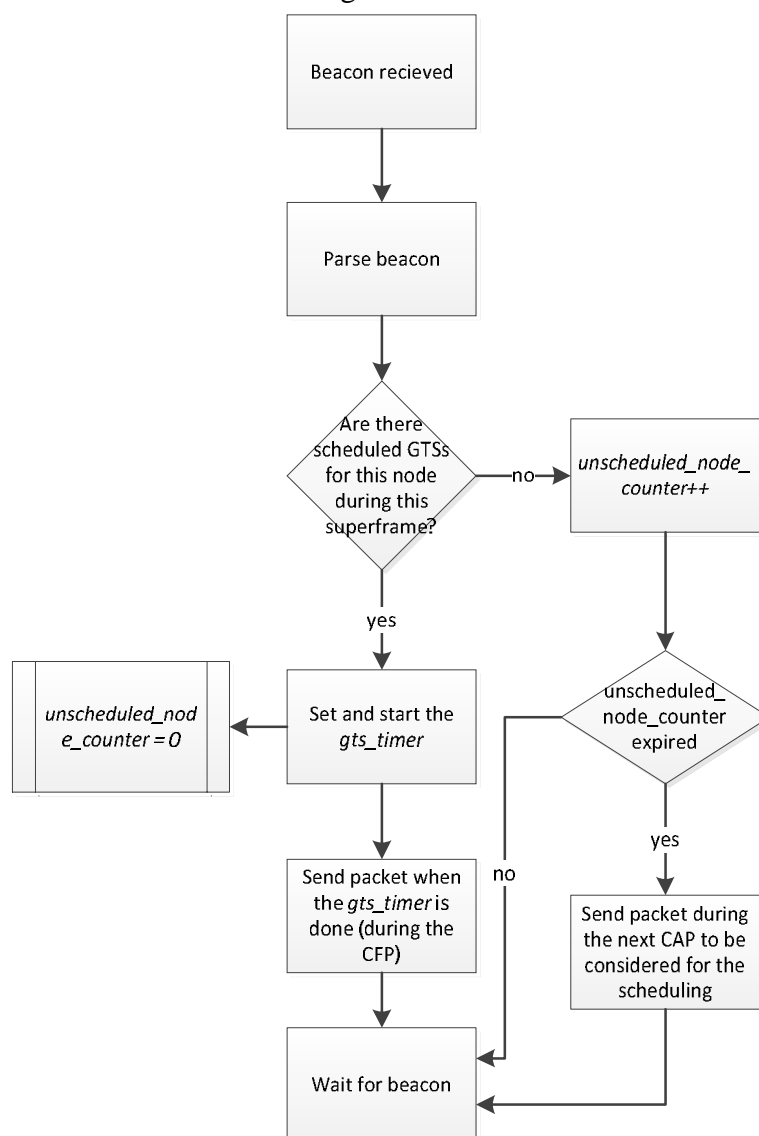


Figure 4.17: Sensor nodes' algorithm flow chart

4.4. Experimental implementation

The following section focuses on the experimental description of the tests required to analyse the performance of the wireless packet scheduling system for 6LoWPAN industrial applications described by this thesis work.

Two different tests were performed:

- Network energy consumption test
- Network throughput and packet loss test

Were three variables were monitored using the network monitoring application described during the Section 4.4.2:

- Network's energy consumption
- Network's throughput
- Network's packet loss

Since the software developed is based on previously existing firmware that does not have beacon-enabled capabilities, the tests evaluate and compare the non beacon-enabled with the packet scheduling system software versions. For this reason, most of the results will be contrasted with the performance of wireless sensor networks using these two different firmware versions at their nodes and coordinators.

4.4.1. Test bed

The FAST WSN, installed at the production line of the FAST laboratory, worked as the test bed for the experimental implementation. This network is described with detail at the Section 3.3.

As it was explained, the FAST WSN is a 6LoWPAN-enabled WSN that consists on Reduced Function Devices nodes connected to an IPv6 network for wireless embedded applications through Full Function Devices in order to enable the use of web-services end-to-end over low power wireless devices employing a passive gateway solution, using Atmel hardware and Contiki operating system ported to Atmel Radio Frequency platform as the core of the embedded firmware.

Due to hardware limitations, the THL sensors were the only sensor nodes used for the tests. In the case of the accelerometers, the limited processing power of the MCU that this sensor uses did not allow the installation of the new firmware as the beacon parsing and the timers handling processes, required to use the IEEE 802.15.4 beacon-enabled mode on the sensors, surpass the capabilities of their micro-controller.

Moreover, the digital-analogue I/O sensors installed at the production line cannot be used for testing as they are critical for other applications running at FASTory and due to the working schedule of the line it was not possible to have them out of service for the tests. Furthermore, they required different and more complex software changes that would not allow them to be used for the scheduling firmware tests.

4.4.2. Network monitoring application

In order to monitor the performance of the scheduling system, a network monitoring application was required. The application was programmed on JAVA programming language. Its main window interface is shown by Figure 4.18. This application is based on the Wireless Asset Management application, programmed at FAST Lab and previously used for the eSON-IA project. The FAST Lab Wireless Network Monitoring tool is a network sniffer that can store the data that it captures in a text file for its analysis.

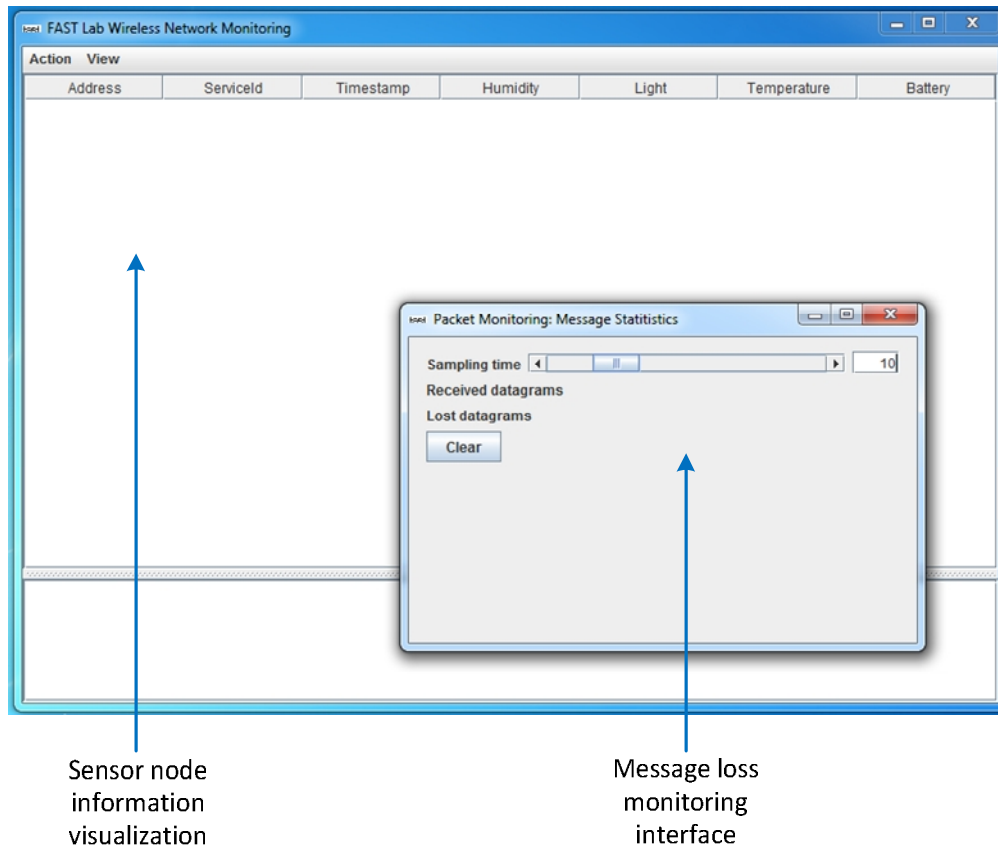


Figure 4.18: FAST Lab Wireless Network Monitoring tool

The application is able to monitor the nodes sending messages to the destination port where it is hosted with the following information:

- Node's address
- Node's service ID
- Message's time stamp
- Message's length
- Sensor node information
- Total number of messages received
- Number of lost messages

In case the sensor sending packets to the destination is a THL sensor, the application will show its temperature, humidity, light and the battery left on it (showed in Volts).

The application is able to show the number of messages received and lost according to a variable sampling time interval that can be manually adjusted. It can create and write a text

file with all the captured data so that it is possible to handle and analyse the information using specialized software for data management and plotting such as Microsoft Excel, Origin or Matlab.

4.4.3. Energy consumption network implementation

The purpose of energy consumption tests was to compare the energy consumption between a WSN using the packet scheduling system and another with the non beacon-enabled firmware version.

For this test, two different wireless sensor networks were sending messages to a coordinator under the same transmission conditions. Figure 4.19 shows the layout of the networks installed at the FASTory line room. Both networks are similarly deployed over the room with their respective WPAN coordinator installed at the same place. Each circle represents a THL sensor located at the place shown. As it can be seen, each of the networks consists on:

- 10 THL sensors
- 1 S1000 RTU with wireless expansion as the WPAN coordinator

The only difference between both networks relies on the firmware version that their sensor nodes and coordinator are using. The network represented by the blue figures is using the real-time packet scheduling system for a 6LoWPAN industrial applications firmware. The network represented by the red figures is using the previously existing non beacon-enabled firmware.

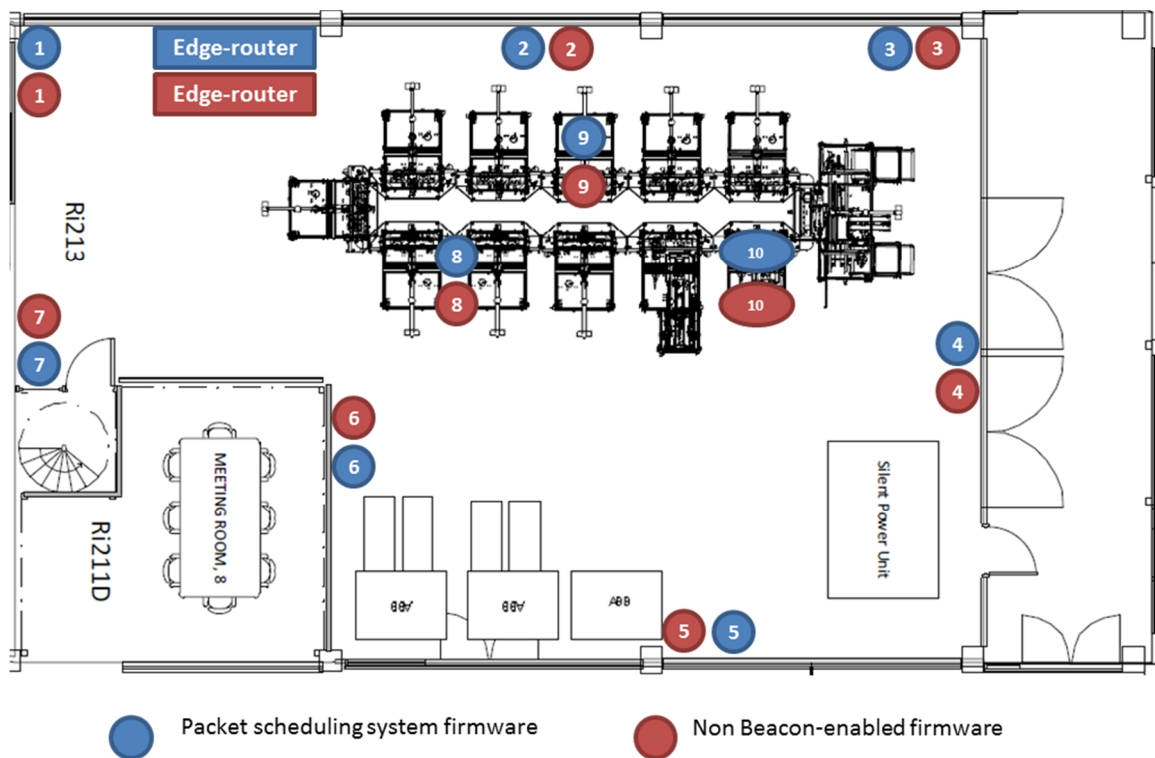


Figure 4.19: WSN deployment over FASTory line room for energy consumption tests

For the energy consumption analysis tests, both networks were transmitting at the same data rate using different channels. The energy consumed was monitored using the network

monitoring tool. More details about the parameters used, along with the results of this test, will be shown at the Section 5.1.

4.4.4. Network throughput and Packet loss network implementation

The purpose of these tests was to monitor the global network throughput and its relation with the packet loss of the system in order to evaluate the efficiency of the packet scheduling system. In the same way as in the energy consumption tests, the results obtained with the packet scheduling firmware are compared with the ones obtained by using a non beacon-enabled network.

Another important purpose of these tests is to evaluate if the scheduling algorithm selection is working as intended. Since there is not any experimental data that proves that the scheduling algorithm selected for each traffic condition is the ideal one, an important test was performed to monitor the behaviour of each scheduling algorithm in order to validate that the scheduling selection is the ideal according to the wireless traffic density.

Due to the renovation works that the FAST laboratory was going through during the summer of 2013, these tests could not be performed at the FASTory line. Instead, they were held at the RM205 room at the Rakennustalo building of the Tampere University of Technology. Figure 4.20 shows the layout of the WSNs deployment used for this test.

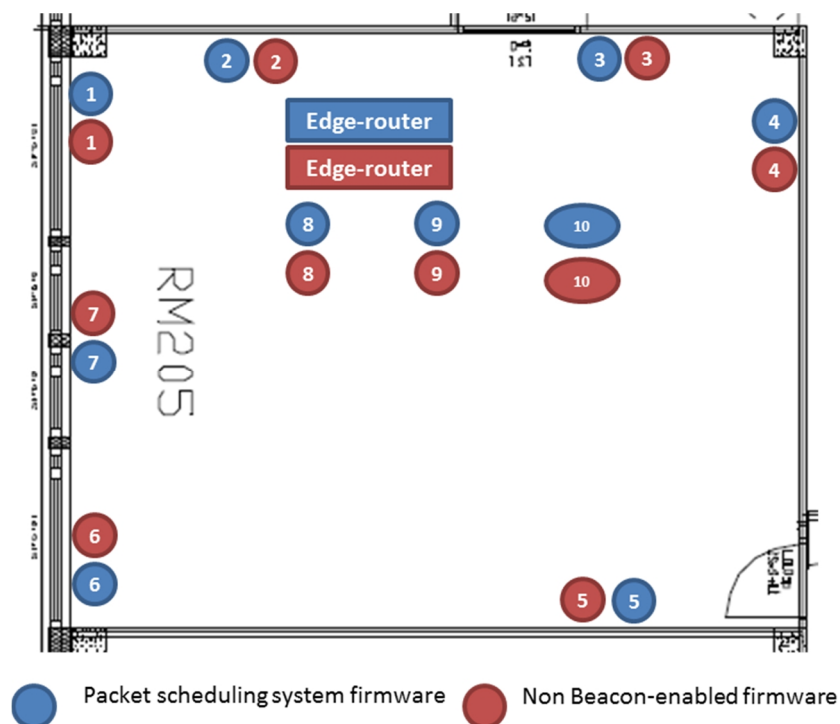


Figure 4.20: WSN deployment over RM205 room for throughput tests

In the same way as in the energy consumption tests, each of the networks used consists of:

- 10 THL sensors
- 1 S1000 RTU with wireless expansion as the WPAN coordinator

The network represented by the blue figures is using the real-time packet scheduling system for a 6LoWPAN industrial application firmware. The network represented by red figures is using the previously existing non beacon-enabled firmware.

For these tests, the following variables were monitored using the network monitoring tool:

- Node's address
- Message length
- Message time stamp

With this data it is possible to obtain the global throughput of the networks by calculating the number of messages successfully received during a fixed time window. This information is required to calculate the throughput versus the packet loss ratio of the network and to evaluate the behaviour of the scheduling algorithms. More details about the parameters used for the test, along with the results of it, will be shown at the Sections 5.3 and 5.4.

5. RESULTS

This chapter presents the experimental implementation results of this thesis work. It contains the outcome from the energy consumption, scheduler's algorithm selection and network throughput analysis tests.

The results are presented with a detailed description of the tests where they were obtained. All the plots were drawn using Microsoft Excel software and some of the data was processed using Matlab software.

5.1. Energy consumption

In order to analyse the energy consumption of the scheduling system, the experiment described at the Section 4.3.3 was implemented, where two networks, one using the packet scheduling system and the other a non beacon-enabled firmware, were transmitting a given amount of messages.

Each network consisted of 10 Inico THL sensor nodes and its coordinator. All the sensors of both networks were configured to send messages with the following periodicity and parameters for the test:

Table 5.1: Energy consumption tests transmission parameters

Message transmission interval [seconds]	Sensing interval [seconds]	Throughput per node [bytes/s]
10	10	64

The selected transmissions parameters correspond to a low speed network (see Section 5.2) where the number of packet collisions is expected to be low, this configuration was selected as the purpose of this test is to monitor the amount of energy required to transmit packets using different bandwidth management mechanisms and not to measure the extra energy consumed by the additional processing tasks that collisions handling may bring to each sensor.

The WPAN coordinator energy consumption is not of relevant for this test as it is a FFD sink node that consumes a relatively high amount of energy and that it is plugged to AC energy socket.

Each node requires two Energizer LR6 batteries of 1.5 Volts, especially designed for industrial use. All the nodes of both networks started the test with full battery, which value corresponds to 3.16 [Volts] per node.

For the tests, the energy value of each THL sensor was captured during 10,000 successfully delivered messages per node using the network monitoring application, meaning that the

WSNs were transmitting during over 13 hours. The message loss percentage per network is presented in Table 5.2:

Table 5.2: Network's message loss percentage for energy consumption tests

Packet scheduling system network	Non beacon-enabled network
3%	9%

The number of packets lost by each network will be later analysed by the throughput and packet loss tests described during the Section 5.3.

With the data obtained during the experiment, the average of the network's battery consumption was calculated and then filtered using Matlab's instruction *medfilt1()*. The filtered data is shown by Figure 5.1

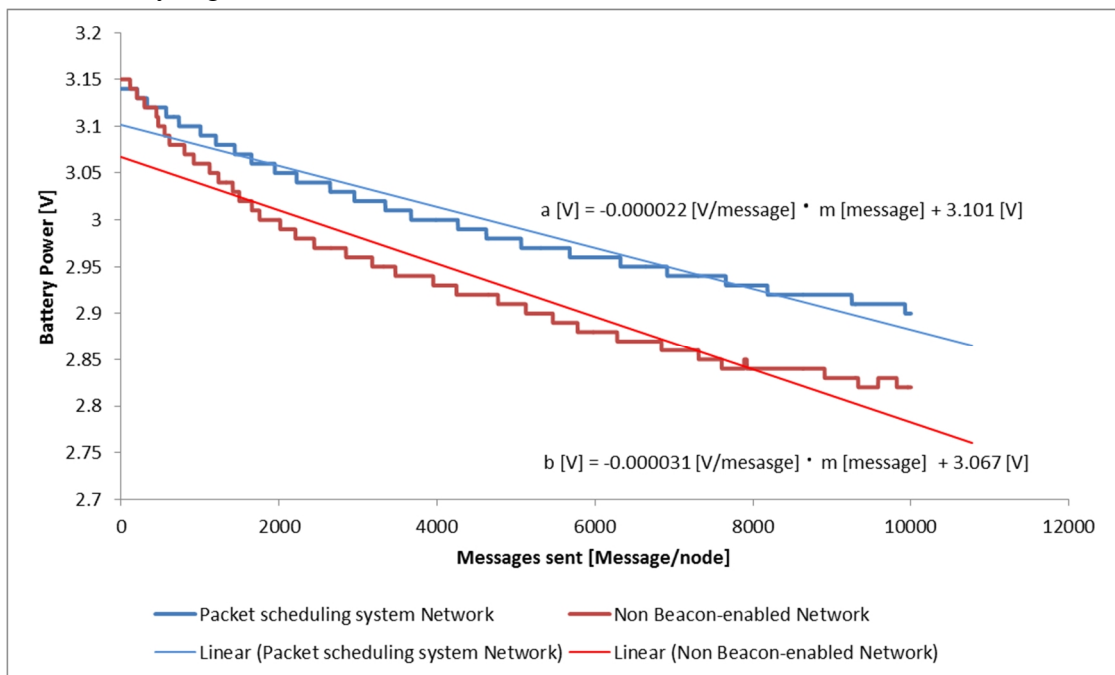


Figure 5.1: Energy consumption comparison plot

The energy consumption of a network using the packet scheduling system, designed for this thesis work, has lower battery consumption than a network using a non beacon-enabled firmware.

Using the linear trend line shown in Figure 5.1 it is possible to forecast that the network using the packet scheduling system firmware will be able to send an approximate of $27,318 \pm 3\%$ messages before its battery reaches the 2.5 Volts value where the sensors will start having sensing problems due to the low power available in the batteries. On the other hand, a network using a non beacon-enabled firmware will be able to send an approximate of $18,290 \pm 9\%$ before the batteries reach the 2.5 Volts threshold value, 40% less messages than the network with the packet scheduling system.

5.2. Networks' performance limits

Since the data rates of any industrial WSN are applications specific and they depend on the wireless transmission protocols they use, the type of sensors utilized and the order of the network, in order to test the FAST WSN it is necessary to obtain its performance limits.

In other words, before starting analysing the performance of the scheduling algorithms and its effectiveness, it is necessary to know which data rate loads are suitable to be used to test the packet scheduling system implemented at the FAST WSN.

These tests were performed using two networks, with the layout described by Section 4.4.4, that were submitted to a high data rate load so that its performance limits can be calculated. The parameters of the test are shown by Table 5.3.

Table 5.3: Performance limits' tests parameters

Rate load	Network's Firmware
640 [bytes/s] per node	Packet scheduling system
	Non beacon-enabled

Figure 5.2 and show the throughput limits of both networks obtained using the previously explained parameters. The throughput limit can be considered as the maximum data rate load to which the network can be submitted to before its performance stops being efficient.

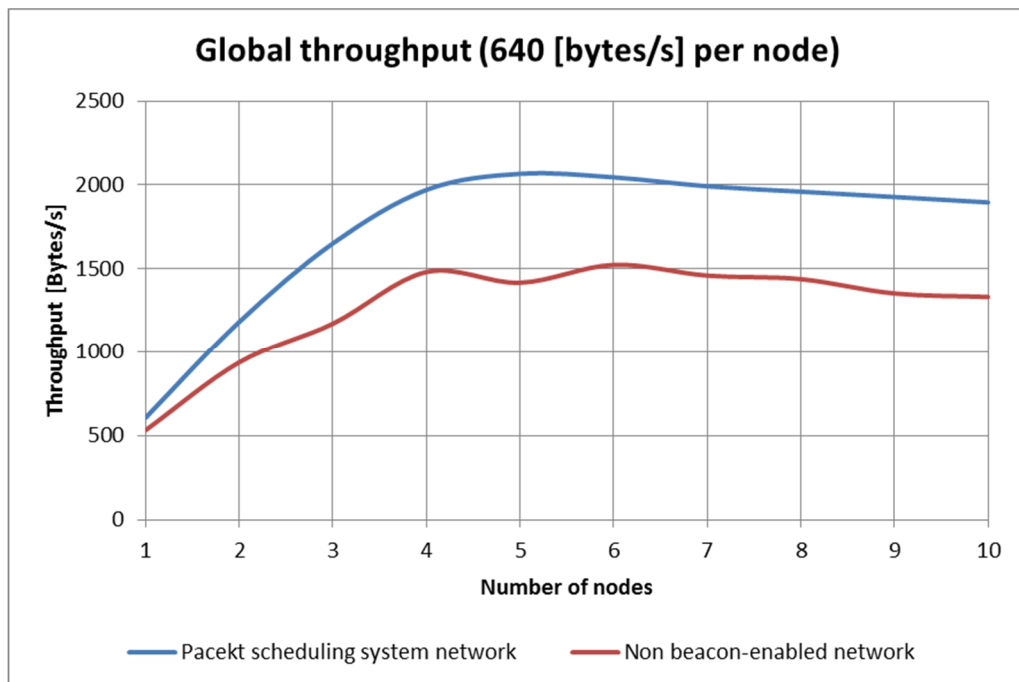


Figure 5.2: Global limit throughput

The limit throughputs for each network are:

- Non beacon-enabled network: 1522 [bytes/s]
- Packet scheduling system network: 2066[bytes/s]

5.3. Scheduling algorithms' performance

The purpose of the scheduling algorithm's performance tests is to give experimental support to the scheduling manager decision algorithm that selects the optimal scheduling algorithm according to the network's traffic condition.

Two networks were deployed with the layout that the Section 4.4.4 describes but only the one with the packet scheduling system is used for these specific tests. The performance of each of the three schedulers algorithms implemented was monitored using three different transmission rates per node, with a number of nodes that ranged from 1 to 10, in order to evaluate if their behaviour is the one that best fits the traffic condition in which the algorithm is used by the scheduling manager.

For each throughput, the network is "forced" to use only one of the scheduling algorithms programmed at a time while the WSN transmits information at a fixed rate per node. The network starts with one node broadcasting messages and it keeps adding nodes until the number of nodes transmitting reaches ten, increasing the data transmission load. The coordinator is "forced" to use only one of the schedulers by modifying the firmware program and disabling the other two algorithms implemented. The data is captured using the network monitoring application so that the network's global throughput and the throughput versus the packet loss ratio can be plotted later.

The throughput versus the packet loss ratio shows the efficiency of the message transmission-reception process and it is obtained by the following equation:

$$\frac{\text{number of messages succesfully recieved by the coordinator}}{\text{number of nodes} \times \text{number of messages sent per node}}$$

As it can be seen, the highest value of the equation can be 1. It can also be considered as a percentage of successfully received messages by the coordinator where a 100% percentage means that there is not any package loss.

Table 5.4 shows the parameters used for the evaluation of the scheduling algorithms tests.

Table 5.4: Scheduling algorithm performance tests parameters

Test	Rate load	Scheduler used
1	210 [bytes/s] per node	Max C/I
		Score Based
		Proportional Fair
2	128 [bytes/s] per node	Max C/I
		Score Based
		Proportional Fair
3	64 [bytes/s] per node	Max C/I
		Score Based
		Proportional Fair

The results of the scheduling algorithm performance test 1, at 210 [bytes/s] per node are shown in Figure 5.3 and Figure 5.4. As it was explained during the Section 5.2, this corresponds to a high data rate load to the network.

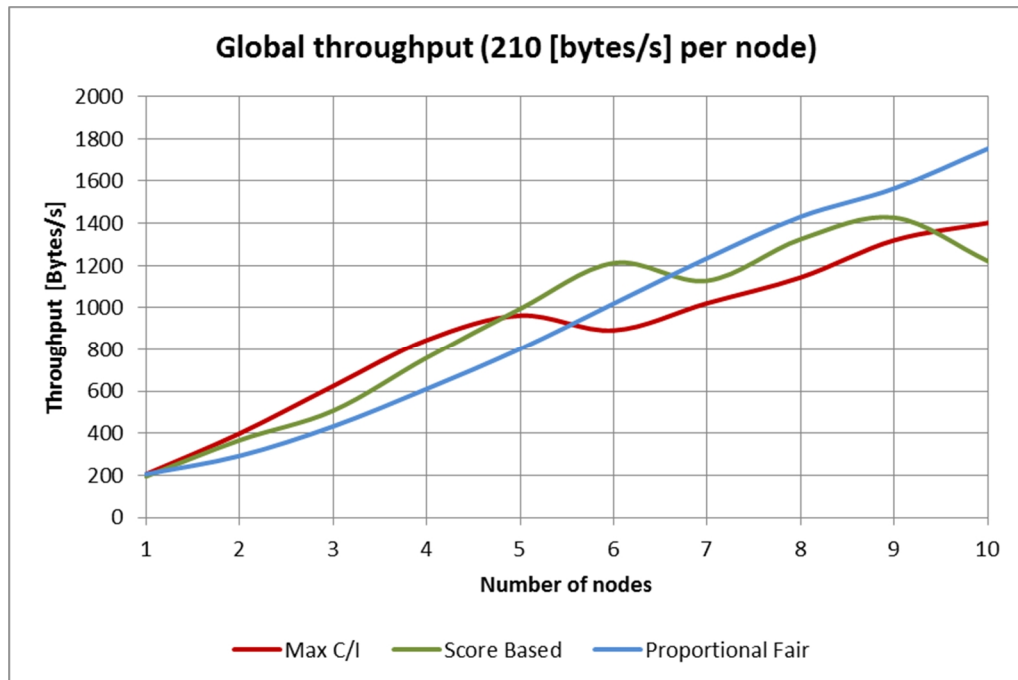


Figure 5.3: Global throughput at 210[bytes/s] per node forcing scheduling algorithms

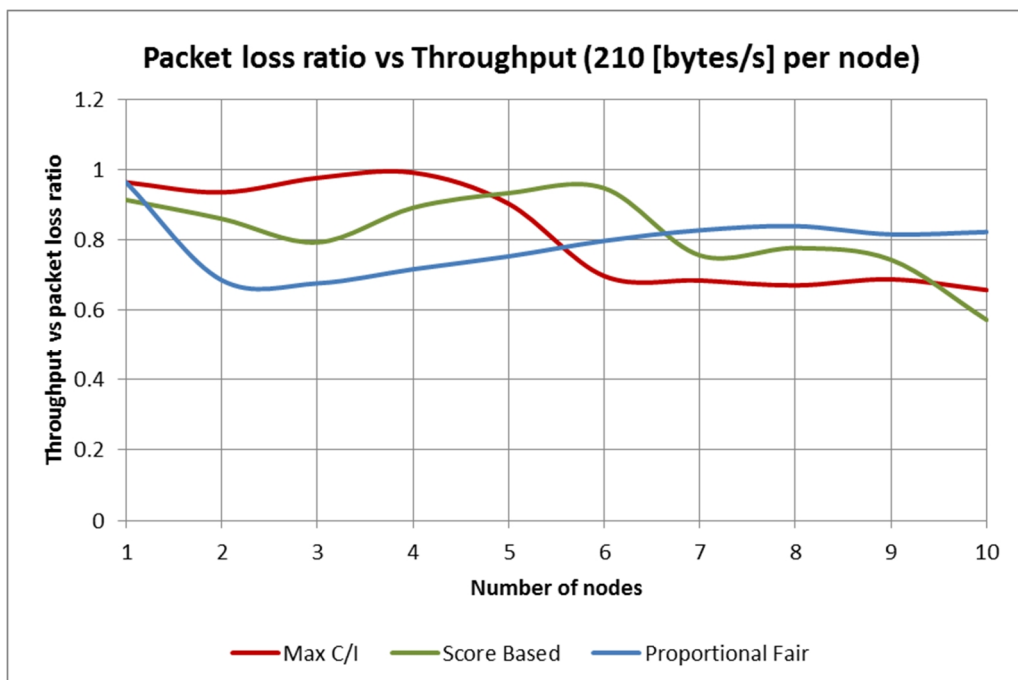


Figure 5.4: Packet loss ratio vs. Throughput at 210[bytes/s] per node forcing scheduling algorithms

For a low data transmission load, the Max C/I scheduler has the best performance as it guarantees both, the highest throughput and the lowest packet losses. As the data load of the network increases, the Max C/I's throughput starts decreasing caused by the packet loss. Similar behaviours can be observed on the Score Based and the Proportional Fair scheduling algorithms for medium and high throughputs respectively.

As the behaviours of the scheduling algorithms cannot be generalized by analysing only one transmission load, it is required to test their performance with a medium and a low load. The results of the scheduling algorithm performance test 2, at 128 [bytes/s] per node are shown in Figure 5.5 and Figure 5.6. As it was explained during the Section 5.2, this corresponds to a network's medium data rate load.

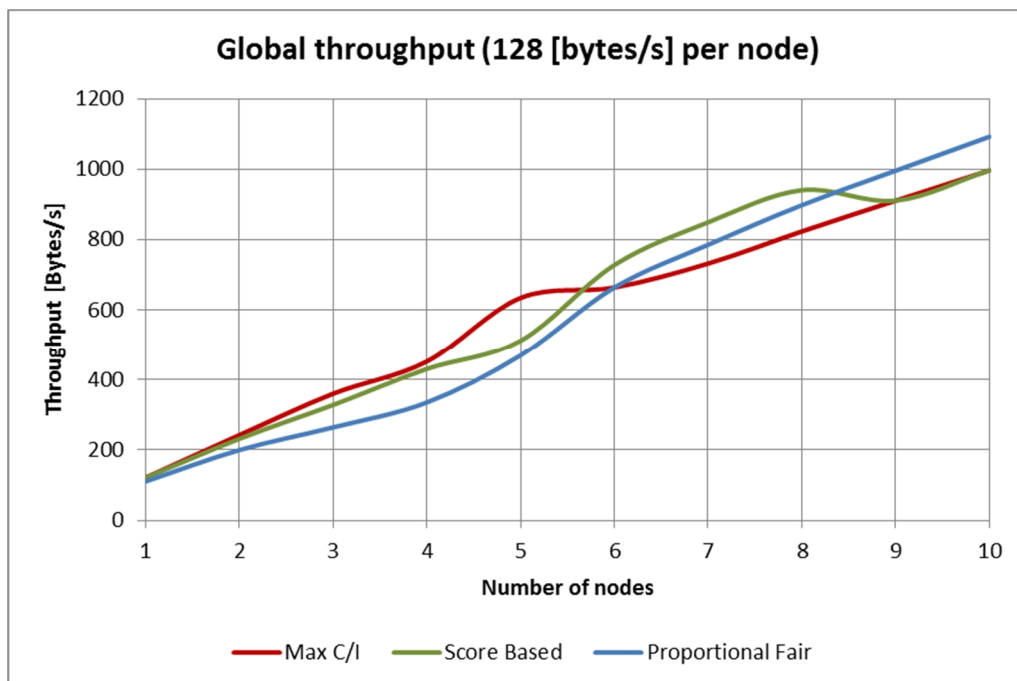


Figure 5.5: Global throughput at 128[bytes/s] per node forcing scheduling algorithms

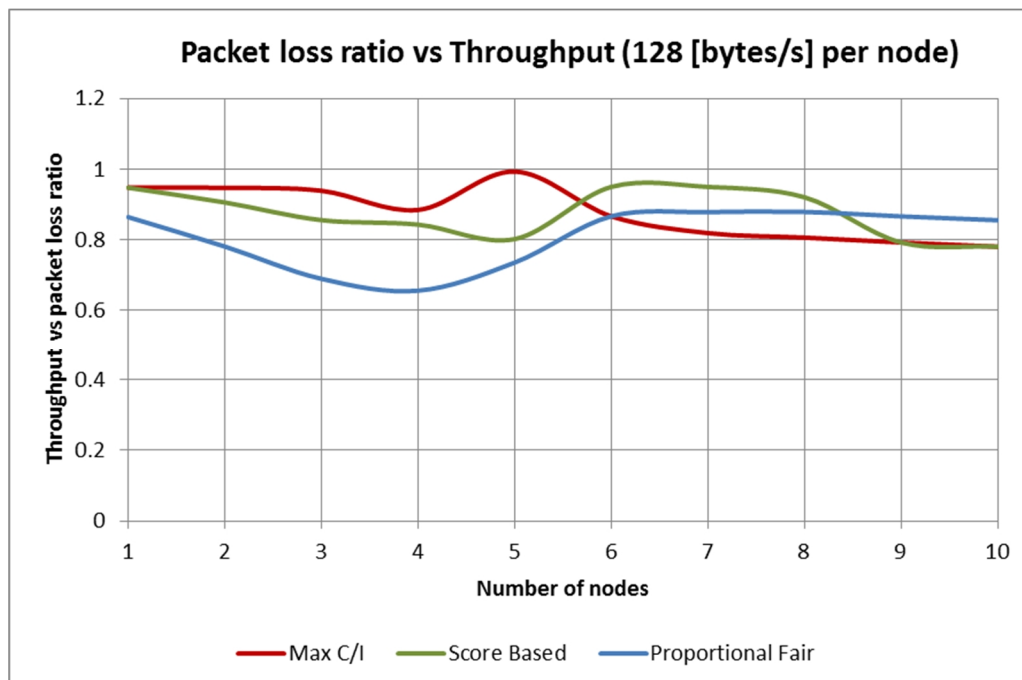


Figure 5.6: Packet loss ratio vs. Throughput at 128[bytes/s] per node forcing scheduling algorithms

The behaviour of the scheduling algorithms under a medium throughput load is similar to the one that the network had with a high transmission load. A noticeable difference is the small displacement on the performance curves caused by the lower load of this test.

Finally, the performances of the scheduling algorithms during the test 3, at a 64 [bytes/s] per node transmission load are shown in Figure 5.7 and Figure 5.8. As it was explained during the Section 5.2, this corresponds to a network's low data transmission rate load.

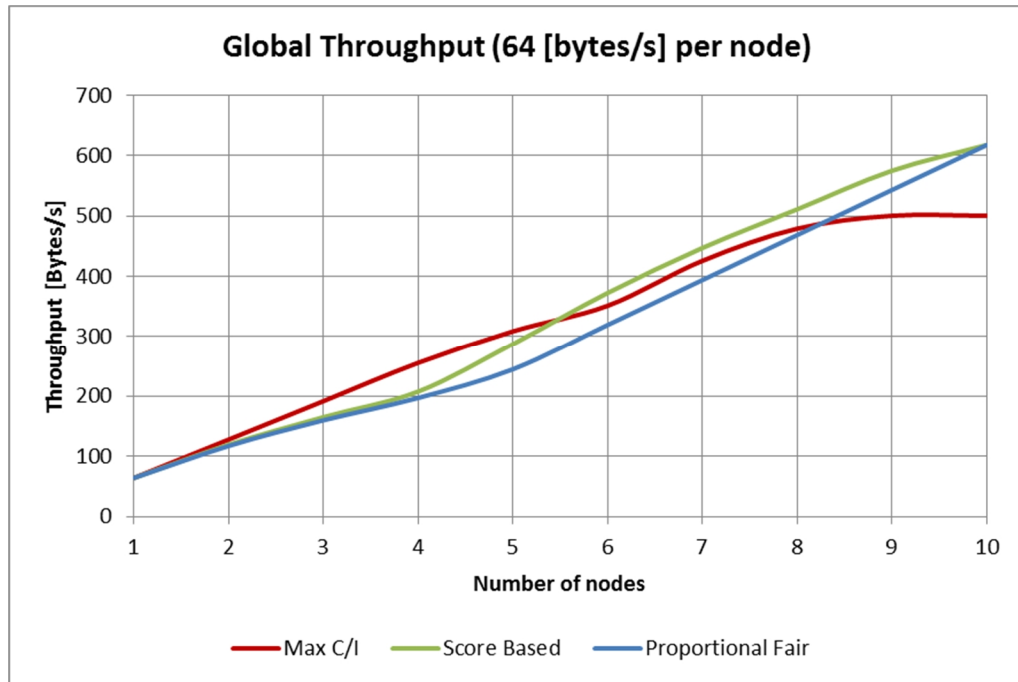


Figure 5.7: Global throughput at 64[bytes/s] per node forcing scheduling algorithms

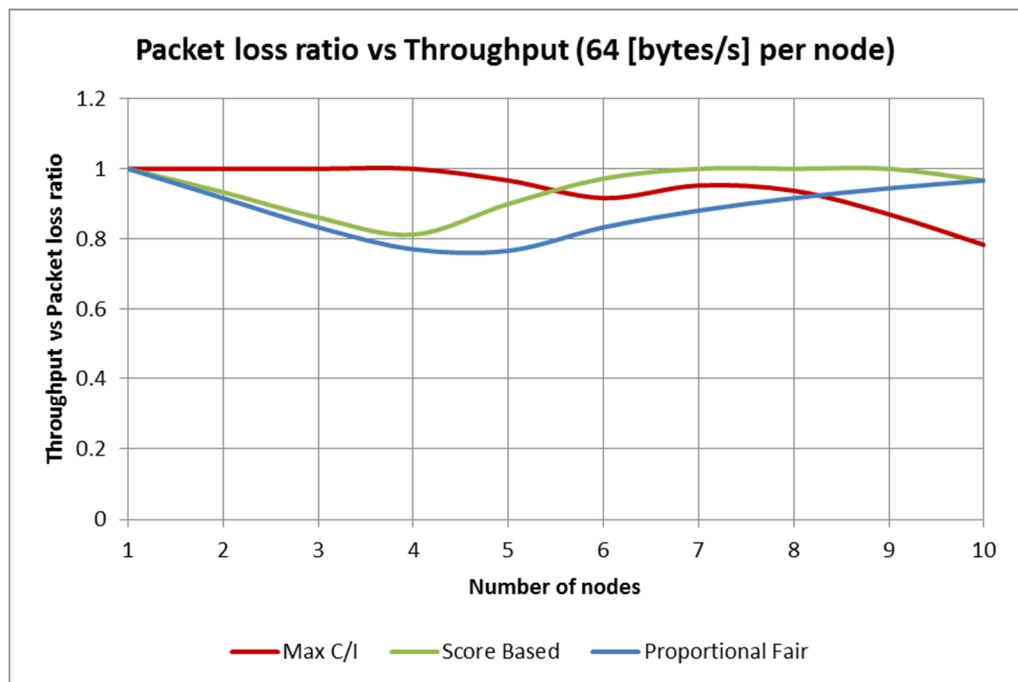


Figure 5.8: Packet loss ratio vs. Throughput at 64[bytes/s] per node forcing scheduling algorithms

At 64 [bytes/s] per node, the load is not high enough for the Proportional Fair scheduling algorithm to be as efficient as the Score Based for the highest values. However, the SB performance seems to be decreasing due to its packet loss increase, at the same time, the PF

performance is about to surpass the SB one, showing a similar behaviour as in the previous tests with higher data transmission loads.

With the results of these tests, it can be concluded that for the network where the real-time packet scheduling system for a 6LoWPAN industrial application is implemented:

- The performance of the Max C/I scheduling algorithm is the ideal for low data transmission loads.
- The performance of the Score Based scheduling algorithm is the ideal for medium data rate transmission loads.
- The performance of the Proportional Fair scheduling algorithm is the ideal for high data transmission loads.

This coincides with the theoretical review scheduling principles presented during previous chapters of this thesis work.

5.4. Networks' performance comparison

This is the most relevant test as it evaluates and compares the performance of the packet scheduling system with the performance of a non beacon-enabled network under different data transmission loads. The layout of both networks is described during the Section 4.4.4. The efficiency of the network's firmware is evaluated by obtaining the global throughput and the amount of data losses during the transmissions. The global throughput and the previously explained throughput versus packet loss ratio were calculated using the monitoring tool.

The performances are tested using three different data transmission rate loads. It is important to point out that the firmware of the packet scheduling system automatically selects the scheduler algorithm that is optimal for the network traffic condition. Table 5.4 shows the parameters used for these tests.

Table 5.5: Network throughput and packet loss tests parameters

Test	Rate load	Network's Firmware
1	210 [bytes/s] per node	Packet scheduling system
		Non beacon-enabled
2	128 [bytes/s] per node	Packet scheduling system
		Non beacon-enabled
3	64 [bytes/s] per node	Packet scheduling system
		Non beacon-enabled

The performance comparisons between the packet scheduling system and a non beacon-enabled network under a 210 [bytes/s] per node data transmission rate load, for a range between 1 and 10 nodes, are shown in Figure 5.9 and Figure 5.10. According to the Section 5.2, this corresponds to a high data rate load.

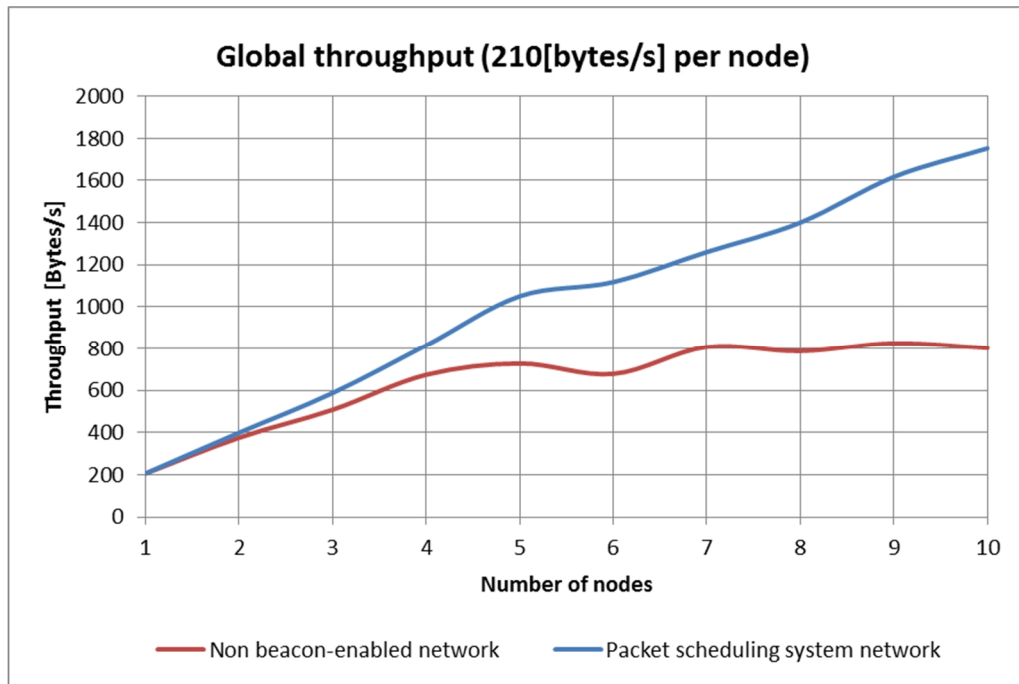


Figure 5.9: Global throughput comparison at 210 [bytes/s] per node

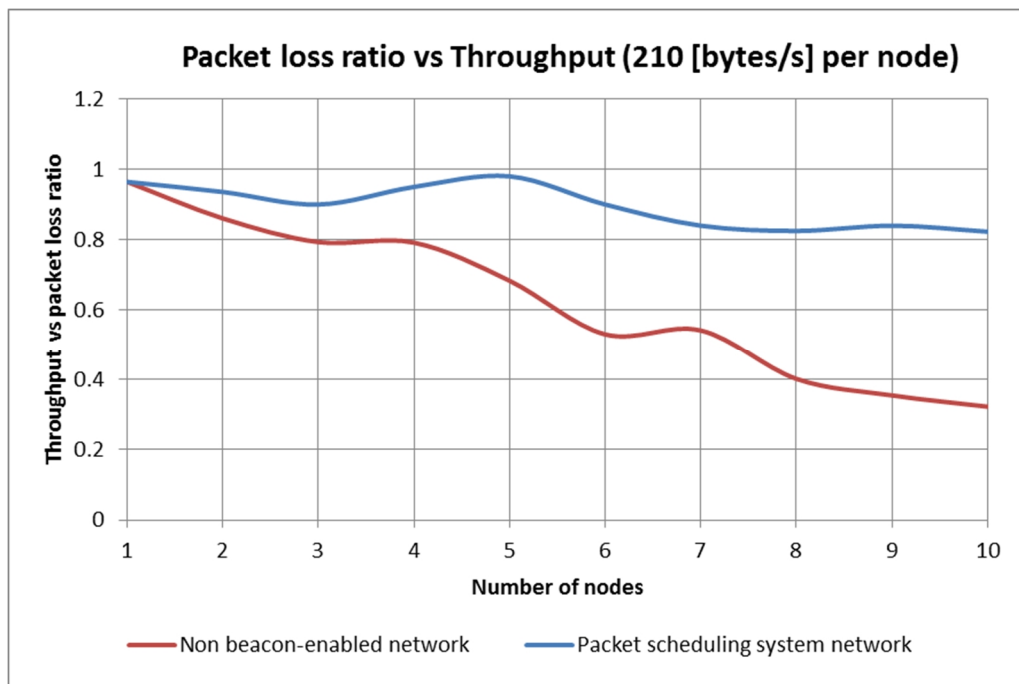


Figure 5.10: Performance comparison at 210 [bytes/s] per node

The results are clear: the network using the packet scheduling system firmware has a better throughput and less package loss ratio than the non beacon-enabled network. The packet scheduling system reaches in some points a package loss of less than 2%, having its maximum losses when the network traffic load increases its throughput almost to the maximum load that this network can handle according to the analysis made during the Section 5.2. The non beacon-enabled network cannot increase its throughput due to the high amount of data losses.

The packet scheduling system is designed to choose the scheduling decision algorithm that best fits the network traffic load. Using breakpoints on the debugging tool of Eclipse IDE, while the coordinator is running the firmware, it is possible to know the exact moment where each of the scheduling algorithms are used and switched by the scheduling selection mechanism.

Figure 5.11 shows the way the algorithm selection mechanism worked during this test. The scheduling manager chooses and changes the algorithm according to the network wireless data load.

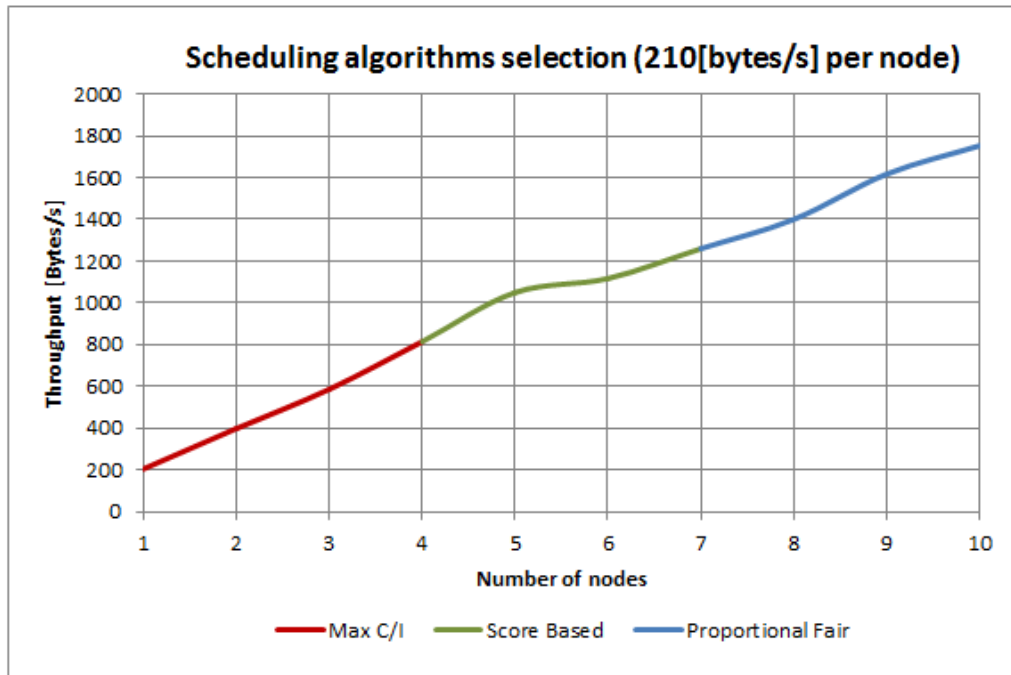


Figure 5.11: Scheduling algorithm selection at 210 [bytes/s] per node

For low throughputs, the Max C/I scheduler is used, the Score Based is used for medium throughputs and the Proportional Fair is used for high ones. This shows that both, the network monitoring module and the scheduling selection algorithm are working as intended.

As it is not possible to generalize only with one test, different data rate loads are required in order to validate the experimental results. The next data rate per sensor is considered to be a medium rate by the concepts presented during the Section 5.2.

Figure 5.12 and Figure 5.13 show the networks performance at 128 [bytes/s] per node for a range from 1 to 10 nodes, a load considered to be a medium data transmission rate load for the packet scheduling system.

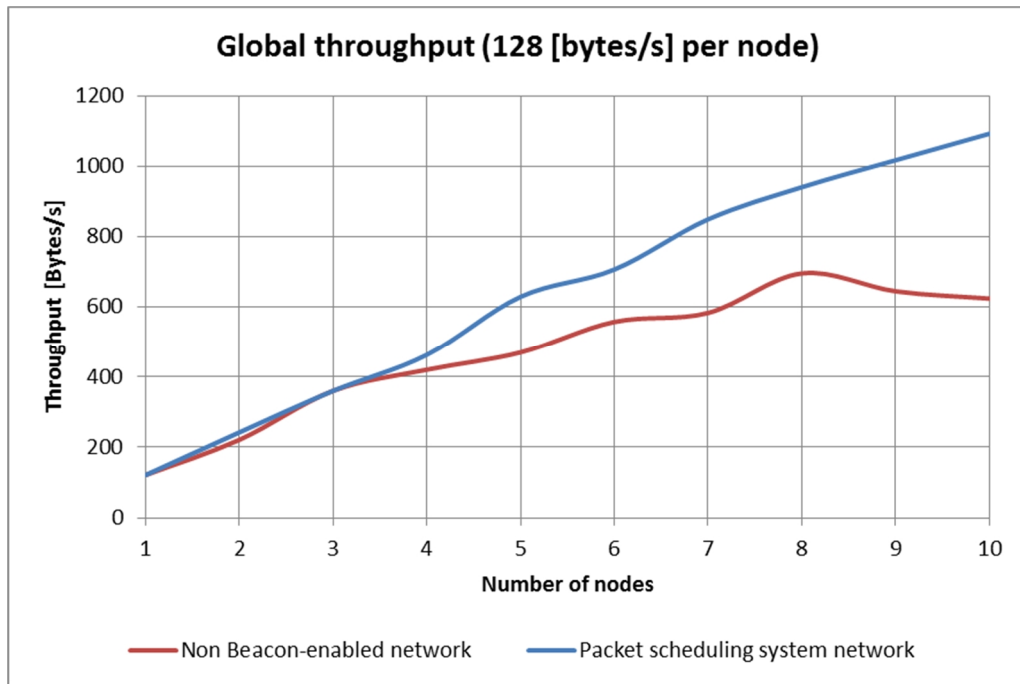


Figure 5.12: Global throughput comparison at 128 [bytes/s] per node

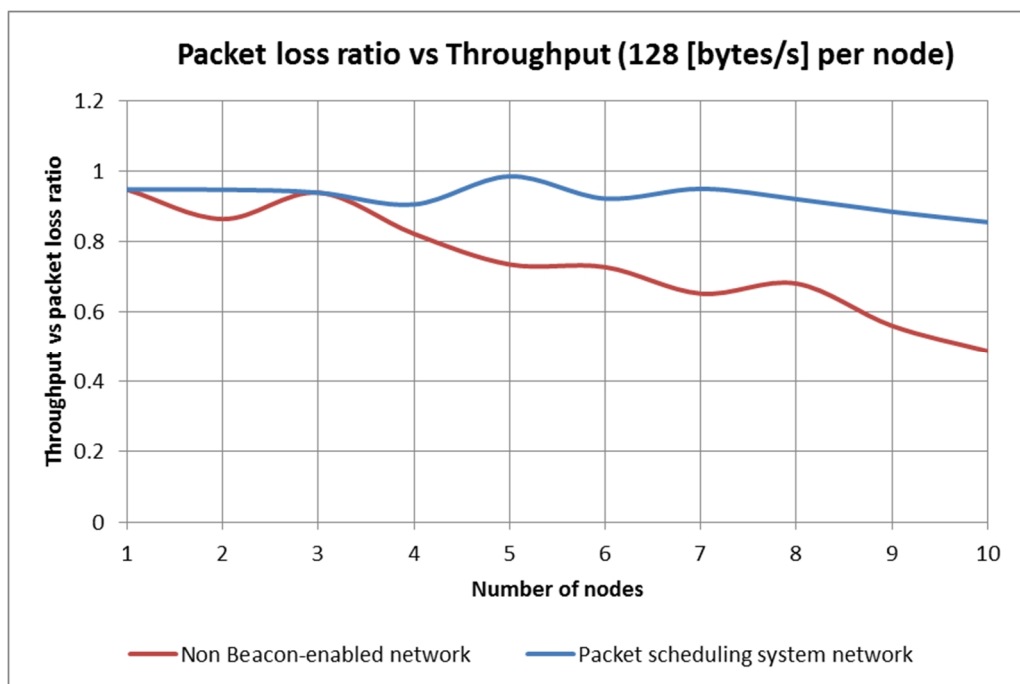


Figure 5.13: Performance comparison at 128 [bytes/s] per node

The same behaviour as in the first performance comparison tests can be observed: the performance of the packet scheduling system is better than the network using a non beacon-enabled firmware. Even if the last one shows an improved performance with a lower data transmission load, the network using the system developed for this thesis work shows a higher throughput due to the low amount of package loss that the scheduled GTSs mechanism offers.

Figure 5.14 shows the way the algorithm selection mechanism of the packet scheduling system, selected the scheduling decision algorithms during this test.

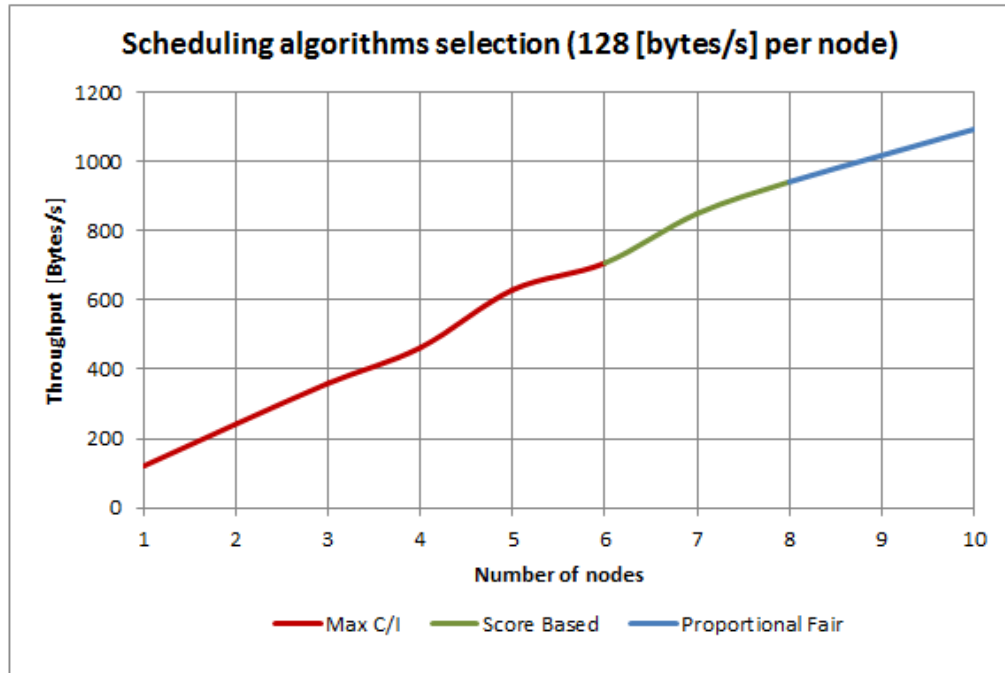


Figure 5.14: Scheduling algorithm selection at 128 [bytes/s] per node

In the same way as in the previous network performance comparison, the scheduling selection mechanism is behaving as intended.

During these tests, where a medium data load was transmitted to the coordinator, the scheduling selection is choosing the Max C/I scheduler for low throughputs, the Score Based for medium throughputs and the Proportional Fair for high ones. Again, this shows that both, the network monitoring module and the scheduling selection algorithm are again working well using different experimental parameters. Notice that the schedulers are selected with a delay of one node compared with the high transmission rate load test.

A final performance comparison tests was held with a low data transmission rate load of 64 [bytes/s] per node. Figure 5.15 and Figure 5.16 show the performance comparison between the networks using different version of the firmware.

Once again, the performance of the packet scheduling system is better than the one from the network using a non beacon-enabled network.

In Figure 5.15, the non beacon-enabled network shows a better performance than with higher data load rates but it is not as good as the one of the packet scheduling system. Figure 5.16 shows that the data loss of the packet scheduling system is almost reduced to 0%.

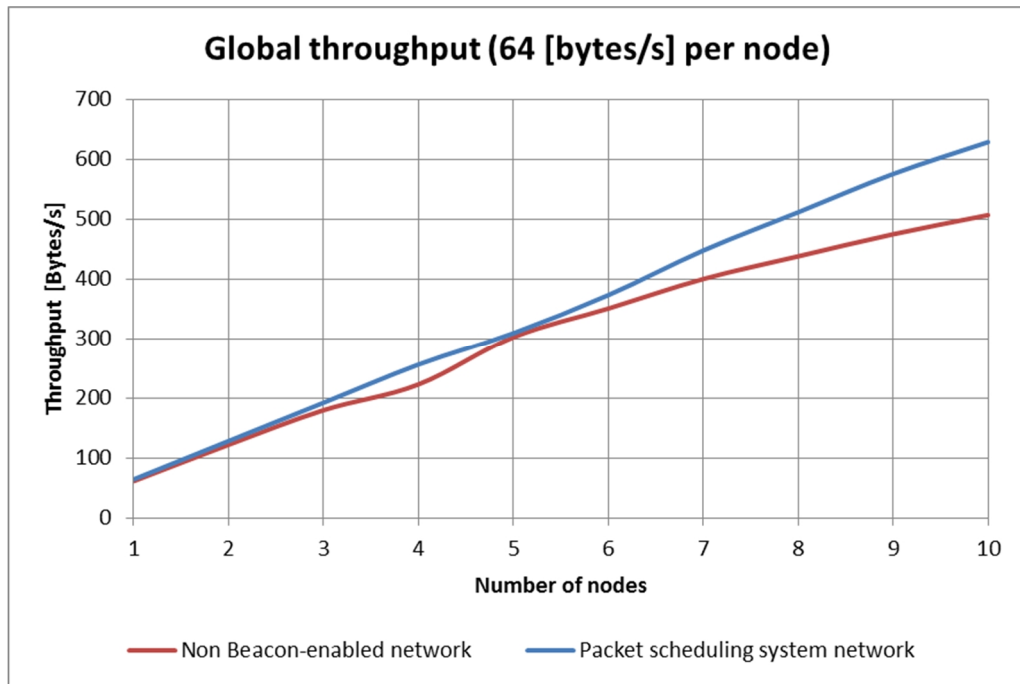


Figure 5.15: Global throughput comparison at 64 [bytes/s] per node

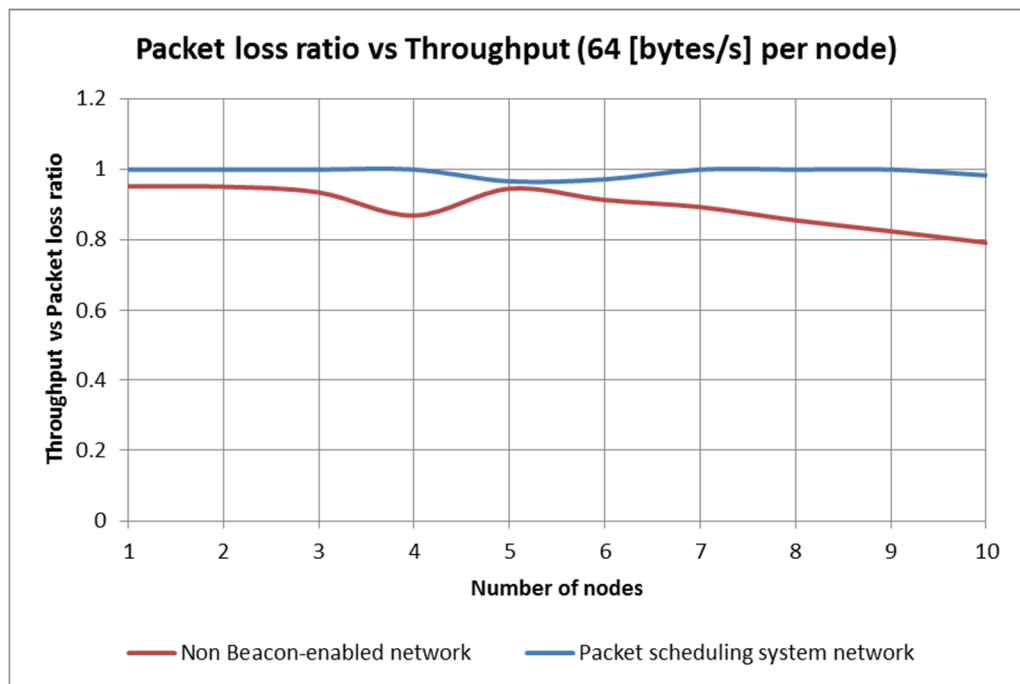


Figure 5.16: Performance comparison at 64 [bytes/s] per node

Figure 5.17 shows the way the algorithm selection mechanism, embedded on the firmware developed, selected the scheduling decision algorithms during this test. It can be seen that the Proportional Fair scheduling algorithm was never selected due to the low data rate load on the system. This was expected as the scheduling algorithm's performance test, showed in Figure 5.7 and Figure 5.8, had the same behaviour in which, due to the low data transmission load, the algorithm selection mechanism did not use the Proportional Fair scheduling algorithm.

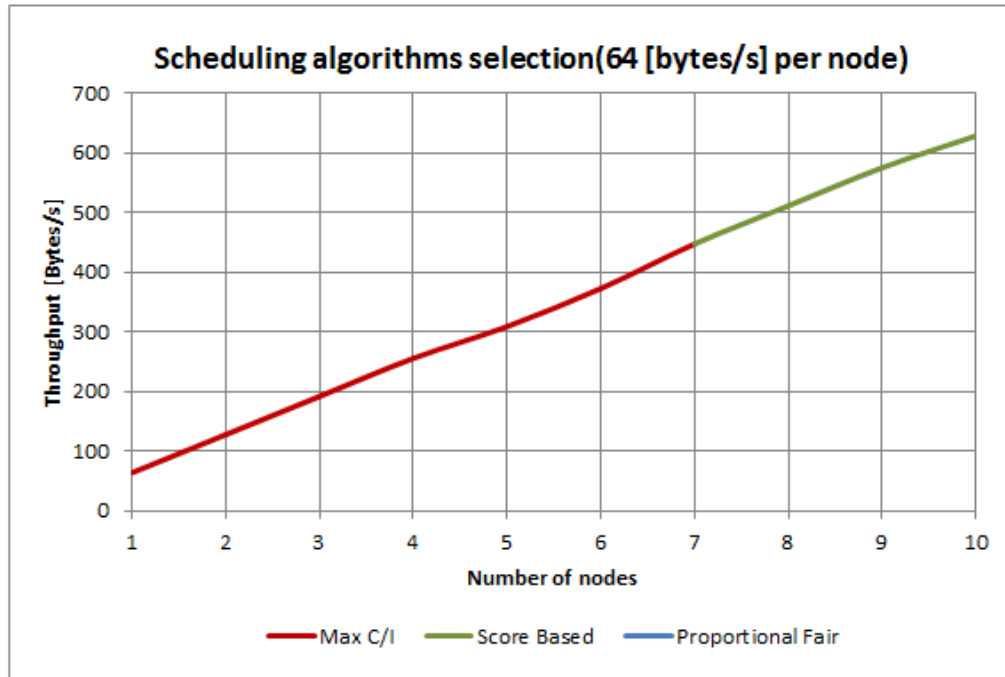


Figure 5.17: Scheduling algorithm selection at 64 [bytes/s] per node

After reviewing the experimental results given by the networks' performance comparison results it can be concluded that:

- The network using the real-time packet scheduling system for a 6LoWPAN industrial application has a better performance, with higher global throughputs and less packet losses, than a network using a non beacon-enabled firmware, especially for high speeds where the scheduling system is improving its performance significantly increasing the global throughput by reducing the packet loss.
- The scheduling algorithm selection mechanism is effectively working and it is using the ideal scheduler for each network data rate load helped by the network monitoring algorithm embedded in the packet scheduling system.

6. CONCLUSIONS

This chapter presents the final conclusions of the implementation developed for this thesis work. It contains the discussion of the results of the tests described and developed to evaluate and compare the performance of the real-time packet scheduling system for a 6LoWPAN industrial application with the previously existing non beacon-enabled firmware for the FAST WSN.

Finally, this work proposes the direction that further research with a similar approach may take from this point onwards.

6.1. Conclusions of the implementation

Different tests were performed in order to evaluate different aspects of the implementation described by this document. These tests utilize different methodologies and parameters to test specific features of the packet scheduling system developed for this thesis work.

The outcome of the tests was previously presented during Chapter 5. The discussion of the results is done during this chapter.

6.1.1. Battery consumption results

The battery consumption tests results showed that a WSN using the packet scheduling system has a considerably larger battery-life compared with a network using the previously existing firmware that used a non beacon-enabled mode at its MAC layer. This is important for a LoWPAN where the limited power availability of the devices is one of the most important factors to be considered.

The use of superframes and scheduled GTSs assignment reduces the sensor's transmission time by limiting the sensor's data packets broadcasting to their corresponding GTSs. This impacts directly to the energy consumption as the data transmission is the most energy demanding task that a sensor node can perform.

With a non beacon-enabled channel access mechanism, whenever a packet collision exists, the nodes need to re-transmit its data. Even if there are not any data losses, this could mean that there was a constant re-transmission of packets. In both cases, the re-transmissions of data have a deep impact on the battery-life as it increases the energy consumed by the sensor.

On the other hand, even if, with the beacon-enabled channel access mechanism, the processing power consumed is higher due to the timers handling and extra superframe parsing tasks required, as the theoretical review showed, the power consumed by processing and even memory writing processes is minimal compared to the one required by the transmission. This

coincides with the data presented by Figure 2.4 that compares the energy consumed by different sensor nodes' tasks.

6.1.2. Scheduling algorithms' performance results

After reviewing the scheduling algorithms' performance tests results, it is possible to state that the behaviour of the scheduling decision algorithms was the expected. Each of the algorithms is designed to assure the maximum throughput at the traffic condition that they are optimized for.

The Max C/I scheduler gives priority to nodes with the maximum instantaneous data rate in the current TTI. This scheduler is designed to maintain the highest global throughput as it always schedules users with the highest instantaneous data rate. Based on the results of the tests, it can be seen that for the packet scheduling system that uses a beacon-enabled mode to manage the channel access, this scheme is very unfair because not all the nodes of a network can get all the resources and some nodes of the network will be starved, making it ideal for low traffic with a low number of nodes transmitting critical data to the coordinator, but for higher transmission traffic, the package loss of the system increases, reducing its throughput. This decreases its performance dramatically making it inefficient.

The Score Based scheduling algorithm is designed to ensure that all simultaneously queued users receive the same average throughput. The tests results show that this scheduler is ideal for medium traffic conditions as, in every TTI, the priority to have channel access is given to the user with the highest average throughput or historical data rate.

As the theoretical review showed, the Proportional Fair scheduling can distribute the channel access equally even under asymmetric networks. The results of the test showed that the scheduler reacts slowly and inefficiently with a low traffic load but it guarantees maximum throughput and lower package loss with a high transmission load, making it ideal for a network with a high wireless traffic.

It is important to mention that each of the schedulers required an extensive tuning process in order to adjust its algorithm and guarantee that they were giving the right scores to the nodes according to their data rate condition. This was an iterative process that consumed a considerable amount of time before its results were the expected.

6.1.3. Networks' performance comparison results

The networks' performance comparison tests results are clear: the real-time packet scheduling system proposed by this work has a better performance than the non beacon-enabled firmware.

The networks' performance limits tests showed that the network using the packet scheduling system increases its maximum global throughput by 25% approximately.

After analysing the tests results of the performance comparison, it can be seen that the scheduled assignation of GTSS, part of the superframe of the IEEE 802.15.4 beacon-enabled mode, implemented on the packet scheduling system improves the global throughput and re-

duces the packet loss caused by the message fragmentation as it assures that all the nodes' transmissions can be "listened" and processed by the coordinator at their assigned GTSs.

The scheduling system's performance was better during all of the three tests implemented to evaluate the behaviour of the FAST WSN using different firmware versions. Being at higher transmission loads where the newest firmware presents the best behaviour compared to the oldest one.

At low transmission loads, the non beacon-enabled network presents an acceptable throughput and a relatively low packet loss but the scheduling system shows a better performance where the throughput is higher due to the almost null data loss.

By analysing the Figure 5.11, Figure 5.14 and Figure 5.17 where the scheduling algorithm selection is showed at different data rate loads, it is possible to state that the scheduling manager, part of the scheduling module shown in Figure 4.9, is working as intended. The traffic evaluation algorithm is monitoring and classifying the traffic load in an accurate way which at the same time, causes that the scheduler selection algorithm choses the scheduler that best fits the traffic condition.

With the previous results discussion it is valid to state that the firmware developed is capable of monitoring the wireless traffic, managing the communication between wireless sensor nodes, and scheduling the channel access of an industrial 6LoWPAN WSN resulting in a more reliable, robust, time-predictable and scalable network with higher data transmission efficiency, a better usage of bandwidth and with less package loss.

6.2. General conclusions

Knowing that it was possible to improve the data transmission and channel access mechanisms of the 6LoWPAN's protocol lower layers, this thesis work focused on the adaptation of scheduling techniques used by the HSDPA technology to the GTSs scheduled allocation of the IEEE 802.15.4 beacon-enabled mode, giving as a result a more reliable, efficient and scalable network for industrial applications.

An industrial network using the real-time packet scheduling system:

- Is more reliable: first, because the data losses due to packets collisions are reduced, assuring that the final users will get a considerably higher amount of data coming from the sensors in real-time. Second, because critical data transmitted by highly important sensors, such as alarm messages, will not be discarded as the coordinator has a scheduled channel access, designed to listen and process all the incoming transmissions.
- Is more efficient: the total throughput of the network was increased due to the packet loss reduction. Moreover, a more energy-efficient system was achieved. The number of successful transmissions from the sensors to the LoWPAN coordinator was increased, reducing the number of re-transmission attempts and therefor, a lower energy consumption. This brings an extended battery life-time, a very important factor for a network of low-power embedded devices.

- Is more scalable: thanks to the transmission scheduling and the increased throughput capacity, the LoWPAN coordinator is able to process the transmission from a higher number of nodes, provided that its limits are not surpassed.

The result of this work is a successful attempt at integrating concepts from different wireless technologies showing that it is possible to adapt them and obtain a design that shows a good performance.

Even if this was a system that was designed for a specific network in a factory floor with specific applications consuming data from the sensors, the results show that the real-time packet scheduling system can be adapted to other industrial applications where the data transmitted is critical for the production plant. The system can be used not only for monitoring but for control as its enhanced reliability qualifies for such use.

6.3. Future work

This thesis work represents one of the first attempts to adapt HSDPA fast and slow scheduling techniques to the 6LoWPAN low-power protocol. However, in order to exploit all the advantages that this integration can bring, it is necessary to adapt and test other scheduling decision algorithms used by this technology, such as the Fair Throughput or the Fast Fair Throughput which algorithms use more complex equations for the nodes classification and scoring.

Regarding the implementation over the FAST WSN, the obvious next step is to deploy this scheduling system on the wireless accelerometers/gyroscopes and on the digital-analogue I/O sensors. Due to the hardware limitations of the accelerometers/gyroscopes and the software constraints of the I/O sensors, it was not possible to implement and test this scheduling system on those devices.

Further performance analysis can be done regarding the behaviour of the system with different messages sizes from the sensors. It is known that, due to the way that 6LoWPAN deals with the message fragmentation and re-assembly at its adaptation layer, the bigger the message size, the highest the probability of losing the packet will be. Even if this scheduling system is designed to deal with a variable size of messages, this particular behaviour and performance is not tested and analysed by this thesis work.

7. REFERENCES

- [1] N. Kushalnagar, G. Montenegro and C. Shumacher, "IETF RFC4919: IPv6 over Low-Power Wierless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement and Goals," 2007.
- [2] G. Montenegro, N. Kushalnagar and D. Hui, "IETF RFC4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks," 2007.
- [3] I. S. Association, "IEEE 802.15.4-2006".
- [4] Z. Shelby, J. Hui, P. Thubert, S. Chakrabarti and E. Nordmark, "Neighbor Discovery for 6LoWPAN," 2009.
- [5] Z. Shelby and C. Bormann, 6LoWPAN: The Wireless Embedded Internet, West Sussex vols. John Wiley & Sons Ltd., 2009.
- [6] P. Thurbert and J. Hui, "LoWPAN fragment Forwarding and Recovery," 2010.
- [7] V. Roca and B. Adamson, "IETF RFC6968: FCAST: Object delivery for the Asynchronous Layered Coding and NACK-Oriented Reliable Multicast protocols," 2013.
- [8] X. Li, C. J. Bleakley and W. Bober, "Enhanced Beacon-Enabled Mode for improved IEEE 802.15.4 low data rate performance," 2011.
- [9] A. Pantelidou and A. Ephremides, "A Review of Scheduling Problem in Wirelss Networks," 2010.
- [10] "Wikipedia: Scheduling," [Online]. Available: <http://en.wikipedia.org/wiki/Scheduling>. [Accessed June 2013].
- [11] "eSONIA EU Research Project," [Online]. Available: <http://www.esonia.eu/>.
- [12] "ARTEMIS Joint Undertaking," [Online]. Available: <http://www.artemis-ju.eu/>.
- [13] A. Willig, K. Matheus and A. Wolisz, "Wireless Tehcnology in Industrial Networks," 2005.
- [14] D. Christin, P. Morge and M. Hollick, "Survey on Wireless Sesnor Network Technologies for Industrial Automation: The Security and Quality of Service Perspectives," *Future internet: Special Issue Security for Next Generation Wireless and Decentralized Systems*, vol. II, 2010.
- [15] J. Janhunen, "Quality Inspection of a production line with a smart camera", Tampere University of Technology, 2012.
- [16] B. Ramis Ferrer, "An ontological approach for modelling configuration of factory-wide data integration systems based on IEC-61499", Tampere University of Technology, 2013.

- [17] A. Vidales Ramos, "Embedded service oriented monitoring, diagnostics and control: Towards the asset-aware and self-recovery factory", Tampere University of Technology, 2011.
- [18] H. Pájaro, "A 3d Real-Time Monitoring System for a Production Line", Tampere University of Technology, 2012.
- [19] Z. Bin, "Design and Implementation of an InfoStore for Key Performance Indicators", Tampere University of Technology, 2012.
- [20] T. Sedlacek, "A Real-Time Positioning System of Manufacturing Carriers Deploying Wireless MEMS Accelerometers and Gyroscopes", Tampere University of Technology, 2012.
- [21] J. N. Martinez Valdez, "Wireless Technologies for Indoor Asset Positioning", Tampere University of Technology, 2011.
- [22] B. Bennet, M. Boddy, F. Doyle, M. Jamshidi and T. Ogunnaike, "Reports on Industrial Controls, Information Processing,," *Assessment Study on Sensors and Automation in the Industries of the Future*, 2004.
- [23] IEEE, "IEEE Standard for a Smart Transducer Interface for Sensors and Actuators – Transducers to Radio Frequency Identification (RFID) Systems Communication Protocols and Transducer Electronic Data Sheet Formats," 2010.
- [24] E. Knapp, *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and other Industrial Control Systems*, 2011.
- [25] F. A., P. Ferrari, D. Marioli, E. Sinsinni and A. Taroni, "Wired and wireless sensor networks for industrial applications," *Microelectronics Journal*, 2009.
- [26] U. D. o. E. O. o. E. a. R. Energy, "Industrial Wireless Technology for the 21st. Century," 2002.
- [27] "RUNES EU Research Project," [Online]. Available: ist-runes.org.
- [28] "ON World," [Online]. Available: <http://www.onworld.com/>.
- [29] C. Burati, M. Martalo', R. Verdone and F. G., *Sensor Networks with IEEE 802.15.4 Systems: Distributed Processing, MAC, and Connectivity (Signals and Communication Technology)*, Springer, 2011.
- [30] X. Lu, "Design of Energy Efficient and Secure Wireless Sensor Networks," 2008.
- [31] V. C. Gungor and H. G., "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches," 2009.
- [32] N. Tsiftes and D. A., "A Database in Every Sensor," 2011.
- [33] M. A. Matin and I. M. M., "Overview of Wireless Sensor Network," 2023.
- [34] J. Zheng and J. A., *Wireless Sensor Network: A Networking Perspective*, Wiley, 2009.
- [35] "AdVantech: A description of wireless standards," [Online]. Available: http://www.advantech.eu/it/edm/2011_WSN/index_flash.htm.

- [36] N. Hunn, *Essentials of Short-Range Wireless*, The Cambridge Wireless Essentials Series, 2010.
- [37] L. Frenzel, "What's The Difference Between IEEE 802.15.4 And ZigBee Wireless?," 22 March 2013. [Online]. Available: <http://electronicdesign.com/what-s-difference-between/what-s-difference-between-ieee-802154-and-zigbee-wireless>.
- [38] I. S. Asociation, "IEEE 802.15 Wireless personal area networks".
- [39] A. Conta, D. S. and M. Gupta, "IETF RFC4443: Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification," 2006.
- [40] "Farlex Online Dictionary: user network," [Online]. Available: <http://www.thefreedictionary.com/user+network>.
- [41] "Wikipedia: Scheduling," [Online]. Available: <http://en.wikipedia.org/wiki/Scheduling>.
- [42] V. Venkataramanan, "On Wireless Scheduling Algorithms for Minimizing the Queue-Overflow Probability," 2010.
- [43] B. Furht and S. Ahson, *HSDPA/HSUPA Handbook*, Boca Raton, FL, U.S.A.: CRC Press, 2011.
- [44] G. Barriac and H. J., "Introducing delay sensitivity into the proportional fair algorithm for CDMA downlink scheduling," *IEEE seventh symp. Spread Spectrum Techniques and Applications*, 2002.
- [45] Y. Ding and S. Hong, "CFP Scheduling for Real-Time Service and Energy Efficiency in the Industrial Applications of IEEE 802.15.4," *Journal of Communications and Networks*, vol. 15, no. 1, 2013.
- [46] J. Lee, "Performance Evaluation for IEEE 802.15.4 for low-rate wireless personal area networks", *IEEE Trans. Consum. Electron*, vol. 52, no. 3, pp. 742-749, 2006.
- [47] C. Burati and R. Verdone, "Performance Analysis of IEEE 802.15.4 non beacon-enabled mode," *Trans. Veh. Technol.*, vol. 58, no. 7, pp. 3480-3493, 2009.
- [48] L. Angrisani, B. M., D. Fortin and A. Sona, "Experimental study of coexistence issues between IEEE 802.11B and IEEE 802.15.4 wireless networks," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 8, pp. 1514-1523, 2008.
- [49] H. Tseng, J. Pang and C. Kuo, "An adaptive contention control strategy for IEEE 802.5.4-based wireless sensor networks," *IEEE Trans Veh. Technol.*, vol. 58, no. 9, pp. 5164-5173, 2009.
- [50] T. Park and M. Lee, "Power saving algorithms for wireless sensor networks on IEEE 802.15.4," *IEEE Connun. Mag.*, vol. 46, no. 6, pp. 148-155, 2008.
- [51] L. Li, R. Maunder, B. Al-Hashimi and L. Hanzo, "An energy efficient error correction scheme for IEEE 802.15.4 wireless sensor networks," *IEEE Trans. Circuits Syst. II Exp. Briefs.*, vol. 57, no. 3, pp. 233-237, 2010.
- [52] A. Koubaa, M. Alves and E. Tovar, "GTS Allocation analysis in IEEE 802.15.4 for real-time wireless sensor networks," *IEEE IPDPS*, 2006.

- [53] E. Toscano and L. Lo Bello, "Multichannel superframe scheduling for IEEE 802.15.4 industrial wireless sensor networks," *IEEE Transactions of industrial informatics*, vol. 8, no. 2, 2012.
- [54] P. Raja and G. Noubir, "Static and dynamic polling mechanisms for FieldBus networks," *ACM SIGOPS Operating System Review*, vol. 27, no. 3, pp. 34-35, 1993.
- [55] H. Kim, J. Song and S. Lee, "Energy-efficient traffic scheduling in IEEE 802.15.4 for home automation network," *IEEE Trans. Consum. Electron*, vol. 53, no. 2, pp. 369-374, 2007.
- [56] Y. Huang, A. Pang and H. Hung, "An adaptive GTS allocation scheme for IEEE 802.15.4," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 5, pp. 641-651, 2008.
- [57] C. Na and Y. Yang, "An optimal GTS scheduling algorithm for time-sensitive transactions in IEEE 802.15.4 networks," *Comput. Netw.*, vol. 52, no. 13, pp. 2543-2557, 2008.
- [58] S. Yoo, P. Chong, D. Kim, Y. Doh, M. Pham, E. Choi and J. Huh, "Guaranteeing real-time services for industrial wireless sensor networks with IEEE 802.15.4," *IEEE Trans, Ind. Electron*, vol. 57, no. 11, pp. 3868-3876, 2010.
- [59] A. Koubaa, A. Cunha, M. Alves and E. Tovar, "TDDBS: A time division beacon scheduling mechanism for ZigBee cluster-tree wireless sensor networks," *Real-time Syst*, vol. 40, no. 3, pp. 321-354, 2008.
- [60] Z. You-Min, S. Mao-Heng and R. Peng, "An enhanced scheme for the IEEE 802.15.4 multi-hop network," *Proc. Int. Conf. Wireless Commun., Networking and Mobile Comput., WiCOM*, pp. 1-4, 2006.
- [61] P. Muthukumar, "Enabling low power multi-hop personal area networks," *Proc. 10th Int. Symp. Wireless Personal Multi-media Commun.*, 2007.
- [62] G. Fiore, V. Ercoli, A. Isaksoon, K. Landermäs and M. Di Benedetto, "Multihop multi-channel scheduling for wireless control in wirelessHART networks," *IEEE Conf. Emerging Technol. Factory Autom.*, pp. 1-8, 2009.
- [63] A. Saifullah, Y. Xu, C. Lu, C. Yixin, P. Soldati, H. Zhang and M. Johanson, "Real-time scheduling for wireless HART networks," *31st. IEEE real-time Syst. Symp.*, pp. 150-159, 2010.
- [64] P. Soldati, H. Zhang and M. Johanson, "Deadline-constrained transmission scheduling and data evacuation in wirelessHART networks," *ECC*, 2011.
- [65] K. Kunert, M. Jonsson and E. Uhlemann, "Exploiting time and frequency diversity in IEEE 802.15.4 industrial network for enhanced reliability and throughput," *15th. IEEE Int. Conf. Emerging Technol. Factory Autom.*, Vols. 1-9, pp. 13-16, 2010.
- [66] M. Jonsson and K. Kunert, "Towards reliable wireless industrial communication with real-time guarantees," *IEEE Trans. Ind. Informat.*, vol. 5, no. 4, pp. 429-442, 2009.

- [67] V. T. Chien, H. Chan and T. Huu, "A comparative study on operating system for Wireless Sensor Networks," in *Advanced Computer Science and Information System (ICACSIS), 2011 International Conference on*, 2011.
- [68] M. Farooq and T. Kunz, "Operating Systems for Wireless Sensor Networks: A Survey," 2011.
- [69] "Tiny OS for WSN," [Online]. Available: <http://www.tinyos.net/>.
- [70] "Wikipedia: Modular Programming," [Online]. Available: http://en.wikipedia.org/wiki/Modular_programming.
- [71] A. Dunkels, B. Grönvall and T. Voigt, "Contiki- a Lightweight and Flexible Operating System for Tiny Networked Sensors," 2004.
- [72] "Contiki: The Open Source OS for the Internet of Things," [Online]. Available: <http://www.contiki-os.org/>.
- [73] "Mantis OS: Multimod AI Networks of In-situ Sensors," [Online]. Available: <http://mantisos.org/index/tiki-index.php.html>.
- [74] "Avrora: AVR simulation and Analysis Framework," [Online]. Available: <http://mantisos.org/index/tiki-index.php.html>.
- [75] "Nano-RK: A Wireless Sensor Networking Real-Time Operating System," [Online]. Available: <http://www.nanork.org/projects/nanork/wiki>.
- [76] "Lite OS," [Online]. Available: <http://www.liteos.net/>.
- [77] V. Roca and B. Adamson, "IETF RFC6968: Object Delivery for the Asynchronous Layered Coding (ALC) and NACK-Oriented Reliable Multicast Protocols".
- [78] "Wikipedia: HSDPA," [Online]. Available: http://en.wikipedia.org/wiki/High-Speed_Downlink_Packet_Access.
- [79] H. Holma and A. Toskala, *HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobiel Communications*, John Wiley & Sons, 2006.
- [80] P. Gutierrez, *Packet Scheduling and Qualit of Service in HSDPA*, 2003.
- [81] T. Kolding, "Link and system performance aspects of proportional fair packet scheduling in WCDMA/HSDPA," *Vehicular Technol. Conf., September 2003*, 2003.
- [82] T. Kolding, F. Frederiksen and M. P.E., "Performance aspects of WCDMA systems with high speed downlink packet access (HSDPA)," *IEEE 56th Vehicular Technol. Conf*, 2002.
- [83] "Inico Technologies," [Online]. Available: <http://www.inicotech.com/>.
- [84] "Atmel AT91SAM7X512 MCU," [Online]. Available: <http://www.atmel.fi/devices/SAM7X512.aspx?tab=tools>.
- [85] S. Technologies, "RF Engine 200 series data sheet," 2012. [Online]. Available: <http://www.synapse-wireless.com/documents/products/Synapse-RF200PD1-and-RF200PF1-Datasheet.pdf>.
- [86] I. Technologies, "S1000 User Manual," [Online]. Available: <http://www.inicotech.com/doc/S1000%20User%20Manual.pdf>.

- [87] "Fluid House OY.," [Online]. Available: <http://www.fluidhouse.fi/>.
- [88] A. Resua Rey, Design of a smart carrier for asset-awareness production, Tampere University of Technology, 2012.
- [89] S. G., "IPSO Challenge 2013: Solutios for Smart Factories," 2013.
- [90] "FAST Laboratory," [Online]. Available: <http://www.tut.fi/en/fast/>.
- [91] "Wkipedia: Integrated Development Environment," [Online]. Available: http://en.wikipedia.org/wiki/Integrated_development_environment.
- [92] "Elcipse IDE," [Online]. Available: <http://www.eclipse.org/>.
- [93] "Instant Contiki IDE," [Online]. Available: <http://sourceforge.net/projects/contiki/files/Instant%20Contiki/Instant%20Contiki%202.5/>.
- [94] G. Santillán, "Inico Development Environment Set-Up" Tampere University of Technology: FAST Laboratory, 2012.
- [95] "Contiki GitHub," [Online]. Available: <https://github.com/contiki-os/contiki>.
- [96] "Contiki 2.6 Sourceforge.net," [Online]. Available: <http://contiki.sourceforge.net/docs/2.6/index.html>.