UNIVERSITÀ
DEGLI STUDI
FIRENZE

Università degli Studi di Firenze
Dipartimento di Ingegneria dell'Informazione (DINFO)
Corso di Dottorato in Ingegneria dell'Informazione
Curriculum: Reti e Sistemi Telematici

Distributed Services Management over Dynamic Networks of Things: towards a Unified SDN oriented Design

Candidate
Michele Bonanni

Supervisors
Prof. Francesco Chiti

Prof. Laura Pierucci

Prof. Tommaso Pecorella

PhD Coordinator
Prof. Fabio Schoen

ciclo XXXV, 2019-2022

Università degli Studi di Firenze, Dipartimento di Ingegneria
dell'Informazione (DINFO).

To my grandparents

Acknowledgments

Nel mio percorso di dottorato, é stato fondamentale l'aiuto e la vicinanza di tante persone che desidero ringraziare apertamente, poiché senza di loro il mio lavoro sarebbe stato sicuramente piú difficile ed incompleto.
Innanzitutto, vorrei ringraziare i miei tutor Prof. Francesco Chiti, Prof.ssa Laura Pierucci e Prof. Tommaso Pecorella per i loro preziosi consigli e la loro continua disponibilitá. La vostra guida é stata fondamentale per la mia attivitá di ricerca e grazie a voi ho potuto accrescere le mie conoscenze e competenze.
Ringrazio infinitamente Francesca con la quale ho condiviso tutte le mie gioie e preoccupazioni e senza il cui supporto non sarei stato in grado di raggiungere questo traguardo.
Non posso non ringraziare gli amici incontrati in questi anni: Dimitri, Marjia, Beatrice, Andrea, Chiara M., Chiara P., Marta e Francesco, con i quali ho condiviso giorni e notti splendide e che, con la loro presenza e parole, hanno alleggerito alcuni dei momenti piú difficili del mio percorso.
Ringrazio i colleghi del DaCoNets, e in particolar modo Adnan e Roberto, per i continui confronti e gli scambi di idee che hanno contribuito a migliorare i miei studi.
Estendo i miei ringraziamenti ad Alessandro e Fabio che mi hanno dato l'opportunitá di lavorare sulla mia ricerca all'interno della loro azienda e per il magnifico rapporto creato con loro. Ringrazio i miei colleghi Elia, Andrea e Leonardo che sono sempre disponibili per un confronto o un suggerimento e con i quali sto crescendo professionalmente.
Infine, un ringraziamento di cuore ai miei genitori, a mia sorella e ai miei vecchi amici Nicola, Valentina e Matteo che mi hanno sempre incoraggiato e che hanno rappresentato un porto sicuro nei momenti piú complessi vissuti in questi anni.

Abstract

The need of new technologies and paradigms capable to overcome the challenges of evolving communication patterns are leading to technologically advanced ecosystems in which human are increasingly immersed without the physical perception of their existence.

The Internet of Things (IoT), being the set of embedded systems interconnected with each other and with Internet represents the largest use of this evolution. An IoT domain can connect a large number of devices with different characteristics, in terms of category, functionality, manufacturer up to adopted communication protocols and nowadays its use spreads from industrial to civil environments.

In this context, the Wireless Sensor and Actuator Networks (WSANs) are the most important element of the increasing trend towards the IoT since they provide a heterogeneous internet-working and allow a more symbiotic interaction with the surrounding physical environment. However, WSANs and in general IoT systems are becoming more complex with growing scales, which leads to significant challenges that need to be addressed, such as their difficult management, the ossification of the networking protocols and the increasing energy consumption. Moreover, emerging IoT applications require more flexible network architecture to best satisfy real-time requirements.

Software Defined Networking (SDN) is an innovative paradigm designed and implemented for wired networks to facilitate their management by overcoming the constraints of device vendors. Thanks to this approach, the handling of network flows is decoupled from data forwarding and the network behaviour can be easily manipulated bypassing the nature of transmitting devices.

Recent studies have tried to import the SDN logic into WSANs with the aim to overcome their limitations and ensure an efficient network management and a smart resources load balancing. This integration, known as Software Defined Wireless Sensor and Actuator Networks (SD-WSAN), has resulted in the adaptation of the SDN protocols to the physical features of the WSAN and has highlighted drawbacks in terms of network availability, energy consumption and robustness in mobility scenarios.

The work presented in this dissertation shows another possible SDN adoption in the WSANs and more in general in the IoT ecosystem, rethinking the integration logic of the two paradigms by surpassing the limitations of the existing approaches. The direct management of the SDN Controller has been

used to dynamically change the routing policy of the WSAN by minimizing the energy consumption of the sensor nodes and optimizing the use of underutilized network resources. Focusing on environments where devices are battery powered and installed in harsh and hardly accessible locations, the Wireless Power Transfer (WPT) technique has been considered to improve the overall networks lifetime.

The resulting SD-WSAN architecture has been enriched by a new virtual level, called Power Plane and the recharging procedure has been combined with the management of the SDN Controller to maximise the services availability and system durability of clustered wireless IoT domains. In this scenario, the benefits of advanced energy-saved techniques and smart network strategies are often limited by Web of Things (WoT) communication protocols which are not optimized for wireless sensor networks (e.g. Message Queue Telemetry Transport (MQTT)). In order to overcome this open issue, the proposed model has been used to modify the default behaviour of existing application protocols and enable innovative traffic flows.

Finally, the study moved from static to dynamic environments and the Internet of Vehicles (IoV) sub-domain has been analysed. IoV allows vehicles to share information about their surroundings, traffic condition, and other relevant data, enabling them to make more informed decisions and improving safety, efficiency and convenience. However, traditional Cloud approaches introduce latencies that make the development of innovative real-time applications infeasible. To this purpose, SDN has been applied in combination with the Fog Computing (FC) paradigm to properly accommodate users mobility and to support complex mobile services with stringent time-space constraints.

Contents

List of Figures

List of Tables

List of Abbreviations

ACO Ant Colony Optimisation

ADC Analog-to-Digital Converter

AP SDN Application Plane

ARP Address Resolution Protocol

BS Base Station

BSS Business Support Systems

CC Cloud Computing

CH Cluster Head

CN Core Network

CoAP Constrained Application Protocol

CP SDN Control Plane

CTP collection Tree Protocol

DBSBA Dual Band Switched Beam Antenna

DM-MQTT Direct Multicast-MQTT

DP SDN Data Plane

DRM Dynamic Routing Module

EC Edge Computing

EH Energy Harvesting

EHN Equivalent Hop Number

EMS Element Management System

ES Energy Supplier

FC Fog Computing

FFD Full-Functional Devices

FJAPSO Fork and Join Adaptive Particle Swarm Optimisation

HTTP Hypertext Transfer Protocol

IIoT Industrial Internet of Things

IoMT Internet of Mobile Things

IoT Internet of Things

IoV Internet of Vehicles

IT Information Technology

IWSAN Industrial Wireless Sensor and Actuator Network

LEACH Low-Energy Adaptive Clustering Hierarchy

LO Lion Optimization

MAC Medium Access Control layer

MEC Mobile Edge Computing

MME Mobile and Management Entity

MQTT Message Queue Telemetry Transport

MQTT-SN Message Queue Telemetry Transport for Sensor Network

NAT Network Address Translation

NB SDN North-Bound interface

ND Network Discovery

NFV Network Function Virtualization

NFV-MANO NFV Management and Orchestration

NVFI NFV Infrastructure

OPC UA Open Platform Communications Unified Architecture

OPNFV Open Platform for NFV

OSS Operations Support Systems

OT Operational Technology

PHY Physical layer

PLR Packet Loss Ratio

PSO particle swarm optimisation

PTA Power Transfer Application

PTM Power Transfer Manager

QoE Quality of Experience

QoS Quality of Service

RAN Radio Access Networks

REST REpresentational State Transfer

RFD Reduced-Functional Devices

RN Rechargeable Node

RPL IPv6 Routing Protocol for Low-Power and Lossy Networks

RTT Round Trip Time

SADFIR Application-aware Distributed adaptive Flow Iterative Reconfiguring

SB SDN South-Bound interface

SBA Switched Beam Antenna

SD-WSAN Software Defined Wireless Sensor and Actuator Network

SDCSN Software Defined Clustered Sensor Networks

SDEAR Software Defined Energy Aware Routing

SDMA Space Division Multiple Access

SDN Software Defined Networking

SEANET Software Defined Energy Harvesting IoT

SGW Serving Gateway

SOHS Self Organizing and Healing SDWSAN

SPIN Sensor Protocol for Information via Negotiation

TDMA Time Division Multiple Access

TTL Time To Live

V2I Vehicle to Infrastructure

V2P Vehicle to Pedestrian

V2V Vehicle to Vehicle

V2X Vehicle to Everything

VANET Vehicular Ad-hoc Networks

VIM Virtualized Infrastructure Manager

VM Virtual Machine

VNF Virtual Network Function

WoIT Web of Industrial Things

WoT Web of Things

WPT Wireless Power Transfer

WSAN Wireless Sensor and Actuator Network

WSN Wireless Sensor Network

Chapter 1

Introduction

The technological advancements of the last two decades and the need of a digital transition have brought considerable changes in the modern world and society [90], [106]. One of the most important trends which has accelerated this transformation is represented by the Internet of Things (IoT). Several IoT definitions have been proposed in the recent literature, as *network* of items-each embedded with sensors-which are connected to the Internet." or an Internet that can concurrently operate among TCP/IP and non-TCP/IP protocols, with Things that are objects identified by unique addresses", to name a few. The mean idea behind IoT is to connect production, medical, automotive, or transportation devices (i.e., the things) with Information Technology (IT) and business-critical information systems in order to provide unprecedented added values in terms of innovation, efficiency, and quality [4]. According to the forecasts of research institutes specialized in this sector, the IoT market, which was already worth 170 billion in 2017, will reach 561 billion by 2023, with a number of connected devices that will be around 125 billion in 2030. This continuous growth is mainly due to the great heterogeneity and flexibility of the IoT itself which can be potentially used in all application areas and sectors [95]. Some of the application fields which the IoT is embracing are:

- Smart Agriculture: Monitoring of micro-climatic parameters to improve the quality of the food products and reduce the resources consumption and the environmental impact.

- Smart Car: Promoting the communication among human, car and

surrounding infrastructure for the incident prevention and detection.

- Smart City: Monitoring and management of the elements of the city (e.g. public transportation, illumination, parking.) in order to improve the liveability.

- Smart Home: Solutions for the automatic and remote management of the home systems and plants with the aim of improving the comfort and reducing the energy consumption.

- Smart Factory and Industry 4.0: Intelligent machinery and improved Machine to Machine and Machine to Humans connections to optimize the production lines, minimize the waste of resourced and increase the safety of workers.

Industries 4.0 are generally heterogeneous environments in which hundreds of smart devices, with different characteristics, cooperate with each other to perform complex tasks without manual assistance [105]. The products within these Intelligent Industries will no longer be passive objects that will undergo management decisions but will participate and become essential for the decision-making process and for the optimization of their production. The main purpose of these industries is to integrate Information Technology in industrial plants and implement customized work processes to increase productivity, meet customer needs, improve industrial technologies, and increase energy efficiency [36].

In this vision, Wireless Sensor and Actuator Networks (WSANs) has been viewed as the enablers of IoT paradigm, since they provide a heterogeneous internet-working and allow a more constructive interaction with the surrounding physical environment, both gathering run-time information from it and executing micro-tasks to modify it [111], [125]. However, in order for the IoT to be implemented realistically, it is essential that devices have enough energy to function properly since they often are placed in harsh locations with difficult management [26]. This is often achieved by giving devices a battery pack, but this solution has limitations such as the need for periodic battery replacement and the potential for performance, durability, and cost trade-offs. To overcome these issues, it may be necessary to consider alternative strategies for meeting energy needs in a factory setting where wireless devices are heavily used and may be difficult to maintain. In principle, two possible methodologies for providing energy to devices can be

considered: Energy Harvesting (EH) and Wireless Power Transfer (WPT) [129]. EH converts every possible energy source available in a specific environment into power to supply WSAN devices; usually, energy-depleted nodes can get energy from a renewable energy sources. Besides a mobile charger is usually employed to provide the energy replenishment through energy downlink, while it can collect the surplus energy from energy-rich nodes and transfer it to energy-deficient nodes. In industrial contexts, it may be the machinery itself that provides *intrinsic* energy sources, such as heat and vibration. An EH system consists of two processes: (1) energy harvesting and (2) data transmission. Since data communications modify the nodes energy, an optimal control of data and related energy flows represent a critical issue for effective communications [107]. On the other hand, WPT enables the energy supply via either a direct device activation or the device battery recharge, respectively. WPT can be attractive for various devices, where the energy can be cordlessly harnessed in air, e.g., from electromagnetic field, microwave, and laser, to charge the battery and prolong the lifetime of WSAN [34]. WPT approach relies on an infrastructure dedicated to the energy transfer embodied in the communications topology, consequently requesting specific routing strategies to optimise both aspects.

To create a general purpose IoT platform to easily set up a distributed application, it has been investigated the adoption of the so called Web of Things (WoT) approach [9], [44]. The information exchange occurring as an WoT domain interacts with the physical world is characterised by specific features, mainly dictated by the constrained devices which suffer of limited processing power, network bandwidth, and intermittent connectivity. In addition, communications are sporadic and often event-triggered updates, with continuous multiple data flows. To address these limitations, the Constrained Application Protocol (CoAP) has been firstly proposed to adapt HTTP for constrained devices and to enable more advanced interaction patterns w.r.t. HTTP, e.g., the support for multicast [85]. Another promising candidate is represented by the MQTT protocol designed for monitoring applications. It is based on the publish-and-subscribe paradigm, by which Publishers (e.g., sensors) transmit data messages to a Broker, which in turn delivers such messages to interested entities, called Subscribers. This flexible approach places complexity in the Broker. Furthermore, MQTT defines a lightweight header format and requires a small code footprint. Recently, the industrial Open Platform Communications Unified Architecture (OPCUA) adopted the publish/subscriber model in its release 14, and the MQTT messages are used for

the broker-based case. Moreover, a version of this protocol, called MQTT for Sensor Networks (MQTT-SN), has been specifically designed to face the constraints of a typical IoT domain with suitable architecture, interfaces and components. However, in case of WSANs, these communication protocols present constraints which limit the potential of the network and degrade the overall performance.

Alongside the IoT, another topic which has captured the interest of the industrial and academic communities over the past few years is the Software Defined Networking paradigm. SDN is a network architecture approach that enables the network to be intelligently and centrally controlled, or *programmed*, using software applications [69]. SDN enables the programming of network behaviour in a centrally controlled manner through software applications using open APIs. This approach separates the two levels of the network devices, moving the control plane, which determines where to send traffic to the software, and leaving the data plane, that actually forwards the traffic, into the hardware [35]. By opening up traditionally closed network platforms and implementing a common control layer, operators can manage the entire network and its devices consistently, regardless of the complexity of the underlying network technology [12].

The SDN principle can be applied to WSANs to encompass their constraints and improve the network flexibility [100]. The resulting networks, known as Software Defined Wireless Sensor and Actuator Networks (SD-WSAN) ensure an efficient management and enable new types of data streams, identified as right and left streams, that overcome the traditional 802.15.4 topologies [19]. Previous methodologies have led to the incorporation of SDN protocols to match the characteristics of WSANs, however, they have exposed limitations in terms of availability, energy usage and stability in mobile environments especially for large scale networks. Therefore, for an effective integration the SDN basis must be rethought and its main protocol, OpenFlow, modified according to the features of the sensor nodes [51].

Focusing on mobility scenarios, the Internet of Mobile Things (IoMT) is helping to drive the development of smart cities, connected healthcare, and other applications that rely on the integration of mobile devices and internet connectivity with the physical world. This scenario is often characterized by fast topological changes and instantaneous data traffic peak which must be considered in the network design [72]. Moreover, many IoT applications require sufficiently low latency in data transfer and processing to guarantee reliable and punctual computing services. This is more evident in the In-

ternet of Vehicles (IoV) subdomain where safety critical applications need response time of less than 10 milliseconds. Thanks to ubiquitous availability as well as capability to offer high data rate and low latency communication, cellular networks are promising to support the vehicle connectivity [87]. This has led to the standardization of Vehicle-to-Everything (V2X) communication, including Vehicle-to-Vehicle (V2V), Vehicle-to-Pedestrian (V2P), and Vehicle-to-Infrastructure/Network (V2I/N) by the 3rd Generation Partnership Project (3GPP) [2]. The LTE platform has been extended in 3GPP Rel-15 to meet the evolving requirements of the automotive industry while in 3GPP Rel-16, potential enhancements of 5G systems have been identified and evaluated to support advanced V2X applications. However, the limited data processing resources of vehicles may not always satisfy the stringent delay requirement of real-time services. One of the existing solution, known as computational offloading, is to transfer the computation tasks from user equipment to a cloud. However, this approach, suffer disadvantages in terms of communication latency and this is not suitable for safety-related services. In contrast, the fog computing paradigm distributes computational resources as close as possible to end users, allowing data to be processed in close proximity to the vehicles and roadside sensors/units. It is worth mentioning that, in addition to the fog computing, the idea of moving cloud servers to the network edge is also referred to as cloudlet and Mobile-Edge Computing (MEC) [41]. This dissertation presents an alternative approach to integrate SDN in IoT domains by proposing an hybrid stack which combines the centralized vision of the SDN Controller with the distributed nature of WSAN. The main objective is to surpass the limitations of existing methods and use the SDN Controller to dynamically change the routing policy of the WSAN, thus minimizing energy consumption and optimizing underutilized network resources. The Wireless Power Transfer (WPT) technique is also considered to improve the overall network lifetime, particularly in environments where devices are battery-powered and installed in harsh and difficult-to-reach locations. The resulting SD-WSAN architecture includes a new virtual level called the Power Plane, and the recharging process is combined with the management of the SDN Controller to maximize service availability and system durability in clustered wireless IoT domains. Additionally, the proposed model addresses the issue of communication protocols not being optimized for wireless sensor networks by modifying the default behaviour of existing application protocols and enabling innovative traffic flows. The study then

moves on mobile environments, specifically the IoV sub-domain, and SDN is combined with the Fog Computing paradigm and advanced Container Migration Techniques to accommodate users' mobility and support complex mobile services with strict time and space constraints.

1.1 Structure

This dissertation is organized in 9 chapters (including this introduction) as follows: Chapter 2 provides basic definitions and concepts related to WSAN, SDN and SDWSAN. It also introduces the Fog/MEC computing approaches and the service dissemination on the network *edges*. Chapter 3 presents the State of the Art of SDWSANs and the main contributes on energy aware routing protocols, WoT and container migration in Fog contexts. In Chapter 4 a novel declarative SDWSAN approach is given. In the provided solution, the distributed nature of WSAN is fully integrated with the centralized vision of SDN and the benefits of both approached are merged together. Chapter 5 introduces an innovative high level application to optimize the network lifetime without worsening the other performances parameters. In Chapters 6 and 7 the WPT technique is integrated with the SDN architecture. The resulting network is used to optimize the energy consumption and to improve the MQTT-SN protocol. Then, in Chapter 8 SDN is integrated with the IoV and Fog computing paradigms to satisfy strict location-aware service requirements. Finally, Chapter 9 summarizes the conclusions and proposes further works related to the presented subject.

Chapter 2

Technical Background

2.1 Software Defined Networking

Software-Defined Networking is a method of managing networks using software to centrally control and customize them. Traditional network architecture offers minimal flexibility to coordinate between fixed function network devices that must be configured manually. A single change can have a cascading effect on the network performance and has the potential to bring down the entire network [69]. The growing volumes of data traffic, complex network architecture, and increasing demands to improve network performance gets obsolete the traditional approach to network management. The static nature of the traditional network architecture model fails to meet the demands of modern business IT. Organizations require network infrastructure that allows the flexibility to scale and support dynamic computing environments based on rapidly evolving technology and business landscapes. A classic communication network architecture consists of three main components:

- Data Plane is the network architecture layer that physically handles the traffic based on the configurations supplied from the Control Plane.
- Control Plane refers to the network architecture component that defines the traffic routing and network topology.
- Management Plane takes care of the wider network configuration, monitoring and management processes across all layers of the network stack.

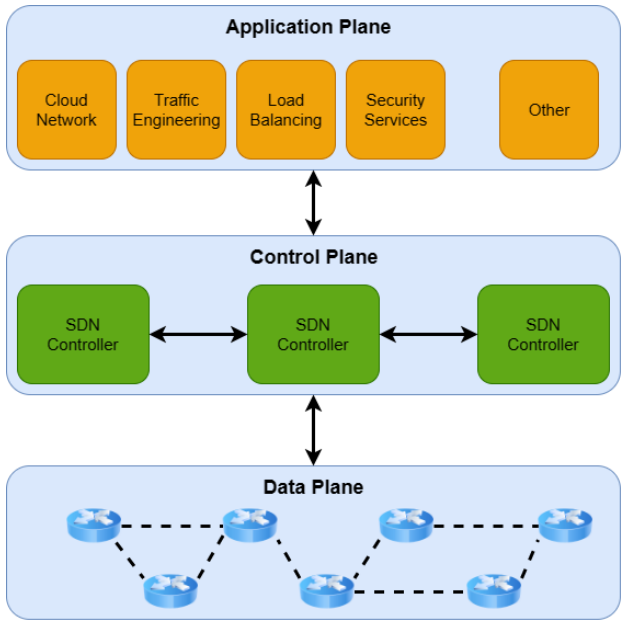


Figure 2.1: Software Defined Networking Logical Architecture.

In traditional network architecture, the control plane and data plane are integrated. Any changes to the system are dependent upon configuring physical network devices, the protocols, and software they support. In contrast, SDN decouples the Control Plane (CP) from the Data Plane (DP) and the overall network logic is centralized in one or more controllers, as shown in Fig. 2.1. Therefore, all the devices of the DP are deprived from the network algorithms and, the execution of forwarding operations is relied on the controller's directives. The logically centralized and decoupled controller operations allow organizations the enhanced agility to automate, extend, monitor, maintain, manage, extend, provision, and troubleshoot the network infrastructure [88]. The controller provides a global view of the network state and allows for more dynamic, scalable, and flexible infrastructure, leading to simplified operations and the ability to explore new business opportunities that were previously limited by network constraints. The major architectural differences between SDN and traditional network infrastructure are identified within the Control and Data layers. However, the programs within the Application layer define the new approach of data communication between

controllers and services that run over the network. The main protocol that the controller uses to communicate to the DP devices and which represents the *de-facto* standard of the lowest level of the SDN architecture is Openflow [59]. Openflow defined a unified and open model that abstracts the network management from the hardware making the higher layers of the SDN network architecture independent of the implementation of the particular vendor and the technologies used in the DP. In OpenFlow the network nodes handle incoming packets as specified in the so-called Flow Table. Each entry of the Flow Table is related to a flow and is composed by the following sections:

- Match fields: These fields specify the characteristics of a flow that a packet must match in order to be processed by the flow. For example, a match field might specify that a packet must have a specific source IP address or be part of a specific protocol.

- Priority: This field specifies the priority of a flow. If a packet matches multiple flows, the flow with the highest priority will be applied.

- Counters: These fields keep track of the number of packets and bytes that have been processed by the flow. This information can be used for monitoring and troubleshooting.

- Instructions: These fields specify the actions that should be taken when a packet matches the flow. These actions might include forwarding the packet to a specific port, modifying the packet's header, or sending the packet to a controller for further processing.

- Timeout: This field specifies how long a flow should remain active before it is automatically removed from the flow table. This can be used to automatically remove stale flows or limit the number of flows that are active at any given time.

- Cookie: This field is an opaque value that can be used to identify a flow. It can be used to modify or delete a specific flow. It is a 64-bit value that is assigned by the controller and it can be any number.

When an OpenFlow switch receives a packet, it checks it against the rules in the first table of the pipeline. If a match is found, the corresponding actions will be executed and the counters will be updated. However, if no matching

rule is found, the packet will either be sent to the controller via a secure communication channel or discarded. This process is carried out using two types of OF packets, "packet-in" and "packet-out". The packet-in packet is sent from the device to the controller and includes the entire unmatched packet, while the packet-out packet is sent by the controller in response and includes the specific forwarding rule required to handle that data stream. The rules installed in the flow table are scanned by priority. The rules with higher priority are consulted first and in the case of mismatches, the rules with lower priority are followed. If there are multiple valid rules with the same priority, the switch is free to choose which one to consider. Organizations adopt software-defined networks in reaction to the constraints of traditional infrastructures. Some of the benefits of software-defined networking include:

- Lower cost: Software-defined networks can be more cost-effective than traditional hardware-based networks because they use standard servers instead of specialized, expensive appliances. They also take up less space as multiple functions can be run on a single server, reducing the need for physical hardware and resulting in cost savings through consolidation of resources such as space, power and overall costs.

- Greater scalability and flexibility: the virtualization of the network infrastructure gives the ability to adjust networking resources as needed, rather than having to purchase additional proprietary hardware. A software-defined network provides significant flexibility and enables self-service provisioning of network resources.

- Simplified management: Adopting a software-defined network results in a more manageable infrastructure as it does not necessitate the need for highly specialized network experts to oversee its operation.

Finally, when SDN is coupled with software-defined storage and other technologies, can comprise an approach to IT infrastructure known as hyperconvergence: a software-defined approach to everything.

2.2 Wireless Sensor Networks

Sensing is the process of gathering information about physical objects or processes, including changes in state such as temperature or pressure. A device that performs this task is called a sensor. Technically, a sensor is

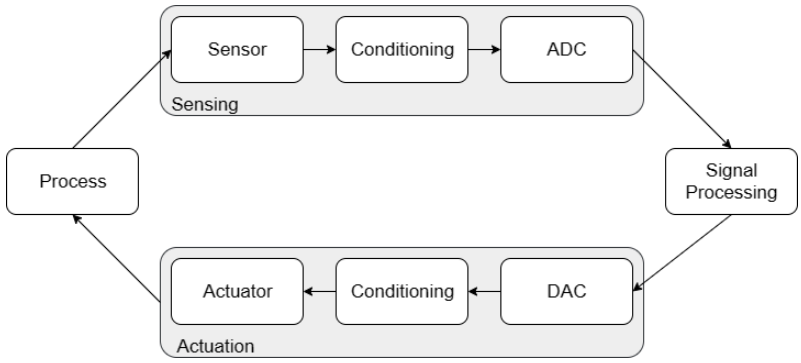


Figure 2.2: closed loop WSN node.

a device that converts physical parameters or events into signals that can be measured and analyzed. The term transducer is also commonly used to describe a device that converts energy from one form to another. A sensor is a type of transducer that converts physical energy into electrical energy, which can then be sent to a computing system or controller. The resulting electrical signals often need to be processed before use, so they go through a stage of signal conditioning, where they are amplified or attenuated and filtered to remove unwanted noise. After that, the analog signal is converted into a digital signal using an analog-to-digital converter (ADC). The signal is now in a digital form and ready for further processing, storage, or visualization [38].

Wireless sensor networks often include actuators, which allow them to directly control the physical environment. A wireless sensor and actuator network (WSAN) receives commands from a processing device (controller) and converts them into input signals for the actuator to interact with the physical process, creating a closed control loop, as shown in Fig. 2.2. While some sensors connect directly to controllers and processing stations, an increasing number of sensors use wireless communication to transmit the collected data to a central processing station. This is crucial for network applications that involve many sensor nodes, which are often located in remote and hard-to-reach places. A wireless sensor not only has a sensing component but also on-board processing, communication, and storage capabilities. When many sensors work together to monitor large physical environments, they form a wireless sensor network (WSN) [63]. Sensor nodes communicate

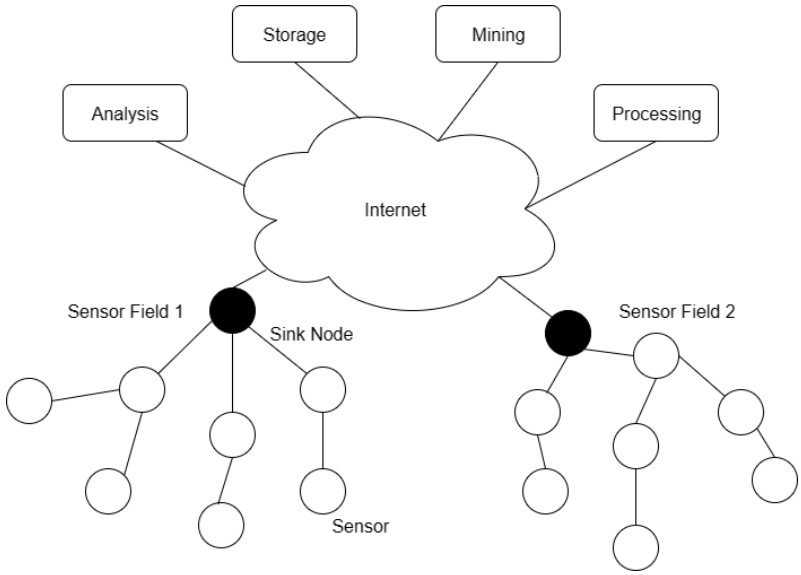


Figure 2.3: Traditional Wireless Sensor Network.

with each other and with a base station (BS) or Sink Node using wireless radios, allowing them to transmit sensor data to remote processing, visualization, analysis, and storage systems, as illustrated in Fig. 2.3. The IEEE 802.11 family of standards, first introduced in 1997, is the most widely used wireless networking technology for mobile systems together with cellular systems. It uses different frequency bands, such as the 2.4-GHz band used by IEEE 802.11b and IEEE 802.11g, and the 5-GHz frequency band used by IEEE 802.11a. This standard was commonly used in early wireless sensor networks and can still be found in current networks when high bandwidth is required. However, its high energy consumption makes it unsuitable for low-power sensor networks, leading to the development of alternative protocols that better meet the needs of low power consumption and low data rates, such as the IEEE 802.15.4 standard. When the transmission range of the sensor nodes' radios is large enough and the sensors can transmit their data directly to the base station, they can form a star topology as shown in the left of Fig. 2.4. In this topology, each sensor node communicates directly with the base station through a single hop. However, sensor networks often cover large areas and radio transmission power needs to be kept to a minimum

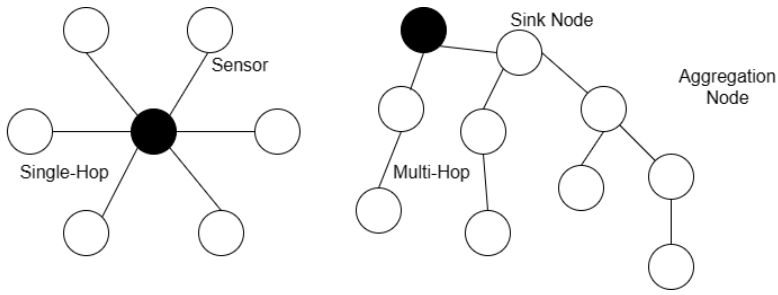


Figure 2.4: WSN Topologies.

to conserve energy, so multi-hop communication is more commonly used. In this mesh topology, sensor nodes not only capture and disseminate their own data but also act as relays for other sensor nodes, collaborating to propagate sensor data towards the base station. The routing problem, which is the task of finding a multi-hop path from a sensor node to the base station, is one of the most significant challenges and has received significant attention from researchers. When a node acts as a relay for multiple routes, it often has the chance to analyse and pre-process sensor data in the network, which can result in the elimination of redundant information or aggregation of data that may be smaller than the original data. WSNs are a type of distributed systems that face unique challenges that affect their design, leading to the development of specific protocols and algorithms. Some of the main design constraints for WSNs are:

- **Energy**: One of the main challenges in designing a wireless sensor network is that sensors have limited energy budgets, typically powered by batteries that need to be replaced or recharged when depleted. For non-rechargeable batteries, the sensor must be able to operate until its mission time is complete or the battery can be replaced. This results in energy efficiency being the primary and most important design challenge for WSNs. This requirement permeates every aspect of sensor node and network design. The energy consumption of CMOS-based processor is mainly due to switching energy and leakage energy:

$$E_{cpu} = E_{switch} + E_{leak} = C_{total} V_{cc}^2 + V_{cc} I_{leakage} \Delta_t \qquad (2.1)$$

  where C_{total} is the total capacitance switched by the computation, V_{cc}

is the supply voltage, $I_{leakage}$ is the leakage current, and Δ_t is the duration of the computation. Downsides of these approaches include the energy overheads and delays incurred by the collisions and recovery mechanisms and that sensor nodes may have to listen to the medium at all times to ensure that no transmissions will be missed. Therefore, some MAC protocols for sensor networks are contention-free, that is, access to the medium is strictly regulated, eliminating collisions and allowing sensor nodes to shut down their radios when no communications are expected. The network layer is in charge of determining the paths for communication between sensor nodes and the base station, considering factors such as the number of intermediate steps (hops), the power required for transmission, and the available energy on the relay nodes, all of which affect the energy consumption of multi-hop communication. Additionally, the aim for energy efficiency has an impact not only on network protocols, but also on the design of the operating system, middleware, security, and applications. For instance, processing data within the network can help to minimize redundant sensor data and combine multiple sensor readings, but this also involves a trade-off between computation and communication, which can be leveraged to save energy.

- **Self Management**: In a lot of cases, sensor networks are deployed in remote areas and harsh conditions, without any infrastructure or the possibility of maintenance and repair. Because of this, sensor nodes must be able to operate independently and manage themselves, they should be able to configure themselves, collaborate with other nodes, adapt to environmental changes and failures, all this without human intervention.

- **Self Organization and Self Healing**: Many wireless sensor network applications do not rely on predetermined and pre-engineered locations for each sensor node. As a result, the nodes must be able to autonomously complete various setup and configuration tasks, including connecting to other nearby sensor nodes, determining their own location, finding the best route to the sink node, and starting their sensing activities. Additionally, they should be able to detect, identify, and respond to network disruptions while maintaining network continuity.

- **Wireless Networking**: when WSNs need to cover large areas, it is more energy-efficient to divide the distance into smaller segments, which requires supporting multi-hop communication and routing. This can be challenging, especially in networks that use a duty cycle to conserve energy. During the inactive periods, sensor nodes cannot receive messages from their neighbours or serve as relays for other sensors.

- **Design Constraints**: Even though traditional computing systems are becoming increasingly powerful, the main objective of wireless sensor design is to create smaller, cheaper, and more energy-efficient devices. Due to the need to run specific applications with minimal energy consumption, sensor nodes typically have low processing speeds and limited storage capacity. These limitations make it difficult to include advanced components and affect software design. However, the lack of advanced hardware also makes it easier to design small operating systems and develop efficient protocols, algorithms, and processing techniques.

- **Security**: Wireless sensor networks often collect sensitive information. Due to their remote and unsupervised nature, sensor nodes are at a higher risk of malicious intrusion and attacks. In addition, wireless communications make it easy for adversaries to intercept sensor transmissions. While there are various techniques and solutions to protect against attacks or minimize their impact on distributed systems, many of these require significant computational, communication, and storage resources which are often not available in resource-constrained sensor nodes. As a result, sensor networks require new solutions for securely establishing and distributing keys, authenticating nodes, and ensuring secrecy.

As previously stated, a major design challenge for wireless sensor nodes is to minimize power consumption. Standards for wireless sensor networks (WSNs) include IEEE 802.15.4, Zigbee, 6LoWPAN, Thread, and NRF24L01. IEEE 802.15.4 is one of the most widely used standards for low-data-rate wireless personal area networks (LR-WPAN) and it was designed for short-range communication applications with strict energy and computational constraints. The goal of the standard is to provide a basic format that other protocols and features can be built upon. It defines the PHY and the MAC

layer of the Open System Interconnection (OSI) model for network operation. The Physical layer of IEEE 802.15.4 standard enables a variety of options for PHY in ISM bands, ranging from 868/915 MHz to 2.4 GHz. The basic structure is designed for a range of 10 meters and a data rate of 250 kbps, but to reduce power usage, even lower data transmission speeds are available. The standard uses direct sequence spread spectrum (DSSS) modulation, which is highly resistant to noise and interference and offers coding gain to improve link reliability. Standard binary phase-shift keying (BPSK) is used in the two low-speed versions, while offset-quadrature phase-shift keying (O-QPSK) is used for the higher-data-rate version. O-QPSK has a constant wave envelope, which allows for the use of more efficient non-linear power amplification techniques to minimize power consumption. With regard to channel access, 802.15.4 uses carrier sense multiple access with collision avoidance (CSMA-CA). This multiplexing approach lets multiple users or nodes access the same channel at different times without interference. Most transmissions are short packets that occur infrequently for a very low duty cycle (¡1

Zigbee is the most widely used enhancement of the IEEE 802.15.4 standard. It builds on layers 3 and 4 to add additional communication features. These enhancements include authentication with valid nodes, encryption for security, and a data routing and forwarding capability that enables mesh networking. Some of the applications that Zigbee is used for include:

- Building automation for commercial monitoring and control of facilities
- Remote control (RF4CE or RF for consumer electronics)
- Smart energy for home energy monitoring
- Health care for medical and fitness monitoring
- Home automation for control of smart homes
- Retail services for shopping related uses
- Telecom services

These upper-layer software additions provide specialized uses for Zigbee that are not present in the IEEE 802.15.4 standard.

6LoWPAN stands for IPv6 over Low-power Wireless Personal Area Networks. It is a standard protocol that enables IPv6 communication on wireless

networks consisting of low-power wireless modules. The 6LoWPAN specification includes packet compression and other optimization mechanisms to support efficient transmission of IPv6 packets on networks with limited power resources and reliability, which makes it possible to use IPv6 over low-power wireless networks. Before 6LoWPAN, various low-power wireless networks have been proposed and implemented, but currently, 6LoWPAN is considered one of the most preferred protocols for the Internet of Things (IoT) as it allows low-power wireless networks to connect to the global network (the Internet) and implement advanced intelligent services that were not possible before [58]. The nRF24L01+ is a wireless transceiver module designed to operate in the 2.4 GHz ISM frequency band worldwide, and it uses GFSK modulation for data transmission. The data transfer rate is adjustable and can be set to 250kbps, 1Mbps, or 2Mbps. The module transmits and receives data on a specific frequency, known as a channel, with a bandwidth of 1 MHz. It is not based on the IEEE 802.15.4 standard and it defines its own communication models and control packets. Specifically, it divides each RF channel into six parallel data channels, known as data pipes, each with a unique address that can be used to receive data packets. This communication technique enables mesh networking, and through the Enhanced ShockBurst packet structure, it provides a transmission control mechanism and reliable communication [112].

2.3 Software Defined Wireless Sensor and Actuator Networks

The increasing interest in wireless sensor networks has led to the development of new protocols and techniques to address the constraints of these networks, but there is currently no single solution that can effectively address all of the issues inherent in WSANs. The use of the SDN approach may be able to overcome this challenge and fully realize the potential of WSANs [20]. Specifically, SDWSAN can offer the following benefits:

- Energy Efficiency: By using SDN, most energy-consuming functions can be moved from the sensor nodes to the controller, resulting in improved overall network lifetime. Routing decisions are chosen by the SDN controller and can be modified based on the energy status of the network. Additionally, existing energy optimization techniques

such as duty cycling, different sleep modes, and data aggregation can be inherited and optimized through the central view provided by the SDN controller.

- Network Management and Configuration: The SDN approach simplifies network management and optimizes provisioning, configuration, and maintenance. By removing the control logic from the network nodes, IT administrators can modify the behavior of the network according to their objectives through the SDN controller and dynamically enable or disable network functions. This feature improves the development of new applications by decoupling network management from vendor-specific nodes.

- Scalability: WSANs become difficult to handle as their size grows and they often have a maximum number of nodes and levels that they can support. SDN has the potential to support network scalability through software-oriented strategies. Specifically, the management of an increasing number of sensor nodes can be addressed by deploying more SDN controllers and splitting the global network into more sub-networks. This feature also improves the availability and load balancing of the WSAN, making it more flexible.

- Wireless Networking: Traditional routing protocols in WSANs can be classified as proactive and reactive. Proactive methods periodically sense the network through specific messages and update their routing table, while reactive methods determine routes only when needed. Both methods are energy-intensive and often result in tree topologies that are not suitable for larger WSANs. With SDN, the routing decision is moved to the controller, which can execute advanced algorithms based on the specific WSAN and dynamically change the routing policy according to the IT goals without heavily interfering with the control logic of the nodes.

- Security: In SDWSAN, network security is centralized, which simplifies the development of computationally intensive security mechanisms. The global view also enables proactive monitoring, which can be used to quickly react in case of an attack. Additionally, the centralized management of the WSAN allows for the temporary isolation and analysis of suspected malicious nodes without human intervention and with minimal degradation of network performance.

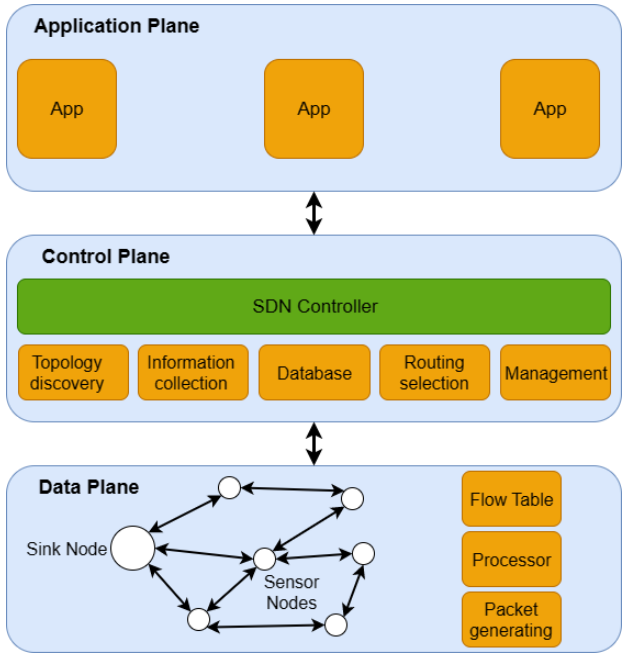


Figure 2.5: SDWSAN Architecture.

- Interoperability: WSANs are often application-specific, which leads to resource underutilization. SDN reduces dependence on manufacturers by allowing devices to be handled from a centralized point through an open standard.

SDWSAN combines the SDN approach with Traditional WSAN, as shown in Fig. 2.5. The architecture consists of three logical layer: (i) data plane, (ii) control plane and (iii) application plane [71]. The Data Plane is composed of the wireless sensor nodes deprived from its control logic. These nodes are typically divided into full-functional devices (FFDs) and reduced-functional devices (RFDs). FFDs are responsible for setting up the network and forwarding packets from other nodes, while RFDs have fewer memory, computing, and energy resources and cannot act as routers. They typically spend most of their time in sleep mode and only wake up to transmit their sensed information or receive data from other nodes. From a hardware perspective, both types of devices have a processing unit, one or more sensing

units, external memory, a power source, and a radio module, which is often based on IEEE 802.15.4. The processing unit coordinates the other modules and makes the sensed data available to the end user by sending it to the sink node using the flow rules saved in the flow table. Unlike traditional WSANs, where these tables are filled using distributed routing protocols, in SDWSAN, this operation is delegated to the SDN controller and is performed using the packet-in/packet-out SDN mechanism. It's worth noting that the protocols and techniques used in wired SDN do not meet the requirements of WSANs, so adaptation is needed before their implementation. In a later section, existing SDWSAN solutions will be analysed and reviewed. The SDN Controller is the central point of the SDWSAN. Its main function is to manage the forwarding of the entire WSAN and install specific flow entries in each wireless sensor node. It must keep an updated view of the network and provide it to the application plane. Depending on the design, some SDN Controllers may come with pre-installed modules and can perform basic functions such as routing algorithms and load balancing. Applications in the application plane communicate with the Controller through API REST and utilize it to achieve their goals. They are completely separated from the WSAN and their logic is based solely on the information provided by the SDN Controller. Although SDWSAN has the potential to solve many problems in traditional WSAN, it introduces new challenges that must be addressed before its full implementation. Unlike wired networks, WSANs have a dynamic structure with a higher failure rate. Therefore, the SDWSAN and the SDN Controller must be able to detect network anomalies and quickly change the forwarding path of the affected nodes. Additionally, SDWSANs heavily relies on the Controller, which represents a single point of failure. Therefore, recovery procedures and/or redundant systems must be taken into consideration to ensure the stability and reliability of the network.

2.4 Web of Things

The Web of Things (WoT) is a term used to describe the approaches and styles of software architecture and programming models that enable real-world objects and devices to become part of the World Wide Web. This allows for the creation of applications for the Internet of Things (IoT) by providing an application substrate. The WoT is based on a set of best practices that can be classified and categorized using the architecture of the

WoT. This architecture consists of four main levels or layers that enable the classification of the various models and protocols used in the WoT [101].

1. Accessibility level: This is the most important layer within the Web of Things infrastructure. The protocols and algorithms belonging to this layer make any object or gadget connected to the Network accessible from the web, making it programmable and usable by web platforms.

2. Searchability level: The protocols that are part of the second level of the Web of Things are necessary to search and find objects within the World wide web. These software constructs are strongly influenced by the semantic web and involve the reuse of semantic web standards to describe gadgets and connected objects and list the possible services offered to the end-users.

3. Sharing level: The Web of Things is largely based on the idea of making large amounts of data available on the web, where it can be analyzed and studied using big data analysis models and algorithms. The sharing layer allows for the large-scale sharing of data generated by things and objects connected to the Web of Things, while ensuring efficiency and safety.

4. Composition level: The role of the fourth and last level is to integrate services and data offered by things into higher-level web tools, such as data analysis software and mashup applications, making it easy to create applications that use the data made available by the protocols of the previous layers and transform them into virtual web services.

Many network-level protocols and standards for connected Things have already been developed, and have millions of devices deployed in the field today. These standards are converging on a common set of transport protocols and transfer layers, but each has peculiar content formats, payload schemas, and data types. Despite using unique formats and data models, the high-level interactions exposed by most connected things can be modeled using the Property, Action, and Event interaction affordances of the WoT Thing Description. Therefore the Wot Things Description can be adapted to the specific protocol or data payload usage across the different standards and enable an application client (a Consumer) to interact, using a consistent interaction model, with Things that expose diverse protocols. In the following sub-sections the most important application protocols which enable

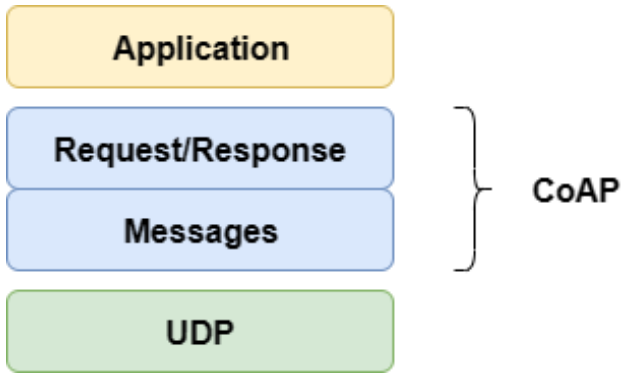


Figure 2.6: CoAP Architecture.

communication and data transfer between connected devices and the web are presented.

2.4.1 CoAP

CoAP stands for Constrained Application Protocol, and it is defined in RFC 7252. It has some similarities with HTTP but it has specifically designed for constrained networks and constrained devices [25]. The main characteristics of CoAP are:

- Web protocol used in M2M with constrained requirements
- Asynchronous message exchange
- Low overhead and very simple to parse
- URI and content-tyoe support
- Proxy and caching capabilities

CoAP is based on the Client - Server approach and within its standard specifies 6 roles:

- Endpoint: An entity that participates in the CoAP protocol. Usually, it is identified with a host
- Sender: The source that sends a message

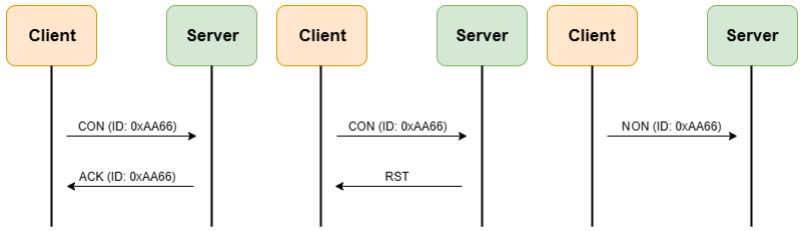


Figure 2.7: Difference between CON and NON messages.

- Recipient: The destination of a message
- Client: The entity that sends a request and the destination of the response
- Server: The entity that receives a request from a client and sends back a response to the client
- Proxy: it is border element of the CoAP domain and, its main task is to enable the interaction with other domains, e.g. HTTP.

As shown in Fig. 2.6, there are two different layers that define CoAP: Messages and Request/Response.The first one deals with UDP and with the asynchronous messages between endpoints while the latter one manages the *Client - Server* interaction based on request and response messages. Specifically, CoAP supports four types of messages: (1) Confirmable (CON), (2) Non-confirmable (NON), (3) Acknowledgment (ACK), and (4) Reset (RST). A confirmable message is used by the client to establish reliable communication with the server. This message is repeatedly sent until the server responds with an ACK message, which contains the same ID as the CON message. If the server is unable to process the request, it can instead send a RST message. In contrast, NON messages are unreliable and do not require a response from the server, and should not be used to transmit critical information.

The request and response process can be completed using both CON and NON messages. If the server can provide an immediate response to a client's CON request, it sends back an ACK message with the response or an error code. If the server cannot provide an immediate response, it sends an ACK message with an empty response. Once the response is ready, the server sends a new CON message to the client with the response. Finally, the client

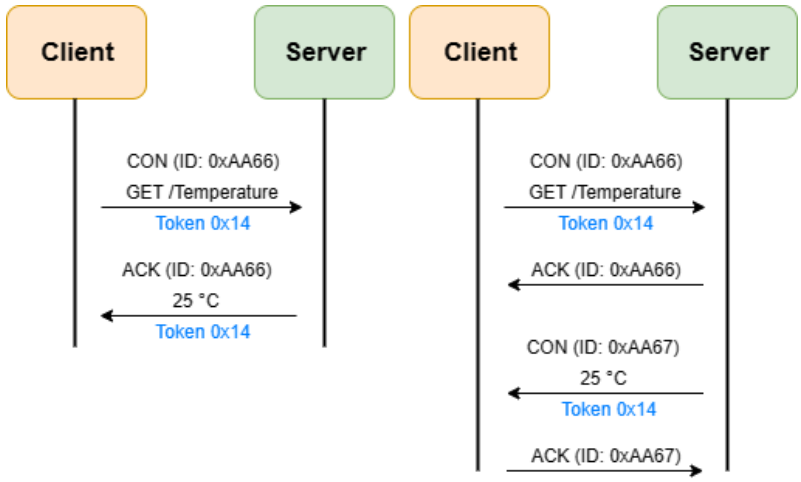


Figure 2.8: CoAP Reequest/Response procedure.

sends back an ACK message. The link between a request and a response is determined by the packet header's unique ID, called the Token. When a client sends a request using a NON message, the server may respond using a NON message, but there is no assurance that the request and response will be delivered correctly to the recipient.

2.4.2 MQTT and MQTT-SN

MQTT is a publish/subscribe messaging transport protocol standardized by OASIS. It is designed to have minimal protocol overhead and is optimized for IoT applications where a small code footprint and/or limited network bandwidth is required. Unlike CoAP, MQTT uses many characteristics of the Transmission Control Protocol (TCP), so the minimum requirement is a functioning TCP stack [115]. MQTT follows the publish/subscribe pattern, which separates clients that send messages (publishers) from clients that receive messages (subscribers). This means that publishers and subscribers do not need to be aware of each other. The central point of this architecture is a third element called a Broker. This entity is known by both publishers and subscribers, and its main task is to filter messages between them as shown in Fig. 2.9.

This decoupling of sender and receiver can be differentiated in three di-

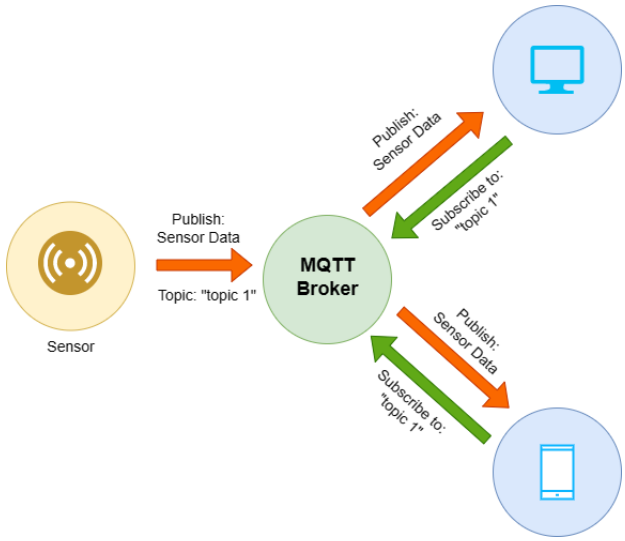


Figure 2.9: MQTT Message Exchange.

mensions:

- Space decoupling: Publisher and subscriber do not need to know each other
- Time decoupling: Publisher and subscriber do not need to be connected at the same time
- Synchronization decoupling: Operations on both components are not halted during publishing or receiving messages.

Each publisher sends its information within a specific Topic. A Topic is a UTF-8 string that the broker uses to filter messages for each connected client. It is also used by subscribers to identify a specific resource and receive information from it. When the broker receives a publish message for a certain topic, it checks its subscribed clients and then forwards the message to them. In contrast to a message queue, a topic is lightweight. There is no need for a client to create a desired topic before publishing or subscribing to it, as the broker accepts any valid topic without prior initialization. The publishing and subscribing operations occur between two phases, called connection and disconnection, as shown in Fig. 2.10.

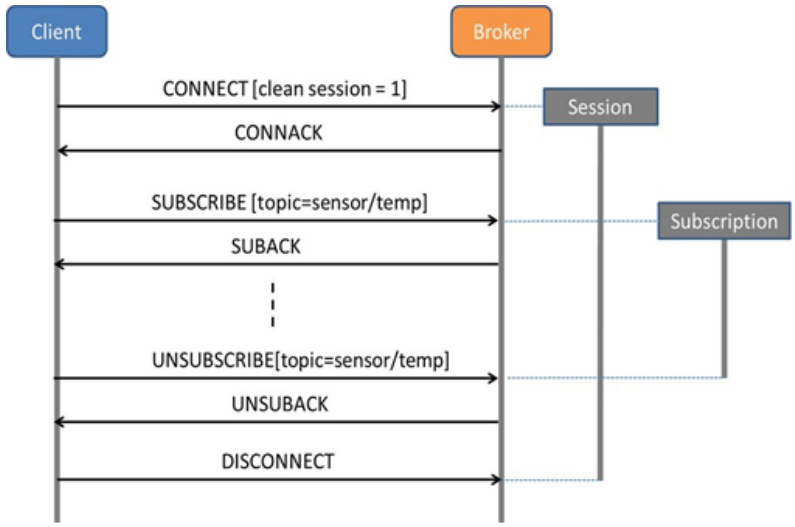


Figure 2.10: MQTT Connection and Disconnection procedure.

A client starts the connection sending a CONNECT message to the broker which contains information about the connection and the client. Moreover, in this phase the client can communicate a "Last Will and Testament" message which will be sent by the broker to the subscribers if that client does not disconnect gracefully. Depending on previous exchanged messages, the CONNECT can open a new session or can recover an existing one. If the Broker is active, it sends back an ACK message called CONACK containing the reason code. The connection will be kept open as long as the client does not send a DISCONNECT message or loses the connectivity. Each MQTT publish is sent with one of three Quality-of-Service (QoS) levels that are associated with different guarantees with regard to the reliability of message delivery. Both client and broker provide additional persistence and redelivery mechanisms to increase reliability in case of network failures, restarts of the application, and other unforeseen circumstances. MQTT relies on TCP, which has reliability guarantees on its own. Historically, QoS levels were needed to overcome data loss on older and unreliable TCP networks. This can still be a valid concern for mobile networks today.

- Level 0: At most once delivery - The sender tries to send the message with best effort and relies on the reliability of TCP. No retransmission

takes place.

- Level 1: At least once delivery - The receiver will get the message at least once. If the receiver does not acknowledge the message, or the acknowledgement gets lost on the way, it will be resent until the sender gets an acknowledgement, called PUBACK. Duplicate messages can occur.
- Level 2: Exactly once delivery - The protocol makes sure that the message will arrive exactly once at the receiver. This increases communication overhead but is the best option when neither loss nor duplication of messages are acceptable.

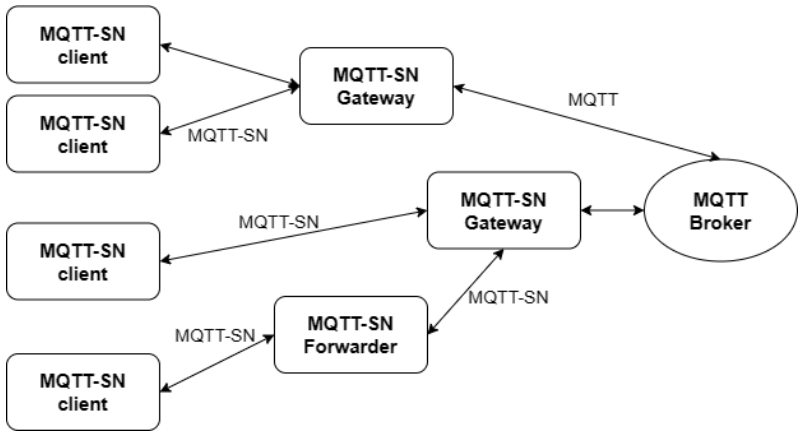


Figure 2.11: MQTT-SN Architecture.

MQTT-SN, or MQTT for Sensor Network, is a variation of the main protocol designed for embedded devices on non-TCP/IP networks [116]. It is tailored to the specific characteristics of wireless environments (low bandwidth, high link failures, short message length, etc.) and optimized for low-cost, battery-powered devices with limited processing and storage resources. Unlike MQTT, it is not dependent on the underlying networking service and can operate on top of the UDP protocol. Moreover, MQTT-SN is characterized by the following differences:

- The CONNECT message is split into three messages. The two additional ones are optional and used to transfer the Will topic and the

Will message to the broker.

- The Topic name in the PUBLISH message is replaced by a short, two bytes long *topic id*
- Pre-defined topic ids and short topic names are introduced
- A discovery procedure has been developed to aid clients to discover an operating server/gateway
- A Keep-alive mechanism has been implemented in order to support duty cycling and sleeping clients.
- Two new devices have been developed: MQTT-SN gateway and MQTT-SN forwarder. The first one is used to convert messages between the MQTT and the MQTT-SN domains while the latter is used to forward MQTT-SN messages to the gateway when it is no present or available in the same sensor network.

Although, MQTT-SN considerably improves the use of MQTT within a WSN, the centralized message management of the Broker can negatively affect the network performance parameters such as, overhead, latencies and energy consumption. In this scenario, SDN can be efficiently used to by-pass the Broker's dependency and create direct paths from Publishers and Subscribers.

2.5 Wireless Power Transfer

One of the most important challenges of IoT is to increase the battery lifetime of the devices without causing inconvenience for the end users. Although the technical advancement of the last two decades have undoubtedly improved the performance of the battery technology, the physical depletion of the limited amount of energy is still an open issue. A wired power supply or an integration with renewable energy systems (e.g., photovoltaic panel) might overcome this problem but often it is not a suitable solution (e.g., in WSN). Wireless Power Transfer (WPT) is a technology which aims to supply a system or a device without using wire for connection [131]. Basically, there are two types of WPT as shown in Fig. 2.12 : (i) near field and, (ii) far-field.

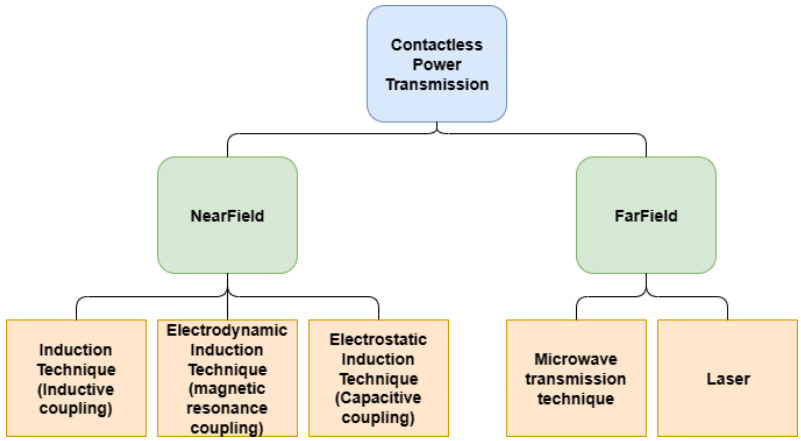


Figure 2.12: WPT Techniques.

The first one involves inductive techniques and magnetic field to transmit energy across short distances (i.e., lower than one meter) while the latter has a long-range coverage and typically use beamed electromagnetic power (lasers, radio-wave transmission and microwave).

Inductive technique

All the electromagnetic energy transfer approaches are based on the Maxwell laws, although in case of inductive technology the most important principle is the variation of the magnetic field which induces an electromotive force (emf) in a secondary winding (e.g., the action of a coupled transformer). In this way energy can be transferred over short distances. This drawback limits its application only for close-range situations.

Electrostatic induction technique

In this case, the oscillation of an High-Frequency (HF) electric fields transfers energy between conductive plates, creating one or more capacitors. At the received side, the load or battery is supplied by the transponder HF capacitive current. Unfortunately, the very high intensity electric field needed by this solution and the limited distance between the transmitter and the receiver restrict the possible applications.

Magnetic resonance coupling

The magnetic resonance coupling occurs when both sender and receiver have the same magnetic resonant frequency. Unlike the inductive coupling, in this way significant power can be transferred over long distances with min-

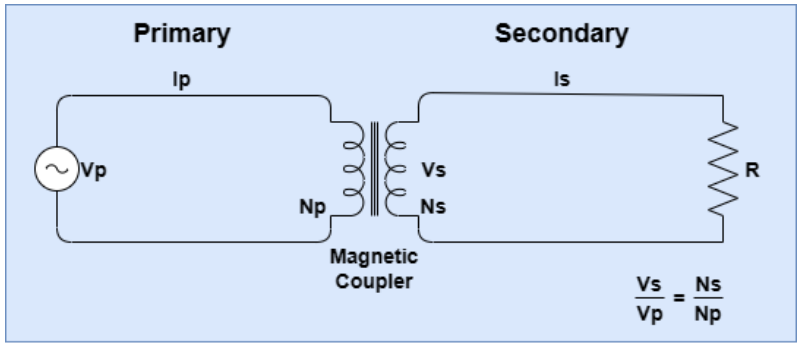


Figure 2.13: Inductive technique.

imum losses. Although both processes apply resonance, the equations that describe the energy transfer process at HF magnetic resonance are different from the equation of the inductive coupling at lower frequency.

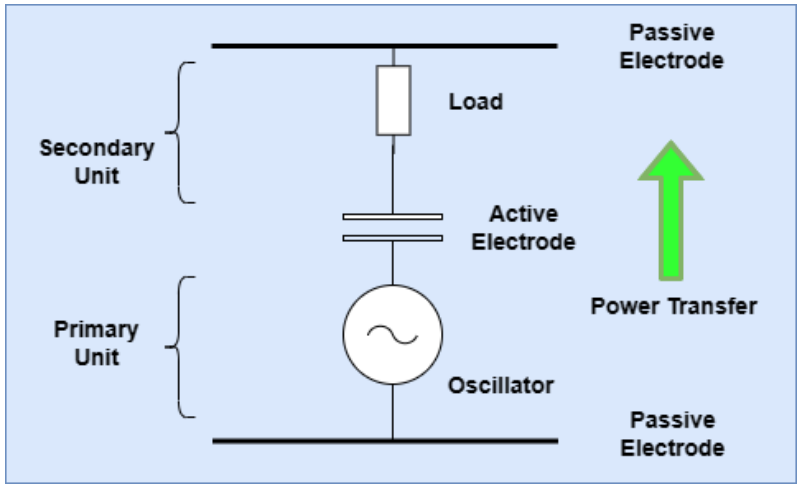


Figure 2.14: Electrostatic induction technique.

Laser beamed power-transmission

The laser emission can be used for long-distance WPT. The laser beam may be oriented on a solar transducer which converts it into electricity with and efficiency of 40-50%. The transformation process is the most problem-

atic part of this technology, since most of the energy is lost during this phase. Before making the method effective, more efficient solar cells must be developed

Microwave power transmission

The energy transmission process of this far-field technique is based on three parts:

- A converter from traditional energy to microwave
- A transmitting antenna
- And a *rectenna* (combination of an antenna and a rectifying unit) on receiver side. This device converts the electromagnetic power in DC electric power.

A small receiver antenna area results in a low amount of received energy, which represents the main drawback. Because of these constraints, the microwave power transmission is suitable for low power applications, for example low power wireless sensor nodes.

2.6 MEC, Edge, Fog Computing and Container Migration

Multi-Access Edge Computing (MEC) shifts the delivery of computing and services from a central location in the cloud to the edge of the network, providing a modular platform which exposes sensing and actuation services close to customers. Instead of sending all data to a central location for processing, the network edge analyses, processes, and stores the data. This results in both applications that are closer to end-users and processing services that are closer to the data generated by the applications. This type of architecture aims to reduce latency, improve the efficiency of network and service delivery, and provide a better user experience. The technical standards of MEC were primarily developed by the European Telecommunications Standards Institute (ETSI) [45]. The MEC architecture enables service providers to bring mobile device workloads closer to the user, thus improving performance and speeding up processing activities. Radio Access Networks (RANs) provide essential connection points between end-user devices and other parts of the operator's network. The RAN connects end-user devices to the services offered by the operator, such as voice, data, and Over-The-Top (OTT) services

like video streaming or telemedicine. The MEC architecture allows application developers and licensed content providers to access the RAN and use Edge computing at both the application layer and the underlying layer of network functions and information processing. Some of the most important application areas of the MEC architecture are:

- Video and data processing
- Localization service
- Internet of Things
- Augmented and virtual reality

5G can be considered an edge computing usage scenario, which in turn supports other usage scenarios at the edge. As part of a 5G deployment, the service provider needs to virtualize network functions to simplify network operations and increase flexibility and availability, enabling them to create innovative services and features. The MEC architecture allows to balance the performance and latency requirements of 5G networks, thus enhancing the customer experience. MEC architecture and 5G can work together to offer new services and applications. The MEC platform is the environment in which the value-added services and intelligent applications (e.g., AI/ML applications) delivered through 5G are executed. Fog computing (FC) extends the network's edge. The OpenFog Consortium defines fog computing as "a horizontal system-level architecture that distributes resources and services for computing, storage, control, and networking anywhere along the continuum from Cloud to Things." Fog computing shares similar advantages to edge computing such as low latency, a focus on storage, and real-time analytics [28]. FC relies on a group of small computing servers that are located near the edge devices and not necessarily on the device itself, whereas edge computing is primarily focused on performing compute processes close to end-users outside the core of the network. Compared to Edge, Fog Computing is more scalable as it allows a centralized system to have a broader view of the network, capturing data from multiple endpoints. The servers are connected to each other and centralized cloud servers, enabling the intelligent flow of information. These small units work together to handle pre-processing of data, short-term storage, and rule-based real-time monitoring. The fog computing architecture reduces the amount of data transported through the system, and improves overall efficiency. Mukherjee et al. [84]

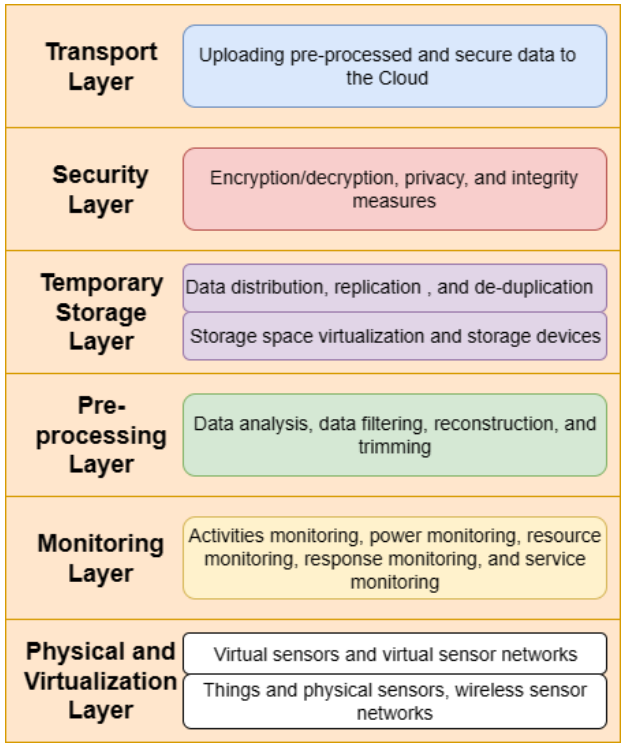


Figure 2.15: Fog Architecture.

classify the overall FC architecture in 6 levels, physical and virtualization, monitoring, pre-processing, temporary storage, security and transport, as show in Fig. 2.15. The physical and virtualization layer is the foundation of the FC architecture and includes physical nodes, virtual sensors, wireless sensor networks, and networks of virtual devices. These nodes are geographically dispersed, and their data is sent to higher levels of the FC stack for further filtering, processing, and analysis. The monitoring layer monitors the performance, resource consumption, and tasks of the sensor nodes and manages the status of services provided by the infrastructure. The collected information is forwarded to the pre-processing layer for analysis, filtering, reconstruction, and trimming operations. Then, the pre-processed data is temporarily saved on storage devices (SAN, NAS, ISCSI, etc) before being uploaded to the Cloud. The security layer ensures privacy and integrity for

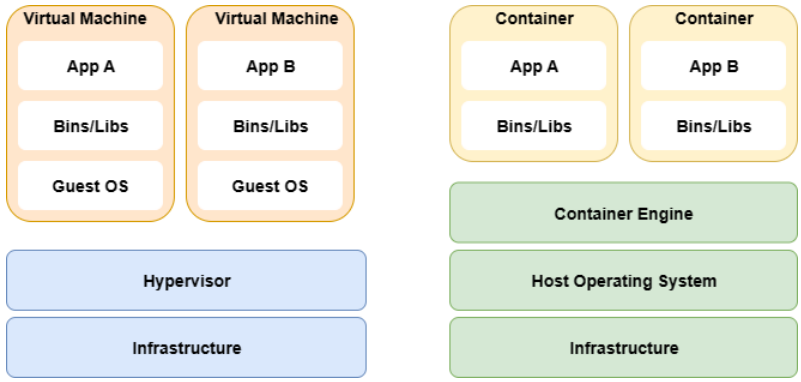


Figure 2.16: Virtual Machine and Container architecture.

the data that is outsourced to the fog nodes. Finally, the transport level uploads the partially processed and finely-secured data to the cloud for permanent storage. For efficiency, only a portion of the data is collected and uploaded, and communication protocols are chosen based on the limited resources of FC. The flexibility of the FC approach can be further enhanced through virtualization techniques such as virtual machines and containers. The key difference between containers and virtual machines is that virtual machines virtualize an entire machine down to the hardware layers, whereas containers only virtualize software layers above the operating system level. The VM technology allows to a single device to host and run multiple operating systems, sharing its resources. Therefore, a VM can be defined as a software that offer the same functionality as physical computers. Like the latter, they run applications and an operating system. However, VM are digital files that run on a physical computer and behave like separate digital systems. The coexistence of multiple VM on the same host system is regulated by a software module called Hypervisor. It acts as a resource broker, and it is logically placed between the hardware of the hosting system and the VMs. Specifically, there are two types of Hypervisors:

- Type 1: it is a thin software layer loaded directly on the physical server. Once installed and configured, the server can support multiple VMs at the same time.
- Type 2: the program is loaded on an existing operating system and after its installation, it allows the presence of multiple VMs.

Although this technique improves the availability and the elasticity of the entire system, it has two main limitations:

- **Iteration speed** Virtual machines are time consuming to build and regenerate because they encompass a full stack system. Any modifications to a virtual machine snapshot can take significant time to regenerate and validate they behave as expected.
- **Storage size cost** Virtual machines can take up a lot of storage space. They can quickly grow to several gigabytes in size. This can lead to disk space shortage issues on the virtual machines host machine.

Containers are lightweight software packages that contain all the dependencies required to execute the contained software application. These dependencies include things like system libraries, external third-party code packages, and other operating system level applications. The dependencies included in a container exist in stack levels that are higher than the operating system. In the container approach there is no hypervisor but there is a system (e.g. docket engine) that inserts the applications / application services in containers. This system creates a level of abstraction between the containers and the host operating system and manages their activation and deactivations. Furthermore, unlike the VM technique, containers share the same operating system kernel and isolate application processes from the rest of the infrastructure. This may represent a security risk since an exploit in one container could break out of the container and affect the shared hardware.

The flexibility of both virtualization techniques is further enhanced by an advanced feature called migration, which allows for the movement of a VM or container from one physical server to another without loss of information. The following summarizes the main migration modes, particularly used in container scenarios.

Migration techniques can be divided into two main classes: stateless or stateful. The first, stateless, does not store the container or VM state. Therefore, once the migration from the source node to the destination node is completed, the container restarts from scratch. It simply consists in the following steps:

- The launch of a new container on the destination node.
- The deletion of the old container on the source node.

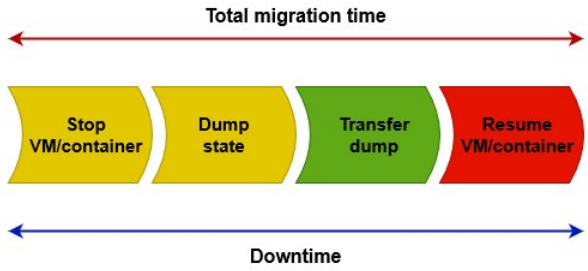


Figure 2.17: Cold Migration strategy.

In Stateful migration, the old container or VM state is also transferred to the destination node. This allows the new container to reboot from the same point where the old one stopped. Stateful migration strategies can be classified into three types: (i) Cold migration, (ii) Pre-copy migration, and (iii) Post-copy migration. As shown in Fig. 2.17, the cold migration strategy is composed of four steps. The yellow steps are executed by the source node, the red step is performed by the destination, and the green one involves both nodes.

First, the source node stops the container to prevent changes to the execution state. Then, its entire state is dumped and sent to the destination. Finally, when the entire state has been correctly transferred, the destination node restarts the container. The cold migration strategy is characterized by a very long downtime, such as the period during which the container is not reachable. It is equal to the total migration interval, which is defined as the time needed to complete the migration. Therefore, this technique is not suitable for services that require low downtime and high efficiency.

The main objective of live migration strategies is to overcome the limitation of cold migration and reduce the overall downtime period. Two different live migration techniques exist for containers: pre-copy and post-copy. The Pre-copy migration procedure, shown in Fig. 2.18, transfers most of the container status before the final dump. This is accomplished through an "iterative or pre-dumping phase" in which only those memory pages that have been modified (dirty pages) during the previous iteration are transferred to the destination node. After the pre-dumping phase, the container is suspended, and the final memory pages along with the execution state are moved to the destination server.

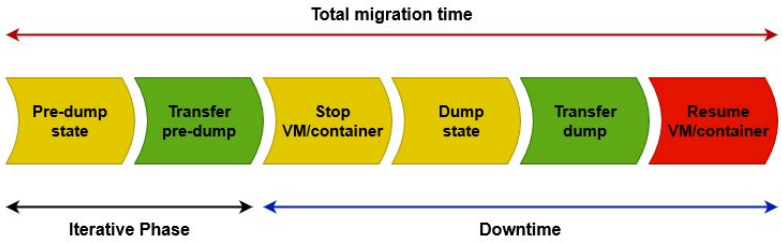


Figure 2.18: Pre-Copy Migration strategy.

Finally, the container with the updated status is resumed on the destination node. Therefore, the overall downtime should be shorter than that for cold migration because less data is transferred after the last dump. This is generally true if the speed at which the service changes the memory pages (page dirtying rate) is low and the network throughput is sufficiently high.

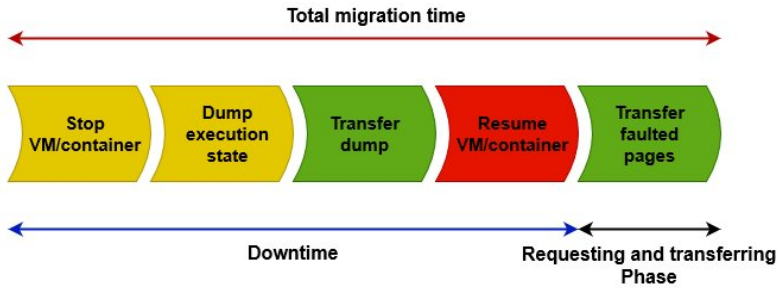


Figure 2.19: Post-Copy Migration strategy.

The Post-copy migration, instead, immediately stops the container on the source and transfers its execution state to the destination node. After that the new container is resumed and finally the missing memory pages are copied, as depicted in Fig. 2.19. The last phase can be performed using multiple techniques. One of those is called demand paging or lazy migration and it is handle by two entities, a lazy page daemon running on the destination node and a page server on the source node. Specifically, each time the resumed container cannot access to a memory page, it generates a page fault. This warning is handled by the lazy pages daemon which will contact the page server to obtain the requested faulted pages. The dump in the post-copy migration is simply the execution state, therefore the overall downtime

is less than that in cold migration. However, the unavailability of the memory pages at the destination node can degrade the service performance, and not meet the requirements of latency-sensitive applications.

2.7 Network Function Virtualization

Network Function Virtualization (NFV) refers to the process of virtualizing network functions by running them as software on virtual machines (VMs). This new approach offers a significant change from traditional methods of designing, deploying and managing network services. With NFV, network functions such as NAT, firewalling, intrusion detection, DNS and caching are separated from hardware appliances and can run on VMs. NFV leverages standard virtualization technologies and extends their use in networking. Virtual machine technology allows the migration of dedicated servers to x86 servers that are commercially available. The same principle can be applied to network devices such as:

- Network function devices: e.g. switches, routers, access points, CPEs, deep packet inspectors.
- Devices that perform network-related computing tasks: e.g. firewalls, intrusion detection systems, network management systems.
- Network-attached storage servers: servers for storing files and databases that are connected to the network.

The resulting VM which executes a specific network function is called, Virtual Network Function (VNF). Traditional networks use proprietary and closed systems for deploying all devices. Devices are enclosed and cannot share hardware, requiring additional hardware for capacity expansion, even if it remains unused when the system is underutilized. In contrast, NFV separates network elements into independent apps that are deployed on a standardized platform using servers, storage devices and switches. This decoupling of software and hardware allows for flexible capacity management, by adding or reducing virtual resources as needed.

The ISG NFV Architectural Framework, depicted in Fig. 2.20, can be viewed in three main layers, including the NFVI and virtual infrastructure manager that provide and manage virtual and physical resources, the VNF layer that provides software network functions with element management

systems and VNF managers, and the management, orchestration, and control layer with OSS/BSS and NFV orchestrator.

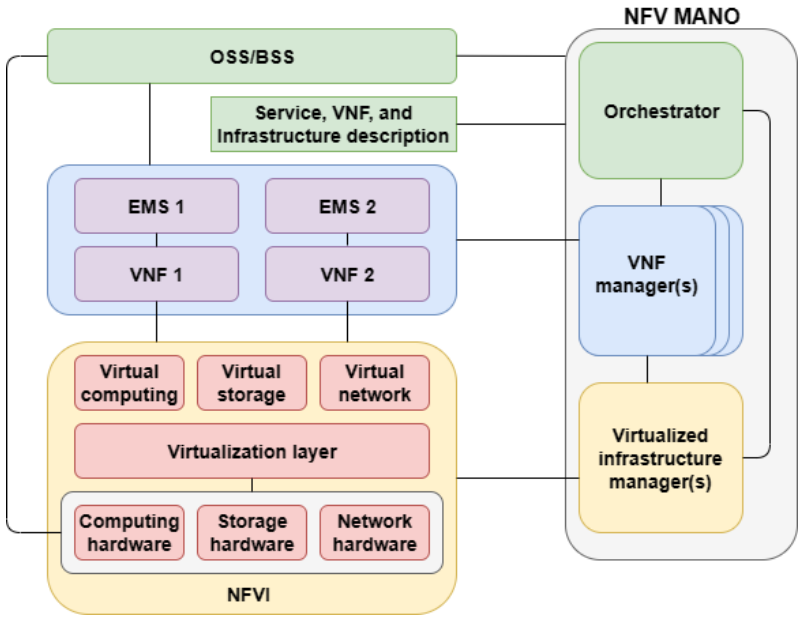


Figure 2.20: ISG NFV Reference Architecture.

Specifically, the ISG NFV architecutre is composed of the following elements:

- NFV Infrastructure (NFVI): The hardware and software that make up the environment where VNFs are deployed. NFVI virtualizes physical computing, storage, and networking and pools them.

- VNF/EMS: Collection of VNFs running on virtualized computing, storage, and networking resources, along with a set of element management systems (EMS) that manage the VNFs.

- NFV Management and Orchestration (NFV-MANO): A framework for managing and orchestrating all resources in the NFV environment, including computing, networking, storage, and VM resources. It is complex to implement and must also interact with existing OSS and

BSS for managing customer environments that combine physical and virtual elements.

- OSS/BSS: Refers to the operator's existing operations and business support functions that aren't part of the current architectural framework, but are expected to exchange information with the NFV-MANO framework. These systems may manage legacy systems and provide complete visibility of services offered by legacy network functions in the operator's network.

Focusing on the NFV management and orchestration system, it is made up of several functional blocks, comprising the NFV orchestrator, VNF manager, and virtualized infrastructure manager (VIM). The NFV orchestrator is in charge of setting up and arranging new network services and VNF packages, managing the lifecycle of network services, controlling global resources, and verifying and granting access to NFVI resources. The VNF manager is responsible for overseeing the lifecycle of VNF instances. The VIM handles the interaction between the VNF and computing, storage, and network resources, as well as their virtualization, with one VIM controlling and managing the NFVI resources within one operator's infrastructure.

For NFV to be successful, it needs to have standards set at relevant interface points and open source software for frequently utilized functions, similar to SDN. The ISG NFV has been dedicated to creating standards for the different interfaces and parts of NFV for a number of years. In 2014, the Linux Foundation announced the Open Platform for NFV (OPNFV) project, which aims to be a carrier-grade, unified platform that can bring new products and services to the industry rapidly.

Chapter 3

Literature review

This chapter gives a brief survey of related works on existing Software Defined techniques and protocols for IoT management and optimization. The first part of the chapter introduces the existing SDN approaches for the network lifetime improving, while the second part focuses on the combination between SDN and WoT communication protocols. In the third sub-section the most important SDWSAN solution are presented and their main criticism are exposed. Finally, the main contributes on service dissemination in hybrid Cloud-Fog scenario are shown in the last part.

3.1 Energy-aware routing proposals

In the following, the main papers which detail energy-aware routing proposals are considered. The authors in [42] present a software-defined clustered sensor networks (SDCSN) arranged into clusters managed by one SDN Controller. About 1000 nodes divided in five clusters exchange messages and forward packets among themselves according to the policies of the SDN Controller. This solution provides a scalable architecture for IoT, since it can route data packets flows among the devices. The same model, i.e., a centralised control and hierarchical cluster structure is proposed in [113]. The Lion Optimization (LO) algorithm creates both the clusters of sensor nodes and the routes to forward data. A hierarchical structure allows a global control of the WSN, making easier the deployment of network-wide management

protocols and on demand applications.

Software defined energy aware routing (SD-EAR) proposed in [14] divides the sensor network into clusters with a SDN Controller associated to each cluster, where Controllers know the topologies of the associated cluster as well as of the border nodes. Each Controller evaluates the best path from the source to destination if the destination falls in its area or, alternatively, it can send a request to discover the route to destination, taking into account the residual energy of nodes. Moreover, SD-EAR adopts sleeping strategy to reduce energy consumption in the network.

The contribution in [96] analyses the application of the SDN architecture to improve security and energy efficiency in the WSN domain. In particular, the SDN Controller executes the Sensor Protocol for Information via Negotiation (SPIN) protocol to generate neighbours table and, then, to select the best next hop, according to its energy level and link quality. Then, the selected node is used by sensor node to forward the information.

Shafique, et al. in [109] consider a reconfigurable distributed protocol, called Application-aware Distributed adaptive Flow Iterative Reconfiguring (SADFIR). It uses multiple distributed SDN Controllers selected among IoT nodes as cluster heads (CH) and computes the residual resources of the IoT devices. Then, SADFIR distributes the selected CHs to the IoT infrastructure allowing distributed cluster formation of heterogeneous network.

Beside, a modified low-energy adaptive clustering hierarchy (LEACH) clustering protocol is proposed in [18]. In particular, the same concept of the LEACH scheme [50], i.e., sensor nodes organised into clusters, is adopted with the selection of a cluster head. The clustering mechanism is applied to distribute supply energy among sensors, according to the time division multiple access (TDMA) based MAC protocol, which is further modified by introducing a threshold value for cluster head selection with the simultaneous switching of different power levels among nodes.

Several papers in the literature consider bio-inspired clustering algorithms. Ant Colony Optimisation (ACO) is presented in [108], where the cluster heads selection is implemented as it happens with the ant pheromone level. The artificial bee colony algorithm is proposed in [65] to monitor the state of the sensor nodes in a typical SDN architecture through the Controller which selects the best path. The wide use of SDWSAN to reconfigure the networks, even after their displacement, leads the authors in [128] to consider an efficient particle swarm optimisation (PSO) algorithm based on

the residual energy of the nodes and the transmission distance. A green routing algorithm using fork and join adaptive particle swarm optimisation (FJAPSO) is addressed in [70] to operate in SDWSAN. The considered scenario presents a classical topology, where sensor nodes are divided into clusters and each cluster has its own CH. However to maximise the lifetime of sensor nodes and to allow the adaptation to scalable networking scenarios, the FJAPSO acts on two levels by optimising jointly the number of control nodes and the best suited position of all the control nodes with respect to the Control Server (CS) and between CHs and sensor nodes. The Whale Optimization Algorithm, which considers the residual energy, communication cost and nodes density, is proposed in [5] to enhance the network lifetime. The aforementioned works typically consider a well-known hierarchical routing algorithms relying on the partition in clusters and the selections of the proper CHs. Moreover, CHs are highly energy consuming nodes, since they are requested to transmit to the Sink over a single hop, especially if they are located far from the Sink.

3.2 SDN and the Web of Things

In the literature, several surveys present the specific challenges and applicability of the SDN principle to meet the application-specific requirements of a typical IoT domain. In particular, SDN-based architectures are coupled with network function virtualization (NFV) to cost-effectively provide the scalability and flexibility necessary for IoT services, as shown in [24]. In [21], an overview of SDN for IoT is provided, highlighting several SDN-based data aggregation and management approaches, with different architectures, i.e., edge, access, core, and data center networking, together with addressing the main requirements and issues to efficiently manage the sensor nodes resources. Moreover, future directions and open research issues are detailed to address aspects as mobility, policy enforcement, hardware platform for SDN-based solutions. Further, another recent survey [62] also discusses the issues of SDN-based Edge Computing (EC) architectures for IoT. Recently, EH technology has been proposed to achieve long-term and self-sustainable functioning for IoT nodes and prolong the lifetime of overall network, especially if operating in extreme environments, where the replacement of batteries is either difficult or costly. Energy-depleted nodes can download energy from a renewable energy sources, such as wind and solar

energy or a mobile charger. In this vision, a SDN architecture is tailored to optimise EH management by adopting a centralised control for both energy and data flows. As shown in [54], [55], a SDN-based architecture considers separated control, data and energy planes to provide high flexibility in energy scheduling and to maximize energy utilisation. The authors specifically propose a software defined energy harvesting IoT (SEANET) architecture with a centralised flow control to optimise network management procedures. In more details, SEANET relay data packets among those nodes with higher reputation values and sufficient energy to reduce the packet loss. Moreover, energy-rich nodes make optimal decisions to upload the surplus energy to the mobile charger.
MQTT protocol, standardised by Oasis [15], is widely used to assist communication between servers and resource-constrained IoT devices. MQTT is a lightweight protocol based on publish/subscribe message exchanging via an intermediate device called Broker. This approach allows delivery guarantees, low overhead and power consumption. Moreover, a device can be easily added/removed to the network without perturbing the existing topology. Recently, a combination of SDN and MQTT has been proposed in literature for the management of complex and evolving networks; in particular, the SDN Controller allows IoT devices to interoperate, especially in the presence of heterogeneous technologies, protocols and standards.

An architecture comprised of a master Broker implemented on the SDN Controller to handle the overall system, and several slave Brokers, each administrating a cluster of IoT devices, is presented in [120]. The centralized SDN Controller uses a multicast scheme to distribute MQTT data across the external wireless network, thus reducing the transmission delay for massive IoT scenarios.

A multicast based transmission scheme among Publisher and Subscriber is also proposed in [92], namely Direct Multicast-MQTT (DM-MQTT), together with the integration along an SDN architecture to minimise the transmission delays between multiple edge subnetworks. DM-MQTT relies on a hierarchical Broker architecture by placing a slave Broker on the very edge node, while a master Broker gathers information to the SDN Controller which sets the data transmission paths among different edge networks to minimise data transfer delays.

Similarly, in [93], the SDN paradigm is applied to a wired network to modify the behaviour of OpenFlow Switches in order to create direct paths

between publishers and subscribers. Besides, MQTT protocol is used for data exchanging among IoT devices by properly integrating within an SDN architecture, pointing out a remarkable performance improvement, as shown in [103].

3.3 Existing Software Defined Wireless Sensor Network solutions

The emerging SDWSAN technology aims to integrate the SDN logic with the traditional WSAN [97] [67]. In this section, an overview of the existing solutions is given. An early effort to construct a SDWSAN was the *Sensor OpenFlow* solution [76], which adheres to the OpenFlow architecture by mandating that nodes keep a flow table with entries of a particular, predetermined format. The primary goal of Sensor OpenFlow is to enable data aggregation, as is often done in WNS for energy conservation. However, it should be noted that Sensor OpenFlow is not equipped to handle the vast array of protocols, both standard and proprietary, that have been proposed for use in WSANs.

In [43], the authors propose an architecture called SDN-WISE, which seamlessly integrates the three levels of SDN and the 802.15.4 wireless sensor network. The network is organized in a tree structure where the sink node is the sole point of contact between the SDN controller and the WSN. The control messages were designed with the limitations of the sensor network in mind, and, in contrast to wired SDN, the forwarding decisions are based on the current state of the sensor node. The SDN-WISE tool provides its own SDN Controller but it can also be integrated with existing solutions such as ONOS or Ryu. The network stack is based on the 802.15.4 PHY and MAC layers and implemented on the Contiki platform. The forwarding decisions for downstream traffic are entirely dependent on the SDN controller while upstream messages are handled using a distributed algorithm. This means that a malfunction of the controller can cause a widespread network failure. Furthermore, high dynamic scenarios cannot be promptly handled by the controller and frequent network changes can increase network overhead and negatively impact overall performance.

The TinySDN tool, as described in [40], is a software-defined networking implementation based on TinyOS. It consists of two main components: the SDN-enabled sensor node, which includes an SDN switch and an SDN end-

user device, and the SDN controller node, where the control plane logic is executed. The SDN-enabled sensor nodes, where the data plane components are located, connect to the SDN controller node through multi-hop wireless communication, enabling interaction between the two planes. The SDN-enabled sensor node runs on sensor motes, providing forwarding services to applications as a network protocol. The SDN controller node, on the other hand, performs tasks such as maintaining a network topology view and applying definitions of SDN applications by creating and managing network flows. Each SDN-enabled sensor node must connect to an SDN controller node using the collection tree protocol (CTP) to send network topology information and request flow specifications as needed. TinySDN supports two types of flows: control flow and data flow. The former is used to forward data packets generated by applications through the sensor network, while the latter is used to forward control packets from the SDN controller node to the SDN-enabled sensor nodes, specifically to establish downward routes for control packets. Upward routes (from controller to sensor nodes) are established using CTP.

Unlike SDN-WISE and TinySDN, IT-SDN [80] has a distinct separation of the protocols used for southbound communication (SB), neighbor discovery (ND), and controller discovery (CD), allowing for the evaluation of different methods for neighbour and controller discovery. Additionally, the IT-SDN architecture allows for configuring multiple nodes with a single packet, leaving it up to the controller to decide whether or not to use it. The IT-SDN architecture comprises three components: a southbound (SB) protocol, a neighbour discovery (ND) protocol, and a controller discovery (CD) protocol. The SB protocol defines the communication procedure between the controller and the SDN-enabled devices, including packet format, processing, and operation workflow. The ND protocol obtains and maintains information about neighbouring nodes, and the CD protocol identifies the next hop to reach the controller. In static networks, the ND protocol is executed during network bootstrapping, while in networks with mobility and changing links, it runs continuously. The CD protocol is primarily used during network bootstrapping, before the node gets routing information from the controller.

The authors of [11] propose a lightweight SDN framework called uSDN designed for the Contiki OS. It builds upon some of the architectural concepts proposed in previous works while incorporating optimization techniques to

reduce control overhead and improve scalability. uSDN sits above the IP layer within the IEEE 802.15.4-12 stack and is designed to be interoperable with legacy nodes in an IEEE 802.15.4 network. uSDN uses the RPL to provide a fall-back communication path to the controller, which can be replaced with any other distributed routing protocol. It also has a modular architecture and API that allows for application-specific features to be separated from core SDN processes. The uSDN protocol follows the main packet types present in traditional SDN protocols such as OpenFlow, including basic flowtable request/set functionality and configuration and metric update packets. To optimize the performance in low-powered wireless networks, uSDN compresses information such as node addresses and uses node configuration tables to control the information sent to and from the controller. The traffic generated by uSDN comes from three processes: controller discovery, node updates, and requests for controller instructions.

3.4 Fog Computing and container migration in mobility scenario

The potential benefits of the FC application to the IoV scenario, especially with cellular connectivity, have been addressed in the literature. In this scenario, mobility represents the most significant challenge as vehicles move between different cells at high speeds, requiring service migration to maintain continuity. This highlights the need to handle mobility effectively to guarantee low latency and high reliability, particularly for real-time service [29]. To minimize negative impact on Quality of Service (QoS), both computation and communication must be considered.

Live migration was first proposed and used in cloud data centers to move running applications between different physical machines without disconnecting the users. It has also been used to hand over computation jobs between fog nodes. Virtual machine (VM) migration [86] and container migration [13] (e.g., Docker [48]) are the two main solutions. In [118], a service migration mechanism in wired networks called Follow Me Cloud (FMC) is proposed. It supports smooth migration of IP services between a data center and mobile device to another data center without redesigning the wireless handover timing procedure (i.e., it follows a standard procedure in traditional cellular networks). In the context of MEC or fog computing, active service applications are encapsulated in Virtual Machines (VMs) or

containers, and several studies have been conducted to address the mobility problem [23] - [124]. In [23], general architectural components supporting VM migration and interactions among such components are defined and discussed. In [77] a general layered approach is proposed, which allows the migrated applications to be decomposed into multiple levels. In this framework, only the layers missing at the destination need to be transferred, thus reducing a big amount of data to be handled during the service migration. A VM handoff mechanism for the service migration is proposed in [47], in which the migration files are compressed before being migrated to adaptively reduce the total migration time. Huang et al. [119] and Wu et al. [122] focused on the mobility pattern of edge computation devices and developed a cost model for the service migration using a Markov decision process based approach. In [124], a time window based service migration is proposed to search the optimal service placement sequence. However, most of the existing studies, e.g. [119] - [124], are based on abstract models, and do not reflect the real situation, where many parameters need to be optimized. An architecture, integrating FC and CC managed by SDN in order to reduce IoV latencies and support mobility, is proposed in [49]. A framework for vehicular CC in which vehicles together form a Cloud resource unit is presented in [8]. This allows to handle a network slice at the WiFi Access Point (AP) and to share bandwidth and transmission opportunities of an AP dedicated to different slices. Khakimov *et al.* combined SDN and FC approaches in an IoT oriented framework in [66], where resources available in the OpenFlow (OF) Switches are not only used for forwarding functions, but also for providing basic services. A protocol architecture for SDN based IoV, which allows data packets forwarding over V2V and V2X links, is proposed in [1]. Iqbal *et al.* presented in [57] a data analytic framework at the Fog Layer of IoV reference model to provide context-aware real-time and batch services at the network edge. A scheme to support seamless handover between different computation APs is presented in [16], implicitly handling task pre-migration mechanisms whenever the handover is expected to happen, however limited to low-to-medium mobility patterns. A comprehensive overview of the migration techniques and a real Fog testbed is addressed in [98]. Li *et al.* proposed in [73] a QoS-aware approach based on the existing handover procedures for real-time services, applying the service migration in a FC enabled cellular network under restrictive hypotheses.

Chapter 4

An innovative Software Defined approach for self-healing and self-organizing Wireless Sensor Network

SDN can bring many improvements in the wireless IoT domain but its correct implementation is still an open issue. The simple translation and adaptation of centralized algorithms within low-power wireless sensor and actuator networks has shown many limits especially in large scale and high dynamic scenarios. Not only the SDN controller can represent a single point of failure for the entire network, but the overhead generated by control messages and the inherited OpenFlow procedure can heavily affect the performance of data traffic. This chapter addresses these challenges by proposing an hybrid architecture, called SOHS, where the distributed nature of the WSANs is combined with the centralized management of SDN.

4.1 Introduction

The Software Defined Networking paradigm was specifically studied and implemented to encourage the interoperability of devices of different vendors

and simplify the management of wired networks. WSAN can also benefits from SDN, but their integration must consider the features and constraints of this kind of networks. The proposed approach aims to adapt the SDN logic to the WSANs requirements overcoming the single point of failure of the SDN controller and, guaranteeing optimal performance also in dynamic scenarios. In fact, even if most of the existing SDWSAN solutions have modified the OpenFlow protocol considering the WSAN constraints (bandwidth, energy, data rate, etc.), the availability and scalability of the SDWSAN are still two open issues, especially in large scale network with frequent topological changes. In these cases, it is infeasible to rely on *centralized* algorithms to implement network management solutions such as topology management or routing. In traditional WSAN, sensor nodes collaborate with their neighbours to make localized decisions, without global knowledge. Focusing on routing decisions, in centralized approaches, a central and powerful node collects information from all sensor nodes, establishes routes that are optimal and informs each node of its route. However, the overhead can be significant, particularly if the topology changes frequently. Moreover, a failure of the central node can lead to a global network unavailability. Instead, a decentralized solution allows each node to make routing decisions based on limited local information (e.g., a list of the node's neighbours, including their distances to the sink node). Therefore, the results of *decentralized* (or distributed) algorithms will not be optimal, but they may be more efficient than centralized approaches. The proposed approach, named SOHS, aims to adapt the SDN logic to the WSANs requirements overcoming the single point of failure of the SDN controller and, guaranteeing optimal performance also in mobile scenarios. Differently by the existing solutions, SOHS combines the distributed nature of the WSAN with the centralized vision of the controller by creating a dual stack hybrid architecture. The resulting approach does not impose the SDN logic in the 802.15.4 networks, but exploits the WSAN to quickly disseminate the forwarding rules and to improve the system availability.

4.2 System Design

The proposed solution leverages the self-organizing and self-healing features of the WSAN to quickly deploy the SDN rules. In traditional SDWSAN, devices can forward packets only if their forwarding table is correctly filled

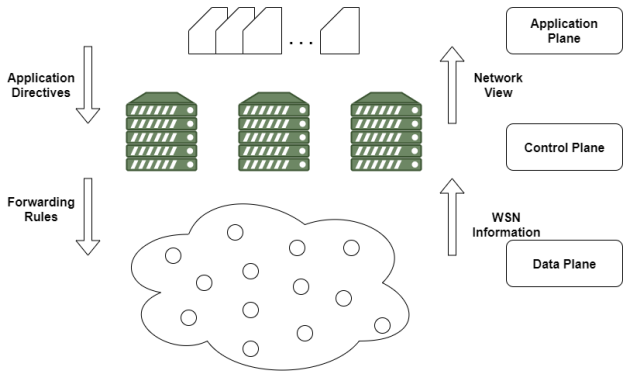


Figure 4.1: Proposed Software Defined Wireless Sensor and Actuator Network Architecture.

by the SDN controller. If the table does not contain the proper forwarding rule, the device has to contact the SDN controller and starts the packet-in/packet-out procedure. This messages exchange, in addition to cause an increase of network overhead, makes the network totally dependent to the SDN Controller, and its fault can result in a network failure. In the suggested approach, instead, the network maintains its characteristics of self-organizing and self-healing but the packet forwarding may be delegated to the SDN Controller. Usually, WSANs are organized in a tree topology (Zigbee, 6LowPAN, NRFl04, ecc.) and use their protocol to populate the device′s forwarding table and enable the downstream and upstream data traffic. In SOHS, every node detects which devices are in its range and communicates this information to the SDN Controller, which will be able to deploy its network view. Finally, applications use this representation to perform their operations, as network routing optimization, firewalling or network segmentation. The directives are translated by the SDN controller in forwarding rule for the network devices and inserted in specific control messages. These forwarding rules do not overwrite the rules of the running WSAN protocol but are inserted in specific tables with a priority greater than the existing one. Thus, when a node receives a packet first checks the SDN forwarding table and then, if no rules are matched, checks the original table. A show in Fig. 4.1, the proposed hybrid architecture is based on the classical SDN layers: (i) Data Plane, (ii) Control Plane and (iii) Application Plane. However, unlike existing SDWSAN solutions, the devices within the DP are

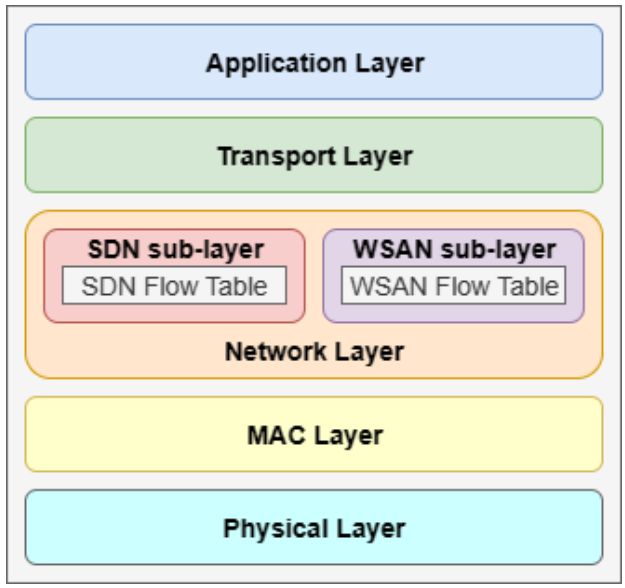


Figure 4.2: SDWSAN Hybrid Dual Stack.

not completely dependent to the SDN controller but maintain their control logic. The Network Layer is decomposed in two parts, SDN and WSAN sub layer, which share the same MAC and PHY technologies but have different Flow Tables, as depicted in Fig. 4.2. Particularly, the network organization and updating phases are performed using the algorithms and messages of the existing WSAN. During the creation step, the network devices do not know which nodes are in their radio coverage and which is the best route to the sink node or Gateway. As consequence, they have to send specific beacon and control messages to announce themselves and detect the neighbours presence. This is usually accomplished by distributed algorithms in which every nodes update its route using the information sent by other nodes. For example, in IT-SDN this is called Collection Tree Protocol (CTP) while in SDN-WISE this is known as Topology Discovery Protocol. In both cases, the presence of the sink is broadcasted to all nodes of the network and the distance to it is updated at each retransmission. This information is locally saved in appropriate routing tables and used to handle the network traffic. It should point out that nodes only have local information and therefore their routing tables are limited to portions of the network. Finally, the periodic

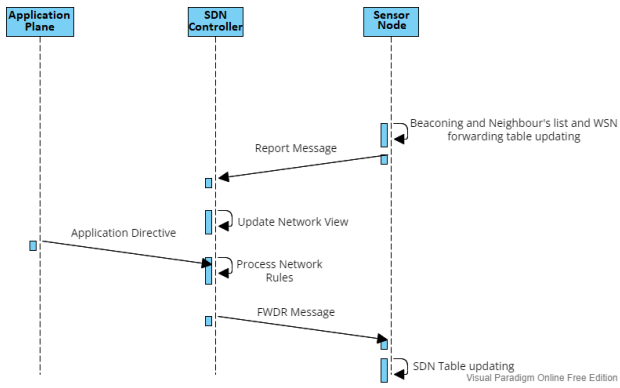


Figure 4.3: Sequence Diagram of the proposed SD-WSAN solution.

control messages exchange allows the network to be always updated and to promptly react to internal changes and faults. For a complete integration of SDN with WSAN, in the proposed approach all nodes periodically send their internal information (neighbour′s table, battery level, identity) in apposite *Report* messages to the Sink Node. It re-transmits them to the SDN Controller which can build the network view and run optimized algorithms. As in the wired SDN solution, applications are agnostic of the network fabric and all their directives for the network are opportunely translated in forwarding rules by the SDN controller and included in proper messages called *FWDR*. Each rule is inserted in a specific table called SDN routing table with a priority greater than the traditional WSAN table. Therefore, in case both tables share the same destination, the SDN rule will be used. Unlike the existing SDWSAN solutions, the SDN paradigm collaborate with the WSAN, and all the decision are always top-down. The entire procedure is shown in Fig. 4.3.

4.2.1 SDN Routing Table

As previously described, the sensor nodes forwarding logic is based on two tables: WSAN table and, SDN table. The first one is dependent on the specific WSAN technology, and its entries are created using distributed routing algorithms. The latter one is instead managed by the SDN Controller and contains its forwarding rules. As shown in Tab. 4.1. it is composed of the following fields:

Table 4.1: SDN Forwarding Table.

ID	Source Address	Destination Address	Action	Next Hop Address	TTL
1	aa:bb:cc:dd:ee:01	aa:bb:cc:dd:ee:02	FWD	aa:bb:cc:dd:ee:03	100
2	aa:bb:cc:dd:ee:f4		DROP		80
3		aa:bb:cc:dd:ee:f5	FWD	aa:bb:cc:dd:ee:03	20

Source Address: it identifies the sensor node who sent the packet. This information can be used to filter some network flows and execute specific actions.

Destination Address: it specifies the node that has to receive the packet. As in the source address field, this information can be used to perform specific operation. Source and Destination address fields are called matching fields.

Action: If the packet header matches the source and/or the destination address, an action is performed. Specifically, the allowed operations are: (i) forward the packet to the next node and (ii) drop the packet.

Next Hop Address: it identifies the next node to forward the packet. The concatenation of more next hop node creates a network path which is used to connect a source node with a destination node. If the specified next hop node is no longer reachable by the sensor node, it has to delete the entire flow entry and sends a report control message to the SDN controller which can update its network view.

TTL: This information represents the entry lifetime. It starts from a default value and is updated each time the flow entry is matched. If it reaches the zero values, the flow entry is deleted from the table.

When a node receives a packet, it first checks the header with the matching fields of the SDN Table. If the research provides a result, the related flow action is performed. If instead no results are obtained, the node uses the traditional WSAN table to forward the packet. The full logic is depicted in the diagram of Fig. 4.4.

4.2.2 SDN Messages

The dynamic management performed by the SDN Controller is accomplished through three messages: (i) Report, (ii) Info and, (iii) FWDR which are inserted as payload in the existing and running WSAN protocol packets.

Report Message

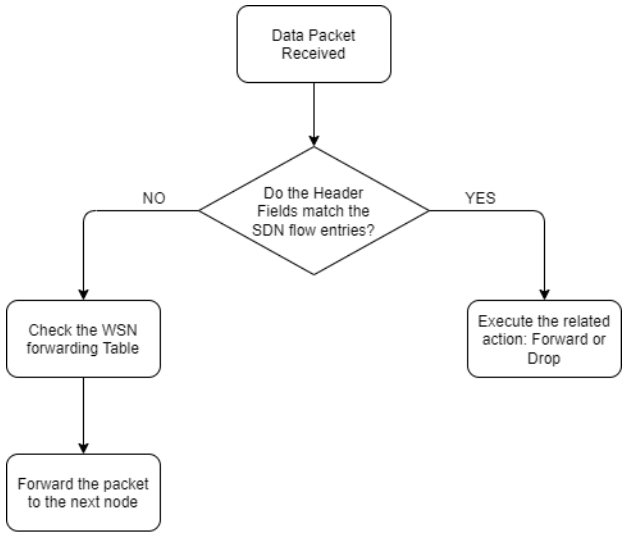


Figure 4.4: Flow chart of the forwarding mechanism.

Table 4.2: Report Message.

Source Address	Battery Level	Address 1	...	Address N

This message, shown in Tab. 4.2 , is periodically sent by each node of the network to update the view of the SDN Controller. It is composed of the following fields:

- Source address: it univocally identifies the sensor node within the network.
- Battery level: is the remaining energy of the node. It can be used by the applications as routing parameter and to optimize the lifetime.
- Neighbour′s list: it contains all the nodes that are in the coverage range of the specific sensor node.

The SDN Controller combines all the received Report messages and creates a virtual representation of the underlying network. Applications will then use this information to perform their tasks and dynamically modify the forwarding paths.

Table 4.3: Info Message.

Reason Code	Data

Info Message

As in traditional SDN architecture, the Controller represents a single point of failure and its fault must be quickly detected. In the proposed system, this is achieved through the *Info* messages and exploiting the tree topology. Specifically, the reachability of the Controller is exclusively checked by the Sink node or Gateway. In case of Controller failure, it broadcasts a Info message to all the WSAN with the specific reason code. As shown in Tab. 4.3 this control message is composed of two fields:

- Reason Code: it indicates the specific message ID and it is used by the network node to execute a specific command. In case of Controller fault, the related code is the number one and the action is to switch to the WSAN forwarding table.
- Data: this field contains a value which is used in combination with the Reason Code to perform the relative function.

The Info message can also be used to modify the node behaviour like the Report sensing frequency or the duty cycling periods.

FWDR Message

The implementation of new routes embraces all the three levels of the SDN architecture. The Application Plane process the new paths, the Control Plane translate the directives in forwarding entries, and the DP inserts them in the SDN Table. The communication of a new flow rule occurs through the FWDR message. It is composed of two main field, as shown in Tab. 4.4:

- Length: it specifies the number of the new flow entries that must be installed.
- Flow entry: It contains the entire flow rule, Source Address, Destination Address, Action, and Next Hop Address. Depending on the specific case, some of these fields can be empty.

4.2.3 Control Tasks

The implementation of the SDN paradigm within the sensor node and its full integration with the WSAN is accomplished by two main tasks called Flow

Table 4.4: FWDR Message.

Length	Source Address	Destination Address	Action	Next Hop Address

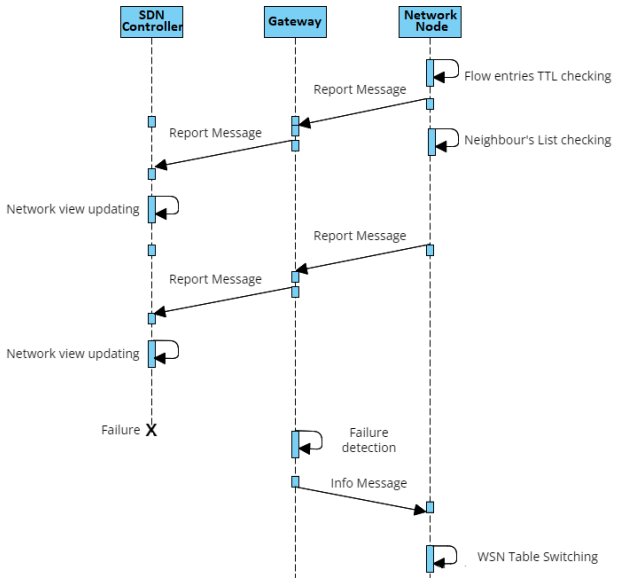


Figure 4.5: Sequence Diagram of the FTU and CU tasks.

Table Updating Task (FTU) and Controller Updating Task (CU). The first one monitors the status of the SDN flow entries and in case of receiving a FWDR Message, it updates the SDN Table adding the new rule. Moreover, it periodically checks the TTL field of each entry and in case of 0 value, it deletes the entire rule. The rule can also be deleted, if the next hop node address is no longer present in the neighbour′s list. After the deletion, the FTU task sends a Report message to the SDN controller with the updated list. The CU Task is used by each node to periodically update the Controller about the node status. The sending period of the Report message can be modified according to the WSAN scenario and policies. For example, in high dynamic environment it should be lower than the static one in order to quick adapts the node′s flow entries to the network changes. Both two tasks are depicted in the sequence diagram of Fig. 4.5.

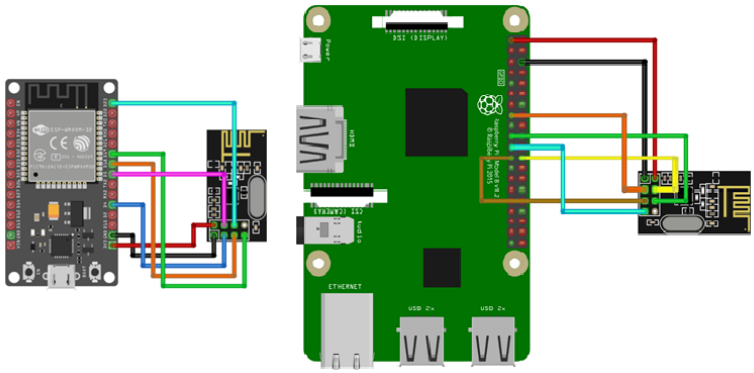


Figure 4.6: ESP32 and NRF24L01 interconnection.

4.3 Testbed and Experimental results

In this section the performance of the proposed approach are compared with existing SDWSAN solutions in terms of Latency, Availability, Overhead and Data Delivery Rate in static and dynamic scenarios.

Experimentation was performed using the Espressif ESP32 microcontroller as network node and the Raspberry Pi 3b+ as local SDN Controller. The wireless communication and the related WSAN, has been implemented using the NRF24L01+ radio transceiver technology. The interconnection of the two modules is shown in Fig. 4.6.

4.3.1 ESP32

Created by Espressif Systems, ESP32 is a low-cost System on Chip (SoC) with dual-mode Wi-Fi and Bluetooth capabilities. The main element of this module is the dual-core or single-core Tensilica Xtensa LX6 microprocessor with a clock frequency of up to 240 MHz. Thanks to power saving features including clock gating, multiple power modes and dynamic scaling of power, ESP32 has extremely low power consumption, making it perfect for IoT applications. Tab. 4.5 lists the main characteristics of the ESP32 DEVKIT V1 DOIT module.

Table 4.5: ESP32 Features.

Core	2
WiFi	2.4 GHz up to 150 Mbits/s
Bluetooth	BLE (Bluetooth Low Energy) and legacy Bluetooth
Architecture	32 bit
Clock Frequency	up to 240 MHz
RAM	512 KB
Pin	From 30 to 36
Peripherals	Capacitive Touch, ADC (Analog to Digital Converter), DAC (Digital to Analog Converter), I2C (Inter-Integrated Circuit), UART (Universal Asynchronous Receiver / Transmitter), CAN 2.0 (Controller Area Netwokr), SPI (Serial Peripheral Interface), I2S (Integrated Inter-IC Sound), RMII (Reduced Media-Independent Interface), PWM (Pulse Width Modulation) and more.

Table 4.6: NRF24L01 Features.

Frequency range	2.4 - 2.5GHz ISM band
Data Rate	250Kbps / 1Mbps / 2Mbps
Maximum output power	0dBm
Power supply	1.9 - 3.6V
Maximum current	12.3 mA
Standby current	22 uA
Logic Input	5V
Coverage	100m (open space)

4.3.2 NRF24L01

The nRF24L01 transceiver module uses the 2.4 GHz band and can operate with transmission rates from 250 kbps up to 2 Mbps. When used in open spaces and at lower baud rates, its coverage can be up to 100 meters. Tab. 4.6 summarizes its parameters. The module can use 125 different channels which give the possibility to have a network of 125 nodes operating independently. Each channel can have up to 6 addresses, and each unit can communicate with up to 6 other units simultaneously. The power consumption of TX operation is about 12 mA while the module operating voltage can vary from 1.9 to 3.6 V. The other pins tolerate 5V logic, and thus can be easily connected to a microcontroller, such as ESP32, without using any logic level converter. Three of these pins are for SPI communication and must be connected to the SPI pins of the ESP32 module.

The NRF24L01 network stack was mainly implemented to give an open source and low cost solution to the Zigbee standard. It provides three layers and communication modes, which are internally connected: (i) Point-to-

Point, (ii) Network and, (iii) Mesh . Specifically, the Mesh network is based on the static network which relies on multiple Point-to-Point communications. The main features of the NRF24L01 network are:

- Network ACKs: Efficient acknowledgement of network-wide transmissions, via dynamic radio acks and network protocol acks.
- Updated addressing standard for optimal radio transmission.
- Extended timeouts and staggered timeout intervals. The new txTimeout variable allows fully automated extended timeout periods via auto-retry/auto-reUse of payloads.
- Optimization to the core library provides improvements to reliability, speed and efficiency.
- Built in sleep mode using interrupts.
- Host Addressing. Each node has a logical address on the local network.
- Message Forwarding. Messages can be sent from one node to any other, and this layer will get them there no matter how many hops it takes.
- Ad-hoc Joining. A node can join a network without any changes to any existing nodes.

The network layer takes advantage of the fundamental capability of the nRF24L01 radio to listen actively to up to 6 other radios at once. The network is arranged in a Tree Topology, where one node is the base (Master), and all other nodes are children either of that node, or of another. In the static version, nodes have a static address which cannot be modified. The mesh layer, instead, provides extended features, including automatic addressing and dynamic configuration of wireless sensors. Specifically, nodes are assigned a unique number ranging from 1 to 255, which is used to communicate at a high level within the RF24 communication stack and will generally remain static.

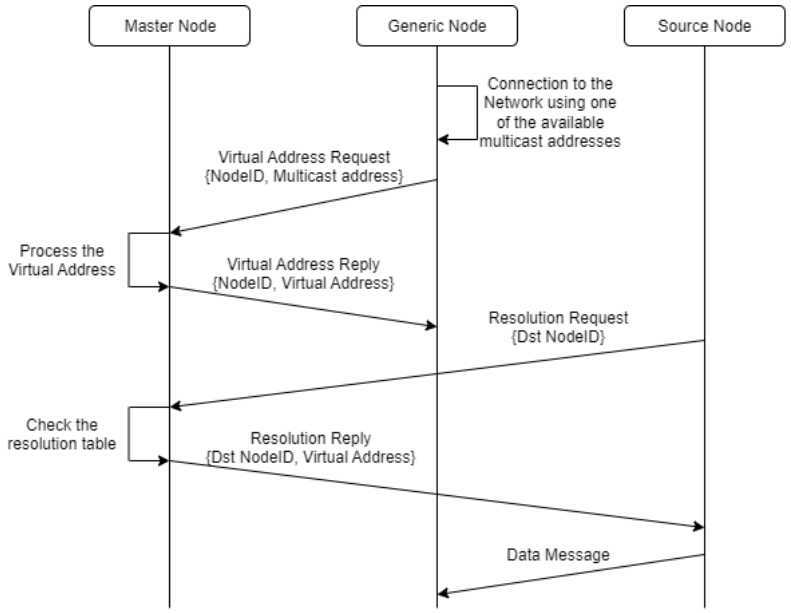


Figure 4.7: NRF24L01 Joining and resolution process.

At the network layer, the physical radio addresses, similar to MAC addresses, are allocated as nodes move around and establish connections within the network. The Master node keeps track of the unique nodeIDs and the assigned Network addresses. When a node is moved physically, or just loses its connection to the network, it can automatically re-join the mesh and reconfigure itself within the network. In the mesh configuration sensors/nodes can move around physically, far from the Master using other nodes to route traffic over extended distances. Addressing and topology is reconfig-

Table 4.7: Address Resolution Table.

ID	Node ID	Destination Address	TTL
1	001	05555	100
2	005	02334	80
3	....	....	...

ured as connections are broken and re-established within different areas of the network. The Mesh network takes advantage of functionality and fea-

tures within the P2P and Network layers, so everything from addressing, routing, fragmentation/re-assembly (very large payloads) are handled automatically with processes designed to support a multi-node radio network. For each transmission, the source node must know the virtual address of the destination node. This is accomplished through a procedure similar to the Address Resolution Protocol (ARP), which is depicted in the sequence diagram of Fig. 4.7 . Specifically, the request containing the nodeID and the multicast address used for the network connection, is sent by the source node to the master node, which replies with the virtual address of the destination node. In case of intermittent transmissions, this procedure increases the traffic overhead and for large WSAN this cannot be acceptable. In our model this issue has be resolved, enabling the locally saving of the resulted destination virtual address in a internal resolution table, as show in Tab. 4.7. Each entry has a timeout which defines its lifetime and can be updated only if the transmission is correctly acknowledged. In case of three consecutive no acknowledged transmissions, the source node has to start a new resolution process.

4.3.3 Performance Evaluation

The performance of the proposed approach depends on the number of the enabled SDN flows, the network size and the tree levels. The graphs of Fig. 4.8 and Fig. 4.9 have been obtained using a network with the listed characteristics of Tab. 4.8.

Table 4.8: Network features

Network Size	20 nodes
Report message period	10 seconds
Beacon message period	5 seconds
FWDR message period	5 minutes
Testing Period	30 minutes

Specifically, Fig. 4.8 shows the overhead of the control packets sent every minute in the network for a time interval of 30 minutes. The average number of report, beacon and FWDR messages per minute is about 365 with a maximum value of 389. The SDN controller periodically enables a SDN flow between a source and a destination node. The path is randomly chosen,

Table 4.9: WSN and SDN Forwarding Rules Percentage

SDN paths [%]	**WSN paths [%]**
100	0
50	50
25	75
0	100

and it is activated by sending the FWDR messages to all the nodes of the selected route.

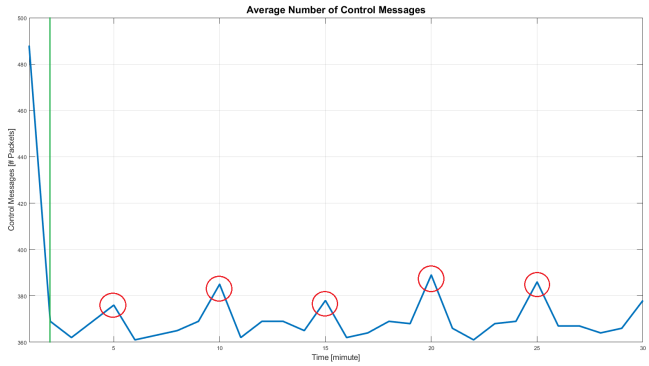


Figure 4.8: Control messages overhead.

The increment of the exchanged control messages caused by the SDN flow activation is highlighted in the figure by the red circles. Finally, the green line defines the end of the network building process which represent the phase with the maximum number of exchanged control messages. Fig. 4.9 shows the data delivery rate in case of a dynamic scenario. Specifically, the results have been obtained choosing the same source node and varying the number of installed SDN paths to the other nodes of the network, as shown in Tab. 4.3.3. The green line specifies the instant in time in which the source node changes its physical position.

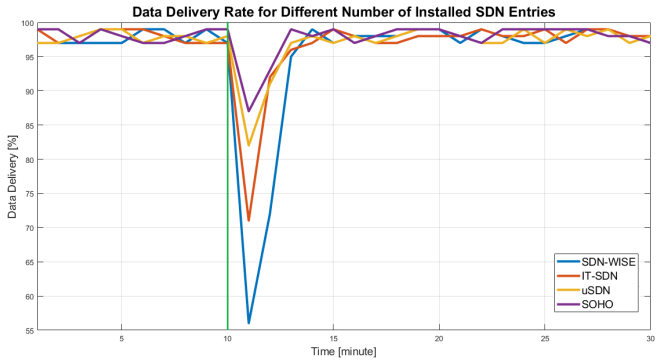


Figure 4.9: Data Delivery Rate in case of Controller failure with different number of installed SDN rules.

In can be notice that the packet loss grows as the enabled SDN paths increasing. This is mainly caused by the higher convergence time of the SDN paradigm than distributed algorithm since the Controller has to update its network view and send new forwarding rules.

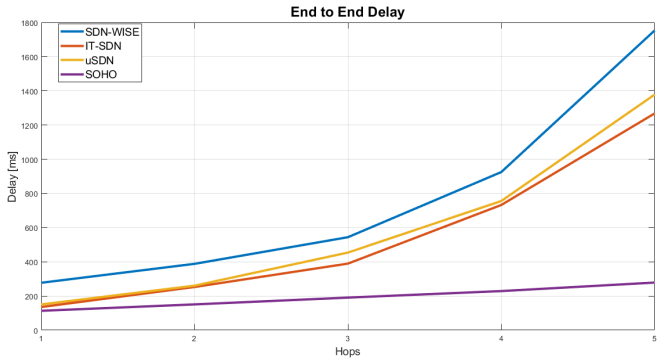


Figure 4.10: Comparison of the end to end delay.

Vice versa, the traditional WSAN promptly recognizes the network change and limits the number of loss packets. The following results have been obtained comparing the proposed model with three existing solutions, SDN-WISE, IT-SDN and, uSDN. Fig. 4.10 depicts the end-to-end delay in a multi-hop network of 25 nodes. Specifically, it shows the delay between a source

and a destination node for different hops in case of a new communication. Unlike the existing approaches, the proposed one is built on a WSAN and its functioning is independent to the SDN Controller.

Figure 4.11: Comparison Data Delivery Rate.

The communication between two nodes is performed without the Controller interaction and therefore the delay due to the packet in packet out process is cancelled. In the other solutions, the source node has to request the route to the controller, and this degrades the overall performance. It has to be noticed, that after the SDN rules installation all the models approximately reach the same results in terms of delay and data delivery as shown in Fig. 4.11. In Fig. 4.12 the control packet overhead of different network sizes is shown. The results has been obtained evaluating the average number of control messages exchanged during a simulation of 30 minutes. In order to consider the SDN impact, a predefined source node has been set up to periodically send a data message to a random destination node. The IT-SDN solution has the worst result while the growth of control packet of the proposed approach is lower than the others. This is mainly due to the lower impact of new SDN flows installation to the overall overhead and to the lower number of control messages of the mesh network. The decoupling between the SDN logic and the underlying WSAN improves the global network availability and its robustness to Controller failure. This benefit is shown in Fig. 4.13 where the Data delivery rate of each approach in case of Controller malfunction is analysed. Specifically, the resulting graph has been processed considering a periodic sending of data packet from a specific source node to

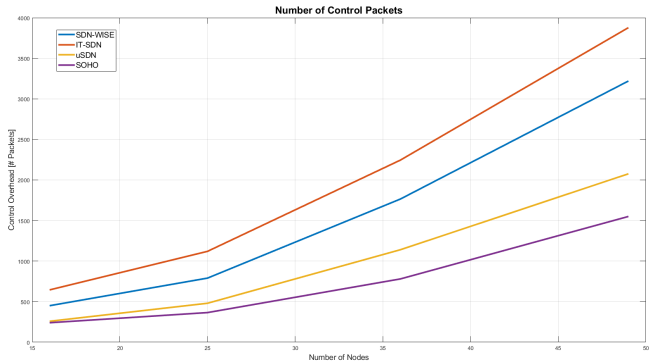


Figure 4.12: Comparison of the control message overhead.

another random node of the network. After the Controller fault, identified by the vertical green line, the SDN-WISE, IT-SDN and, uSDN solutions cannot correctly provide a path from the source to the destination node and therefore the delivery rate drastically decreases. However, the proposed approach detects the failure and automatically switches to the traditional WSAN flow management ensuring an higher data delivery rate. The management of the

Figure 4.13: Comparison of Data Delivery rate in case of SDN controller Failure in static scenario.

Controller failure has been finally tested in dynamic scenario. In Fig. 4.14

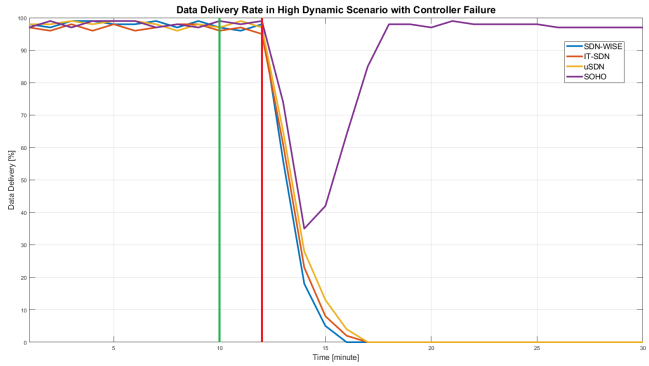


Figure 4.14: Comparison of Data Delivery Rate in case of Controller Failure in dynamic scenario.

the Controller breakdown (green line) is followed by a topology change (red line) in which the source node moves to another physical position. Like the previous graph, the SDN-WISE, IT-SDN and uSDN solutions cannot resolve a controller failure while the proposed approach can guarantee continuity of the operation with a limited performance degradation. Particularly, in SOHS the network changes are promptly detected and the packet loss per node is limited to a fixed number of unacknowledged messages. The achieving of the above-mentioned threshold triggers the request of the virtual address of the destination node which is then used for the correct transmission.

4.4 Summary of the contribution

This chapter proposed an alternative SDWSAN scheme which rethinks the integration of SDN within the wireless sensor and actuator networks. Unlike existing approaches which are completely dependent on the SDN Controller, the presented SOHS solution combines the distributed procedures of WSANs with the centralized management of SDN by proposing an hybrid dual stack architecture. The results show a significant performance improvements w.r.t the main SDWSAN systems in terms of availability, overhead and, end-to-end delay especially for large scale networks with rapid topology changes. Although, the presented model has been implemented and tested using spe-

cific devices and modules, it can be extended to other WSAN technologies. To this purpose, SOHS is not only a powerful SDWSAN solution but represents an innovative design philosophy to efficiently integrate SDN in the WSAN domain.

Chapter 5

Dynamic Software Defined Optimization in wireless IoT scenarios

SDN provides a new perspective for the Internet of Things (IoT), since, with the separation of the control from the data planes, it is viable to optimise the traditional networks operation management. As a consequence, devices, usually facing limited communications and computing resources, are relieved of the route selection task in a distributed and, thus, suboptimal way. This chapter proposes a SDN-IoT architecture, specifically focusing on the Controller design, which dynamically optimises in real time the end-to-end flows delivery. In particular, the dynamic routing policy adaptation is based on the real-time estimation of the network status and it allows to jointly minimise the end-to-end latency and energy consumption and, consequently, to improve the network life time. The performance of the proposed approach is analysed in terms of the average latency, energy consumption and overhead, pointing out a better behaviour in comparison with the existing distributed approaches.

5.1 Introduction

The Internet of Things (IoT) represents a unifying paradigm that aims at automating business processes, since *smart* things are connected at any place/time via a *network of networks* and empowered with data analytic features [17, 114, 126]. This concept could be actualised to several areas, including smart homes/buildings, smart cities, intelligent healthcare, smart traffic, smart environments to name a few [117]. To this end, many enterprises leverage on IoT devices to collect relevant and heterogeneous data and better manage their processes, however, two major challenges need to be faced for the expansion of the IoT.

First of all security issues become more complex in the IoT use case, because (i) usually unattended and constrained devices are extremely vulnerable, (ii) access networks can be violated to enter the system, and (iii) an attack to the Cloud services can alter the end-to-end (e2e) application logic. The other main challenge is represented by the network connectivity, since it is required for each of the devices to be able to connect in an efficient way.

The above interrelated challenges can be answered by applying the software-defined networking approach [64,83]. The advantages achievable by the SDN paradigm rely on the fact that the Controller maintains a *global view* of the underlying network, so it can dynamically optimise e2e data flows according to the state of the network, integrating both Cloud and Fog computing domains [6, 74].

The SDN principle could be perfectly paired with the IoT as an effective way to solve its actual limitations [81, 99]. In particular, IoT devices could be secured when they are operated on a separate network. This is because it makes more sense to secure the devices at the network level, rather than at the individual device level. Specifically, the application of the SDN principle allows to dynamically partition a *physical* network into different *virtual* networks, as it has been previously achieved through Virtual Local Area Network (VLAN) protocols to manage broadcast domains in an efficient and flexible way. Moreover, SDN represents a more general model for separating at a more higher level, i.e., to translate service requirements into an actual network configuration [104]. In addition to this, many of the IoT applications are time-sensitive, and, consequently, the optimisation of connectivity with the storing/processing resources is a priority performed, while achieving the goal of load balancing. This is effective especially when it is expected to have the computing nodes *near* where the data are being used, due to

severe jitter or latency requirements [91]. In this perspective, the combined use of IoT with SDN is one of the main catalysts for pushing resources to the *edge* of the network and simplifying the interaction with a distributed and dynamic Application Plane (AP).

However, the main constraint of a typical IoT domain is concerned with energy limitation, as smart devices are requested to perform battery intensive sensing/communications tasks, likely to greatly increase, due to the wide and pervasive adoption of this paradigm [132]. It compels the adoption of efficient energy management policies, which is commonly addressed as *green* IoT, where this means a reduction of devices energy consumption of IoT in order to fulfill a smart world with sustainability [10, 30].

To this purpose, the proposed approach effectively introduces the SDN concept into a typical IoT ecosystem in order to optimise the e2e flows delivery by means of a centralised and agile multifaceted forwarding strategy. In particular, the formulated objective function trades off flows delivery latency and overall network survivability, as it focused on typical IoT constraints, mainly in terms of residual battery available at each device. As a consequence, the SDN Controller, upon continuously monitoring the underlying network status (mainly in terms of both available communications links and remaining battery level) and QoS requirements, is able to evaluate the best path by executing the Dynamic Routing Module (DRM) and to update the forwarding tables of the involved devices with a limited overhead, when compared with traditional distributed approaches, regardless the nodes deployment.

The rest of the chapter is organised as follows: the proposed architecture and forwarding strategies are characterised in Section 5.2, whereas its performance is validated in Section 5.3 by means of numerical results. Finally, in Section 5.4, conclusions are drawn.

5.2 Proposed Approach

The proposed SDWSAN architecture, combining the SDN paradigm with the main traditional WSAN elements, is shown in Fig. 5.1, where it is possible to notice the classical three-layers arrangement (i.e., DP, CP and AP). Differently from traditional WSANs, in which nodes resort to distributed routing protocols, the SDWSAN architecture can dynamically adopt centralised and specific forwarding strategies by simply allowing the SDN Controller to *anno-*

tate the *network graph* for improving the overall system flexibility, reliability, and lifetime. In particular, the proposed approach is focused on dynamically adjusting the routing strategy based on the real-time network state for enhancing the network *responsiveness* and its *energetic footprint*. It is worth noticing that this policy being application-oriented is able to support time varying and heterogeneous QoS requirements usually originating when the Cloud and Fog/Edge architectures are effectively integrated. To take into ac-

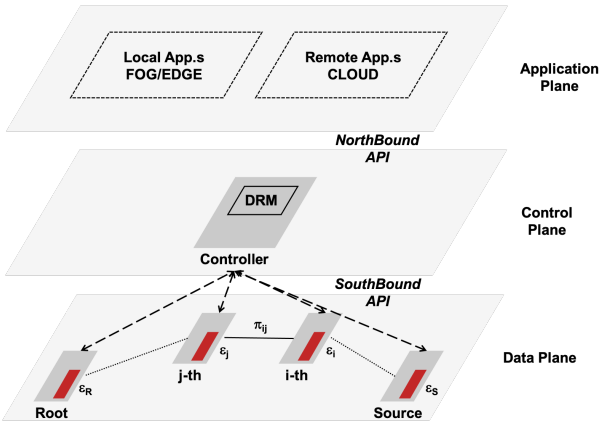


Figure 5.1: Reference SDN-IoT architecture, where it is pointed out the presence of Application, Control and Data Planes. AP is comprised of both local and remote applications, while CP is composed of a Controller with endowed the routing module (DRM). Finally, generic sensor nodes are within DP, where only Root is physically connected to the Controller.

count the above mentioned goals, the typical one way latency requirements of time sensitive applications have been combined with the energy constraints of a green IoT domain, by focusing on the following optimisation problem:

$$\begin{aligned} \underset{w_{i,j}}{\text{minimize}} \quad & \frac{\lambda(s,d)}{\theta(\pi_{s,d})} \\ \text{subject to:} \quad & s,d \in \mathcal{G} \\ & i,j \in \pi_{s,d} \\ & \phi_{s,out} = \phi_{d,in} \end{aligned} \tag{5.1}$$

where $\mathcal{G}$ represents the overall network graph, s,d a generic couple of source-destination nodes, $\pi_{s,d}$ a possible path connecting s to d, and i,j two generic adjacent nodes along $\pi_{s,d}$. Moreover, $\lambda(s,d)$ represents the *one way* latency needed to deliver a messages *flow* from s to d nodes, while $\theta(\pi_{s,d})$ is the minimum time interval in which all the nodes along the path $\pi_{s,d}$ are jointly available. Finally, the last constraint is imposed to preserve the end-to-end flow from the source ($\phi_{s,out}$) toward the destination ($\phi_{d,in}$). It is worth recalling that in principle the solution of the optimisation problem (5.1) can be easily obtained by applying the Djkstra algorithm, once a proper set of weights $\mathbf{w}$ is applied to every link. To this purpose, a generalised time varying weight $\mathbf{w}(t) \doteq \{w_{i,j}(t)\}$, associated to the link connecting the i-th and j-th nodes, has been introduced as:

$$w_{i,j}(t) = w_\tau(t)\frac{\tau}{\tau_{\max}} + w_\epsilon(t)\frac{\epsilon}{\epsilon_{\max}} \tag{5.2}$$

where τ is the number of hops associated to the e2e path, ϵ is the residual battery level of the destination node, $\tau_{\max}$ is the maximum number of hops, $\epsilon_{\max}$ is the maximum battery level, while $w_\tau(t)$ and $w_\epsilon(t)$ represent the weighting factors to each terms, such that $\forall t$, $w_\tau(t) + w_\epsilon(t) = 1$. In particular, $w_\tau(t)$ and $w_\epsilon(t)$ could be dynamically updated according to two different schemes:

- *reactive* mode: an application can directly communicate with the SDN Controller and asynchronously compel the adoption of a shortest path, due to real time constraints. In this case it is accomplished with $w_\tau(t) \gg w_\epsilon(t)$, whereas the condition $w_\tau(t) \ll w_\epsilon(t)$ is usually verified.

- *proactive* mode: the SDN Controller autonomously modifies the weighting factors based on its own dataset of network related information.

It is worth noticing that, according to the proposed unifying formulation, the hop-based mode and the energy-forwarding strategies represent two special cases that can be easily implemented setting $w_\tau(t) = 1$ and $w_\epsilon(t) = 0$, or $w_\tau(t) = 0$ and $w_\epsilon(t) = 1$, respectively.

Since battery consumption is mainly affected by packets transmission and reception, the SDN takes into account this drawback by adapting the *control* messages rate, typically reducing it when the residual battery is below a certain critical level. Specifically, this is gradually accomplished using two thresholds:

- *soft*: indicates that the node is no longer in a full energy state and, consequently, a moderate control message reduction can be applied.

- *critical*: in this case the node has a limited residual battery and a more stringent energy management is necessary to avoid its shut down.

When T_s is reached, the sending periods of `Beacon` and `Report` messages are exponentially increased. Further, when T_c is overcome, the above mentioned periods are again increased according to an exponential rule. This approach allows the nodes to maintain an updated *upstream* path to the Sink node (and to the Controller) and, publish their data to an external data center (e.g., a Cloud), without significant performance degradation. The network view achieved by the SDN Controller depends mainly on the sending period of the `Report` messages. A high sending rate reduction can cause an update delay of the Controller's view and remarkable differences from the actual WSAN. This situation may involve a considerable Packet Loss Ratio (PLR) and, a greater Round Trip Time (RTT), especially in a mobile scenario.

Additionally, the SDN Controller can also manage the node *duty cycle* (DC) by varying the duration of it *active* and the *sleep* states, where the latter one usually is increased as the residual battery decreases. This procedure is usually adopted when the critical threshold is approached.

In order to enable the DC procedure, the SDN Controller sends a specific message, called `Info` packet, which contains the sleeping period. Upon receiving this message, a node replies with an `ACK`. After the `ACK` reception, the Controller supposes that the specific node is going to sleep mode. In the suggested implementation, the sleep interval of DC has been set equal to the transmission period of `Data` messages, thus, nodes will be active only for few seconds needed for data transmission. There are several routing policies that an SDN Controller is able to adopt, depending upon the choice of the time varying weights $w_{i,j}(t)$ associated to each link, as expressed in the optimisation problem (5.1). An affordable solution could consider those weights as *constant* over time, resulting in a *static* strategy. Focusing on the QoS constraints expressed in the optimisation problem (5.2), two opposite routing schemes are possible:

- MinHop: this algorithm minimises only the number of hops between source and destination nodes (i.e., $w_\tau(t) = 1$, $w_\epsilon(t) = 0$, $\forall t$), thus being efficient especially in terms of latencies and overhead, but without

taking into account the effect of energy consumption.

- MaxBattery: this approach merely monitors each node residual battery level and minimises its consumption in order to optimise the overall network lifetime, that is $w_\tau(t) = 0$, $w_\epsilon(t) = 1$, $\forall t$. It also includes power-saving additional techniques, such as the packet rate reduction and the duty cycling.

In general, when taking into account both the energy consumption and delivery latency, a hybrid strategy is to be adopted, but always following a *dynamic* approach. Specifically, this scheme is integrated on board of the SDN Controller within the so called Dynamic Routing Module (DRM), which performs a real-time estimation of the network status, integrating the local information received by each node (i.e., neighbours list, link quality indicator and battery level), with a sampling frequency which depends on the e2e control signalling delay. Besides, the DRM is capable of updating the global weights and to apply the correct strategy, as again detailed along the Algorithm 1.

Initially, the SDN Controller modifies the traffic flows minimising the path length and updates the network status according to the control messages sent by the nodes. Specifically, it changes the status of the nodes following the T_s and T_c thresholds. If the sum of nodes in soft or critical status is greater than the remaining ones, the DRM deactivates the default strategy and starts to consider the battery levels in the traffic path evaluation. Finally, when the soft nodes will be lower than the number of nodes in critical status, the algorithm mostly optimises the residual energy and accordingly performs traffic redirection. It could be noticed that the proposed approach is close to the MinHop one if the number of nodes in soft or critical state is lower than the remaining ones, while it is similar to the MaxBattery strategy, whenever the number of nodes with a critical battery level is greater than the ones with soft energetic level. In conclusion, the proposed DRM aims at jointly increasing the network lifetime, achieving a more uniform battery consumption, and keeping as low as possible the communication latencies between any couple of nodes.

Require: Sensor nodes, and soft (Ts) and critical (Tc) thresholds
Ensure: Tradeoff between latency and energy consumption
1: Initialization of the MinHop Strategy
2: Setting $w_\tau = 1$ and $w_\epsilon = 0$
3: Initialization of a Network Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.
4: A Battery state vector B is set as an empty array. Each next element b_i in B is a list of two elements (address a, battery state s).
5: **while** *TRUE* **do**
6: update $\mathcal{G}$ and B with the information received by the i-th node (neighbours list, link quality indicator and its battery level b_l)
7: **if** $b_l > Ts$ **then**
8: $b_{is} = 0$
9: **else if** $b_l \leq Ts$ and $b_l > Tc$ **then**
10: $b_{is} = 1$
11: **else**
12: $b_{is} = 2$
13: **end if**
14: **if** $\sum_{b_i \in B}^{N} b_{is=1} + \sum_{b_i \in B}^{N} b_{is=2} < \sum_{b_i \in B}^{N} b_{is=0}$ **then**
15: Continue with the MinHop Strategy
16: **else if** $\sum_{b_i \in B}^{N} b_{is=1} + \sum_{b_i \in B}^{N} b_{is=2} \geq \sum_{b_i \in B}^{N} b_{is=0}$ and $\sum_{b_i \in B}^{N} b_{is=2} < \sum_{b_i \in B}^{N} b_{is=1}$ **then**
17: Deactivate the MinHop Optimization Strategy
18: Update w_τ and w_ϵ values
19: Update node's flow entry of active flows
20: **else if** $\sum_{b_i \in B}^{N} b_{is=1} + \sum_{b_i \in B}^{N} b_{is=2} \geq \sum_{b_i \in B}^{N} b_{is=0}$ and $\sum_{b_i \in B}^{N} b_{is=2} \geq \sum_{b_i \in B}^{N} b_{is=1}$ **then**
21: Setting $w_\tau = 0$ and $w_\epsilon = 1$
22: Activate the MaxBattery Strategy
23: Update node's flow entry of active flows
24: **end if**
25: **end while**

Algorithm 1: Dynamic Routing Module.

5.3 Performance Analysis

The performance of the proposed DRM strategy is evaluated by means of numerical simulation resorting to Cooja simulator. The nodes are spatially distributed according to a *bottleneck* topology which represents the worst case scenario for network lifetime. Specifically, only few nodes fall inside the Sink coverage area, thus they are supposed to forward to it the messages originated by other devices, which, instead resort to a multi-hop communications. As consequence, the nodes belonging to the *first* tier are prone to

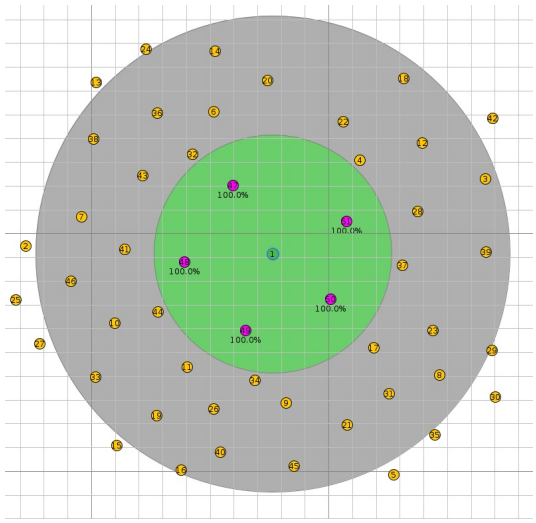


Figure 5.2: Bottleneck Topology with 50 randomly deployed nodes.

quickly deplete their batteries. The simulation playground is 200 m^2 and the transmission range of sensor nodes is assumed to be 100 m. An example of bottleneck topology with 50 nodes in the playground is shown in Fig. 5.2. All nodes are characterised by an identifier (ID) and a colour. The green node represents the Sink while purple and yellow elements identify all the other devices. Specifically, purple nodes are the only ones inside the Sink coverage area and subject to a higher packet retransmission. In particular, the system has been modelled by considering the MICAz mote model in terms of CPU states, transmit and receive power levels, as reported in Tab. 5.1. In addition to this, it has been assumed that the `Beacon` and

`Report` messages rates are increased respectively to 20 and 40 seconds, as soon as the T_s threshold is matched, while, when T_c is overcome, the above mentioned rates are, respectively, updated to 30 and 60 seconds. Moreover,

Table 5.1: MICAz Mote Current Consumption

Microcontroller		Transceiver	
Active	1.8 mA	RX	19.7 mA
Idle	20 uA	TX(0dBm)	17.4 mA
Sleep	5 uA	TX(-5 dBm)	13.9 mA
		TX(-10 dBm)	11.2 mA
		TX(-15 dBm)	9.9 mA
		TX(-25 dBm)	8.5 mA

after an exhaustive measurement campaign, the obtained results have shown that, in static/quasi-static scenarios, the maximum applicable sending period of `Report` messages, which does not entail a considerable degradation of PLR and RTT, is about 60 seconds. The evaluation started by focusing on the average residual energy that the proposed routing policies running on SDN Controller can achieve. In deriving the results, we assumed that all nodes present the same initial energy value except for the Sink node, which is not battery powered and, therefore, is not considered in the performance evaluation.

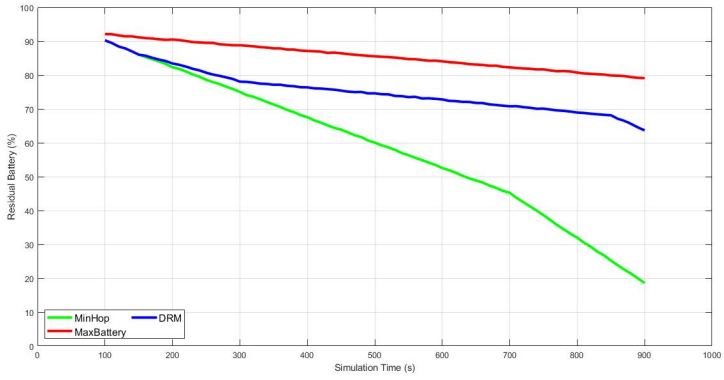


Figure 5.3: Average Residual Energy achieved by the proposed strategies MaxBattery, DRM, and MinHop.

In Fig. 5.3, the residual energy is presented with respect to the simulation time ranging from 100 to 900 s for the MaxBattery, DRM, and MinHop strategies. It is pointed out a better behaviour of the MaxBattery and DRM approaches; in particular, the battery saving achieved by these two alternatives positively affects the network life time. From Fig. 5.3, it is evident also that the performance gap between DRM and MaxBattery keeps constant, while the other strategy gets worst with time. For instance, after 800 s the gain between DRM and the MinHop strategy reaches a value nearly equal to 55%.

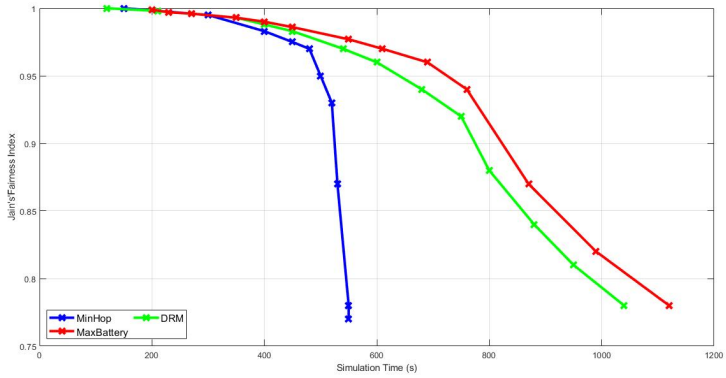


Figure 5.4: Jain's Fairness Index for the proposed strategies: MaxBattery, DRM, and MinHop.

In addition, it is important to evaluate the fairness of the proposed DRM strategy in uniforming the energy depletion of the nodes, since this property affects the network partitions. To this regard, the so-called Jain's Fairness Index is considered in the validation campaign, defined as [60]:

$$J(\epsilon) \doteq \frac{\left(\sum_{i=1}^{N} \epsilon_i\right)^2}{N \sum_{i=1}^{N} \epsilon_i^2} \tag{5.3}$$

where N is the number of nodes and ϵ_i denotes again the battery level of node i. The Jain's index approaches the unitary value when the residual battery level values of the nodes are extremely close. In Fig. 5.4, this parameter is depicted as a function of time to compare the proposed different

routing strategies in the case of bottleneck topology. As shown in Fig. 5.4, the energy-based strategies, i.e., MaxBattery and DRM, both achieve Jain's fairness index values closest to the unit value and, consequently, they are able to enhance the network lifetime by minimising the variance of battery consumption among nodes. Besides, the MaxBattery strategy points out an asymptotically better behaviour followed by the DRM strategy, while MinHop strategy presents a remarkable performance worsening, since it does not consider any energy consumption related information. Another relevant issue to be addressed is the SDN Controller WSAN nodes management. Specifically, the overall latency needed to monitor, update, evaluate and notify the selection of the best path represents a key parameter to be evaluated. Moreover, the process efficiency results in a better energy utilisation and, consequently, in a greater network lifetime.

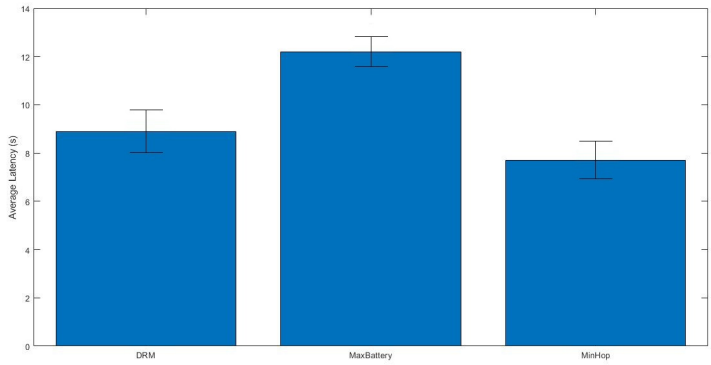


Figure 5.5: Comparison of proposed strategies MaxBattery, DRM, and MinHop in terms of Average Latency.

To this end, in Fig. 5.5, the average latency performance of the proposed strategies is shown, together with the related range of measured values. In more details, the average latency has been evaluated considering N unicast communications between the Sink node and the farthest node from it. It is easy to point out that the MaxBattery approach presents a remarkably higher latency w.r.t. the other ones, since it only takes into account the battery nodes level, while DRM dynamically adapts its strategy by considering

more the multi hop distance (in terms of w_τ) as long as the battery level of the majority of nodes is not critical.

Table 5.2: Packets Features

Packet	Type	Sending Period (s)	Lenght (Bytes)	Description
Beacon	Control	10	2	It is sent by every nodes to inform its neighbours about the distance to the Sink node and its battery level.
Report	Control	20	[3 , 108]	It is sent to the Controller to report information about battery level, distance to the Sink node and the neighbours list associated with their link quality indicator.
Data	Data	50	[1 , 116]	It can be sent by every device including the Controller. Its size cannot exceed 116 Bytes.

In Fig. 5.8, the overhead introduced by the various strategies is shown. Specifically, the results have been obtained using the three different packets shown in Tab. 5.2.

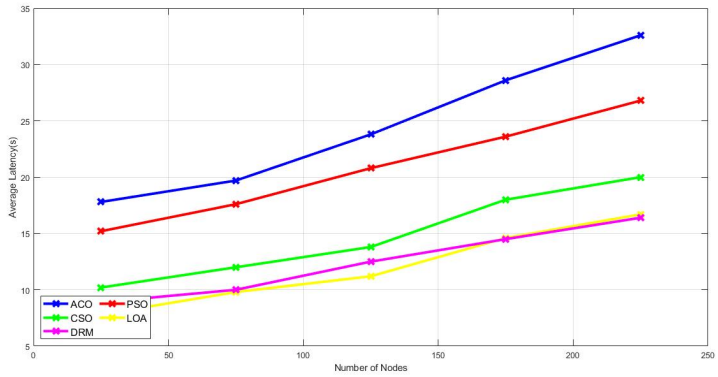


Figure 5.6: Average latency comparison for different network size.

It can be noticed that the DRM approach is slightly higher than the ones requested by the HopMin and MaxBattery solutions. This is due to the more

complex control operated by the SDN Controller in order to modify the node flow table and, hence, increasing the overall network lifetime.

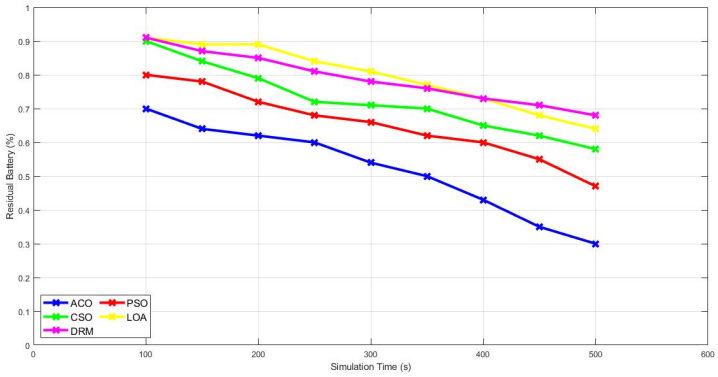


Figure 5.7: Average energy consumption comparison.

From the previous analysis it can be intuitively argued that the proposed DRM approach achieves a better trade-off between lifetime and latency.

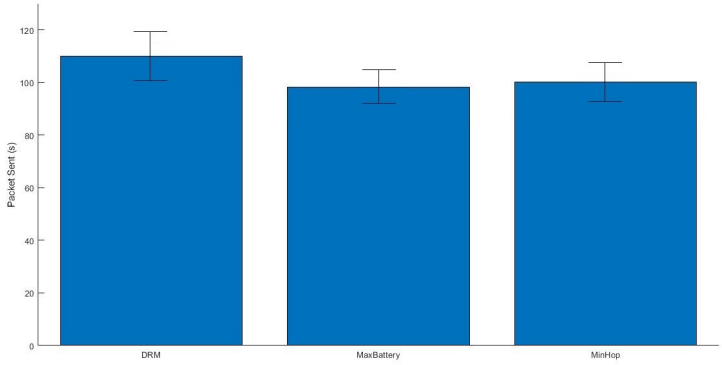


Figure 5.8: Overhead comparison for the proposed strategies: MaxBattery, DRM, and MinHop.

Specifically, DRM can reach a network lifetime comparable with the Max-Battery approach, by maintaining low latency and overhead, which represent two of the most important QoS parameters for WSANs in IoT domains. Therefore, its performance is further compared against the existing techniques (ACO [108], CSO [68], cooperative PSO [52, 128], and LOA [113]) in terms of average latency and average energy consumption, as shown in Fig. 5.6 and Fig. 5.7. In particular, in Fig. 5.6, the average latency values achieved by DRM approach and the above strategies are compared varying the network size from 25 to 225 nodes. The results points out that the DRM approach outperforms all the other strategies when the number of nodes is greater than 175. This minimal latency value is obtained by an adaptive use of the weight w_τ within the DRM, as shown in Algorithm 1.

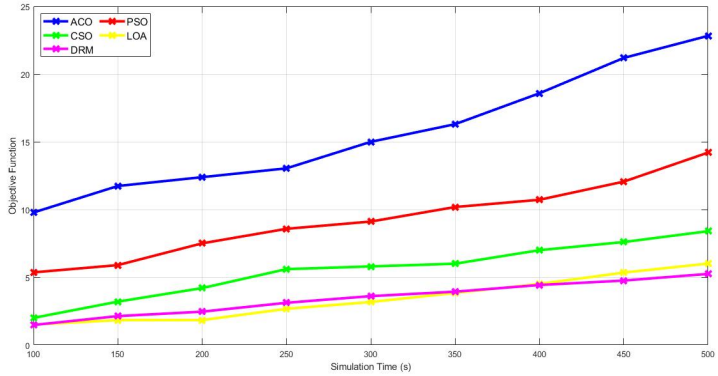


Figure 5.9: Objective Function Analysis

In Fig. 5.7 the comparison of DRM approach against the existing techniques in terms of average energy consumption is depicted for the worst case scenario of a network comprised of 225 nodes. It is possible to notice that the residual energy of all the approaches monotonically decreases. The LOA algorithm outperforms all the other solution for most of the time, but the best asymptotic performance is achieved by the DRM solution. It mainly depends on the activation of energy saving procedures operated by DRM approach, after which it is more focused on the network lifetime rather than the hops optimisation, thus reducing its decreasing slope.

Finally, Fig. 5.9 highlights the objective function values for the optimisation problem (5.1) achieved by all the approaches. Specifically, the results has been obtained with a network topology of 225 nodes. Although, the LOA strategy initially presents the better performance, DRM is, however, able to attain the best asymptotic performance.

5.4 Discussion

This chapter provided an SDN oriented framework to improve the green management of an IoT domain. In particular, it addressed on a general architecture design with a specific focus on the forwarding strategy operated by the SDN Controller, which dynamically adapts it to the network status in order to optimise the network responsiveness and energetic footprint. To this purpose, we characterise an optimisation problem, jointly based on nodes energy consumption and e2e flow delivery and we addressed a practical solution implement in the SDN Controller in the DRM, with a limited control message overhead. We conducted a comprehensive and realistic simulation campaign, pointing out that the performance achieved by the proposed approach is better than SDN alternatives based on a hierarchical CP, as in LOA. Specifically, the SDN-DRM Controller presents the best trade-off between delay and battery consumption, without excessive overhead for comparable network sizes. Therefore, the proposed framework introduces a more effective and more flexible CP for a reference SDN IoT architecture. Future developments of the present work are planned mainly in terms of introduction of network related Machine Learning techniques to predict future operative conditions to optimise in advance the weighting factors of the DRM approach.

Chapter 6

Wireless Power Transfer in a Publish-Subscribe oriented Industrial WSAN

The foundation of an energy sustainable Web of Industrial Things (WoIT) is facing several open issues due to the constraints imposed by the involved devices, the technological heterogeneity and the complex interactions and, hence, communications patterns. Towards this goal, in this chapter, a general framework inspired by the Publish-Subscribe principle have been proposed, in order to jointly optimize the service requirements and the network availability. In particular, in this chapter we focus on a holistic design with the objective to manage power budget distribution, in order to support applications that extend the basic publish-and-subscribe scheme.The involved WoIT nodes functionalities, interfaces and hardware architectures have been designed, with a special focus on control protocols. The introduced integrated solution has been validated in scenarios minimising and possibly balancing the power consumption. The achieved results show an average improvement of 45% for the communications performance with the wireless power management.

6.1 Introduction

IoT ecosystem could be considered an extremely relevant and promising component of the future industrial processes, that are alternatively indicated as the *Industrial Internet* or the *Industry 4.0*. Industry 4.0 is converting traditional industrial automation and control systems into cyber-physical manufacturing systems by integrating Information Technology (IT) and Operational Technology (OT). In this vision, industrial wireless sensors and actuators networks (IWSANs), may contribute to overcame the drawbacks and constraints of the wired communications technologies usually implemented in industrial scenarios. IWSANs decrease the installations and maintenance costs since they do not requires to deploy or protect cables even in hostile industrial areas. Moreover, they allow easy reconfiguration and, possibly, the support for devices mobility, and, consequently, the control of robot. However, IWSANs and in general the Industrial loT (IIoT) systems are becoming more complex with growing scales, which leads to a number of significant challenges that need to be addressed, such as increasing energy consumption [123, 127].

In a smart manufacturing environment, actuators and sensors are usually battery powered. This requires a specific maintenance cycle which can became a daunting task with hundreds of devices installed in harsh and hardly accessible locations. As a consequence, an energy-saving approach for IWSANs should be considered to improve the networks lifetime. In principle, two possible methodologies for supplying energy to devices can be considered: Energy Harvesting (EH) and Wireless Power Transfer (WPT) [53].

Both the approaches represent promising methodology but differ conceptually. EH converts every possible available energy source existing in a specific environment into power to supply the device. An optimum implementation of this approach requires an accurate identification of both spatial and temporal energy distribution in the operation environment in order to power up the devices in accordance with the functional requirements [7]. As a consequence, the knowledge of these parameters drives the overall system design in term of performance, complexity and functionalities [75]. Recently an EH approach based on the use of renewable energy sources has been proposed to achieve long-term and self-sustainable operations for nodes. This translates in prolonging the lifetime of IoT network especially if operating in extreme environments where the replacement of batteries can be difficult. The latter represents a great improvements with respect of applications where a mobile

charger is usually employed to provide the energy replenishment through energy downlink *energy downlink* collecting surplus power from energy-rich nodes and transfer it to energy-deficient nodes. An EH system consists of two processes: (1) energy harvesting and (2) data transmission. Since data communications modify the nodes energy, an optimal control of data and related energy flows represents a critical issue for effective communications.

On the other hand, WPT supports both battery-less and backup-battery approach, giving a higher degree of freedom for the system designers [82]. In particular, WPT enables the energy supply with a direct device activation, or the device battery recharge, respectively.

The present work is focused on an overall system architecture design for application over an IIoT scenarios. In particular, it has been considered that industrial devices (i.e., sensor and actuators) are expected to act as a group to perform homogeneous operations in terms of manufacturing and quality assessment. Then they can be logically clustered and the possible interactions among clusters needs to be optimised mostly under energetic constraints which affects their availability and life-time. In order to establish intra- and inter-cluster(s) communications, the entire system resorted to a WoT protocol, in particular, following the publish-and-subscribe pattern, according to the MQTT-SN framework. This allows to set-up extremely flexible end -to-end (e2e) sessions; to this purpose, it has been derived a solution that allows the joint management of the data flows and power budget management. Specifically, the aim is to evaluate e2e paths among Publisher(s) and Subscriber(s) minimising the power consumption. Accordingly, the proposed architectures and interfaces have been characterized, further extending this approach by taking into account also the presence of a subnetwork to handle the WPT procedure, that is aimed to dynamically distribute the available power to that nodes involved in communications. The involved functional modules and the device level architectures have been described and harmonised within the overall system.

Summarising, the main contribution of this chapter consists in:

- an high-level architecture design, integrating a typical IIoT scenario on a publish-subscribe oriented messaging framework through proper interfaces and protocols
- individual module modelling and design, with a specific focus on the power supplying process, both on a component and procedure point of view.

- development of an ad hoc simulation framework including different approaches performed over realistic scenarios.

The chapter structure is at it follows: the reference system model is presented in Sec. 6.2 in terms of both the general architecture and specific communications and control protocols, with a characterisation of hardware components. The overall performance is investigated in Sec. 6.3 for different device deployments, pointing out a remarkable gain in terms of network life-time. Finally, the conclusions are drawn in Sec. 6.4.

6.2 System Model

The proposed solution aims at optimising the overall energy consumption of a clustered IIoT scenario, leveraging the WPT concept and, consequently, maximising the services availability and system durability to support typical industrial applications. Specifically, this ha been accomplished by introducing the following elements: (i) an Energy Supplier (ES) node, which is in charge of handling the WPT procedure and possibly acting as a communication gateway toward external networks, (ii) a Rechargeable Node (RN) which represent a IIoT able to be properly recharged thank to the harvester block, and (iii) the Power Transfer Manager (PTM) which optimise the power transfer by enforcing a suitable strategy.

In Fig. 6.1, the high level architecture is depicted, where each IIoT cluster is arranged in a tree-wise topology, according to the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [31], which has been modified to select the root and lower *rank* nodes among the RNs. It is worth noting that the power consumption is mainly due to packet transmission and reception; as a consequence in each cluster the lower rank nodes are mainly affected by power constrains. Typically, the root node (rank =0) that forwards all the messages from/to all the other nodes is likely to firstly deplete their battery. For this reason, it is fundamental that the lower rank nodes were possibly placed closer to ES in order to be recharged when needed to handle an amount of messages greater than the other nodes. According to the proposal, PTM leveraging on the publish-and-subscribe message exchanging, can monitor the residual battery level for the devices that subscribed to the WPT services (i.e., RNs), schedule a WPT cycle on a weighted round robin basis for each cluster, and properly coordinating the ES circuitry. In order to

implement the proposed system architecture two specific configurations for both the ES and RN have been conceived, as pointed out in the following.

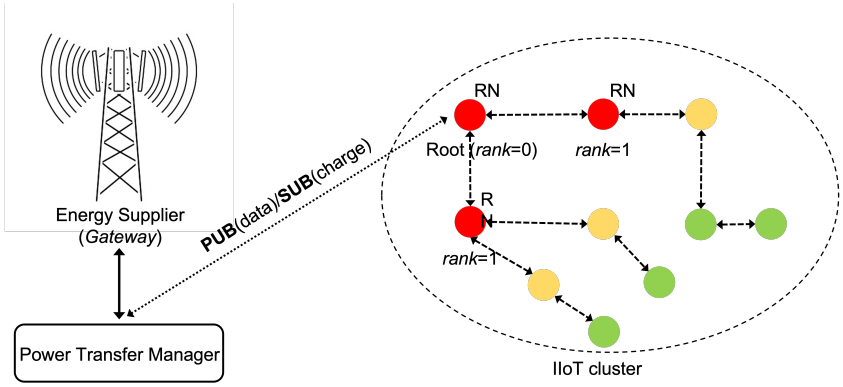
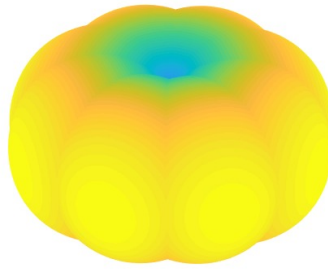


Figure 6.1: High level architecture comprising the Energy Supplier node and the Power Transfer Manager which jointly handle the WPT procedure for an IIoT cluster arranged in a tree-wise topology, in the presence of Rechargeable Nodes.

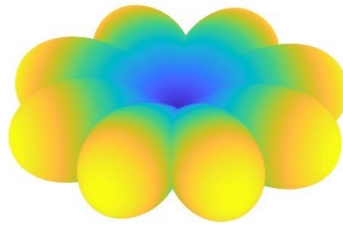
6.2.1 Energy Supplier node architecture

The ES node has been conceived to act primarily as a WPT device for supplying energy to the WSAN IoT devices and as a communication gateway. The proposed architecture is based on a dual frequency eight flat face Switched Beam Antenna (SBA) operating at both 2.45 GHz and 5.8 GHz in circular polarization (CP). CP has been implemented in order to provide robustness in both indoor and outdoor scenarios, since it can contrast multipath (indoor) and Faraday rotation (outdoor). Furthermore, CP is not affected by the terminals relative orientation, this avoiding antenna polarisation mismatching in the power link, that consequently could nullify WPT advantages. However, a single antenna cannot be an effective radiator, since the transmitting node, in order to achieve high power transfer efficiency, should focus the transmitted power to the receiver through a narrow beamwidth. In a dynamic scenario, the relative position of transmitter and receiver are not a priori known. Hence, an adequately narrow beam should be dynamically steered to adaptively direct all the available power towards the target receiver in far or near field.

SBAs are cost-effective radiative systems capable to cover an omnidirectional domain as a whole, but able to selectively switch-on the transmission toward specific directions, hence operating as spatial filter according to the Space Division Multiple Access (SDMA) paradigm. SBAs are typically controlled by means of a multiplexer, thus it is simpler and more cost-effective than electronically steered array. The better solution for both communication and WPT purposes is to adopt a dual band, where each operation is carried out at different frequency. In view of these considerations, a Dual



(a) 2,45 GHz



(b) 5.8 GHz

Figure 6.2: 3D SBA composite radiation patterns of the Energy Supplier node.

Band Switched Beam Antenna (DBSBA) has been designed and simulated. The basic antenna element considered in this chapter is based on an approach already demonstrated in developing a dual band antenna for indoor

communication [32]. The latter being designed for dual band WiFi operation working 2.45 GHz and 5.2 GHz [78]. The SBA final arrangement is obtained introducing the octagonal shape, which is sufficient to cover the entire 2D plane considering the performance of the antenna. Fig. 6.2 shows the 3D simulated radiation pattern for the SBA at the considered frequencies.

The block diagram of the ES proposed architecture is shown in Fig. 6.3, where it is pointed the presence of both the communication and power transfer chains operating at the two frequencies before introduced. The above presented solution defined for the ES is capable to supply power to RNs and particularly to the root node, as well as to communicate with the cluster. The two chains interact with the SBA through eight parallel double frequencies Diplexer driven by two single-pole eight-trough SP8T switch. These ones are connected, respectively, in the 2.45 GHz communication chain to a half-duplex transceiver, while in the 5.8 GHz one, first to a 27 dB gain Power Amplifier, with an output power of about 47 dBm, then to a 5.8 GHz transceiver used to generate a CW signal with an output power level of 20 dBm. In order to demonstrate the feasibility of such architecture, the performance of each functional block corresponds to a commercially available components, as illustrated in Fig. 6.3, where the characteristics of the block match with its specific datasheet.

As a result, ES has the ability to supply power to a target node, as well as communicate with it, while the two SP8T switch both the power and the TX/RX signals to the corresponding active lobe. In this way, the active lobe of the SBA rotates and depict the 3D radiation patterns as in Fig. 6.2. The switching time of SP8T, which correspond to 150 nsec, does not introduce significant latency in both power transmission and communication, due to his inherently negligible impact on the typical on/off time interval.

6.2.2 Rechargeable node architecture

In order to properly characterize the proposed WPT based approach, also the RN architecture has been defined. The related block diagram is described Fig. 6.4. Each RN is equipped with a 2.45 GHz communication chain adopting an omnidirectional antenna and a 5.8 GHz harvester chain using a linear directional patch antenna designed according to a non canonical optimum impedance matching. The 5.8 GHz RF power, supplied by the ES during the WPT task, is collected by the RN and turned into DC power supply. The energy conversion within the harvesting chain is obtained by means of

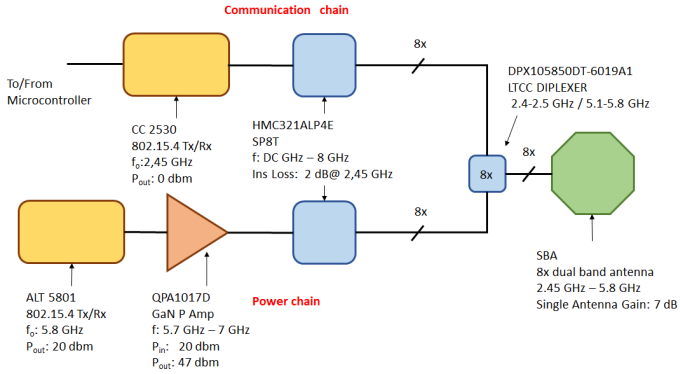


Figure 6.3: Energy Supplier node block diagram.

a solution proposed in [33].

The latter architecture, which is described in Fig. 6.4, makes use of an 5.8 GHz linear optimum matched patch antenna connected to a voltage triplier and followed by a commercial DC-DC Boost BQ25570. The optimum matching impedance is obtained by means of a source-pull optimization carried out on Keysight Advance Design System (ADS) integrating the diode model and a custom BQ25570 behavioural model. The optimum impedance being the one maximizing the BQ25570 input current and, consequently, the power transfer performance.

In the proposed solution, the antenna shows directly the optimum impedance to the voltage triplier minimizing the insertion loss and maximizing the DC-DC boost output current IDC. Taking into account a configuration as the one illustrated in [94], and considering for the present node the same output current parameter, a planar distribution of the current is made available to the node by the WPT block.

6.2.3 WPT modelling

To completely characterise the WPT process, a model describing the current which is effectively available to RN has been developed and implemented

by taking into account the results obtained from electromagnetic and ADS simulations, as well as resorting to a set of measurement illustrated in [33].

The overall final model consists in the 2D spatial current distribution illustrated in the Fig. 6.5. The procedure followed to obtain it starts with the simulated 5.8 GHz radiation pattern of the SBA Fig. 6.2. As described in Sec. 6.2.1, it has been evaluated through an EM simulation that consider the total SBA antenna as a whole. This radiation pattern has been translated in a 2D RF power distribution that depends from the planar (x,y) coordinates. The current distribution has been calculated by considering the RF to DC conversion efficiency of the RN and taking into account the ES antenna gain. In particular, the RF to DC conversion performances have been calculated basing on the measurements illustrated in [33], that correlate the RF power available at the input of the voltage triplier to the DC current. The inclusion of the ES antenna gain into the SBA 2D RF power planar distribution maps it in the 2D RF planar distribution at the input of the RN voltage triplier. The results is the 2D current planar distribution are illustrated in Fig. 6.5.

Finally, the derived current distribution has been included in a Matlab framework in order to describe the energy made available from the ES to each RN as a function of its 2D position within the described area [78, 79].

6.2.4 Power Transfer Manager

The PTM application is mainly focused on the optimisation of the power transfer process, and, consequently, the battery recharging. As previously introduced, this technique allows the GW nodes to receive the energy from ES; in particular they are equipped with ad-hoc circuitry to recharge their battery. In order to avoid energy waste and optimise the recharge process, a synchronisation procedure between the ES and the specific cluster is necessary. Each IIoT cluster is arranged in a tree topology where a unique root node that is interconnected with outer networks, and, during a proper time interval, all the GW nodes belonging to it can receive energy from ES. This is accomplished by PTA which schedules the cluster charging operations. Every GW node sends to PTA its battery level relying on the on the publish-and-subscribe communications framework, together with its Cluster ID and the current tree level (i.e., rank); this information is used to update the network view and perform the cluster selection. The selection is based on the weighted mean residual battery level of an IIoT cluster, where the weighting factor is inversely proportional to the node rank. Once the cluster

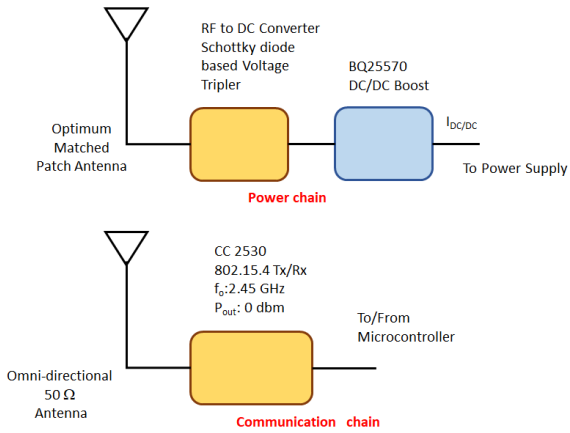


Figure 6.4: Rechargeable Node power and Tx/Rx chaines.

has been identified, PTA notifies to the intended cluster nodes the starting of the recharging process by sending a specific publish message.

6.3 Experimental Results

The performance of the proposed approach has been evaluated by means of numerical simulations conducted in the MATLAB framework. Specifically, the simulated nodes are based on IEEE 802.15.4 standard and are divided in sensors and actuators. A sensor is capable to observe homogeneous physical parameters, while an actuator can gather data from other devices and perform specific tasks. The simulated IWSAN nodes have been modelled considering the data sheet of a commercial sensor node as indicated in Tab. 6.1. Specifically, we considered a initial battery level of 2500 mAh, corresponding to 2 AAA batteries, a keep alive consumption of 1 μA, a radio transmission consumption of 17 mA and finally a reception consumption of 20 mA.

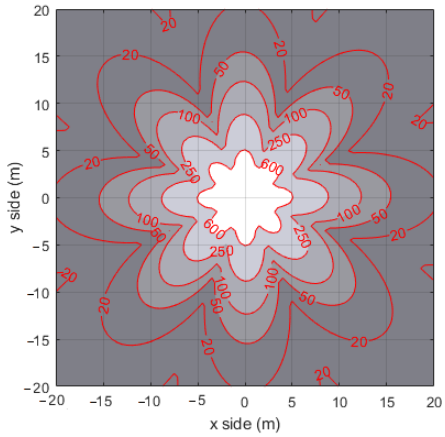


Figure 6.5: Planar distribution of RF power converted in DC (I_{DC} in uA).

Table 6.1: Parameters adopted to model a IWSAN node in simulation campaign.

	Min	**Nom**	**Max**	**Unit**
Supply voltage during radio operation	2.1		3.6	V
RF Frequency Range	2400		2483.5	MHz
Transmit bit	250		250	Kbps
Current consumption: Radio Trasmitting at 0 dBm		17.4		mA
Current consumption: Radio receiving		19.7		mA
Current consumptiom: Radio on, Oscillator on		365		μA
Current consumption: Idle mode, Oscillator off		20		μA
Current consumption: Sleep Mode			1	μA
Active current at Vcc = 3V, 1MHz		500	600	μA
Current consumption during sensing operation		15.4		mA
Voltage regulator current draw	13	20	29	μS

The transmission power mainly depends on the transmission duration, which is correlated to the packet size and to the adopted data rate. The former was set to 50 bytes, while the latter was set equal to the maximum rate allowed in the IEEE 802.15.4 standard, that is 250 kbps. Finally, the overall node consumption has been completed adding a fixed value of 15 mA for sensing operations. The CPU consumption has not been considered because it is negligible with respect to the transmission and reception power

consumption. The communication channel has been implemented by considering a static environment without obstacles. Therefore, the variation of the received signal power over distance has been characterized by mainly considering the path loss component. The WPT has been modelled considering the result shown in Fig. 6.5 and a constant recharging period of 5 seconds. This value has been evaluated considering an information publishing rate equal to 2 packet/sec and a maximum tolerable packet loss of 10 packets. Moreover, the MQTT-SN standard has been adopted as the application protocol since it represents a solution suited for IIoT in the presence of resources constrained devices, but also the WPT module has been embodied in the design of the proposed WPT-MQTT-SN. For the sake of comparison, the CoAP protocol performance have been considered.

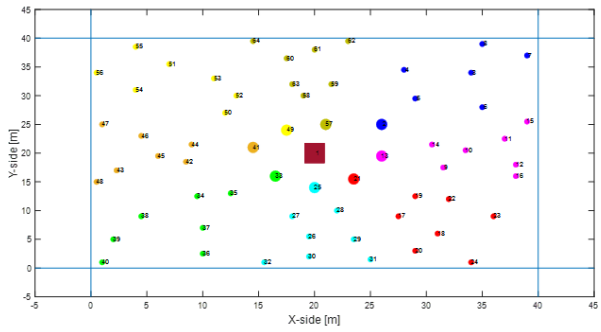


Figure 6.6: Random Mesh nodes deployment.

The network has been partitioned in logical clusters, each managed by its own CH that specifically handles the traffic from or to the ES, where the PTA is executed. The overall system performance has been evaluated in terms of network lifetime, which indicated the network duration. In particular, a strict definition has been adopted, that is the time elapsed since the first node in the network depletes its battery. Results have been obtained considering a initial battery level of 150 mAh. Furthermore, the proposed system has been tested over three different nodes deployment: (i) Random Mesh, (ii) Two-Tier, and (iii) Unbalanced Random Mesh and, for each of them it has been evaluated how the data transmission frequency impacts on the WPT procedure. Finally, the overall number of devices has been assumed to be equal to 40, in compliance with most common IIoT scenarios.

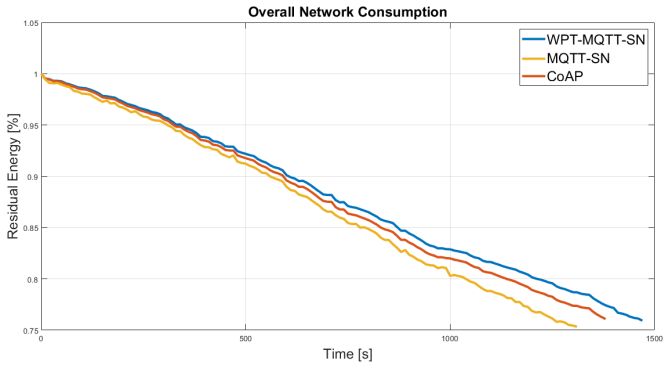


Figure 6.7: Normalised average residual battery level achieved by the CoAP, MQTT-SN and WPT-MQTT-SN protocols in a Balanced Random Mesh topology nodes deployment.

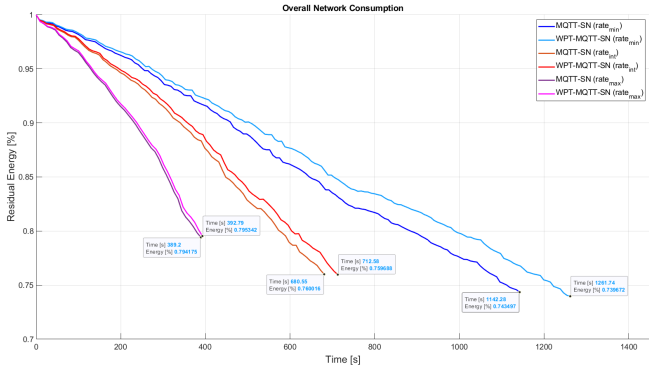


Figure 6.8: Comparison of WPT for different Publishing rates in Balanced Random Mesh topology.

First of all, the performance in the presence of a Random Mesh nodes deployment has been evaluated, which is the most common IoT scenario. In this case, a mesh topology network is originated, in which the number of hops depends on the nodes spatial distribution with a constant transmission range, as depicted in Fig. 6.6. Fig. 6.7 shows the benefits of the proposed system in terms of overall network lifetime. Specifically, it can be noticed an increasing

gain w.r.t MQTT-SN and CoAP, approaching the maximum value of 11.6% and 8.4%. Since power consumption is mainly due to packet transmission and reception, Fig. 6.8 represents the effect of different publishing rate on the overall network lifetime. Three different publishing rates ($PR_{\max}$, PR_{int} , $PR_{\min}$) have been considered, where $PR_{\min} = 2$ Publish pkt/s, $PR_{int} = 2PR_{\min}$ and $PR_{max} = 2PR_{int}$.

It can be noted that the greater improvement is achieved with the lowest rate, whereas higher rates have a negligible impact on the recharging process due to the increased nodes battery consumption. The maximum achievable network lifetime gain is finally summarized in Tab. 6.2.

Table 6.2: Network lifetime gain in a Balanced Random Mesh topology for different publishing rates.

Publishing Rate	**Gain[%]**
Minimum	9.5
Intermediate	4.7
Maximum	0.8

Lastly, Fig. 6.9 shows the residual energy of every CH, which represents the WSAN nodes that perform more radio transmissions and, thus, consume more energy. The average improvement of the WPT is about 37.9%, with a maximum value of 54.2%.

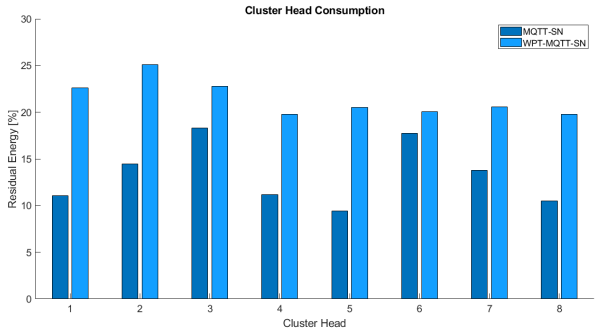


Figure 6.9: Cluster Head Energy Comparison in Balanced Random Mesh topology.

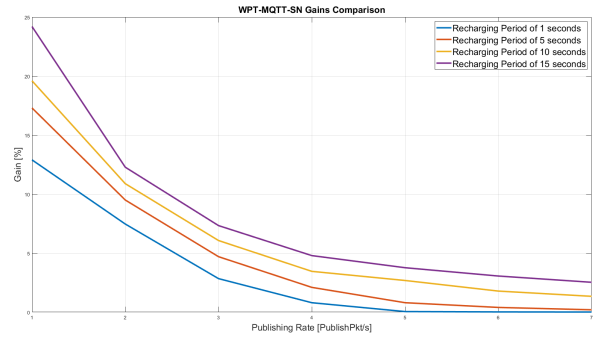


Figure 6.10: Overall Network Lifetime Gain comparison with different publishing rate and recharging period.

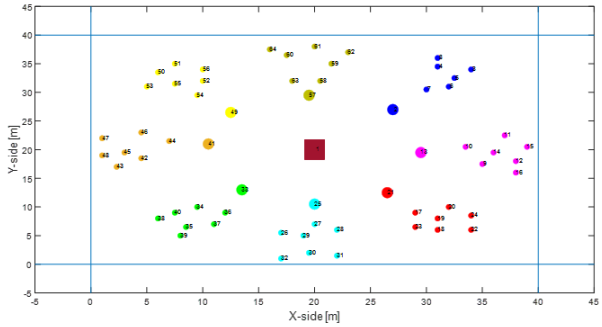


Figure 6.11: Two-Tier star-based nodes deployment.

For the sake of comparison, Fig. 6.10 shows the gain achieved by the proposed approach in terms of overall network lifetime, varying the recharging period of the RNs from 1 to 15 seconds. It can be noticed that, the WPT-MQTT-SN performance decreases at the increasing of publishing rates. However, the better result is obtained with the highest recharging period, even though this approach causes an increase of packet loss, since during the charging period, RNs cannot communicates with internal or external nodes. An alternative scenario that has been investigated consists in a Two-Tier star-based nodes deployment, in which the ES is located at the barycentre of the network surrounded by a first tier of CHs, where all the other nodes

are one-hop distant from them, as represented in Fig. 6.11.

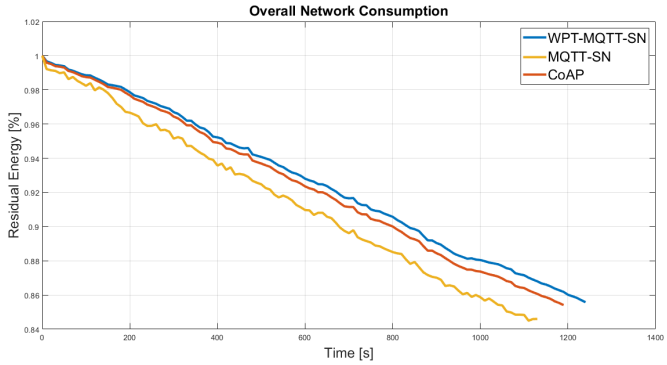


Figure 6.12: Normalised average residual battery level achieved by the CoAP, MQTT-SN and WPT-MQTT-SN protocols in a Two-Tier star based topology nodes deployment.

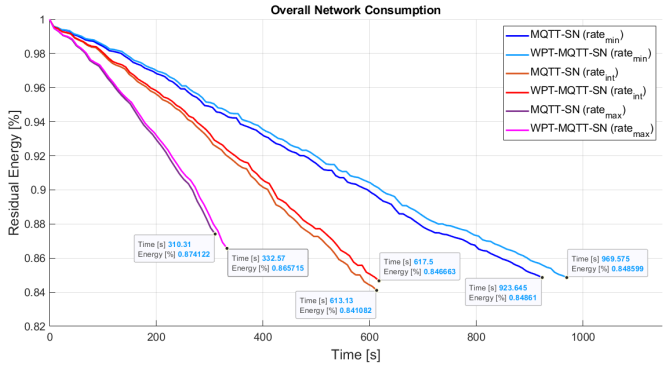


Figure 6.13: Comparison of MQTT-SN protocols without and with WPT for different Publishing rates in Two-Tier star based topology.

Simulation performed on this scenario have highlighted a reduction of the network lifetime, as depicted in Fig. 6.12. This results depends mainly on the node spatial distribution, as nodes that represent bottleneck, i.e. CHs, are more stressed. As a matter of fact, the traffic from/to single cluster is entirely managed by its CH, without the possibility of offloading the workload

according to the nodes battery level. In this case, the difference between the residual energy of the classic MQTT-SN without WPT and the proposed approach is 8.9%, while CoAP loses approximately 4.5%. Moreover, as the Random Mesh topology the performance of MQTT-SN-WPT degrades with the increasing of the publishing rate, as shown in Fig. 6.13. For the sake of completeness, the proposed solution have been tested over an Unbalanced Random Mesh nodes deployment, as shown in Fig. 6.14, where the nodes are arranged as in Random Mesh topology, but the ES is positioned at the edge of the network.

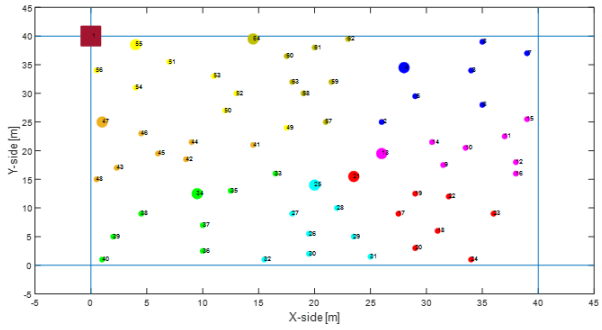


Figure 6.14: Unbalanced Random Mesh nodes deployment.

It is worth noticing that this represents a worst-case scenario, where performances degrades, but it is interesting to evaluate the effects of WPT. Since most of the CHs are far from the ES or are not in its coverage range, the network nodes cannot properly recharge their battery and, thus the overall network consumption of the proposed approach is similar to the MQTT-SN standard, as shown in Fig. 6.15. Moreover, increasing the publishing rate the network consumption gets worse for all the solution and the effect of WPT becomes irrelevant, as depicted in Fig. 6.16. In conclusion, the proposed WPT-MQTT-SN approaches is able to achieve remarkable efficiency in the presence of controlled topology where the ES node deployment can be spatially optimised, which represent the most common case in IWSANs.

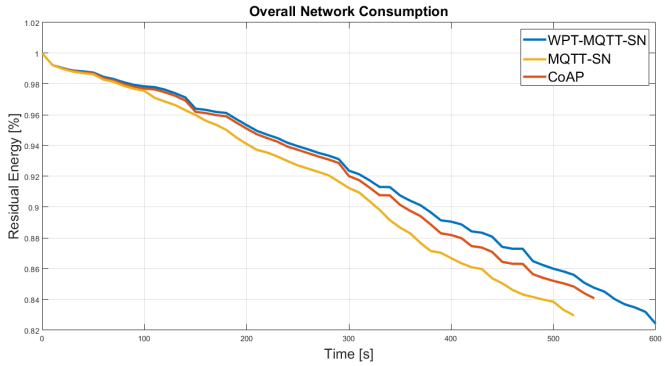


Figure 6.15: Normalised average residual battery level achieved by the CoAP, MQTT-SN and MQTT-SN-WPT protocols in a Unbalanced Random Mesh topology nodes deployment.

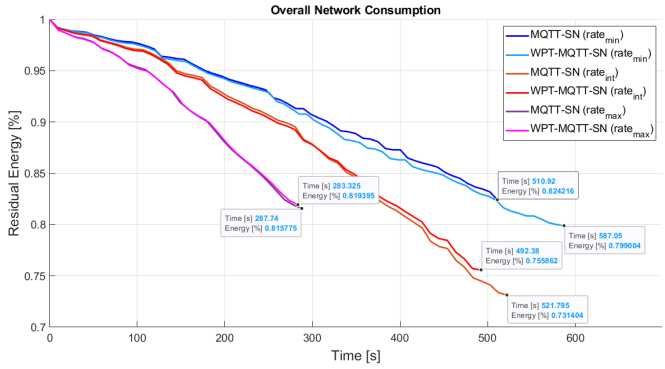


Figure 6.16: Comparison of WPT for different Publishing rates in Unbalanced Random Mesh topology.

To improve the accurateness of the results analysis, the performance achieved by the considered protocols (i.e., CoAP, MQTT-SN and WPT-MQTT-SN) over three different topologies (i.e., Random Mesh, 2-Tier and Unbalanced Random Mesh) in the same simulation time interval have been compared. The results, in terms of normalised lowest/highest residual energy and lowest/highest network lifetime are pointed out in Tab.s 6.3, 6.4,

respectively, all confirming the previous considerations within a more reliable min-max confidence interval.

Table 6.3: Normalised residual energy comparisons for different topologies and protocols.

	Random Mesh		**Two-Tier**		**Unbalanced Mesh**	
	Min	Max	Min	Max	Min	Max
WPT-MQTT-SN	0,78%	0,81%	0,83%	0,90%	0,81%	0,92%
CoAP	0,75%	0,79%	0,82%	0,89%	0,79%	0,91%
MQTT-SN	0,72%	0,76%	0,79%	0,85%	0,76%	0,83%

Table 6.4: Overall network lifetime comparisons for different topologies and protocols.

	Random Mesh		**Two-Tier**		**Unbalanced Mesh**	
	Min	Max	Min	Max	Min	Max
WPT-MQTT-SN	1043	1375	732	985	409	737
CoAP	909	1153	664	923	381	702
MQTT-SN	805	1014	596	898	305	624

6.4 Discussion

This chapter deals with the design of an IoT ecosystem for industrial applications based on Web-like protocols to accommodate complex communications schemes, and to face the technological heterogeneity. In addition, the usual constraints faced by the IoT devices are considered to provide system availability and reliability, despite the limited energetic budget. Towards this goal, a general framework inspired by the integration of the Publish-Subscribe and WPT principles have been proposed, in order to jointly optimise the QoS requirements and the network lifetime. In particular, the chapter focused on specific functionalities to handle data flows and to manage power budget scheduling to each node. The system has been accurately characterised in terms of interfaces, communications/control protocols with a specific focus on hardware architectures and components. This allowed the extension of the basic MQTT-SN protocol towards the WPT-MQTT-SN approach, that is able to optimise the energy distribution among cluster of devices, depending on the mutual interaction patterns. The integrated solution have been validated in practical scenarios and compared with existing

standard, always pointing out remarkable performance in terms of network lifetime.

Chapter 7

Software Defined MQTT-SN for Wireless Sensor and Actuator Networks

This chapter derives a general SDN-based architecture to jointly handle data flows and allocate a power budget, in order to support IoT applications that extend the basic publish-and-subscribe scheme. Specifically, the MQTT-SN standard has been considered, since it represents a solution suited for IoT in the presence of resources constrained devices. In addition to this, the Wireless Power Transfer (WPT) approach described in the previous chapter, has been introduced to effectively balance the energy consumption among the clustered devices of a wireless sensors and actuators network domain. The proposed integrated solution has been validated in practical scenarios, pointing out remarkable performance w.r.t. the end-to-end message delivery latency and the overall network energy consumption, always with limited overhead.

7.1 Introduction

According to an operation perspective, IoT is supposed to perform a *chain* of tasks by iteratively involving different groups of devices, which can be

arranged in a logical time-varying *clustered* topology. This is aligned with the Fog Computing principle which supports the design of novel services beyond the current centralised Cloud concept, where their logic distributively runs in the proximity (i.e., on board of close Fog Servers), though they are still able to interact with applications running in the Cloud [39, 89].

One of the main challenges of the previously introduced distributed services is related to the set-up, maintenance, and ad hoc integration, which can be effectively achieved by adopting the so called Web of Things (WoT) approach [102]. Indeed, Web Services based on REpresentational State Transfer (REST) have been recently adopted to achieve interoperability, particularly in the industrial Open Platform Communications Unified Architecture (OPC UA), that is widely considered one of the main reference communication protocols for information exchange among industrial applications [27, 37, 46].

OPC UA architecture integrates those protocols specifically designed for the manufacturing and automation industry, such as fieldbuses and real-time Ethernet with the Web oriented protocols, i.e., Constrained Application Protocol and Message Queue Telemetry Transport. Specifically, OPC UA Release 14 is mainly referred to the publish-and-subscribe broker-based model. However, the intrinsic nature of the publish-subscribe approach represents a bottleneck for the Wireless Sensor and Actuator Networks and can limit the benefits introduced by innovative network strategies and energy-saved techniques (e.g. WPT).

Instead, a crucial requirement in a realistic implementation of the IoT paradigm is represented by the energy available to devices in order to make them to correctly operate [56]. Wireless communications implicitly require a non-wired power source, and a usual approach consists in endow devices with a battery pack. The battery sizing is, indeed, inevitably a trade-off among different parameters, such as durability, performance and, system complexity or cost. Furthermore, this approach requires a specific maintenance cycle for battery replacements. The latter aspect is not fully compatible with an extremely pervasive wireless devices deployment in a factory environment often hard to access; as a consequence, alternative policies are expected to manage the energy requirements [3].

The Software Defined Networking paradigm can be efficiently applied in this context to both optimize the logic of existing WoT communication protocols and optimize the overall WPT procedure. To this purpose, the chapter focuses on a system level architecture design based on SDN paradigm to a

joint data flows and power budget management. In particular, wireless IoT devices (i.e., sensors and actuators) are logically clusterized, according to the homogeneous operations performed in terms of manufacturing and quality check. To address the possible intra- and inter-cluster(s) communications we resort to the WoT approach and, in particular, we adopt the publish-and-subscribe feature of the MQTT-SN framework to set up extremely flexible end-to-end (e2e) sessions. However, their management, if performed in a distributed and reactive way, does not match the specific requirements of wireless IoT services. To this purpose, we derived a solution inspired by the SDN paradigm to specifically evaluate e2e paths among Publisher(s) and Subscriber(s), jointly minimising both latency and power consumption, in order to prolong the devices' availability and the overall network lifetime.

In deriving a comprehensive design, we specifically focused on the control protocols and the related overhead, with special attention devoted to the SDN Controller procedures, which include also the management of a WPT subnetwork to dynamically optimise the available power distribution, while matching the QoS requirements.

The chapter structure is as follows: The reference system model is presented in Sec. 7.2 and Sec. 7.3, in terms of both the general architecture and specific communications and control protocols. The overall performance is investigated in Sec. 7.4 for different device deployments, pointing out a remarkable gain in terms of network lifetime and, e2e latency, with a limited overhead, and, consequently, a good scalability. Finally, the conclusions are drawn in Sec. 7.5.

7.2 Proposed Integrated Architecture

The proposed system integrates the SDN concept with clustered IoT domains (i) to manage the system partition into non-overlapping clusters and (ii) to optimise the overall energy consumption and, consequently, maximise the system availability and durability to support typical IoT applications. This can be accomplished by optimising the packets forwarding jointly with advanced WPT technology. As shown in Fig. 7.1, the SDN oriented architecture layers have been designed towards these objectives and divided into: Application Plane (AP), Control Plane (CP), Device Plane (DP) and, Power Plane (PP). Following the architecture of Chapter 6, the PP is composed of devices acting as *Energy Suppliers* (ES), whose main function is, indeed, to

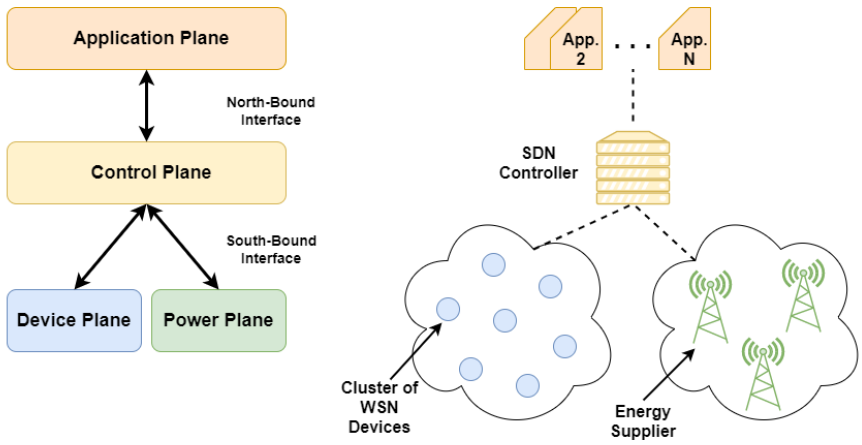


Figure 7.1: Proposed architecture in terms of abstract layered model (left side) or deployment view (right side).

provide energy to the DP nodes that need it, according to WPT paradigm. The CP consists of a logically unique SDN Controller, which has a complete view of the underlying network and can directly manage both the DP and PP devices. Finally, the AP is composed of network applications, which use the CP to reduce the network energy consumption and to manage the battery charge.

The communication protocol used by nodes to exchange data is based on the MQTT-SN standard, where the packets forwarding is handled by the SDN Controller and optimised by the MQTT-SDN Adapter Application which is logically placed on the AP. Finally, the WPT process is accomplished by the PTM Application introduced in 6.2.3. Unlike the original one, in this instance, it is integrated with the SDN ecosystem and some of its tasks are offloaded to the SDN Controller. In order to better evaluate the performance , the approach with only the SDN optimization and the one with the WPT process have been splitted and classified as SD-MQTT-SN and WP-SD-MQTT-SN. The SDN planes and their components are described in the following.

7.2.1 SD-MQTT-SN Communication Protocol

MQTT-SN, unlike the basic MQTT protocol, has been adapted to the impairments of the wireless environment (low bandwidth, high link failure, short message length, etc.) and to the constraints of typical devices in terms of low-cost, battery-powered and, limited computational and storage. Furthermore, this protocol is not necessarily based on TCP, but is completely agnostic of the underlying Transport Layer. As in the MQTT protocol, it is mainly characterised by three elements:

- *Publisher* device which publishes messages under a certain topic.
- *Subscriber* device which registers to a specific topic and, consequently, is willing to receive all the related messages.
- *Broker* device which allows the communication between Publishers and Subscribers.

In MQTT-SN all the Publish messages are sent by the Publisher to the MQTT Broker which forwards them to the Subscriber(s), as shown in Fig. 7.2(a). Thus, even if Publisher(s) and Subscriber(s) are close in terms of hops, Publish messages are requested to reach the MQTT Broker before being delivered to the Subscriber(s), as shown in Fig. 7.2(b). Differently, according to the proposed SD-MQTT-SN approach, the decision to forward or not a Publish message is made by the MQTT-SDN Adapter in accordance to the SDN Controller indications. Specifically, a message is forwarded only if the distance between the Publisher(s) and the Subscriber(s) is lower than the overall distance among Publisher(s), Broker and Subscriber(s), as depicted in Fig. 7.2(c).

In other terms, given a Publisher P, a Subscriber S and a Broker B in the considered network $\mathcal{N}$, the basic MQTT-SN approach finds the best path by solving the following constrained optimisation problem:

$$\begin{aligned} \min_{\pi(P,S)\in\mathcal{N}} \quad & \| \pi(P,S) \| \\ \text{s.t.} \quad & B \in \pi(P,S) \end{aligned} \tag{7.1}$$

where $\pi(P,S)$ is a possible path connecting P and S, and the operator $\| \bullet \|$ evaluates its length. Conversely, the SD-MQTT-SN, is able to identify a better optimum path is it removes the previous constraint and it focuses on the simplified problem:

$$\min_{\pi(P,S)\in\mathcal{N}} \quad \| \pi(P,S) \| \tag{7.2}$$

According to this approach, devices between Broker and Subscriber are not requested to process and transmit unnecessary messages and can save their energy.

7.2.2 Device Plane

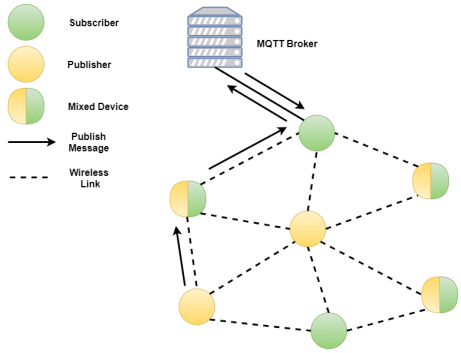
All the WSAN nodes are grouped into cluster and logically inserted in the lower level of the SDN architecture. It can be pointed out that the set up of each cluster is delegated to its devices, without directly involving the SDN Controller. The upstream control messages are directed along the originated topology, usually in a tree fashion to effectively support both multipoint-to-point and point-to-multipoint communications patterns, while downstream control and data messages, including MQTT publish messages, follow a path dynamically chosen by the SDN Controller. The forwarding table of every node, shown in Tab. 7.1, extends the table discussed in Cap. 4 by adding the *Topic ID* field. It is compared with the topic identifier contained within the publish messages and, as the other fields, is directly ,handled by the SDN Controller. In this context, when a node receives a packet, it checks the related entries in its forwarding table. If the `MAC SOURCE ADDRESS` and the `TOPIC ID` of the Publish message are verified, then the node forwards that packet to the next node indicated in the `NEXT HOP MAC ADDRESS` field and updates the `TTL` value. If no entry is satisfied, then the node forwards the packet to the CH.

Table 7.1: Node Forwarding Table.

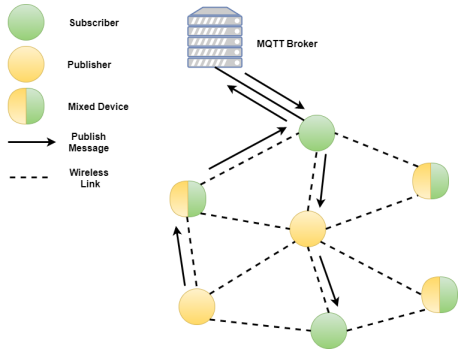
MAC SOURCE ADDRESS	TOPIC ID	NEXT HOP MAC ADDRESS	TTL
C0:36:F0:22:50:6A	1	2B:1B:22:E7:E7:B0	100
0D:20:A5:10:C5:69	2	8F:37:F1:C0:10:18	100

7.2.3 Power Plane

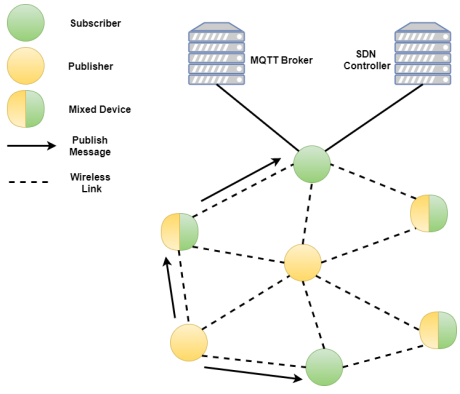
The ES devices, which has the capability to recharge the DP devices, belongs to this plane. In order to accomplish this procedure, several ESs are possibly deployed to effectively supply energy to all the clusters, and directly managed by the SDN Controller. Since the power consumption is mainly due to packet transmission and reception, it is fundamental that every node activates the transceiver only when strictly necessary. Therefore, PTM is in charge of



(a) Publish Messages



(b) Unnecessary Publish Messages



(c) SDN MQTT optimized Publish Message forwarding

Figure 7.2: Comparison of publishing procedure in MQTT-SD and MQTT-SDN.

synchronising a cluster and the ES and, avoiding unnecessary wasting of energy, as described in Sec. 7.2.5. Within a generic cluster, RNs perform multiple radio operations, especially those ones closer to the CH, since they forward messages from/to all the *child* nodes.

The ES node has been conceived to act both as a communication gateway as well as a WPT driver in order to supply energy to the WSAN IoT devices. Its architecture is based on a dual frequency eight flat Switched Beam Antenna (SBA) as deeply described in Sec. 6.2.1.

7.2.4 Control Plane

The CP consists of an SDN Controller, which has a complete view of the underlying network composed of DP and PP nodes. This centralised view allows the Controller to modify the data flows of the DP devices, selecting an e2e path in order to jointly minimise the energy consumption and optimise the ES procedure. According to the proposed approach, the SDN Controller periodically receives control messages from the DP devices indicating (i) the battery level, (ii) the one-hop neighbouring nodes, and (iii) the distance from the CH. Besides, it receives from PP devices messages indicating the Power Transfer time schedule, i.e., the list of CHs and the related recharging period. Using this information, the SDN Controller is able to build and keep updated the view of the network and, consequently, to optimise both the forwarding of MQTT-SN messages and the Power Transfer operations.

7.2.5 Application Plane

AP is mainly composed of *two* applications: (i) MQTT-SDN Adapter (ii) PTM, as depicted in Fig. 7.3. The first one communicates directly with the MQTT Broker, which in turn can locally or remotely run in another Server or in a Cloud. MQTT-SDN Adapter maintains a local *topic table* on which it records the Publishers list, their topics and the related Subscribers list, as shown in Tab. 7.2. Moreover, it resorts to the SDN Controller to evaluate the existence of a multi-hop path connecting Publisher and Subscribers and its cost in terms of number of hops (or of generalised cost), according to (7.1) and (7.2). This information is then used by the MQTT-SDN Adapter to perform the decision previously described in Sec. 7.2.1. It is worth pointing out that a Subscribers can receive a Publish messages from the Publisher in a *direct* way, i.e., not involving the Broker, is labelled as *Direct Mode* (DM)

in the table. In case direct communication is preferred, the SDN Controller changes the forwarding rules of the nodes belonging to the path. The entire procedure is shown in Fig. 7.4.

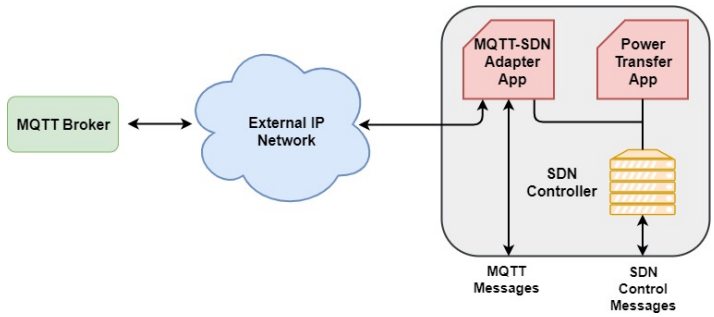


Figure 7.3: Application Plane architecture.

Table 7.2: Publisher and Subscriber Table of MQTT-SDN Adapter Application.

Publisher	**Topic Name**	**Subscribers**	**Direct Mode**
P_1	Topic_1	S_1	No
		S_2	No
		S_3	Yes
		S_n	No
P_2	Topic_2	S_1	Yes
		S_3	No
		S_5	Yes
		S_m	No
P_N	Topic_N	S_i	Y/N
		$\cdots$	Y/N
		$\cdots$	Y/N
		S_M	Y/N

The PTM application allows specific nodes equipped with ad-hoc circuitry (Sec. 6.2.2) to receive the energy from the Energy Supplier, and recharge their battery. Since ES node cannot recharge all the other devices at the same time, the network is partitioned into multiple clusters. Each cluster is topologically connected to the Gateway through a unique CH,

and, within a proper time interval, all the nodes belonging to that cluster can receive energy from ES to recharge their battery.

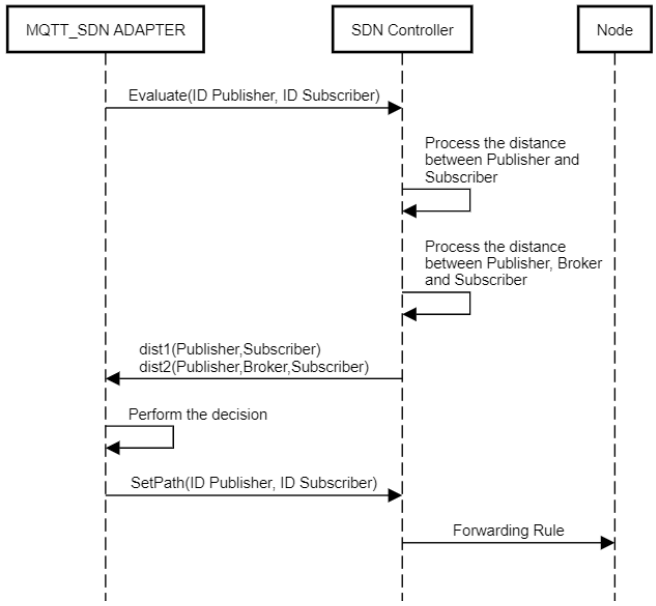


Figure 7.4: Sequence Diagram of MQTT-SDN Application.

Unlike the original version, in this instance, PTM leverages the complete view of the network provided by the SDN Controller (i) to select the proper cluster and (ii) to schedule the re-charging operations. The selection is based on the weighted mean residual battery level of a cluster, where the weighting factor is inversely proportional to the node tree level (where level 0 is conventionally associated to the root). If there is more than one candidate, the cluster with the lowest average energy limited to level 1 nodes is selected; otherwise, a cluster is randomly chosen. In more formal terms, the cluster selection is performed according to the following optimisation problem:

$$SC = \arg\min_{c} \sum_{l=1}^{l_c} \sum_{n=1}^{n_l} \frac{\bar{b}_{n,l}}{l} \quad , \quad 1 \leq c \leq C \tag{7.3}$$

where c indicates a specific cluster, out of C, that is arranged in tree-wise fashion with l_c levels and n_l nodes at the l-th level ($1 \leq l \leq l_c$), each of them has a mean residual battery level equal to $\bar{b}_{n,l}$.

Table 7.3: MQTT-SN Messages

MsgType Field Value	MsgType	MsgType Field Value	MsgType
0x00	ADVERTISE	0x01	SEARCHGW
0x02	GWINFO	0x04	CONNECT
0x05	CONNACK	0x0A	REGISTER
0x0B	REGACK	0x0C	PUBLISH
0x0D	PUBACK	0x0E	PUBCOMP
0x0F	PUBREC	0x10	PUBREL
0x12	SUBSCRIBE	0x13	SUBACK
0x14	UNSUBSCRIBE	0x15	UNSUBACK
0x16	PINGREQ	0x17	PINGRESP
0x18	DISCONNECT		

Once the cluster has been identified, PTM notifies to the intended cluster nodes the starting of the recharging process by sending a multicast control packet.

7.3 Data Flows Management and Forwarding

As previously described, the SDN Controller can directly manage the forwarding rules of the DP devices in order to dynamically change the forwarding path for each data flow. The messages exchanged inside the network can be divided in: (i) SDN-WSAN messages, and (ii) MQTT-SN messages. The SDN-WSAN messages already described in Cap. 4, are mainly used to modify the nodes flow entries according to the optimised e2e paths while the MQTT-SN messages are compliant to the MQTT-SN standard. Since the proposed approach is focused on the optimisation of the energy consumption and on the data flows forwarding, we further specify the MQTT-SN messages as summarised in Tab. 7.3. Specifically, we do not consider the *Will* messages, which are mainly used to alert the Subscriber nodes whenever a topic is no more available, or the Publisher node is disconnected.

7.4 Obtained Results

The performance of the proposed SDN oriented communications and energy management protocol has been evaluated using the MATLAB framework by means of numerical simulations, where the involved nodes can be either sensors, actuators or hybrid devices. The simulated wireless sensor node has

been modelled considering a real IEEE 802.15.4 commercial module with the parameters summarized in Tab. 6.1. Additionally, the initial node battery level has been set up to 2500 mAh, corresponding to 2 AAA batteries, while the transmission current consumption was modelled assuming a fixed packet's payload of 50 bytes and a data rate of 250 Kbps which is equal to the maximum allowed by the IEEE 802.15.4 standard. The network has been partitioned in logical clusters (equal to 8 in the considered test scenario), each managed by its own CH that specifically handles the traffic from or to the GW.

The overall system performance has been evaluated in terms of the following metrics and compared with existing protocols:

- Lifetime: this parameter measures the network duration; a strict definition has been adopted, that is the time elapsed since the first node in the network depletes its battery.
- Overhead: it represents the cost of maintaining up to date the flow tables, and it is generally given by the ratio between control and overall messages (including Publish packets). For SD-MQTT-SN ($\eta_{SD\text{-}MQTT\text{-}SN}$) also *FWDR* and *Report* messages need to be included.
- Equivalent Hop Number (EHN): it is intended here as the total number of transmissions needed to deliver a specific Publish packet. Particularly, all the transmissions through the e2e Publisher-Broker and Broker-Subscriber paths for all the Subscribers associated to a specific topic are considered. In case of direct communications, the transmissions between Publisher and Subscriber not involving the Broker are also taken into account.
- Latency (δ): it measures e2e latency needed to deliver a message by considering all the transmissions delays along the path.

Moreover, the proposed architecture has been tested over three different nodes deployments: (i) Random Mesh, (ii) Two-Tier, and (iii) Unbalanced Random Mesh. The overall number of devices has been assumed to be equal to 40, in compliance with most common wireless IoT scenarios. For the sake of comparison, the CoAP and the MQTT-SN performance have been considered as the benchmark, because they represent the two most relevant WoT standards adopted in practice.

Table 7.4: Overhead required by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Random Mesh nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$\eta_{MQTT\text{-}SN}$	0.662	0.675	0.66	0.685	0.676	0.672	0.665	0.665	0.684	0.646	0.669
$\eta_{SD\text{-}MQTT\text{-}SN}$	0.698	0.724	0.707	0.729	0.709	0.679	0.702	0.763	0.762	0.681	0.709
η_{COAP}	0.639	0.656	0.634	0.667	0.656	0.653	0.647	0.643	0.662	0.631	0.649

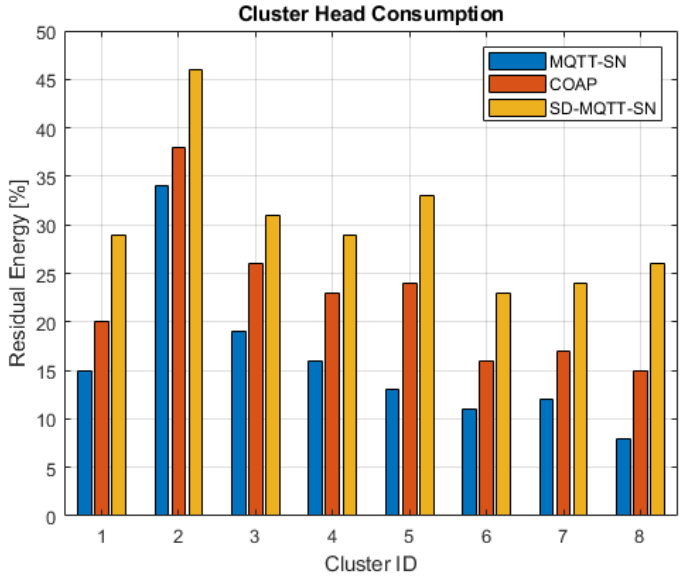


Figure 7.5: CHs normalised residual battery level achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Random Mesh nodes deployment.

Firstly, the performance of the SD-MQTT-SN approach over a Random mesh topology has been evaluated, in which the number of hops for a specific e2e path depends on the nodes' spatial distribution, while a fixed transmission range has been assumed for simplicity. It can be preliminarily noticed that SD-MQTT-SN sporadically resorts to the Broker mediation. As a consequence, the battery consumption results more homogeneous because the majority of transmissions happen at the network edge, while traffic load is reduced on the network core, especially on the CHs side, with a consequent lifetime improvement, as shown in Fig. 7.5.

Table 7.5: EHN achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Random Mesh nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$EHN_{MQTT\text{-}SN}$	14	17	18	18	16	15	15	19	20	16	17
$EHN_{SD\text{-}MQTT\text{-}SN}$	13	15	15	16	14	13	12	16	16	14	14
EHN_{COAP}	14	16	16	17	15	14	14	17	18	14	16

Table 7.6: Latency achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Random Mesh nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$\delta_{MQTT\text{-}SN}$ [ms]	0.130	0.179	0.237	0.336	0.197	0.408	0.396	0.441	0.394	0.465	0.318
$\delta_{SD\text{-}MQTT\text{-}SN}$ [ms]	0.112	0.138	0.156	0.266	0.188	0.358	0.307	0.372	0.327	0.418	0.264
δ_{COAP}	0.121	0.152	0.192	0.302	0.193	0.386	0.356	0.398	0.362	0.447	0.291

From Tab. 7.4, it can be noticed that the overhead required by the SD-MQTT-SN scheme is slightly higher than the one of the basic MQTT and COAP. Approximately 40% additional messages per Publish packet sent are, indeed, needed, i.e., roughly one control message out of two Publish packets. Despite this, in Tab. 7.5 is pointed out that the introduction of SD-MQTT-SN in the considered scenario allows a drastic reduction in the number of transmissions to be performed to deliver every Publish Packet to all the possible destinations, with a performance gain of approximately 11%. On the average, it is possible to save two transmissions for every Publish Packet generated in the network, achieving a considerable reduction of the overall consumption, which completely compensates the additional control messages. Moreover, the SD-MQTT-SN scheme has a positive influence on latency, as shown in Tab. 7.6, where it can be pointed out an increasing e2e delay reduction due to the path optimisation, i.e., a lower number of hops needed to reach the destination.

An alternative scenario that has been investigated consists in a Two-Tier star-based nodes deployment, in which the GW is located at the barycentre of the network surrounded by a first tier of CHs, where all the other nodes are one-hop distant from them. Simulations performed on this scenario have highlighted that the amount of configuration messages necessary to keep the flow tables updated is considerably reduced compared to the Random mesh topology, since those messages are forwarded only by one intermediate node, i.e., the CH. When applied, the SD-MQTT-SN can opti-

Table 7.9: Latency achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Two-Tier nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$\delta_{MQTT\text{-}SN}$ [ms]	0.13	0.14	0.185	0.265	0.175	0.275	0.27	0.315	0.27	0.32	0.234
$\delta_{SD\text{-}MQTT\text{-}SN}$ [ms]	0.13	0.14	0.185	0.251	0.175	0.257	0.246	0.315	0.27	0.296	0.226
δ_{COAP}	0.13	0.14	0.185	0.260	0.175	0.268	0.253	0.315	0.27	0.316	0.231

mise inter-cluster or, less frequently, intra-cluster paths. Furthermore, even though SD-MQTT-SN approach reduces CHs energy consumption and, consequently, increases network lifetime, these improvements are more limited than those ones achievable in the Random Mesh configuration, as depicted in Fig. 7.6.

Table 7.7: Overhead required by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Two-Tier nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$\eta_{MQTT\text{-}SN}$	0.27	0.303	0.248	0.3	0.306	0.274	0.21	0.234	0.297	0.213	0.265
$\eta_{SD\text{-}MQTT\text{-}SN}$	0.33	0.359	0.335	0.362	0.379	0.349	0.327	0.336	0.352	0.317	0.345
η_{COAP}	0.22	0.294	0.239	0.296	0.304	0.268	0.17	0.229	0.292	0.210	0.252

It is interesting to note that, despite the considerable reduction of control messages, the lifetime gain is reduced too: this depends on the node spatial distribution, as nodes that represent bottlenecks, i.e., CHs, are more stressed.

Table 7.8: EHN achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Two-Tier nodes deployment.

Sim Run	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*Avg*
$EHN_{MQTT\text{-}SN}$	12	13	13	15	12	12	12	14	14	12	13
$EHN_{SD\text{-}MQTT\text{-}SN}$	10	12	12	14	12	11	11	13	14	12	12
EHN_{COAP}	11	13	13	14	12	13	12	14	14	12	13

As a matter of fact, the traffic from/to a single cluster is entirely managed by its CH, without the possibility of redistributing the workload according to the nodes' battery level. Moreover, the control messages overhead is illustrated in Tab. 7.7. As already mentioned, since any node is at most two hops far from the Broker, there is a limited number of messages necessary for the setup of direct routes between two nodes; as a consequence, the overhead is less than the one needed for the Random Mesh deployment but

still higher than COAP protocol. Besides, the energetic consumptions are clearly unbalanced, with consequent performance degradation.

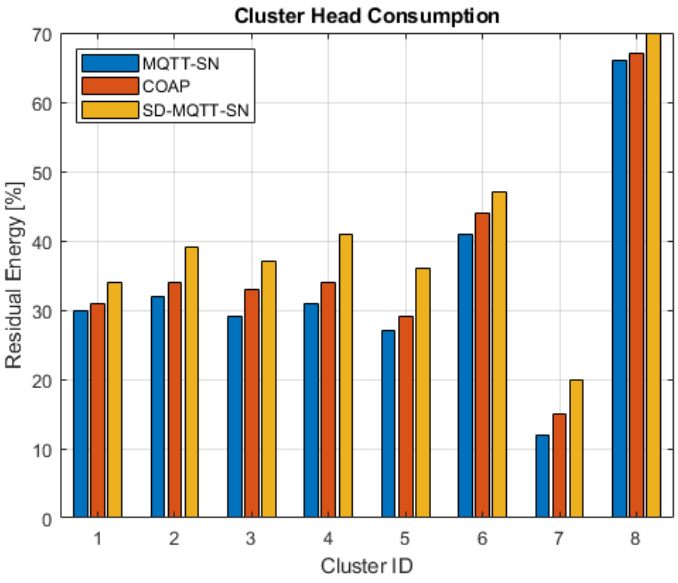


Figure 7.6: CHs normalised residual battery level achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in a Two-Tier nodes deployment.

The equivalent hop number is considerably reduced, and this depends on the fact that there are at most two hops between a Publisher and the Broker, and the same distance between the Broker and subsequent Subscriber. As shown in Tab. 7.8, the introduction of direct communications does not bring substantial improvements, due to the limited number of direct messages exchanged. Finally, from the results shown in Tab. 7.9, it can be seen that latency is significantly lower than the one occurring in the Random Mesh configuration, even though the SD-MQTT-SN approach does not lead to any remarkable improvement because of the limited number of direct transmissions.
For the sake of completeness, the proposed solution has been tested over an Unbalanced Random Mesh nodes deployment,where the GW is positioned at the edge of the playground. It is worth noticing that this represents a worst-case scenario, where performance degrades, but it is interesting to evaluate

the effects of SD-MQTT-SN.

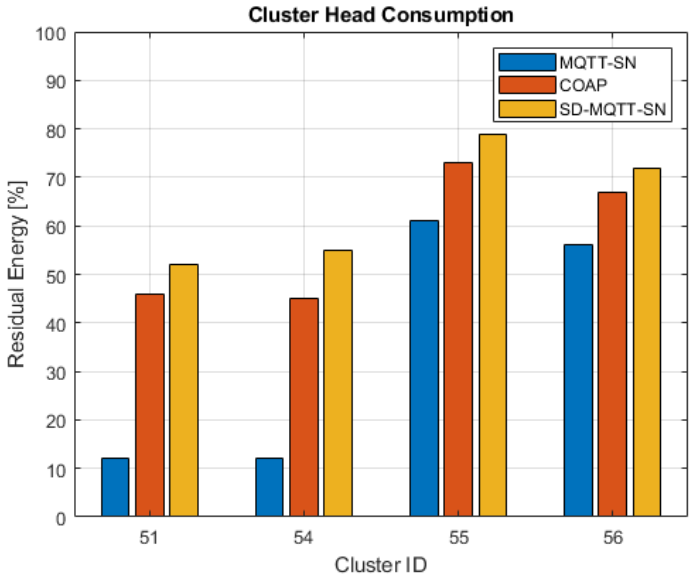


Figure 7.7: Normalised residual battery level of the one-hop from GW nodes achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in an Unbalanced Random Mesh topology nodes deployment.

It can be noticed the impact of an extremely unbalanced topology. First of all, there are fewer nodes one hop away from the GW than in the balanced Random Mesh deployment previously introduced, but they still manage a similar amount of traffic.

Table 7.10: Performance improvements due to the introduction of SD-MQTT-SN approach in an Unbalanced Random Mesh nodes deployment.

	COAP	*MQTT-SN*	*SD-MQTT-SN*
η [%]	0.845	0.851	0.856
EHN [hop]	21	28	17
δ [ms]	0.309	0.466	0.257

Furthermore, there are less chances to optimise the paths in the proximity of the GW and, therefore, lifetime is reduced.

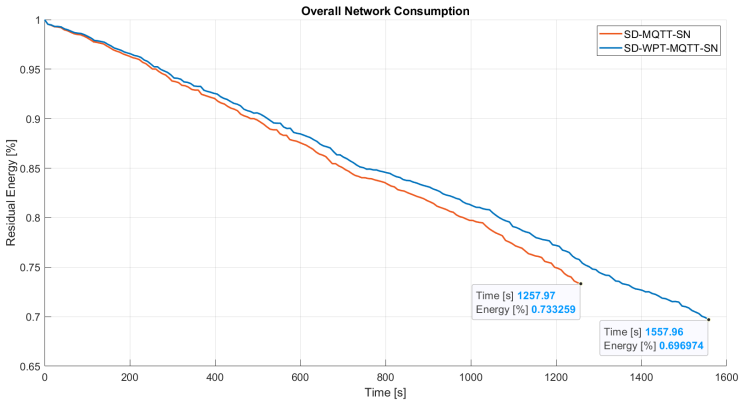


Figure 7.9: Normalised average residual battery level achieved by the SD-MQTT-SN and SD-WPT-MQTT-SN protocols in a Balanced Random Mesh topology nodes deployment.

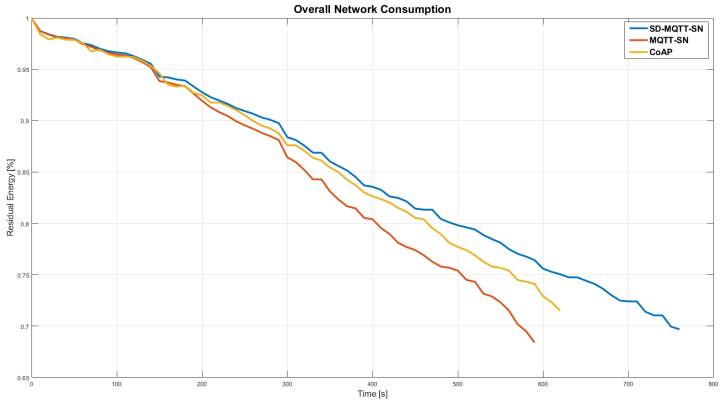


Figure 7.8: Normalised average residual battery level achieved by the CoAP, MQTT-SN and SD-MQTT-SN protocols in an Unbalanced Random Mesh topology nodes deployment.

On the other hand, in the zone opposite to the GW, it is more likely to optimise the routes and this implies an overhead increase needed for the flow tables configuration, with a consequent increase of the equivalent hop number, due to the greater (average) distance between nodes and the GW, and also of the e2e latency. Despite this, it is observed that SD-MQTT-SN is beneficial for energy consumption, as highlighted in Fig. 7.7. Besides, the performance in terms of overhead, equivalent hop number and e2e latency are represented in Tab. 7.10, where it can be clearly pointed out the improvements provided by the SD-MQTT-SN protocol for what concerns latency and the equivalent hops number.

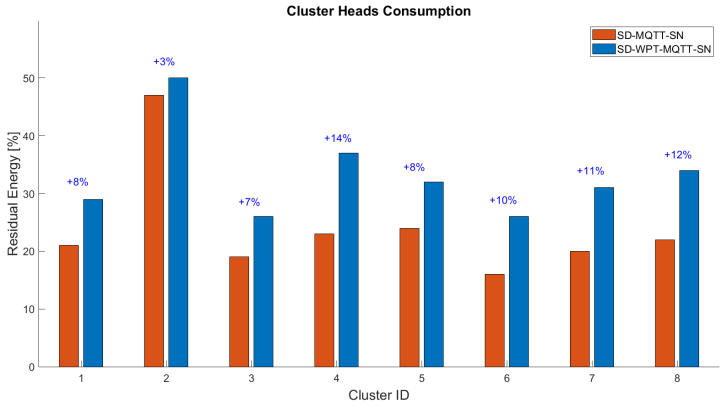


Figure 7.10: Normalised CHs residual battery level achieved by the SD-MQTT-SN and SD-WPT-MQTT-SN protocols in a Balanced Random Mesh topology nodes deployment.

This is achieved with a comparable overhead, while in the previously investigated deployments, the SD-MQTT-SN scheme requires an overhead of about 6% (Random Mesh) or even 30% (Two-Tier). In Fig. 7.8, the normalised average residual battery level for COAP, MQTT-SN and SD-MQTT-SN schemes is presented, pointing out a progressive improvement over time, which is directly correlated with the overall consumption and the network lifetime. To conclude this simulation campaign, the SD-WPT-MQTT-SN approach has been introduced, where the ES functions have been assumed to be installed in the GW node. First of all, the benefits provided

to the CHs in terms of energetic supply in the case of a Balanced Random Mesh nodes deployment, which is the most common IIoT scenario, have been evaluated in Fig. 7.10. It can be noticed a remarkable gain due to their proximity with the GW, while the furthest nodes present lower benefits. Despite this, the network achieves a lifetime increase, as pointed out in Fig. 7.9, where the overall normalised battery level is shown over time.

7.5 Discussion

In this Chapter, a comprehensive architecture has been proposed targeted to IoT ecosystem for industrial applications. The model adopted a Web-oriented communications paradigm to enable complex interaction among sensors and actuators in the presence of technological heterogeneity. In order to provide service availability and reliability, despite the limited energetic budget, a general framework inspired by the SDN principle has been proposed. It has been accurately presented in terms of component and communications/control protocols with a specific focus on the SDN Controller design by characterising its specific functionalities to handle data flows and to manage power budget distribution. This allowed the extension of the basic MQTT-SN protocol towards the SD-WPT-MQTT-SN approach, that is able to jointly optimise the QoS requirements and the network lifetime. The integrated solution has been validated in practical scenarios always pointing out remarkable performance w.r.t. the e2e latency and the overall power consumption, with a limited overhead, and, consequently, a good scalability.

Chapter 8

Fast Fog Services Dissemination in the Internet of Vehicles

This chapter introduces the innovative Mobile Mist Computing (M^2C) concept by addressing a reference architectural model for vehicular networks capable of supporting complex mobile services with stringent time-space constraints. Cellular technology, thank to its coverage, data rates and latencies, is generally considered the key enabler to support real-time vehicular applications. Moreover, their integration with the emerging Fog Computing paradigm can reduce the of the core network, thus minimising the overall service delay. In this scenario, vehicular mobility represents the most relevant challenge to guarantee service continuity, while providing Quality of Service (QoS), especially when Fog Servers (FSs) have limited resources. To this purpose, Fog Computing has been firstly integrated with the Software Defined Networking and Network Functions Virtualization concepts and, subsequently the dynamic replacement of vehicular services (i.e., VNF migration) among FSs has been investigated to properly accommodate users mobility. Finally, an optimised strategy, called swapping migration, that can optimise both resources utilisation and outage rate, has been introduced. The proposed scheme has been validated by means of numerical simulations and compared with several benchmarks over realistic scenarios by pointing out latency and reliability as a function of the services request rate

and the transmission capacity.

8.1 Introduction

The ever-increasing number of vehicles and their densification within typical mega-cities pose a remarkable challenge to the automotive industry and to public administrations in order to provide transportations safety and efficiency in a Smart City by relying on *smart* cars [121]. To this purpose, the emphasis should be put on *group* intelligence, rather than on a single device capabilities, which, indeed, is constrained and not able to face time-varying and unpredictable operative conditions. Recently, information and communication technologies (ICTs) have been regarded as the way to revolutionise vehicular networks domain. Connecting vehicles via Vehicular Ad-hoc Networks (VANET) represented an early attempt to support this vision, where closer vehicles are allowed to communicate with each other via Vehicle-to-Vehicle (V2V) or Vehicle-to-Infrastructure (V2I) interfaces to notify or collect traffic-related information, respectively [121]. Thanks to their high data rates and low latency communications, 5G Cellular Networks (CNs) represent a promising enabling technology for VANETs, and the standardisation of the Vehicle-to-Everything (V2X) interface is paving the floor for advanced vehicular services [110]. The Internet of Vehicles (IoV) concept has been proposed to extend VANETs towards heterogeneous communication systems to provide complex applications, mainly vehicle (automated) driving, urban traffic management, or even logistics [61]. In addition, it is required a paradigm shift from the traditional Cloud Computing (CC) toward hybrid architectures integrating Fog Computing and Networking (FCN) to support real-time and location-aware processing and control. In particular, FCN relies on a set of Fog Servers (FSs) generally deployed *closer* to the end users and able to provide *ad hoc* and opportunistic services. FCN typically avoids to upload *all* the data to a remote data center and, therefore, communication latencies and network congestions are reduced.

One of the most important challenges to be faced when FCN technology is adopted in a vehicular context is the specific mobility patterns. This requires *generalised* handover policies, where not only users, but also services are transferred *among* cells, to maintain service continuity and real-time requirements. This procedure, known as service migration, is widely used by the Network Function Virtualization technology. NFV is a disruptive

paradigm according to which, *network functions* become software applications, i.e., Virtual Network Functions, that can be instantiated on generic servers by means of Virtual Machines or containers. VNFs can be dynamically created/destroyed, migrated from one physical device to another and, finally, *composed* together to offer innovative services. This approach, previously introduced in CC, still represents an open issue within the IoV [133], mainly in terms of vehicular VNFs placement and migration among FSs. Specifically, the reference network architecture requires a decoupling from Data Plane (DP) and Control Plane (CP) to allow a *Controller* to get a comprehensive *view* of the underlying network, and, hence, to apply specific policies, according to the Software Defined Networking approach [130]. SDN application to IoV scenarios can provide several benefits in terms of re-programmability, agility, scalability, elasticity and flexibility. Moreover, it allows the automatic and fast development of network applications and complex services.

Most of the existing investigations only proposed strategies to optimise the service latency under specific time constraints, without properly considering the potentialities of service migration in IoV environment, or limiting it to simplified scenarios. To face the above introduced problem, in Sec. 8.2, we propose a framework, called Mobile Mist Computing (M^2C) - since applications *condense* from the FSs directly *onto* the user path - to support innovative time-space constrained services for a group of (semi) automated vehicles in dynamic scenarios. More in details, the main contributions of this chapter include: (1) the design of an architecture for vehicular communication which combines SDN, NFV and FC and to enable the service migration; (2) a proactive handover based on SDN; (3) a Service Management and Orchestrator implementation; (4) a swapping migration scheme, enhancing the existing IoV service migration strategies; and (5) a detailed performance evaluation of the proposed scheme in comparison with several benchmarks, which is carried out in Sec. 8.3, by means of numerical simulations performed over realistic scenarios in terms of end-to-end (e2e) delay, reliability and resources utilisation at the FSs.

8.2 Basic Idea

The proposed M^2C architecture, combining FC, SDN and NFV concepts, is depicted in Fig. 8.1, where it is evident the classical three-layer arrangement

(DP, CP, and AP). In the proposed architecture, DP is comprised of eNBs and OF Switches, the latter only being connected to the CP through the OF protocol. FSs belong to an intermediate level, called Fog Plane (FP), as represented in Fig. 8.1; they are connected to OF Switches and pre-loaded via the application's *images* provided by the Service Provider (SP), which usually adopt Operational and Business Support Systems (OSS/BSS). According to a *publish & subscribe* pattern, vehicles select a specific service to be personalised which is executed by a specific container.

It is worth noting that the building elements have been properly adapted to the vehicular ecosystem requirements, where the wireless access is provided by the LTE Advanced access network. In particular, it involves specific APs (i.e., eNodeBs), but, unlike to the classical LTE architecture, in which they are directly connected to the Mobility Management Entity (MME) and the Serving Gateway (SGW), they are, instead, paired with an OF Switch. In the architecture, the entities of the LTE Core Network (CN), e.g., MME and S-GW, are virtualized via VNFs and placed into the Cloud. However, IP-based routing procedure are managed by one (or more) SDN Controllers. The SDN principle gives to the Controller(s) a complete network view to optimally manage the system in terms of (i) traffic flows forwarding and (ii) ad-hoc deployment of innovative applications and algorithms without replacing the devices.

In the proposed approach, Controller(s) have additional capabilities to support *generalised* handover related to a *group* rather than individual mobility, if compared with the features available on MMEs and eNBs of traditional 4G cellular system [22]. The handover related VNF (hVNF) running on a Controller collects data from vehicles and acts as a *proxy* between the eNB and the CN; in particular, each eNB triggers its hVNF to manage an handover. In case of INTRA-MME/SWG handover over X2 interface, hVNF only collects signalling. Instead, if the X2 interface is not enabled, hVNF autonomously manages this procedure without any support of CN entities, but invoking the SDN Controller to modify data flows towards the destination eNB. Finally, in case of INTER-MME or INTER-MME/SGW handovers, hVNF transparently forward this request to the CN, which handles it, as in LTE Advanced [22]. As a consequence, SDN Controller can proactively support handover procedures, thus reducing overhead and latency. Furthermore, SDN Controller can involve the Vehicular Functions and Services Management and Orchestrator (VFS MANO) entity to manage FSs resources and

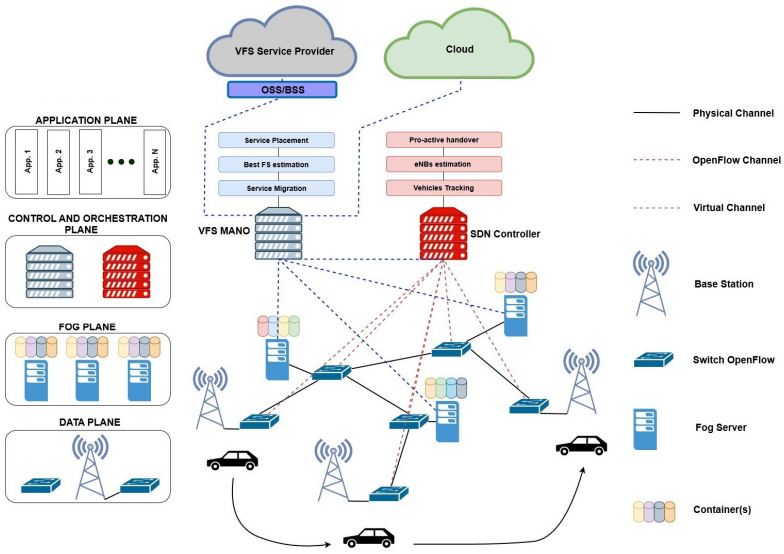


Figure 8.1: Proposed comprehensive system architecture, highlighting actors and interfaces of vehicular ecosystem.

to support services set-up, tear-down and migration. The latter operation aims at maintaining a service close its client, then minimising the latency between its present eNB and the FS hosting the service. However, it requires an additional delay to transfer service *status* (usually a file or a page) to the target FS; when this information is not negligible, the overall delay increases, so that the the service migration should not be performed every time a vehicles initiates a handover, as pointed out in [98].

In this chapter, the benefits provided by a correct service migration are investigated by analysing two schemes, that is, no migration (Scheme 1) and the proposed optimized pre-copy migration for mobile scenarios (Scheme 2). Scheme 1 is introduced only as benchmark, since no service migration is proactively performed, despite vehicles are moving. To maintain the service continuity, the vehicle are requested to always connect with the FS where the requested service is running. Consequently, the e2e latency, which includes the propagation, transmission and queuing delay, becomes larger as the car leaves this FS. To address this drawback, an optimized pre-copy mi-

gration mechanism is introduced in Scheme 2. In order to select the *best* FS to perform the service migration on, the VFS MANO predicts vehicle(s) position(s): it is usually based on the previously evaluated journey since this represents the core service for a traffic *navigation* application, which is also regularly updated by the underlying *tracking* service. Once obtained the vehicle path and the eNBs list along it, the VFS MANO applies the Djkstra algorithm over the network view owned by the SDN Controller to evaluate the best FS to migrate the service. Then the VFS MANO starts a *pre-copy* container migration by means of an iterative *pre-dumping* phase on the selected FS. This procedure transfers most of the container *status* to the destination node, before the complete container dumping. After the pre-dumping phase, the container is suspended and the modified pages are migrated to the destination FS, where the container with the updated status is reactivated. Resorting to the SDN Controller, the VFS MANO can track the vehicle position and activate the last phase of the pre-copy migration when it is close *enough* to the target FS, while performing the handover. Differently from the pre-copy migration conventionally used for mobile networks [73], in which the entire migration starts when the vehicle connects to the new eNB, in this optimized scheme only the *last* phase of the pre-copy migration is executed during the handover, therefore, the overall downtime is reduced. In particular, this delay is composed of the handover downtime and migration downtime. In Scheme 2, both are overlapped and, thus, the total downtime is lower than the conventional approach. The main drawbacks of this solution is the scalability: when the service requests are more than resources available in all the FSs, the system drops new requests or redirects them to CC, hence, increasing the service outage or the e2e delay, respectively.

Therefore, Scheme 2 (and also Scheme 1) is not able to properly support hard time constrained services, and, consequently, an additional swapping migration scheme (Scheme 3), optimising the overall delay and the services availability at each FS, is introduced. Scheme 3 proposes an optimised *swapping-migration*, which is activated *only* if it is necessary to free up resources on a certain FS to be used for *another* container. The active container *releasing* consists in migrating it to another appropriate FS capable of hosting the service, without degrading its performance, but guaranteeing QoS requirements. Specifically, the VFS MANO is aware of the containers/services placement within FSs, together with the resources needed to sat-

isfy QoS requirements. Moreover, by interacting with the SDN Controller, the VFS MANO gets a complete view of the underlying network features so that it can: (1) evaluate the best target FS to migrate the active container and, (2) avoid the migration not match in the QoS requirements of the running service.

8.3 Performance Analysis

The integrated management approach presented in Sec. 8.2 has been evaluated by means of numerical simulations resorting to (i) an ad hoc Python 3 framework to model the container migration and the VFS MANO, (ii) Simulation of Urban MObility (SUMO) to generate vehicular traffic and, (iii) Mininet WiFi to emulate the wireless SDN. The test scenario is composed of 20 eNBs, 30 FSs, 20 OF Switches, 1 SDN Controller and 1 VFS MANO, representing a realistic urban case study, with 2500 vehicles over this area. To allow service migration, FSs has been supposed to be connected via high-capacity links, such as based on fiber or millimeter-wave.

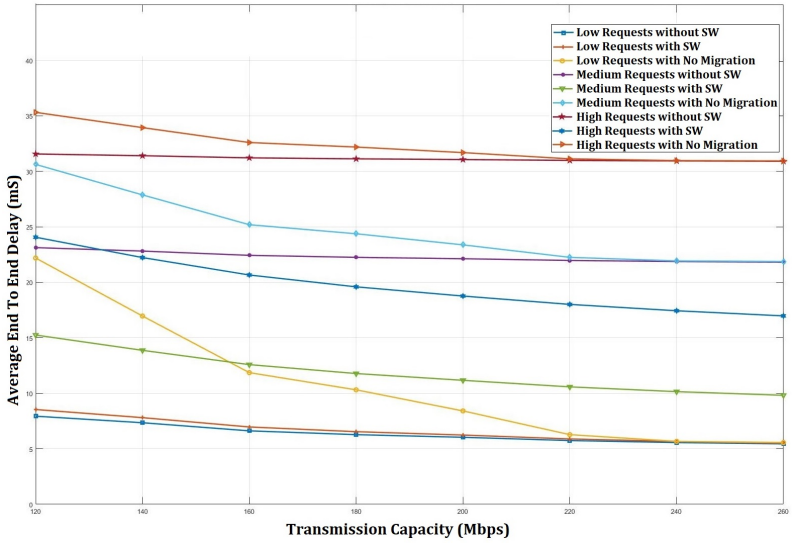


Figure 8.2: Average e2e service completion latency for different transmission capacities and service request rates.

The eNBs maximum coverage radius is 8 km, while the length of the wired links (dotted lines in Fig. 8.1) does not exceed 15 km and with a capacity of 1Gbps. Moreover, there are 30 services offered by the VFS SP, which are different in terms of the resources (RAM, CPU) requested for their execution, whereas FSs are characterised in terms of available resources so that they can accommodate only a variable number of containers. More in detail, the size of each container hosting the service is comprised of 10 to 100 Mbits and the maximum number of containers that a server can host is 500. The maximum number of service requests is 15000, in order to saturate the resources available on the FSs and verify the presented approach. Vehicles trajectories are randomly generated, and, in the suggested optimized pre-copy migration scheme, they are known together with the set of eNBs along with them. The

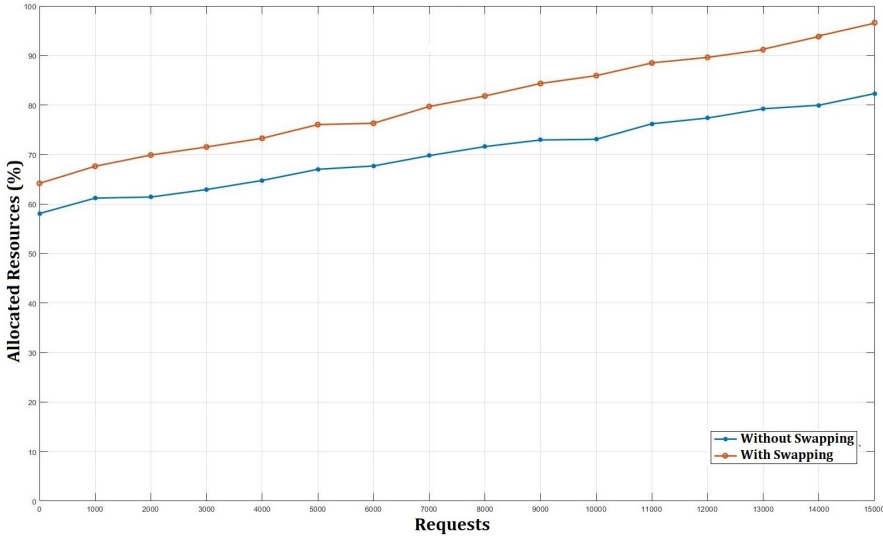


Figure 8.3: Average normalised resource utilisation per FS.

proposed Schemes, described in Sec. 8.2, are besides compared in terms of service e2e latency, normalised resource utilisation (η) and service success probability. In Fig.8.2, the average e2e service completion latency for the considered three Schemes with different transmission capacities and services request rates, is sketched. It is worth pointing out that this delay decrease as the Fog network capacity increases, since higher transmission rates reduce

packet transmission time and queuing delay, thus leading to smaller overall delay. However, the three Schemes behave in a different way at the increasing of vehicular service request rate, as it causes both a decrease of available resources at FSs and a more intense use of the traditional CC, therefore, a worsening of the e2e delay performance. The optimised pre-copy migration (Scheme 2) outperforms the one without migration support (Scheme 1), especially for low transmission capacity, since the service is *deployed* along the trajectory of the requesting vehicle, thus lowering the overall delay.

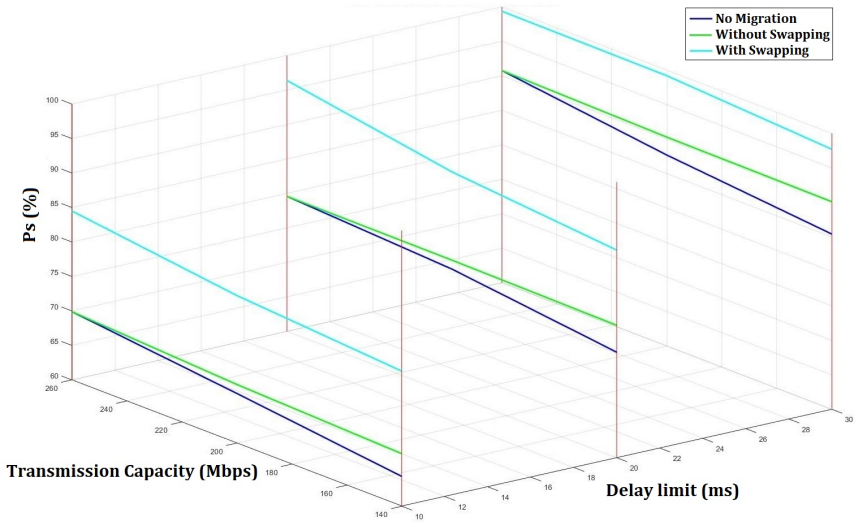


Figure 8.4: P_s achieved by the three different schemes for high service request rate, variable transmission capacities and different delay limit δ^* values.

The swapping oriented approach (Scheme 3) follows the same procedure of the optimised pre-copy migration (Scheme 2) but, when resources are limited, it tries to satisfy the request using the FC instead of the CC. As explained in Sec. 8.2, Scheme 2 migrates an active container to another FS without violating QoS requirements; once the migration is completed, some resources of the selected FS are released for hosting new services. Even though this procedure requires some additional latency to be accomplished, the average e2e delay remains lower than the previous two approaches, es-

pecially for high service request rate.

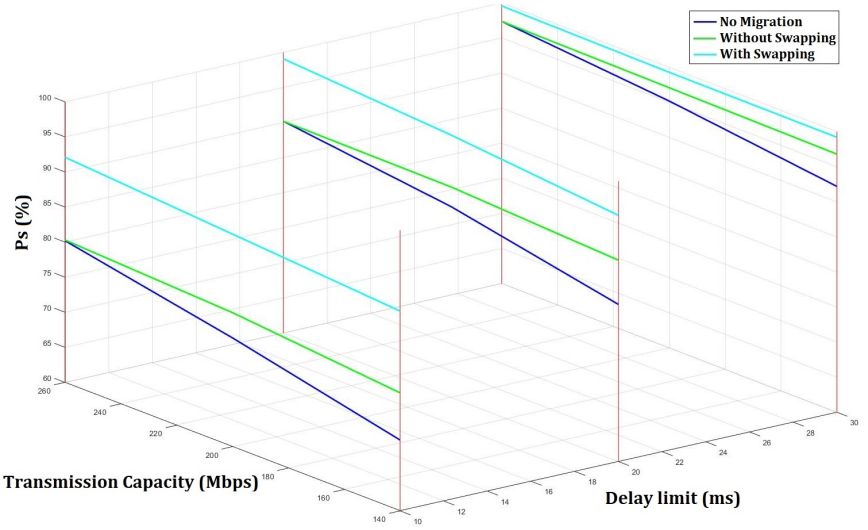


Figure 8.5: P_s achieved by the three different schemes for medium service request rate, variable transmission capacities and different delay limit δ^* values.

From Fig. 8.3 it is evident that the swapping-migration increases η within FSs of about 13%, w.r.t. the migration without swapping. As previously described, this is due to the migration of active containers into available FSs and, consequently, the hosting of additional services. It is worth noting that the gap between the curves increases as the service request rate grows. For low-to-medium request rates both Scheme 2 and 3 mainly resort to resources available in the FSs making η to increase. For high request rate values, instead, Scheme 2 resorts to CC more than to FC, causing a resource wasting in the FSs. Conversely, Scheme 3 thank to the swapping procedure still uses FSs resources in an optimal way approaching the maximum value of η. In the case of strictly real time vehicular services, a most relevant performance metric is represented by the success probability P_s, which is defined as the probability that the e2e service completion latency does not exceed a maximum value δ^*. In Fig. 8.4, Fig. 8.5 and, Fig. 8.6, P_s is investigated for all the three different schemes for incremental service request rate, vari-

able transmission capacities and δ^* equal to 10, 20 and 30 ms, respectively. Particularly, the three figures show the P_s in case of high, medium and low service demand, respectively. It is evident in the Figures that Scheme 3 always outperforms the other ones, with a more remarkable gain in the case of both low delay constraints δ^* and transmission capacities.

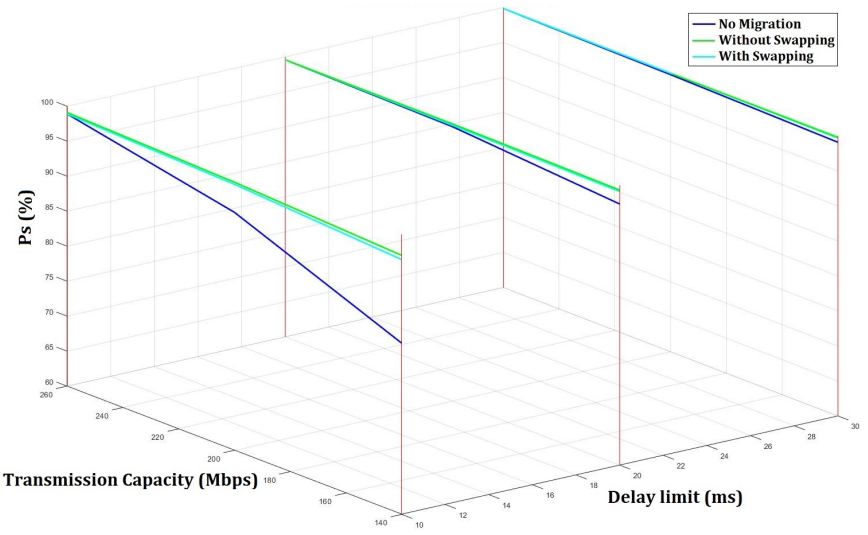


Figure 8.6: P_s achieved by the three different schemes for low service request rate, variable transmission capacities and different delay limit δ^* values.

In Fig.8.4, Scheme 3 achieves a maximum P_s value of 80% when δ^* is equal to 10 ms and capacity is at its lower value, which is significantly higher than the 16% and 20% from the no swapping and static schemes, respectively. Fig.8.5 shows that Scheme 3 reaches a maximum P_s of 99.6% with an e2e delay limit of 30 ms and transmission capacity of 260 Mbps, outperforming the other two solutions with a maximum of 97.4%. In Fig.8.6, Scheme 2 and 3 have comparable performance with a probability higher than 96.4%. The static approach, however, requires a higher transmission capacity or lower constraint delay limits to match their performance.

8.4 Discussion

This chapter proposes a vehicular services oriented framework supporting mobile real time services. A general network architecture based on LTE Advanced technology, which properly combines the innovative concepts of SDN, NFV and FC, has been derived. Specifically, various service migration approaches have been evaluated and a near optimal procedure, called swapping-migration, is introduced. In order to maximise the FSs resources utilisation, the proposed scheme performs active services migrations between FSs. Performance evaluation in comparison with other approaches available in the literature, has been conducted in terms of e2e service completion delay, success probability and resource utilization by means of realistic simulations over an urban area. All the tests pointed out that the proposed swapping-migration scheme outperforms the alternatives especially for high service request rate and low capacity, since it always keeps the service topologically *close* to the requesting clients, while optimising the use of the Fog Network.

Chapter 9

Conclusion

This chapter summarizes the contribution of the thesis and discusses avenues for future research.

9.1 Summary of contribution

This dissertation presented a novel approach based on Software Defined Networking, named SOHS, to control IoT devices. The solution has been extensively tested in different WSAN scenarios and has been shown to have many advantages compared to other existing solutions. The results of the experiments demonstrate that in both static and mobile conditions, the proposed scheme is superior to traditional methods, regardless of the network size, payload size, traffic generated, and performance metric being considered. SOHS not only boasts strong capabilities as an SDWSAN, but also introduce an innovative philosophy to seamlessly integrating SDN into the wireless IoT domain. The resulting architecture has been subsequently used to design a forwarding strategy operated by the SDN Controller, which dynamically adapts it to the network status in order to optimize the network responsiveness and energy consumption. To achieve this, an optimization problem based on both nodes' energy consumption and end-to-end flow delivery was characterized and a practical solution, named DRM, was implemented in the SDN Controller, with a limited control message overhead. Through a comprehensive and realistic simulation campaign, it was found that the performance achieved by the proposed approach is better than SDN alternatives based on a hierarchical CP, as in LOA. Specifically, the SDN-DRM Controller

presents the best trade-off between delay and battery consumption, without excessive overhead for comparable network sizes. Therefore, the proposed framework introduces a more effective and more flexible CP for a reference SDN IoT architecture. In addition, the constraints faced by IoT devices were taken into account to provide system availability and reliability, despite the limited energy budget. Towards this goal, a general framework inspired by the integration of the Publish-Subscribe and WPT principles was proposed, in order to jointly optimize the QoS requirements and the network lifetime. The system was accurately characterized in terms of interfaces, communication/control protocols with a specific focus on hardware architectures and components. A Web-oriented communication paradigm was adopted to enable complex interactions among sensors and actuators in the presence of technological heterogeneity. To provide service availability and reliability, despite the limited energy budget, a general framework inspired by the SDN principle was proposed. It was accurately presented in terms of component and communication/control protocols, with a specific focus on the SDN Controller design by characterizing its specific functionalities to handle data flows and to manage power budget distribution. This allowed the extension of the basic MQTT-SN protocol towards the SD-WPT-MQTT-SN approach, which is able to jointly optimize the QoS requirements and the network lifetime. Finally, the concepts of SDN and Fog Computing with the LTE Advanced Access Network has been combined in order to effectively handle vehicular services with strict real-time and location-based requirements. In the resulting architecture, a migration method, named swapping-migration, has been developed and executed. The performance of this approach has been evaluated in comparison with other methods in the literature in terms of end-to-end service completion delay, success probability and resource utilization through realistic simulations over an urban area. The results has indicated that the proposed swapping-migration scheme outperforms the alternatives, particularly for high service request rates and low capacity.

9.2 Directions for future work

Based on these findings, future work could focus on further integrating the SOHS approach within existing IoT solutions, both in terms of IoT Operating Systems such as Contiki and RIoT, and protocols like ZigBee and 6LoWPAN. Another direction could be to expand the software-defined ap-

proach to the physical layer of the communication stack, by enabling a full control on the transmission channels. Additionally, it is crucial to investigate and implement security-related aspects to ensure a secure and authenticated communication channel between the nodes and the controller. Moreover, machine learning techniques related to networking might be used to predict future operational conditions and optimize the weighting factors of the DRM algorithm. Focusing on the IoV and IoMT contexts, the proposed architecture could be further investigate to promote an heterogeneous wide area mobile network in which devices with different capabilities could coexist and collaborate to provide advanced applications. Finally, ML algorithm should be applied to perform accurate vehicular traffic and service request predictions.

Chapter 10

Publications

This research activity has led to several publications in international journals and conferences. These are summarized below.[1]

International Journals

1. Bonanni, M., Chiti, F., Fantacci, R. (2020). Mobile mist computing for the internet of vehicles. Internet Technology Letters, 3(6), e176

2. Bonanni, M., Chiti, F., Fantacci, R., Pierucci, L. (2021). Dynamic control architecture based on software defined networking for the internet of things. Future Internet, 13(5), 113.

3. Bartoli, C., Bonanni, M., Chiti, F., Pierucci, L., Cidronali, A., Collodi, G., Maddio, S. (2022). Toward the Web of Industrial Things: A Publish-Subscribe Oriented Architecture for Data and Power Management. Sensors, 22(13), 4882.

Submitted

1. Michele Bonanni, Francesco Chiti, Giovanni Collodi, Stefano Maddio, Giuseppe Pelosi, Laura Pierucci. IEEE Transactions on Network and Service Management. The Alliance of SDN and MQTT for Data and Power Management in Industrial Web of Things.

2. Michele Bonanni, Francesco Chiti, Laura Pierucci. IEEE Internet of Things Journal. SOHS: Self Organizing and self Healing smart solution for SD-WSN.

[1]The author's bibliometric indices are the following: H-index = 2, total number of citations = 12 (source: Google Scholar on Month 01, 2023).

International Conferences and Workshops

1. Bonanni, M., Chiti, F., Collodi, G., Maddio, S., Pelosi, G., Pierucci, L. (2022, July). A Parasitic Loaded Rectenna for Energy Harvesting Applications for the Internet of Things. In 2022 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting (AP-S/URSI) (pp. 2008-2009). IEEE.

Technical Reports

1. Bonanni M., Chiti F., Fantacci R. Software Defined Fog/Edge Networking for Internet of Vehicles: a Service-Oriented Reference Architecture. CNIT Technical Report-05: Internet of Things: Technologies, Challenges and Impact, pp. 183-206, ISBN 978-88-949-8238-1.

2. Barletti L., Bonanni M., Chiti F., Pecorella T., Picchi R., Rashid A., Raspini F. Securing IoT: Benefits and Challenges of SDN Approach. IEEE ComSc - Communications and Information Security Technical Committee Newsletter, pp. 3-5

Other Publications

1. F. Chiti, M. Bonanni - Internet: Prospettive, Architetture e Applicazioni. Chapters: Internet of Things, Web of Things and Software Defined Networking. Third edition, Esculapio, 2023.

Bibliography

[1] M. T. Abbas, A. Muhammad, and W.-C. Song, "Sd-iov: Sdn enabled routing for internet of vehicles in road-aware approach," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–16, 2019.

[2] K. Abboud, H. A. Omar, and W. Zhuang, "Interworking of dsrc and cellular network technologies for v2x communications: A survey," *IEEE transactions on vehicular technology*, vol. 65, no. 12, pp. 9457–9470, 2016.

[3] I. F. Akyildiz, A. Kak, and S. Nie, "6g and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133 995–134 030, 2020.

[4] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[5] T. A. Al-Janabi and H. S. Al-Raweshidy, "Efficient whale optimisation algorithm-based sdn clustering for iot focused on node density," in *2017 16th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2017, pp. 1–6.

[6] C. Alippi, R. Fantacci, D. Marabissi, and M. Roveri, "A cloud to the ground: The new frontier of intelligent and autonomous networks of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 14–20, 2016.

[7] A. A. Alkheir, I. S. Al-Anbagi, and H. T. Mouftah, "A statistical analysis of rf-energy harvesting in wireless networks," in *2015 IEEE International Conference on Ubiquitous Wireless Broadband (ICUWB)*, 2015, pp. 1–5.

[8] M. Aloqaily, V. Balasubramanian, F. Zaman, I. Al Ridhawi, and Y. Jararweh, "Congestion mitigation in densely crowded environments for augmenting qos in vehicular clouds," in *Proceedings of the 8th ACM Symposium*

on Design and Analysis of Intelligent Vehicular Networks and Applications. ACM, 2018, pp. 49–56.

[9] F. Antoniazzi and F. Viola, "Building the semantic web of things through a dynamic ontology," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 560–10 579, 2019.

[10] E. Baccarelli, F. Chiti, N. Cordeschi, R. Fantacci, D. Marabissi, R. Parisi, and A. Uncini, "Green multimedia wireless sensor networks: distributed intelligent data fusion, in-network processing, and optimized resource management," *IEEE Wireless Communications*, vol. 21, no. 4, pp. 20–26, 2014.

[11] M. Baddeley, R. Nejabati, G. Oikonomou, M. Sooriyabandara, and D. Simeonidou, "Evolving sdn for low-power iot networks," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 71–79.

[12] A. C. Baktir, A. Ozgovde, and C. Ersoy, "How can edge computing benefit from software-defined networking: A survey, use cases, and future directions," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2359–2391, 2017.

[13] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables devops: Migration to a cloud-native architecture," *Ieee Software*, vol. 33, no. 3, pp. 42–52, 2016.

[14] A. Banerjee and D. Hussain, "Sd-ear: Energy aware routing in software defined wireless sensor networks," *Applied Sciences-MDPI*, vol. 8, 2018.

[15] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, "Mqtt version 5.0," *OASIS Standard*, 2019. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[16] W. Bao, D. Yuan, Z. Yang, S. Wang, W. Li, B. B. Zhou, and A. Y. Zomaya, "Follow me fog: Toward seamless handover timing schemes in a fog computing environment," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 72–78, 2017.

[17] P. Barnaghi and A. Sheth, "On searching the internet of things: Requirements and challenges," *IEEE Intelligent Systems*, vol. 31, no. 6, pp. 71–75, 2016.

[18] T. M. Behera, U. C. Samal, and S. K. Mohapatra, "Energy-efficient modified leach protocol for iot application," *IET Wireless Sensor Systems*, vol. 8, no. 5, pp. 223–228, 2018.

[19] S. Bera, S. Misra, S. K. Roy, and M. S. Obaidat, "Soft-wsn: Software-defined wsn management system for iot applications," *IEEE Systems Journal*, vol. 12, no. 3, pp. 2074–2081, 2016.

[20] S. Bera, S. Misra, and A. V. Vasilakos, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, 2017.

[21] ——, "Software-defined networking for internet of things: A survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1994–2008, 2017.

[22] Y. Bi, G. Han, C. Lin, M. Guizani, and X. Wang, "Mobility management for intro/inter domain handover in software-defined networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1739–1754, Aug 2019.

[23] L. F. Bittencourt, M. M. Lopes, I. Petri, and O. F. Rana, "Towards virtual machine migration in fog computing," in *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*. IEEE, 2015, pp. 1–8.

[24] N. Bizanis and F. A. Kuipers, "Sdn and virtualization solutions for the internet of things: A survey," *IEEE Access*, vol. 4, pp. 5591–5606, 2016.

[25] C. Bormann, A. P. Castellani, and Z. Shelby, "Coap: An application protocol for billions of tiny internet nodes," *IEEE Internet Computing*, vol. 16, no. 2, pp. 62–67, 2012.

[26] H. Byun, "Mobile collector-based cost balancing scheme for uniform data gathering delay and energy consumption in wireless sensor actuator networking systems," *IEEE Sensors Journal*, vol. 20, no. 8, pp. 4260–4268, 2019.

[27] S. Cavalieri, "A proposal to improve interoperability in the industry 4.0 based on the open platform communications unified architecture standard," *Computers*, vol. 10, no. 6, 2021. [Online]. Available: https://www.mdpi.com/2073-431X/10/6/70

[28] S. Chen, T. Zhang, and W. Shi, "Fog computing," *IEEE Internet Computing*, vol. 21, no. 2, pp. 4–6, 2017.

[29] M. Chiang and T. Zhang, "Fog and iot: An overview of research opportunities," *IEEE Internet of things journal*, vol. 3, no. 6, pp. 854–864, 2016.

[30] F. Chiti, R. Fantacci, and L. Pierucci, "Energy efficient communications for reliable iot multicast 5g/satellite services," *Future Internet*, vol. 11, no. 8, 2019. [Online]. Available: https://www.mdpi.com/1999-5903/11/8/164

[31] ——, "A green routing protocol with wireless power transfer for internet of things," *Journal of Sensor and Actuator Networks*, vol. 10, no. 1, 2021.

[32] A. Cidronali, G. Collodi, S. Maddio, M. Passafiume, and G. Pelosi, "2-d doa anchor suitable for indoor positioning systems based on space and frequency diversity for legacy wlan," *IEEE Microwave and Wireless Components Letters*, vol. 28, no. 7, pp. 627–629, 2018.

[33] G. Collodi, S. Maddio, and G. Pelosi, "Design of a compact and highly efficient energy harvester system suitable for battery-less low cost on-board unit applications," *Electronics*, vol. 10, no. 1, 2021.

[34] R. Correia, N. B. Carvalho, and S. Kawasaki, "Continuously power delivering for passive backscatter wireless sensor networks," *IEEE Transactions on Microwave Theory and Techniques*, vol. 64, no. 11, pp. 3723–3731, 2016.

[35] J. H. Cox, J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley, and H. L. Owen, "Advancing software-defined networks: A survey," *Ieee Access*, vol. 5, pp. 25 487–25 526, 2017.

[36] P. Dallasega, "Industry 4.0 fostering construction supply chain management: Lessons learned from engineer-to-order suppliers," *IEEE Engineering Management Review*, vol. 46, no. 3, pp. 49–55, 2018.

[37] M. Damm, S.-H. Leitner, and W. Mahnke, "Opc unified architecture," *Springer Berlin, Heidelberg*, 2009.

[38] W. Dargie and C. Poellabauer, *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[39] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," *Computer*, vol. 49, no. 8, pp. 112–116, 2016.

[40] B. T. de Oliveira and C. B. Margi, "Tinysdn: enabling tinyos to software-defined wireless sensor networks," *XXXIV Simpósio Brasileiro de Redes de Computadores. Bahia*, pp. 1229–1237, 2016.

[41] M. Emara, M. C. Filippou, and D. Sabella, "Mec-assisted end-to-end latency evaluations for c-v2x communications," in *2018 European conference on networks and communications (EuCNC)*. IEEE, 2018, pp. 1–9.

[42] O. Flauzac, C. Santamaria, F. Nolot, and I. Woungang, "Application-aware sdn-based iterative reconfigurable routing protocol for internet of things (iot)," *International Journal on Communications Systems.IET*, vol. 33, no. 7, May 2020.

[43] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "Sdn-wise: Design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks," in *2015 IEEE conference on computer communications (INFOCOM)*. IEEE, 2015, pp. 513–521.

[44] F. Ganz, D. Puschmann, P. Barnaghi, and F. Carrez, "A practical evaluation of information processing and abstraction techniques for the internet of things," *IEEE Internet of Things journal*, vol. 2, no. 4, pp. 340–354, 2015.

[45] F. Giust, X. Costa-Perez, and A. Reznik, "Multi-access edge computing: An overview of etsi mec isg," *IEEE 5G Tech Focus*, vol. 1, no. 4, p. 4, 2017.

[46] S. GrÃ¼ner, J. Pfrommer, and F. Palm, "Restful industrial communication with opc ua," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 5, pp. 1832–1841, 2016.

[47] K. Ha, Y. Abe, Z. Chen, W. Hu, B. Amos, P. Pillai, and M. Satyanarayanan, "Adaptive vm handoff across cloudlets," *Technical Report CMU-CS-15-113*, 2015.

[48] T. Harter, B. Salmon, R. Liu, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau, "Slacker: Fast distribution with lazy docker containers," in *14th USENIX Conference on File and Storage Technologies (FAST 16)*, 2016, pp. 181–195.

[49] X. He, Z. Ren, C. Shi, and J. Fang, "A novel load balancing strategy of software-defined cloud/fog networking in the internet of vehicles," *China Communications*, vol. 13, no. 2, pp. 140–149, 2016.

[50] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, vol. 2, 2000, p. 10.

[51] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and openflow: From concept to implementation," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 2181–2206, 2014.

[52] Y.-F. Hu, Y.-S. Ding, L.-H. Ren, K.-R. Hao, and H. Han, "An endocrine cooperative particle swarm optimization algorithm for routing recovery problem of wireless sensor networks with multiple mobile sinks," *Information Sciences*, vol. 300, pp. 100–113, 2015.

[53] J. Huang, Y. Zhou, Z. Ning, and H. Gharavi, "Wireless power transfer and energy harvesting: Current status and future prospects," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 163–169, 2019.

[54] X. Huang, R. Yu, J. Kang, Y. Gao, S. Maharjan, S. Gjessing, and Y. Zhang, "Software defined energy harvesting networking for 5g green communications," *IEEE Wireless Communications*, vol. 24, no. 4, pp. 38–45, 2017.

[55] X. Huang, R. Yu, J. Kang, Z. Xia, and Y. Zhang, "Software defined networking for energy harvesting internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1389–1399, 2018.

[56] C.-L. I, S. Han, Z. Xu, S. Wang, Q. Sun, and Y. Chen, "New paradigm of 5g wireless internet," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 474–482, 2016.

[57] R. Iqbal, T. A. Butt, M. O. Shafique, M. W. A. Talib, and T. Umer, "Context-aware data-driven intelligent framework for fog infrastructures in internet of vehicles," *IEEE Access*, vol. 6, pp. 58 182–58 194, 2018.

[58] I. A. Ismaili, A. Azyat, N. Raissouni, N. B. Achhab, A. Chahboun, and M. Lahraoua, "Comparative study of zigbee and 6lowpan protocols," in *ICCWCS 2019: Third International Conference on Computing and Wireless Communication Systems, ICCWCS 2019, April 24-25, 2019, Faculty of Sciences, Ibn Tofaïl University-Kénitra-Morocco*. European Alliance for Innovation, 2019, p. 264.

[59] N. A. Jagadeesan and B. Krishnamachari, "Software-defined networking paradigms in wireless networks: A survey," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, pp. 1–11, 2014.

[60] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared systems," Technical Report DEC-TR-301, Digital Equipment Corporation, 1984.

[61] F. Jameel, Z. Chang, J. Huang, and T. Ristaniemi, "Internet of autonomous vehicles: Architecture, features, and socio-technological challenges," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 21–29, August 2019.

[62] S. Jazaeri, S. Jabbehdari, P. Asghari, and et al., "Edge computing in sdn-iot networks: a systematic review of issues, challenges and solutions," *Cluster Comput*, vol. 24, p. 53187Ã¢ÂÂ3228, 2021.

[63] S. R. Jondhale, R. Maheswar, J. Lloret, S. R. Jondhale, R. Maheswar, and J. Lloret, "Fundamentals of wireless sensor networks," *Received Signal Strength Based Target Localization and Tracking Using Wireless Sensor Networks*, pp. 1–19, 2022.

[64] K. Kalkan and S. Zeadally, "Securing internet of things with software defined networking," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 186–192, 2018.

[65] C. Ke, M. Wu, W. Hsu, and C. Chen, "Discover the optimal iot packets routing path of software-defined network via artificial bee colony algorithm," in *Wireless Internet. WiCON 2019. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering,*, vol. 317. Springer, Cham., 2020.

[66] A. Khakimov, A. A. Ateya, A. Muthanna, I. Gudkova, E. Markova, and A. Koucheryavy, "Iot-fog based system structure with sdn enabled," in *Proceedings of the 2nd International Conference on Future Networks and Distributed Systems*. ACM, 2018, p. 62.

[67] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A survey on software-defined wireless sensor networks: Challenges and design requirements," *IEEE access*, vol. 5, pp. 1872–1899, 2017.

[68] L. Kong, J.-S. Pan, P.-W. Tsai, S. Vaclav, and J.-H. Ho, "A balanced power consumption algorithm based on enhanced parallel cat swarm optimization for wireless sensor network," *International Journal of Distributed Sensor Networks*, vol. 11, no. 3, p. 729680, 2015.

[69] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2014.

[70] N. Kumar and D. P. Vidyarthi, "A green routing algorithm for iot-enabled software defined wireless sensor network," *IEEE Sensors Journal*, vol. 18, no. 22, pp. 9449–9460, 2018.

[71] B. B. Letswamotse, R. Malekian, C.-Y. Chen, and K. M. Modieginyane, "Software defined wireless sensor networks (sdwsn): a review on efficient resources, applications and technologies," *Journal of Internet Technology*, vol. 19, no. 5, pp. 1303–1313, 2018.

[72] J. Li, X. Li, J. Yuan, R. Zhang, and B. Fang, "Fog computing-assisted trustworthy forwarding scheme in mobile internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2778–2796, 2018.

[73] J. Li, X. Shen, L. Chen, D. P. Van, J. Ou, L. Wosinska, and J. Chen, "Service migration in fog computing enabled cellular networks to support real-time vehicular communications," *IEEE Access*, vol. 7, pp. 13 704–13 714, 2019.

[74] D. S. Linthicum, "Connecting fog and cloud computing," *IEEE Cloud Computing*, vol. 4, no. 2, pp. 18–20, 2017.

[75] J. Lu, S. Liu, Q. Wu, and Q. Qiu, "Accurate modeling and prediction of energy availability in energy harvesting real-time embedded systems," in *International Conference on Green Computing*, 2010, pp. 469–476.

[76] T. Luo, H.-P. Tan, and T. Q. Quek, "Sensor openflow: Enabling software-defined wireless sensor networks," *IEEE Communications letters*, vol. 16, no. 11, pp. 1896–1899, 2012.

[77] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Communications*, vol. 25, no. 1, pp. 140–147, 2017.

[78] S. Maddio, A. Cidronali, M. Passafiume, G. Collodi, M. Lucarelli, and S. Maurri, "Multipath robust azimuthal direction of arrival estimation in dual-band 2.45?5.2 ghz networks," *IEEE Transactions on Microwave Theory and Techniques*, vol. 65, no. 11, pp. 4438–4449, 2017.

[79] S. Maddio, M. Passafiume, A. Cidronali, and G. Manes, "A distributed positioning system based on a predictive fingerprinting method enabling submetric precision in ieee 802.11 networks," *IEEE Transactions on Microwave Theory and Techniques*, vol. 63, no. 12, pp. 4567–4580, 2015.

[80] C. B. Margi, R. C. Alves, G. A. N. Segura, and D. A. Oliveira, "Software-defined wireless sensor networks approach: Southbound protocol and its performance evaluation," *Open Journal of Internet Of Things (OJIOT)*, vol. 4, no. 1, pp. 99–108, 2018.

[81] A. Mavromatis, C. Colman-Meixner, A. P. Silva, X. Vasilakos, R. Nejabati, and D. Simeonidou, "A software-defined iot device management framework for edge and cloud computing," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1718–1735, 2020.

[82] S. A. Mirbozorgi, P. Yeon, and M. Ghovanloo, "Robust wireless power transmission to mm-sized free-floating distributed implants," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 3, pp. 692–702, 2017.

[83] P. Mishra, D. Puthal, M. Tiwary, and S. P. Mohanty, "Software defined iot systems: Properties, state of the art, and future research," *IEEE Wireless Communications*, vol. 26, no. 6, pp. 64–71, 2019.

[84] M. Mukherjee, L. Shu, and D. Wang, "Survey of fog computing: Fundamental, network applications, and research challenges," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 1826–1857, 2018.

[85] N. Naik, "Choice of effective messaging protocols for iot systems: Mqtt, coap, amqp and http," in *2017 IEEE international systems engineering symposium (ISSE)*. IEEE, 2017, pp. 1–7.

[86] M. Nelson, B.-H. Lim, G. Hutchins *et al.*, "Fast transparent migration for virtual machines." in *USENIX Annual technical conference, general track*, 2005, pp. 391–394.

[87] Y. Ni, L. Cai, J. He, A. Vinel, Y. Li, H. Mosavat-Jahromi, and J. Pan, "Toward reliable and scalable internet of vehicles: Performance analysis and resource management," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 324–340, 2019.

[88] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," *IEEE Communications surveys & tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[89] S. M. A. Oteafy and H. S. Hassanein, "Iot in the fog: A roadmap for data-centric iot development," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 157–163, 2018.

[90] L.-S. Otu, "The internet of economic things: The socio-economic transformation value of the internet of things (iot)," in *2019 International Conference on Cyber Security and Internet of Things (ICSIoT)*. IEEE, 2019, pp. 109–113.

[91] J. Pan and J. McElhannon, "Future edge cloud and edge computing for internet of things applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 439–449, 2018.

[92] J.-H. Park, H.-S. Kim, and W.-T. Kim, "Dm-mqtt: An efficient mqtt based on sdn multicast for massive iot communications," *Sensors*, vol. 18, no. 9, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/9/3071

[93] J. H. Park, S. Lee, S. Yun, H. Kim, and W.-T. Kim, "Dependable fire detection system with multifunctional artificial intelligence framework," *Sensors*, vol. 19, no. 9, p. 2025, 2019.

[94] M. Passafiume, G. Collodi, and A. Cidronali, "Design principles of battery-less transponder for vehicular dsrc at 5.8 ghz," *IEEE Journal of Radio Frequency Identification*, vol. 4, no. 4, pp. 491–505, 2020.

[95] C. Perera, C. H. Liu, and S. Jayawardena, "The emerging internet of things marketplace from an industrial perspective: A survey," *IEEE transactions on emerging topics in computing*, vol. 3, no. 4, pp. 585–598, 2015.

[96] Pradeepa R and M. Pushpalatha, "Sdn enabled spin routing protocol for wireless sensor networks," in *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016, pp. 639–643.

[97] J. A. Puente Fernández, L. J. García Villalba, and T.-H. Kim, "Software defined networks in wireless sensor architectures," *Entropy*, vol. 20, no. 4, p. 225, 2018.

[98] C. Puliafito, C. Vallati, E. Mingozzi, G. Merlino, F. Longo, and A. Puliafito, "Container migration in the fog: A performance evaluation," *Sensors*, vol. 19, no. 7, p. 1488, 2019.

[99] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing iot services through software defined networking and edge computing: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 3, pp. 1761–1804, 2020.

[100] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing iot services through software defined networking and edge computing: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1761–1804, 2020.

[101] D. Raggett, "The web of things: Challenges and opportunities," *Computer*, vol. 48, no. 5, pp. 26–32, 2015.

[102] ——, "The web of things: Challenges and opportunities," *Computer*, vol. 48, no. 5, pp. 26–32, 2015.

[103] M. Raikar, M. S M, and M. Mulla, "Software defined internet of things using lightweight protocol," *Procedia Computer Science*, vol. 171, pp. 1409–1418, 01 2020.

[104] B. Rauf, H. Abbas, A. M. Sheri, W. Iqbal, and A. W. Khan, "Enterprise integration patterns in sdn: A reliable, fault-tolerant communication framework," *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6359–6371, 2021.

[105] M. N. H. Reza, C. A. N. Malarvizhi, S. Jayashree, and M. Mohiuddin, "Industry 4.0–technological revolution and sustainable firm performance," in *2021 Emerging Trends in Industry 4.0 (ETI 4.0)*. IEEE, 2021, pp. 1–6.

[106] K. Routh and T. Pal, "A survey on technological, business and societal aspects of internet of things by q3, 2017," in *2018 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*. IEEE, 2018, pp. 1–4.

[107] T. Ruan, Z. J. Chew, and M. Zhu, "Energy-aware approaches for energy harvesting powered wireless sensor nodes," *IEEE Sensors Journal*, vol. 17, no. 7, pp. 2165–2173, 2017.

[108] S. Selvakennedy, S. Sinnappan, and Y. Shang, "A biologically-inspired clustering protocol for wireless sensor networks," *Computer Communications*, vol. 30, no. 14, pp. 2786 – 2801, 2007.

[109] A. Shafique, G. Cao, M. Aslam, M. Asad, and D. Ye, "Application-aware sdn-based iterative reconfigurable routing protocol for internet of things (iot)," *Sensors*, vol. 12, no. 3521, 2020.

[110] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5g for vehicular communications," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 111–117, Jan 2018.

[111] J. Shi, M. Sha, and Z. Yang, "Distributed graph routing and scheduling for industrial wireless sensor-actuator networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1669–1682, 2019.

[112] S. Sonavane, V. Kumar, and B. Patil, "Designing wireless sensor network with low cost and low power," in *2008 16th IEEE International Conference on Networks*. IEEE, 2008, pp. 1–5.

[113] P. Srinivasa Ragavan and K. Ramasamy, "Software defined networking approach based efficient routing in multihop and relay surveillance using lion optimization algorithm," *Computer Communications*, vol. 150, pp. 764 – 770, 2020.

[114] R. D. Sriram and A. Sheth, "Internet of things perspectives," *IT Professional*, vol. 17, no. 3, pp. 60–63, 2015.

[115] O. Standard, "Mqtt version 5.0," *Retrieved June*, vol. 22, p. 2020, 2019.

[116] A. Stanford-Clark and H. L. Truong, "Mqtt for sensor networks (mqtt-sn) protocol specification," *International business machines (IBM) Corporation version*, vol. 1, no. 2, pp. 1–28, 2013.

[117] X. Sun and N. Ansari, "Edgeiot: Mobile edge computing for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 22–29, 2016.

[118] T. Taleb and A. Ksentini, "Follow me cloud: interworking federated clouds and distributed mobile networks," *IEEE Network*, vol. 27, no. 5, pp. 12–19, 2013.

[119] T. Taleb, A. Ksentini, and P. A. Frangoudis, "Follow-me cloud: When cloud services follow mobile users," *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 369–382, 2016.

[120] R. Tamria and S. Rakrak, "The mi-sdn system to manage mqtt data in an interoperable iot wireless network," *Turkish Journal of Computer and MathematicsEducation*, vol. 12, no. 5, 2021.

[121] E. Uhlemann, "Connected-vehicles applications are emerging," *IEEE Vehicular Technology Magazine*, vol. 11, no. 1, pp. 25–96, March 2016.

[122] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. Chan, and K. K. Leung, "Dynamic service migration and workload scheduling in edge-clouds," *Performance Evaluation*, vol. 91, pp. 205–228, 2015.

[123] K. Wang, Y. Wang, Y. Sun, S. Guo, and J. Wu, "Green industrial internet of things architecture: An energy-efficient perspective," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 48–54, 2016.

[124] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung, "Dynamic service placement for mobile micro-clouds with predicted future costs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1002–1016, 2016.

[125] Y.-C. Wang, C.-C. Hu, and Y.-C. Tseng, "Efficient placement and dispatch of sensors in a wireless sensor network," *IEEE transactions on mobile computing*, vol. 7, no. 2, pp. 262–274, 2007.

[126] M. Weyrich and C. Ebert, "Reference architectures for the internet of things," *IEEE Software*, vol. 33, no. 1, pp. 112–116, 2016.

[127] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[128] W. Xiang, N. Wang, and Y. Zhou, "An energy-efficient routing algorithm for software-defined wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7393–7400, 2016.

[129] H. Zhang, Y.-x. Guo, Z. Zhong, and W. Wu, "Cooperative integration of rf energy harvesting and dedicated wpt for wireless sensor networks," *IEEE Microwave and Wireless Components Letters*, vol. 29, no. 4, pp. 291–293, 2019.

[130] Y. Zhang, H. Zhang, K. Long, Q. Zheng, and X. Xie, "Software-defined and fog-computing-based next generation vehicular networks," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 34–41, Sep. 2018.

[131] Z. Zhang, H. Pang, A. Georgiadis, and C. Cecati, "Wireless power transferâan overview," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 2, pp. 1044–1058, 2018.

[132] C. Zhu, V. C. M. Leung, L. Shu, and E. C. . Ngai, "Green internet of things for smart world," *IEEE Access*, vol. 3, pp. 2151–2162, 2015.

[133] M. Zhu, J. Cao, Z. Cai, Z. He, and M. Xu, "Providing flexible services for heterogeneous vehicles: an nfv-based approach," *IEEE Network*, vol. 30, no. 3, pp. 64–71, May 2016.