Mechanisms for integration of MEC and NFV for 5G networks in dynamic and heterogeneous scenarios

by

Kiril Antevski

A dissertation submitted by in partial fulfillment of the requirements for the degree of Doctor of Philosophy in

Telematics Engineering

Universidad Carlos III de Madrid

Advisor:

Dr. Carlos J. Bernardos

June 2022

“There is nothing impossible to him who will try.”

— Alexander the Great

ACKNOWLEDGEMENTS

The final chapter of my PhD journey is close to the end. Many people I have met prior and during the journey have contributed towards moving forward.

First, and foremost, I want to thank my mentor Carlos Jesus for the opportunity and trust given to join the UC3M team into following the PhD path. My exceptional gratitude for the research freedom, and the continuous support for application of new ideas and concepts. I learnt a lot on how to be more patiently determined in reaching every goal while taking positive, organized and cautious approach. I have always admired your calm, measured and diplomatic leadership in every collaboration or project we worked on as well as the humble, cheerful, and fun personality that you radiate. As well, thanks to Antonio for sharing the mentoring role with a very real and a direct approach which helped into strengthening the vision, the work, and me as a person.

I want to thank all the colleagues from the all the projects we worked together. Special thanks to Jordi, Josep, Juan, Konstantin, Denis, Farhana, Xi, and Andres. It has been a great pleasure to work and learn from you while also enjoy the face-to-face meetings we used to have.

To the friends and colleagues from the o ce with whom we shared together all the painful and all the happy moments: Milan, Sergio (herma), Stefano, Jorge, Luigi, Guimarães, Winnie, Victor, Borja, Luca, Gines, Marco, Patri, Miguel, Nuria, Oscar, Jonas, Lewis, Pelayo, and Cristina. As well the fresh research roster: Khasa, Amir, Gonzalo, Dulce, Raul, and Natalia. Thanks to all for the warm, fun and cheerful environment, I will never forget all the lunch topics about football, politics, and life in general as we grew together. Sorry for all the bragging about crypto & Bitcoin, politics, and conspiracy theories coming from me. *I tata bi sine* and *Rulmen* football teams were epic.

Thanks to Filip, Tina, Ana, Natasha, Ivan for the true friendship and support. Special thanks to Milan, for being my friend and colleague since the very start for more than 15 years ago. We changed two countries together while growing thorough passing exams, looking for jobs, performing experiments, publishing papers.

Thanks to Bal, my girlfriend, with whom I shared a significant part of the PhD journey, living through very happy moments and lock-downs. Thanks for all the love, support, cheerfulness, and making me less worried, more positive, clear-minded, and confident person in every step forward.

Finally, thanks to my family, for the unconditional love, support, critics and expectations. Thanks for the best advises, directions, questions, doubts, and celebrations. Although far away, I have always felt closely your strong support, enthusiasm and trust in me. Truly, without you, I wouldn't be able to climb through the PhD journey.

PUBLISHED AND SUBMITTED CONTENT

Conforming to the Law 14 2011 about plagiarism and Code of Good Practices of the UC3M Doctoral School, I hereby detail a list of articles and other contributions I have (co)authored that are included as part of this thesis and have been published or otherwise submitted for publication.

Published content

Antevski, K., Bernardos, C. J., Cominardi, L., de la Oliva, A., & Mourad, A. (2020). On the integration of NFV and MEC technologies: architecture analysis and benefits for edge robotics. Computer Networks, 175, 107274.

- This work is wholly included in Chapter 1, 2 and 5.
- The role of the author of this thesis was to evaluate, compare, tackle the integration problems and provide tutorial view of integration of MEC in NFV.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., Bernardos, C. J., (2022). Federation in Dynamic Environments: Can Blockchain Be the Solution?. *IEEE Communications Magazine*. IEEE.

- This work is wholly included in 5.
- The role of the author of this thesis was to define federation and its characteristics, emphasize the federation issues in dynamic enviroments, propose solution with applying Blockchain, implement experimental scenario and analyze the obtained results.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., Martín-Pérez, J., Garcia-Saavedra, A., Bernardos, C.J., Li, X., Baranda, J., Mangues-Bafalluy, J., Martnez, R. and Vettori, L., 2020, June. A Q-learning strategy for federation of 5G services. In ICC 2020-2020 IEEE International Conference on Communications (ICC) (pp. 1-6). IEEE.

- This work is wholly included in Chapter 6.
- The role of the author of this thesis was to design and implement the Q-learning solution, assist with the experiments.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., Groshev, M., Baldoni, G., & Bernardos, C. J. (2020, November). DLT federation for Edge robotics. In 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) (pp. 71-76). IEEE.

- This work is wholly included in Chapter 3 and 5.
- The role of the author of this thesis was to analyze the federation problem, design the solution, design and execute the experiments, and analyze the results.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., Girletti, L., Bernardos, C. J., de la Oliva, A., Baranda, J., & Mangues-Bafalluy, J. (2021). A 5G-based eHealth monitoring and emergency response system: experience and lessons learned. IEEE Access, 9, 131420-131429.

- This work is wholly included in Chapter 4.
- The role of the author of this thesis, in collaboration with other co-authors, was to design the scenario solution, assist in development of the mobile application, establish the end-to-end experimental setup, assist the experiments, partial obtain of the results.
- The material from this source included in this thesis is not singled out with typographic means and references.

Javed, F., **Antevski, K.**, Mangues-Bafalluy, J., Guipponi, L., and Bernardos, C. J. (2022). Distributed Ledger Technologies For Network Slicing: A Survey. IEEE Access.

- This work is partially included in Chapter 5.
- The role of the author of this thesis was to analyze the application of Blockchain in vertical industries, provide insight of how Blockchain technology works, what the limitations are, and how can be applied to network slicing.
- The material from this source included in this thesis is not singled out with typographic means and references.

Martín-Pérez, J., **Antevski, K.**, Garcia-Saavedra, A., Li, X., & Bernardos, C. J. (2021). DQN Dynamic Pricing and Revenue Driven Service Federation Strategy. IEEE Transactions on Network and Service Management, 18(4), 3987-4001.

- This work is partially included in Chapter 6.
- The role of the author of this thesis was to assist in defining the business scenario, and implementation of the Q-learning algorithm.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., Groshev, M., Cominardi, L., Bernardos, C. J., Mourad, A., & Gazda, R. (2018, December). Enhancing edge robotics through the use of context information. In Proceedings of the Workshop on Experimentation and Measurements in 5G (pp. 7-12).

- This work is partially included in Chapter 2.
- The role of the author of this thesis was to assist and execute the experiments, design the algorithm.
- The material from this source included in this thesis is not singled out with typographic means and references.

Antevski, K., & Bernardos, C. J. (2020). Federation of 5G services using distributed ledger technologies. Internet Technology Letters, 3(6), e193.

- This work is partially included in Chapter 5.
- The role of the author of this thesis was to analyze the federation problem, design the solution, design and execute the experiments, and analyze the results.
- The material from this source included in this thesis is not singled out with typographic means and references.

Submitted content

Antevski, K., Bernardos, C. J., (2022). Applying Blockchain consensus mechanisms to Network Service Federation: Analysis and performance evaluation.

- This work is wholly included in 5.
- The role of the author of this thesis was to define federation and its characteristics, emphasize the federation issues in dynamic enviroments, propose solution with applying Blockchain, implement experimental scenario and analyze the obtained results.
- The material from this source included in this thesis is not singled out with typographic means and references.

OTHER RESEARCH MERITS

This chapter provides a list of additional publications where I have participated as an author or co-author which are related to this thesis. In addition, it provides an overview of the several EU-funded projects where I was directly involved by applying the work described in this thesis. It underlines my participation in a standardization body white paper, and a book chapter.

Related publications and submitted works

Antevski, K., Martín-Pérez, J., Molner, N., Chiasserini, C. F., Malandrino, F., Frangoudis, P., ... Gharbaoui, M. (2018, September). Resource orchestration of 5G transport networks for vertical industries. In 2018 *IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)* (pp. 158-163). IEEE.

- The role of the author of this thesis was to provide the federation content.
- The material from this source included in this thesis is not singled out with typographic means and references.

Giust, F., Verin, G., **Antevski, K.**, Chou, J., Fang, Y., Featherstone, W., ... & Zhou, Z. (2018). MEC deployments in 4G and evolution towards 5G. *ETSI White paper*, 24 (2018), 1-24.

- The role of the author of this thesis was to provide contribution into the management of the MEC application.
- The material from this source included in this thesis is not singled out with typographic means and references.

Li, X., Garcia-Saavedra, A., Costa-Perez, X., Bernardos, C. J., Guimarães, C., **Antevski, K.**, ... & López, D. R. (2021). 5Growth: An end-to-end service platform for automated deployment and management of vertical services over 5G networks. *IEEE Communications Magazine*, 59(3), 84-90.

- The role of the author of this thesis was to provide introduction, and assist on the gap analysis.
- The material from this source included in this thesis is not singled out with typographic means and references.

Li, X., Mangues-Bafalluy, J., Pascual, I., Landi, G., Moscatelli, F., **Antevski, K.**, ... & Ksentini, A. (2018, April). Service orchestration and federation for verticals.

In 2018 *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (pp. 260-265). IEEE.

- The role of the author was to provide the federation content.
- The material from this source included in this thesis is not singled out with typographic means and references.

Hortiguela, J. B., Mangues-Bafalluy, J., Martinez, R., Vettori, L., **Antevski, K.**, Bernardos, C. J., & Li, X. (2020). Realizing the network service federation vision: Enabling automated multidomain orchestration of network services. *IEEE Vehicular Technology Magazine*, 15(2), 48-57.

- The role of the author of this thesis was to assist in the federation solution.
- The material from this source included in this thesis is not singled out with typographic means and references.

Valcarenghi, L., Martini, B., **Antevski, K.**, Bernardos, C. J., Landi, G., Capitani, M., ... & Tomakh, K. (2018, December). A framework for orchestration and federation of 5G services in a multi-domain scenario. In *Proceedings of the Workshop on Experimentation and Measurements in 5G* (pp. 19-24).

- The role of the author of this thesis was to provide the federation solution.
- The material from this source included in this thesis is not singled out with typographic means and references.

Baranda, J., Mangues-Bafalluy, J., Vettori, L., Martínez, R., Landi, G., & **Antevski, K.** (2019, July). Composing services in 5G-TRANSFORMER. In *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing* (pp. 407-408).

- The role of the author of this thesis was to assist in the federation design solution.
- The material from this source included in this thesis is not singled out with typographic means and references.

Cominardi, L., Abdullaziz, O. I., **Antevski, K.**, Chundrigar, S. B., Gdowski, R., Kuo, P. H., ... & Zabala, A. (2018, April). Opportunities and challenges of joint edge and fog orchestration. In 2018 *IEEE Wireless Communications and Networking Conference Workshops (WCNCW)* (pp. 344-349). IEEE.

- The role of the author of this thesis was to address the challenges of the federation mechanisms.
- The material from this source included in this thesis is not singled out with typographic means and references.

Baranda, J., Mangues-Bafalluy, J., Vettori, L., Martínez, R., **Antevski, K.**, Girletti, L., ... & Gharbaoui, M. (2020, July). NFV service federation: Enabling multi-provider eHealth emergency services. In IEEE INFOCOM 2020-IEEE *Conference on Computer Communications Workshops* (INFOCOM WKSHPS) (pp. 1322-1323). IEEE.

- The role of the author of this thesis was to assist in the federation solution, and implementation of the use-case.
- The material from this source included in this thesis is not singled out with typographic means and references.

Baranda, J., Mangues-Bafalluy, J., Martínez, R., Vettori, L., **Antevski, K.**, Bernardos, C. J., & Li, X. (2020, June). 5G-TRANSFORMER meets Network Service Federation: design, implementation and evaluation. In 2020 6th IEEE *Conference on Network Softwarization (NetSoft)* (pp. 175-179). IEEE.

- The role of the author of this thesis was to contribute into design of the federation workflow and solution.
- The material from this source included in this thesis is not singled out with typographic means and references.

Groshev, M., Martín-Pérez, J., **Antevski, K.**, de la Oliva, A., & Bernardos, C. J. (2021, June). COTORRA: COntext-aware Testbed fOR Robotic Applications. In Proceedings of the *1st Workshop on Serverless mobile networking for 6G Communications* (pp. 7-12).

- The role of the author of this thesis was to contribute into design of the COTTORA architectural design and DLT experimental scenario.
- The material from this source included in this thesis is not singled out with typographic means and references.

Participation and role in EU funded projects

5G-TRANSFORMER

The *5G-TRANSFORMER* project (2017-2019) was an EU-funded project (H2020-ICT-2016-2 grant agreement 761536) aiming at transforming today's mobile transport network into an SDN NFV based Mobile Transport and Computing Platform (MTP), which brings the network slicing paradigm into mobile transport networks by provisioning and managing 5G-TRANSFORMER slices tailored to the specific needs of vertical industries across single or multiple domains. Since one of the key novelty of 5G is the creation of tailor-made infrastructure to meet vertical industries' requirements, the 5G-TRANSFORMER concept was driven by the automotive, eHealth, and media and entertainment vertical

industries. The technical approaches to realize the use-cases were: (i) enable vertical industries to meet their service requirements within customised 5G-TRANSFORMER slices, and (ii) aggregate and federate transport networking and computing fabric, from the edge all the way to the core and cloud, to create and manage 5G-TRANSFORMER slices throughout a federated and constituent virtualized infrastructure.

The role and the activities of the author of this thesis during the lifetime of the 5G-TRANSFORMER project were:

Active participation since the start of the project (September 2017) until the end of the project (December 2019).

Mainly involved in WP1, WP4, WP5: design of the 5G-TRANSFORMER architecture; design of the 5G-T Service Orchestrator.

Task leader of the T4.3 APIs and Service Federation.

Driving the design, and execution of the federation concept.

Coordinating the design, implementation, execution, and results reporting on e-Health use-case.

Contribution and editorial roles in several deliverables (D1.2, D1.3, D4.1, D4.3, D4.4, D5.2, D5.3).

Demonstration of the e-Health use-case demo[1]

5G-Coral

The 5G-CORAL project was based on the ability of edge and fog computing in the Radio Access Network (RAN) to enable access convergence. This was envisioned by the means of an integrated and virtualized networking and computing solution. Virtualized functions, context-aware services, users and third-party applications are integrated together to o er enhanced connectivity and excelling quality of experience. The proposed solution consisted of two major building blocks: (i) the Edge and Fog computing System (EFS) compiled with all the edge and fog computing substrate o ered as a shared hosting environment for virtualized functions, services, and applications; and (ii) the Orchestration and Control System (OCS) responsible for managing, orchestrating and controlling the EFS, including its interworking with external (non-EFS) domains.

The role and the activities of the author of this thesis during the lifetime of the 5G-Coral project were:

Active participation since the start of the project (September 2017) until the end of 2018 (December 2018).

[1]https: www.youtube.com watch?v BGHzJ1SMbzQ

Mainly involved in WP1, WP3: design of the 5G-Coral architecture; design of the federation concept for 5G-Coral.

Involved in the design of the 5G-Coral federation concept.

Contribution to a deliverable (D3.2).

5Growth

The vision of the *5Growth* project is to enhance vertical industries such as Industry 4.0, Transportation, and Energy with an AI-driven Automated and Shareable 5G End-to-End Solution allowing them to seamlessly achieve the key performance targets. 5Growth automates the support process for diverse industry verticals through (i) a vertical portal which is a simple User Interface (UI) in charge of receiving verticals service requirements and translating to deploy the respective network slices on top of 5Growth or other 5G End-to-End platforms, (ii) closed-loop automation and SLA control for vertical services lifecycle management and (iii) AI-driven end-to-end network solutions to jointly optimize Access, Transport, Core and Cloud, Edge and Fog resources, across multiple technologies and domains.

One of the main objectives of 5Growth is the business and technical validation of 5G technologies from the verticals' perspective, followed by a field-trial approach on vertical sites (TRL 6-7). Multiple use cases of vertical industries have been field-trialed on four vertical-owned sites in close collaboration with vendors and operators participating in the 5Growth project. 5Growth leverages on 5G-PPP Phase 2 projects where slicing, virtualization and multi-domain solutions for the creation and provisioning of vertical services has been developed and validated, e.g. 5G-MONARCH and 5G-TRANSFORMER. Two of the ICT-17-2018 5G End-to-End platforms, 5G EVE and 5G-VINNI, have been selected for the field trials in order to demonstrate the 5Growth specific vertical use cases.

The role and the activities of the author of this thesis during the lifetime of the 5Growth project are:

Active participation since the start of the project (June 2019) until the end of the project (February 2022).

Mainly involved in WP1, WP2, WP3, WP4: design of the 5Growth architecture; gap analysis; Innovalia pilot design, planning and implementation.

Involved in Inovation 6 (I6) End-to-End Orchestration. Federation and Inter-Domain.

Driving the design, and execution of the Distributed Ledger Technology (DLT) federation concept.

Contribution to several deliverables (D2.1, D2.3).

Books

Communication Networks and Service Management in the Era of Artificial Intelligence and Machine Learning - (2021, September), *John Wiley Sons, Ltd*

– Contributor to Chapter 4 – Self-Managed 5G Networks

ABSTRACT

The 5G technology presents a significant leap into making the Information and Communication technology and integral part of the industries, and societies. Enhanced connectivity features unlock a range of di erent applications that provide unique user experience such as virtual and augmented reality, or mission-critical communications that improve the healthcare and environmental protection, digital twin for optimizing the production lines, etc. Besides the new radio technology, the virtualization technology is the major enabler of most of the exciting novel applications. Viritualization enables service providers to customize and shape the existing computing, networking and storage infrastructure to accommodate the requirements of the di erent range of customers often referred as vertical industries.

The Network Function Virtualization (NFV) with the Software-defined Networking (SDN) are the key technologies that enable deployment of multiple isolated and customized networks on top of a single administrative domain infrastructure. The Multi-access Edge technology revamps carrier's infrastructure with application-oriented capabilities feeding applications with context information to elevate the user experience. Even though initially projected as a mobile operator technology, it is applicable to any service provider.

This thesis departs from the point on how to integrate both, NFV and MEC, for different environments and scenarios. The MEC technology is not virtualized intrinsically hence the first part of the thesis explores the integration of MEC in NFV environment. Initially a MEC application and the utilization of radio context information is showcased through an Edge robotics scenario. Later the full integration of virtualized MEC components within an NFV infrastructure is elaborated through categorization, and proposed solution in tackling integration issues. Further, a tutorial is presented on how the exemplary Edge robotics would be deployed, terminated and managed in an MEC in NFV environment. The elaborated procedures present high compatibility and readiness for MEC in NFV future deployments. The findings are compared with existing works on the similar topic.

The joint, or horizontal, integration of MEC in NFV is referred to a single administrative domain. The rest of the thesis is focusing on how administrative domains are able to fulfill vertical requirements by deploying end-to-end services across multiple domains. One of the thesis contribution is towards the definition, characterization and classification of federation - the process of deploying NFV services across multi-domain scenarios. Further the federation scenario is showcased in a static environment for a novel mission-critical eHealth application. All the federation functionalities are demonstrated in a real-case experimental emergency scenario for a patient su ering from a heart-attack. The assumption is that the federation occurs between two administrative domains enabling

end-to-end AR VR emergency services spread across two NFV based infrastructures. The obtained experimental results provide improvement in the future emergency events while leveraging on novel technologies such as AR VR. The drawbacks are evaluated accordingly.

The use of both MEC and NFV enables better user experience, customized networks and it is a big step towards automation, and reactive network life-cycle management. The following part of the thesis focuses on how to apply the federation concept in dynamic environments - where the conditions change rapidly, the resources are volatile and the relationships between administrative domains are established on-the-fly or unexpectedly broken. Blockchain as a Distributed Ledger Technology (DLT) is applied to facilitate and build trust in the brief negotiation process between mutually unknown administrative domains. A concrete step-by-step process is proposed which its application, in orchestration and life-cycle management (e.g., healing process), of emulated NFV service has been experimentally evaluated across multiple Blockchain platforms. Additionally, the Blockchain solution is applied in a small-scale Edge robotics experimental scenario. The Edge robotics service is a MEC-in-NFV based remote control application for mobile robots which leverage the DLT federation to extend the robot driving range by deploying radio network extension on top of an external domain infrastructure, without any interruption or downtime of the end-to-end Edge robotics service.

In the last part of the thesis the focus is set on how service provides or telco operators may increase their profit margins by leveraging the federation process and using Machine Learning algorithms to generate a profitable decision of whether to federate a service or deploy the service over the constituent infrastructure. The application of Reinforcement learning algorithms such as Q-learning provides a promising near-optimal results. These are improved with the application of Deep Q-learning techniques through the use of real dynamic price fluctuations for service o erings.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

3GPP 3rd Generation Partnership Project

5G-PPP 5G Infrastructure Public Private Partnership

4G 4th Generation of mobile systems

5G 5th Generation of mobile systems

5GS 5G System

5GC 5G Core

5GT 5G-TRANSFORMER

5GT-VS 5GT Vertical Slicer

5GT-SO 5GT Service Orchestrator

5GT-MTP 5GT Mobile Transport and Computing Platform

5GT-MON 5GT Monitoring

ADMM Alternating Direction Method of Multipliers

AD3 Alternating Directions Dual Decomposition

AD Administrative Domain

AI Artificial Intelligence

AN Access Network

AP Access Point

API Application Programming Interface

AR Augmented Reality

AR VR Augmented Virtual Reality

C-RAN Centralized Cloud RAN

CAPEX Capital Expenditure

CDF Cumulative Distribution Function

CN Core Network

CPU Central Processing Unit

CTTC Centre Tecnològic Telecomunicacions Catalunya

CU Central Cloud Unit

DLT Distributed Ledger Technology

DQN Deep Q Network

E2E End-to-end

EBI Eastbound Interface

E WBI Eastbound Westbound Interface

EFS Edge and Fog computing System

eMBB enhanced Mobile Broadband

eNB Evolved Node B

EPC Evolved Packet Core

ETSI European Telecommunications Standard

FPS Frames Per Second

GPS Global Positioning System

GS Group Specification

GPU Graphics Processing Unit

IaaS Infrastructure-as-a-Service

IoT Internet of Things

IIoT Industrial Internet of Things

IP Internet Protocol

ISG Industry Specification Group

ITU International Telecommunication Union

KPI Key Performance Indicator

LCM Life Cycle Management

LTE Long Term Evolution

M2M Machine-to-Machine

MANO Management and Orchestration

MEAO Mobile Edge Application Orchestrator

MEC Multi-access Edge Computing

MEPM-V Mobile Edge Platform Manager - NFV

MIMO Multiple-Input Multiple-Output

MIoT Massive Internet of Things

MME Mobility Management Entity

mMTC massive Machine Type Communications

mmWave millimeter Wave

ML Machine Learning

MRTK Mixed Reality Toolkit

MTC Machine-Type of Communications

multi-RAT Multiple Radio Access Technologies

NaaS Network-as-a-Service

NBI Northbound Interface

NB-IoT Narrowband IoT

NFV Network Function Virtualization

NFVI Network Function Virtualization Infrastructure

NFVO Network Function Virtualization Orchestrator

NFVO-NSO NFVO Network Service Orchestrator

NFVO-RO NFVO Resource Orchestrator

NGMN Next Generation Mobile Network

NFV-IFA NFV Infrastructure and Architecture Working Group

NFV-NS NFV Network Service

NG-RAN Next Generation Radio Access Network

NPN Non-Public Networks

NPU Network Processing Unit

NR New Radio

NS Network Service

NSD Network Service Descriptor

NSF Network Service Federation

NSO Network Service Orchestrator

NSA Non-Standalone

NTP Network Time Protocol

NTN Non-Terrestrial Networks

OAM Operation Administration and Maintenance

OCS Orchestration and Control System

ONAP Open Network Automation Platform

OPEX Operational Expenditure

OS Operating System

OVS Open Virtual Switch

PaaS Platform-as-a-Service

PDL Permissioned Distributed Ledger

P-GW Packet Data Network Gateway

PHY Physical Layer

PNF Physical Network Function

PoA Proof-of-Authority

PoP Point of Presence

PoW Proof-of-Work

RAN Radio Access Network

QoS Quality of Service

ROI Return on Investment

RO Resource Orchestrator

SA Stand Alone

SDN Software Defined Networking

SEAL Service Enabler Architecture Layer

SLA Service Level Agreement

SLAP Structured Local Address Plan

URLLC Ultra-Reliable Low Latency Communications

V2X Vehicle-to-Everything

VC Vehicular Communication

VIM Virtualized Infrastructure Manager

VM Virtual Machine

VNF Virtual Network Function

VNFM Virtual Network Function Manager

VNF-FG VNF Forwarding Graph

VR Virtual Reality

VxLAN Virtual Extensible Local Area Network

WAN Wide-area Network

WBI Westbound Interface

WiFi Wireless Fidelity

WSN Wireless Sensor Network

XR eXtended Reality

1. INTRODUCTION

1.1. 5G networks

1.1.1. Early expectations and requirements

At the beginning of 2015, the requirements, goals and challenges were laid over for the 5G technologies [1]–[3]. Departing from the 4G, the initial predictions for 5G included di er-ent use-cases. A massive adoption of Internet of Things (IoT) through the Wireless Sensor Networks (WSNs), critical control of remote devices, Vehicular Communication (VC) or Vehicle-to-Everything (V2X). With the increase of end-user bandwidth thanks to the next-generation radio use-case such as the *Broadband and Media availability Anywhere-Anytime*.

The Next Generation Mobile Network (NGMN) Aliance identified vertical industries, consumers and enteprises to be the main drivers for 5G adoption [4]. NGMN imag-ined a transformation in the business models by operators extensive support as asset providers, connectivity providers or partner service providers to vertical industries and enterprises. Namely, operators pose their infrastructure as their asset. Enterprises or in-dustries often need to deploy their own infrastructure to deliver their product to the final customers. In the case of 5G, the goal is to enable operators to provide Infrastructure-as-a-Service (IaaS), Network-as-a-Service (NaaS) or Platform-as-a-Service (PaaS). On top of that, operators are envisioned to provide vertical industries vital connectivity for mission-critical applications (in Industry 4.0, eHealth, Catastrophy management, etc.).

The 5G Infrastructure Public Private Partnership (5G-PPP) proposed several require-ments back in 2016 [5]. Similar to NGMN, in order to accelerate the service delivery for verticals, the idea is to enable service providers to access the underlying infrastructure of operators or infrastructure providers. Note that service providers may not own any infrastructure, which evolves the eco-system as a multi-tenancy and multi-service sup-port adding additional stakeholders. Service providers would be eligible to o er services through multiple mobile operators or infrastructure providers. Researchers agreed that 5G should enable highly e cient data processing and transmission. Low latency solutions have been explored that would reduce the control plane overhead by placing network functions closer to the edge of the network. This has laid foundation for the planning of the Multi-access Edge Computing. To arrive 5G-PPP envisoned the use of di erent new paradigms, such as NFV and SDN, in redefinition of the network boundaries into domains: edge, access, transport, core, services. Additionally new radio technologies should be introduced which would significantly increase the bandwidth through Multiple Radio Access Technologies (multi-RAT) and e cient interworking with Long Term Evo-lution (LTE). Frequencies above 6 GHz, in particular millimeter Wave (mmWave) fre-

quencies are intended in the 5G frontier. Despite the limited propagation and increased loss in signal penetration, the focus to mitigate the challenges was set in the use of various radio methods: massive Multiple-Input Multiple-Output (MIMO), beam steering, beam tracking, small-cells and self-backhauling.

1.1.2. 5G trends and 5G classes

The early expectations were summarized by International Telecommunication Union (ITU) back in 2015 [3]. The ITU dissected the technology, user and application trends. User and application trends are grouped as:

Very low latency and high reliability human-centric applications - where the support for instantaneous one-click behavior would be reflected in di erent application from various areas such as health, safety, entertainment, o ce environments, etc.

Very low latency and high reliability machine-centric communication - critical for designing a Machine-to-Machine (M2M) communication in a real-time scenarios.

High user density - the support for multimedia applications in very dense areas (sport events, shopping malls, festivals, etc.).

High quality at high mobility - applications should not degrade the content quality even with high mobility of the users or devices vehicles.

Application convergence - similar applications are grouped and converged together to provide and maintain the Quality of Service (QoS).

Ultra accurate positioning algorithms - expansion and improvement of the location-based services as well for the navigation services.

Internet of Things - more objects and devices are expected to be connected using 5G technology. Most of the devices contain sensors, cameras and actuators enabling optimized energy-saving usage. Especially vehicles, smart grids, agriculture and healthcare are targeted for IoT growth.

Additionally ITU proposed the technology trends that would drive the innovation of 5G technology. Besides novel radio techniques that would significantly enhance the spectral e ciency, increase the signal-to-interference ratios, and e cient use of radio resources, the use of Software Defined Networking (SDN), NFV, and Centralized Cloud RAN (C-RAN) were selected as the key technologies that would improve the operational e ciency of the network while lowering the Operational Expenditure (OPEX) and Capital Expenditure (CAPEX).

Thus, ITU defined the recommendations in several 5G use case classes. The 3rd Generation Partnership Project (3GPP) used these categorization in the development of the releases. The main classes are elaborated in the following.

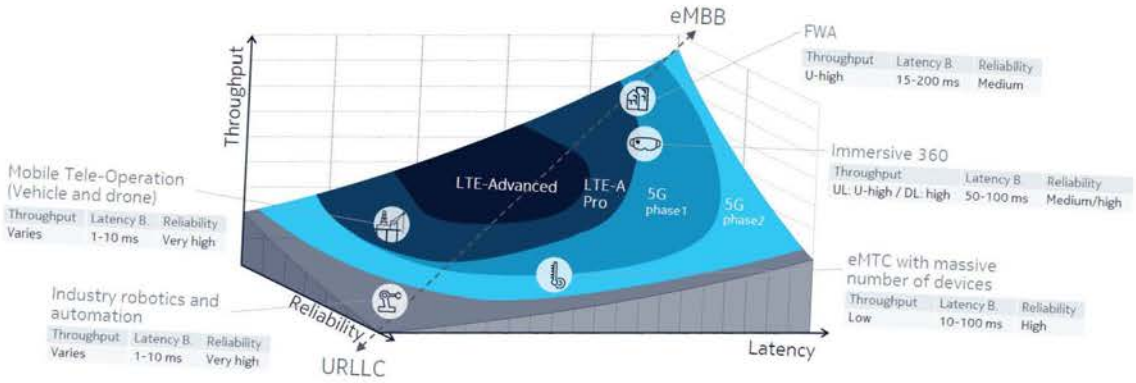


Figure 1.1: 5G categories obtained from [6]

Enchanced Mobile Broadband (eMBB)

The enhanced Mobile Broadband (eMBB) presents the natural extension of the LTE capabilities most obvious for the data ravenous users. The aim is to provide better coverage and seamless user experience to users in multiple di erent scenarios. Scenarios like big public events should benefit from high bandwith capacity, but lower mobility. In the other hand, users traveling in a fast-train should experience stable connection with high mobility and lower bandwidth. Aiming at these scenarios, users should have enhanced web access, video conferencing, usage of Augmented Virtual Reality (AR VR). The eMBB is the first defined class for 5G which was specified in Release 15 of the 3GPP specifications.

Massive Machine-Type Communications (mMTC)

The massive Machine Type Communications (mMTC) class has been partially introduced in Release 13 14, as part of LTE. This 5G class enables the use of Narrowband IoT (NB-IoT) for huge number of low-cost devices and extended coverage. The objective is to provide support for Smart Cities, Smart Homes, Smart Buildings. Additionally, the aim is to provide support for Agricultural, Patient monitoring and Tra c management systems.

Ultra-reliable and Low Latency Communications (URLLC)

The Ultra-Reliable Low Latency Communications (URLLC) is the last 5G class planned for 3GPP Release 16. The URLLC enables vertical industries to deploy services with specific low end-to-end latency, especially for mission-critical communications. Therefore this 5G class opens opportunity for developing enhanced industrial automation, drone control, medical applications (e.g., e-Health), autonomous vechicles through enhancement of V2X communications.

1.1.3. 3GPP Releases

Subsequently of the ITU recommendations for 5G classes [3], the 3GPP developed a roadmap for 5G technology roll-out. There are 3 phases, that are covered by Release 15, Release 16, and Release 17. The following is describes which technologies and features has been rolled-out in each of the release.

Release 15

The 3GPP Release 15 standard is the 5G Phase I roll-out [7]. The main focus to enable features towards the realization of the eMBB vision. Release 15 has been released in 2018, presenting the first full set of 5G standards with the main focus on the NSA 5G radio. The Non-Standalone architecture is a temporary step towards full deployment of the 5G architecture. This temporary step allows for 5G New Radio (NR) to be fully operational and attached to a legacy 4G Core Network (CN) as a 5G Access Network (AN). Beside the definition of the 5G NSA and 5G Stand Alone (SA) architectures, the Release 15 defines the whole 5G System (5GS) that includes the Next Generation Radio Access Network (NG-RAN) and 5G Core (5GC).

The definition of the 5GS is extended with description of Mission-critical communications, Machine-Type of Communications (MTC) and IoT. Use cases are defined for V2X communications and features related to Mobile Communication Systems for Railways, Virtual Reality (VR), multimedia, Operation Administration and Maintenance (OAM) improvements, etc.

Release 16

The 5G Phase II is accomplished by the 3GPP Release 16 [8] In this Release the support for URLLC 5G Class has been the most significant feature. This mainly a ects the introduction of Industrial IoT, Cellular V2X Communications, and **N-PN!** (**N-PN!**). The Release 16 directly addresses the need of the 5GS to be suitable for vertical industries by including Network Service (NS), Edge Computing, Non-Public Networks (NPN) to pave the way for automatized factories, healthcare and public safety.

To achieve this, the release focuses in several features that enhance the Radio Access Network (RAN) through the NR and NB-IoT, as well as adapting the 5GS by defining dedicated application layers such as V2X Application layer support and Service Enabler Architecture Layer (SEAL) for vertical industries.

Release 17

The 3GPP Release 17 is has been locked in March 2022 [9]. It describes the continuous evolution of the 5GS through the 2020s. The initial focus is on the network and appli-

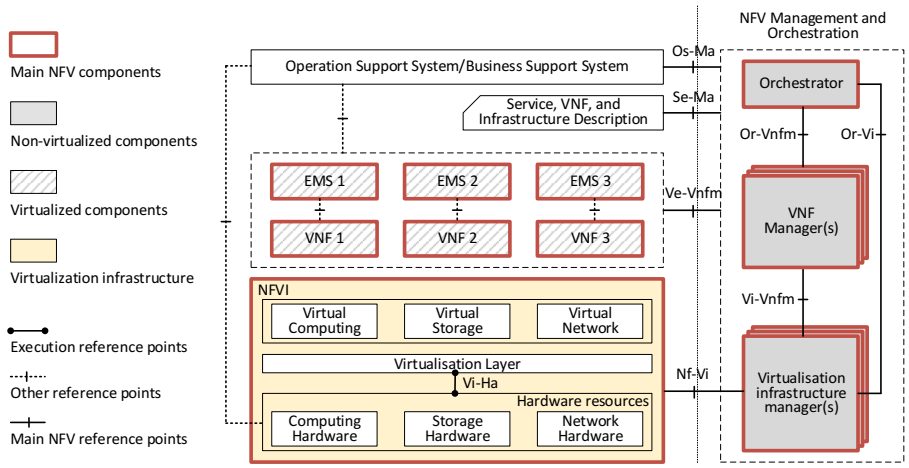


Figure 1.2: NFV architecture framework as proposed in ETSI NFV 002 [11]

cation enablement for vertical industries. According to the locked plan, a major features on the evolution of the radio is in rolling out an NR-light which has low complexity and low power for wearable devices. Additionally plans are laid out for NR operations in the higher frequencies 52.6 - 71 GHz.

Beside the RAN features, there are plans for Network Slicing enhancements and Edge Computing into 5G. These features are planned for improving the Industrial Internet of Things (IIoT), and URLLC particularly for V2X communications, drones. Integration of Open Network Automation Platform (ONAP) and 3GPP 5G management network and enhancement of Network Automation.

Release 18

The work plan of the 3GPP Release 18 is already drafted [10]. This release is labeled as the 5G Advanced, where the main topics are immediate and long-term commercial needs for eMBB and non-eMBB evolution.

The initial plans suggest an improvement of the radio technology by enhancing the MIMO downlink uplink as well as further development in the user mobility. E ort is allocated into improving the positioning localization of the users, and enhancements for the eXtended Reality (XR).

Application of Artificial Intelligence (AI) Machine Learning (ML) technology is planned for the NG-RAN and scenarios where the AI ML brings improvement of the radio performance and the energy savings. Additional focus is planned for the evolution of Non-Terrestrial Networks (NTN) combined with multi-RAT scenarios including NR and IoT.

1.2. Network Function Virtualization

With the advancement of the virtualization technologies and SDN, the NFV was born. NFV enables network nodes and services (e.g., routers, firewalls, and load balancers) to be run on top of general purpose hardware by the use of virtualization. Traditionally these services have been run on proprietary hardware. Thus, the software and hardware are now split which have traditionally been tightly integrated in telco scenarios [12]. More specifically, the general purpose hardware enables the use of Virtual Machines (VMs) or containers. The option for service providers to run their network on conventional servers rather than proprietary ones brings the cloud computing economy of scale to telco providers. This is even more important in a context with increasingly demanding and diverse services.

The NFV combined with the SDN technology has the potential to significantly lower the Operational Expenditure. Additionally, it provides service and telco providers with the capabilities to o er customers isolated dedicated network resources. A combination of isolated networking resources, coordinated over several networking services and networking domains (e.g., computing, radio access) that operate within strictly defined requirements in terms of latency, bandwidth, or scalability are referred to as *network slices*. Each vertical industry, based on its requirements, can be served by a vertical specific network slice which is guaranteeing the end-to-end QoS (e.g., latency, jitter, etc.).

The main architectural framework for NFV widely adopted by the industry is the one defined by the the ETSI Industry Specification Group (ISG) NFV. It is composed of three domains:

Virtual Network Function (VNF), running over the NFV Infrastructure (NFVI).

NFV Infrastructure (NFVI), including the diversity of physical resources and how these can be virtualized. NFVI supports the execution of the VNFs.

NFV Management and Orchestration (MANO), which covers the orchestration and life-cycle management of physical and or software resources that support the infrastructure virtualization, and the life-cycle management of VNFs. NFV Management and Orchestration focuses on all virtualization specific management tasks necessary in the NFV framework.

This NFV architectural framework identifies functional blocks and the main reference points between such blocks as shown in Figure 1.2. Some of these are already present in current deployments, whilst others might be necessary additions in order to support the virtualization process and consequent operation. The functional blocks are explained in the following paragraphs.

The *NFV Infrastructure (NFVI)* includes the hardware and virtualized resources, and the Virtualization Layer. It encompasses the HW and SW resources that create the envi-

ronment in which VNFs are deployed. The NFVI virtualizes physical computing, storage, and networking and places them into resource pools.

The *Virtualized Infrastructure Manager(s) (VIM)* is a functional block with the main responsibility for controlling and managing the NFVI compute, storage and network resources. It controls and manages the interaction of a VNF with computing, storage, and network resources under its authority, in addition to their virtualization.

The *NFV Orchestrator (NFVO)* is a functional block with two main responsibilities: the orchestration of NFVI resources across multiple VIMs, fulfilling the Resource Orchestration (RO) role, and the lifecycle management of Network Services (NS), fulfilling the Network Service Orchestration (NSO) role. The NFVO is responsible for installing and configuring new network services (NS) and virtual network function (VNF) packages, NS lifecycle management, global resource management, and validation and authorization of NFVI resource requests.

The *VNF Manager(s)* is a functional block with the main responsibility for the lifecycle management (e.g., instantiation, update, query, scaling, termination) of VNF instances.

The *Service, VNF and Infrastructure Description* is a data-set providing information regarding the VNF deployment template, VNF Forwarding Graph, service-related information, and NFV infrastructure information models. These templates descriptors are used internally within NFV Management and Orchestration. The NFV Management and Orchestration functional blocks handle information contained in the templates descriptors and may expose (subsets of) such information to applicable functional blocks, as needed.

The *Operations and Business Support Systems (OSS BSS)* provide the operational and business support systems implemented by the VNF service provider.

As it can be evinced in Figure 1.2, some of these functional blocks are virtualized while others can be deployed in any form according to the operator's requirements (i.e., non-virtualized, virtualized, bare metal, etc.).

1.3. Multi-access Edge Computing

ETSI MEC was designed as a technology to be deployed in a virtualization environment, taking advantage of all the features in terms of flexibility, scalability options and ease management provided by the ETSI Network Functions Virtualization (ETSI NFV) framework [11]. However, its development started when ETSI NFV was not mature enough and ETSI MEC and NFV have been evolving in parallel for some time without a tight coordination among them. The ETSI MEC started an initiative to clarify all the open points that need to be addressed in order to run the ETSI MEC framework in an ETSI NFV environment without hassle, finding several open issues that require e ort to be solved [13].

The main architectural framework for edge computing widely adopted by industry is

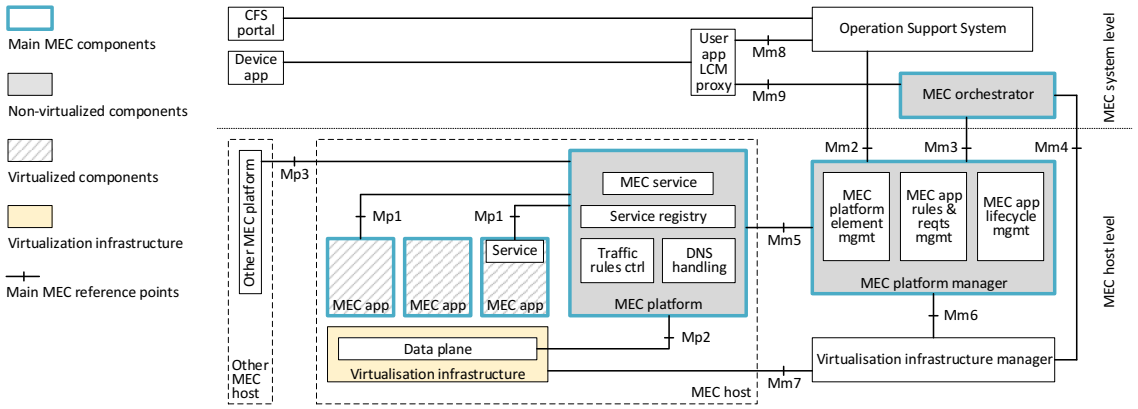


Figure 1.3: MEC architecture framework as proposed in ETSI GR MEC 003 [14]

the one defined by the ETSI ISG Multi-access Edge Computing (MEC) [14] and shown in Figure 1.3. The MEC framework is composed of two domains: *i)* the MEC Host level and *ii)* the MEC System level.

Similarly to the ETSI NFV case, the domains contain functional blocks that are either virtualized or that can be deployed in any form (i.e., non-virtualized, virtualized, bare metal, etc.). The main MEC component at host level is the *MEC Host*, which can be seen as an edge data center, and it consists of the *i)* Virtualization infrastructure, *ii)* the MEC Platform (MEP) and *iii)* the MEC Applications (MEC Apps).

The *virtualization infrastructure* provides compute, storage, and network resources for the MEC Applications. The virtualization infrastructure includes a data plane for routing the tra c among applications, services, local external networks, and MEC Platform. The configuration of the data plane is done via the Mp2 reference point and it is managed by the tra c rules controller in the MEC Platform.

The *MEC Platform* o ers an environment (i.e., service registry and DNS handling) where the MEC Applications can discover, advertise, consume and o er MEC Services.

MEC Applications run on top of the virtualization infrastructure provided by the MEC Host, and can interact with the MEC Platform to consume and publish MEC Services via the Mp1 reference point. How these MEC Applications retrieve the data to be published as a service is left unspecified by current ETSI MEC specifications. Finally, the MEC Host level encompasses two management components: *i)* the Virtualization Infrastructure Manager (VIM) and *ii)* the MEC Platform Manager. While the *Virtualization Infrastructure Manager* is in charge of controlling and managing the virtualized infrastructure, the *MEC Platform Manager* is in charge of controlling, managing, and configuring the MEC Platform for what concerns MEC Applications and MEC Services authentication and authorization.

While the MEC Host level operates on the single edge data center, the MEC System level operates across multiple MEC Host levels and its main components are *i)* the MEC Orchestrator, *ii)* the Operation Support System (OSS) and *iii)* the User application life-

cycle management proxy.

The *MEC Orchestrator* is the main component for the MEC System level management and is in charge of maintaining an overall view of the MEC System based on deployed MEC Hosts, available resources and available MEC Services as well as on-boarding, instantiating, and terminating the applications. The *Operation Support System (OSS)* refers to the OSS of an operator. It receives requests via the Customer Facing Service (CFS) portal and from Device applications for instantiation or termination of applications, and decides on the granting of these requests. The Device applications can request life-cycle operations (on-boarding, instantiation, termination, modification mobility) via the User application Life-cycle Management proxy. Granted requests are forwarded to the MEC Orchestrator for further processing, which in turn may contact the MEC Platform Manager via the Mm3 reference point to manage the application life-cycle and enforce the application rules.

Ultimately, ETSI MEC defines a set of MEC Services such as the Radio Network Information Service, the Location Service and the Bandwidth Management Service. The Radio Network Information Service (RNIS) [15] provides radio network-related information, such as up-to-date radio network conditions, measurement and statistics information related to the user plane, and information related to users served by the radio nodes. While the original RNIS service was designed for 3GPP networks, a new service is being defined to cover also WiFi networks [16]. The Location Service [17] provides location-related information about the users (e.g., all of them or a subset) currently served by the radio nodes. The location information can be geo-location, Cell ID, etc. Finally, the Bandwidth Management Service [18] allows the allocation of bandwidth to certain tra c routed to and from MEC Applications and the prioritization of certain tra c. Additional services can be then defined upon necessity based on the same MEC framework.

1.4. Thesis overview

This thesis tackles the integration of MEC and NFV in di erent heterogeneous and dynamic scenarios for 5G networks. Hence the fist part of the thesis elaborates the joint integration of MEC in NFV while the second part is elaborating how a network services (that may contain MEC application) can be deployed in heterogeneous multi-domain static or dynamic environments through a federation process. The final part of the thesis elaborates how the federation feature may provide extended benefit to an arbitrary service provider.

The study of MEC in NFV integration starts with a background and state of the art overview of how this integration would bring value to multiple vertical industries. Taking the Cloud and Edge robotics as an exemplary use case, an Edge robotics scenario was deployed at the University where we (together with the co-authors) have demonstrated the benefits that MEC is bringing to the Edge robotics. Departing from these insights, we have analyzed and proposed how a general MEC in NFV integration may happen by

classifying and proposing solution to bridge the integration gaps (identified in [13]). The proposed solution of how to solve most of the identified gaps is later showcased for the Edge robotics scenario. We have detailed a step to step tutorial on how a Edge Robotics MEC application would be deployed in a MECinNFV platform. Before concluding the first part, we compare the insights with the state-of-the-art solutions.

The second part of the thesis focuses mainly on federation - the process of orchestrating network services or resources across multiple administrative domains. First, the federation concept is defined along with the state-of-the-art usages. Note that the author has been actively and directly involved in exploring, designing and defining the federation concept in several research works and projects. Some of the project federation designs are presented as examples of federation in static and dynamic environments. Next, the thesis describes the federation process in a static environment. In this case administrative domains are known to each other, and they strike pre-established agreements that define their interactions in case of federation of services or resources. For a static environment, the author has been coordinating the deployment of an e-Health scenario as part of 5G-Transformer project. Additionally the design, implementation and results of the eHealth use-case are presented.

Following the static environment, the realization of the federation feature in dynamic environments is detailed, mainly through the application of Blockchain technology, as a trust enabling technology between multiple unknown domains. First, we provide the insight of what is Blockchain and how Blockchain is used in vertical industries. Then the challenges for the federation in dynamic environments are laid out. Having in mind these challenges, we propose a solution of applying Blockchain for service federation in NFV environments. To justify this solution, we present an experimental scenario of showcasing federation procedure using multiple Blockchain platforms. The results of the federation execution time and resource profiling are presented to provide better idea of the applicability of the solution. Additionally, the solution is applied in the Edge robotics scenario, where a robot is seamlessly reconnecting to a federated Access Point (AP) without interruption of the service. The new federated AP is deployed in a previously unknown domain using the Blockchain federation.

The federation part is finalized by applying Reinforcement learning algorithms to showcase how an operator would increase the profit by using the federation feature. The first scenario considers static service pricing while an extended scenario is tackling a real scenario with dynamic service pricing (obtained from a cloud provider). The solution is compared with additional reinforcement learning algorithms such as Deep Q-learning.

Finally in the last section of the thesis the conclusions and final remarks are laid over.

2. MEC IN NFV

This chapter describes the benefits of the integration of MEC in NFV. To grasp the horizontal integration, we first focus on how di erent vertical industries would benefit from the MEC in NFV integration with specific focus on Edge robotics. For better insights of how MEC is applied use case of Edge robotics, we describe an Edge Robotics system and an experimental scenario that we deployed at the University. Performed experiments show how the robotics systems can benefit from the use of MEC technology. Additionally, we performed analysis of how the Edge robotics use case scenario can be adapted and used in a MEC in NFV scenario. We performed a step-by-step analysis of what can be the potential integration issues and what is the potential solution. At the end of the chapter, we conclude the work by providing a comparison with existing works and future research directions.

2.1. Vertical industries that benefit from MEC in NFV integration

2.1.1. Intelligent video acceleration

Radio information can be dynamically used to adapt a video downstream application according to the estimation of available throughput in the radio downlink interface. MEC enables this *intelligent video acceleration* by locating a radio analytics application in the radio access network (RAN), which monitors the radio downlink interface and sends the monitoring data to a video server. The video server uses the information to prevent TCP congestion and allow the application-level coding to adapt to the radio downlink capacity. This way of acceleration boosts up the users' quality of experience by adapting the video coding so it uses the full capacity of the radio links. Besides, multiple applications may use the radio monitoring data in parallel, justifying the need of enabling it as a MEC service. The deployment of MEC in NFV is clearly useful in this kind of video streaming scenario, by facilitating the deployment of on-demand video caches closer to the highly loaded regions. This enables saving network resources and avoiding some tra c congestion events that might occur when video content is not located closer to the video consumers.

2.1.2. Video stream analysis

Today's video surveillance systems (such as the ones deployed in public areas, parking areas, highways, private properties, etc.) make an increasing use of intelligent video recognition systems (e.g., of faces, license plates, etc). A classical approach is to locally identify patterns and send the video data to a cloud system where it can be stored and a

more powerful analysis can be performed. Sending all this raw video data to a centralized cloud server is expensive and ine cient, even more considering new 4K or even 8K resolution cameras which are starting to be commercialized. By using a MEC approach, where the video stream analysis can be performed locally, on the vicinity of the cameras, the cost can be reduced by performing a local computation (close to the access network), where small pieces of information are extracted and processed from each video upstream. Further on, the MEC application can decide to upload the processed information to a cloud monitoring service or act immediately (e.g., in case of emergency). Here also the use of NFV technology proves to be helpful, as video processing applications can also be deployed and scaled on-demand (e.g., if an emergency situation arises where more computing power is needed to perform image recognition) thanks to NFV.

Note that there is another video-related use case, called collaborative multi-bitrate video caching, which can also benefit from the use of MEC and NFV.

2.1.3. Augmented reality

Various types of events, such as visits to museums, galleries, music shows or sport events, could benefit from using event-tailored mobile applications capable of o ering live additional content related to what the mobile device is pointing at (e.g., using the device's camera). The actual recording of the mobile phone's camera can be enriched with additional content, which is normally referred to as *augmented reality (AR)*. In order to operate properly, the augmented reality application processes the camera input, the precise device location and sight direction (using the accelerometer sensor from the mobile device). With this input, the applications generates the additional information, which is displayed in real-time over the content that the user is seeing. This requires, in order to meet a satisfactory performance, data processing at a very high rate with low latency. Therefore, implementing AR as a MEC application is clearly a feasible and optimal approach, as MEC can provide the exchange of fast data rate, high computing power with low-latency in localized areas. Actually, the localized nature of the augmented reality application (e.g., at a museum, sport event) allows the use of additional MEC services such as localization. Equivalently to what happens with video streaming applications, the demand of AR in terms of computing resources very much depends of the event itself and the number of users, which makes NFV a very suitable technology to allow for dynamic scaling and better use of shared resources.

2.1.4. Connected vehicles

With the advent of autonomous vehicles, the need for network connectivity for vehicles is very fast increasing through the years. Forecasts predict the drastic increase of data flows from and towards the sensors or processors of the connected vehicles. Cars will communicate and exchange data among themselves and also with the road infrastructure,

Table 2.1: Advantages of applying MEC and NFV for each use case and application

Benefits Use-cases	Intelligent video acceleration	Video stream analysis	Augmented reality	Connected vehicles	Collaborative MEC	IoT gateway	Cloud Robotics
Radio-link quality	X	X			X		X
Low-latency	X	X	X	X	X		X
Local Computation	X	X	X		X	X	X
Scaling		X			X	X	
Mobility				X	X		X
Real-time analysis	X	X	X	X	X	X	X

so drivers will be aware of the status of the roads, road accidents, etc. Additionally, more value-added services like infotainment, car finder, parking location, etc., would be also provided. MEC is a key tool here, as it can be used to distribute large portions of the services closer to the access network, as well as to enable fast processing of information coming from multiple connected vehicles or sensors and deliver the necessary information to the vehicles or road-assisting units (e.g., in case of an accident). MEC applications would reside on servers deployed in small-cell sites or Long-Term Evolution(LTE) 5G base stations close to the roads. As vehicles move along the roads, the use of "follow-me" features will allow MEC applications to maintain Quality of Service (QoS) and connectivity to the connected vehicles while migrating through the hosting systems in the direction of the movement. It is clear that this "follow-me" features require NFV technology to become feasible, as network virtualization mechanisms support migration of functions and applications within the infrastructure.

2.1.5. Collaborative MEC

With the placement of MEC at the RAN part of an operator's network, opens an opportunity for a collaborative framework referred as collaborative MEC [19]. The realization of the framework enables end-users to split small tasks of the applications and perform them in the upper-layer, preserving the latency and accuracy requirements of the applications. The aim of the collaborative MEC is to have a horizontal collaboration between the intermediate layer of MEC nodes and a vertical collaboration between the lower end-user devices and upper cloud nodes. Through joint orchestration the tasks execution can be decided dynamically based on the network state, execution requirement or devices power consumption. The scalability and mobility of the collaborative MEC framework can be further increased as well as accomplishing infrastructure agnostic realization can be achieved through the integration of MEC in NFV.

2.1.6. IoT gateway

The continuous evolution of the Internet of Things (IoT) demands upgrades to the gateway devices connecting the di erent "things". Gateways need to scale to support more devices while keeping a level of QoS and security. Di erent protocol families and radio technologies are currently used for IoT, which requires the role of IoT gateway: a low-latency aggregation point that can support various types of protocols, radio-technologies

and real-time processing of monitoring data. MEC is a considered a suitable approach to meet the requirements of various IoT setups, by benefiting from low-latency local processing of monitoring data and enabling the application of real-time analytics. Again, the integration of MEC with NFV in this scenario would allow a faster, cheaper and more agile adaptation by dynamically deploying IoT gateways to support the di erent IoT devices present on a given deployment.

2.1.7. Cloud robotics

Cloud Robotics leverages and integrates Cloud computing, Cloud storage, and other Internet technologies, into industrial and commercial robotics applications. Cloud technologies enable robot systems to be endowed with powerful capability by leveraging the powerful computation, storage, and communication resources available in the Cloud. Consequently, it is possible to build lightweight, low cost, and smarter robots by placing an intelligent brain in the Cloud which o ers a converged infrastructure that can be also used to share services and information from various robots or agents. To that end, Cloud infrastructure for robots shall support the sharing of data between various robots and agents connected to the Cloud, such as images, maps, robot outcomes, trajectories, and control policies [20]. Although robots can benefit from various advantages of Cloud computing, this presents several limitations when applied to the Cloud Robotics field [21]. Cloud facilities traditionally reside far away from the robots and while the Cloud providers can ensure certain performance in their infrastructure, very little can be ensured in the network between the robots and the Cloud, especially when multiple Internet providers are involved. As a result, Cloud-based applications can su er from high-latency or unpredictable jitter in the network. This is exacerbated for applications relying on real-time data from the robot and the surrounding environment (e.g., Automatic Guided Vehicles). Given the challenges of assuring the network performance at infrastructure level, the applications are hence required to adapt their operations depending on the network conditions. However, accessing information related to the network (e.g., on the radio channel) is equally challenging when multiple domains are involved. Moreover, network operators are not allowed to publish such sensitive data on the Cloud for regulatory and privacy reasons.

The use of MEC clearly helps in reducing latency and using context information to help in controlling the robots. In this context, integration with NFV is also considered helpful, as di erent robot control mechanisms can be deployed on demand upon necessity. This also involves, for example, the setup of replicated functions to improve resiliency and reliability, which are critical in this kind of use case. Summarizing, an integrated MEC and NFV system o ers a simplified management (provided by the operator) to cloud robotics which, in turn, can significantly reduce deployment costs thanks to the hardware pooling and the virtualization of white boxes.

Based on the works [22] [23] [24] [25] [26] [27] [28], we generated the Table 2.1 that

summarizes the main identified advantages we envision through the application of MEC in NFV for each of the analyzed use cases and applications.

2.2. Edge robotics - MEC deployment

Over the last few years, Edge computing has arisen as a promising paradigm in the telecommunication industry in response of the ever increasing tra c demands and stringent requirements expected in forthcoming 5G networks [29]. The Edge computing vision foresees the deployment of computing capabilities directly in the operator's access network, which would enable the provisioning of applications and network services closer to the users compared to the traditional Cloud computing. As a result, operators can offer low latency services to the users whilst simultaneously o oading their core network. Moreover, Edge computing aims at exploiting the context information available locally in the access by making it available to the applications through services. By doing so, applications can subscribe to those services and consume the context information to optimize their functionalities.

Driven by these needs and opportunities, ETSI created the MEC Industry Specification Group (ISG) with the goal of standardizing the Edge computing ecosystem. Such ecosystem aims at achieving convergence of IT and telecommunications networking to enable new vertical business segments and services for consumers and enterprise customers. The evolution of Cloud robotics towards Edge robotics lies among these services.

Hence, we tried to emulate an Edge robotics use case through placing the brain of the robots in the Edge rather than in the Cloud. This way is possible (*i*) to ensure low latency between the robots and their brains due to the shorter distance, and (*ii*) to consume context information on the access network in order to adapt the robotics operation to the context, including the communication links status. It is worth highlighting that in case of wireless access, network performance can only be ensured within certain limits and transmission failures are still likely to happen. Consequently, applications can benefit from the context information about the network to adapt to such cases. In the next section, the aims are to showcase the benefits for robotics applications of adapting their operations to context information available locally at the Edge. Later, a real-life experimentation is performed in a small-scale environment where the movement of one remotely-controlled mobile robot is adapted in accordance with the wireless information available at the Edge.

2.3. Edge robotics system overview

This section describes the Edge robotics system we used for experimentation at 5TONIC [30] laboratory. The system is divided in two subsystems: the robotic subsystem, and the MEC subsystem. The system components are shown on Fig. 2.1.

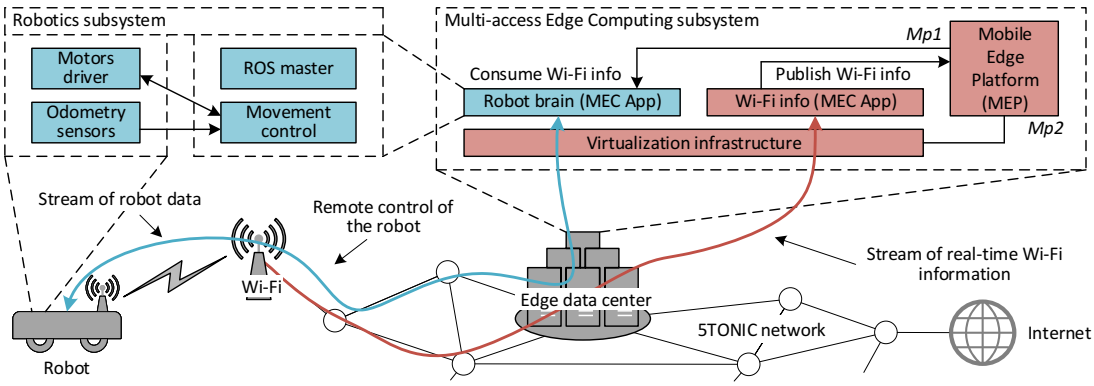


Figure 2.1: Edge robotics system used for experimentation

2.3.1. Robotics subsystem

Today's robotics systems require the deployment of dedicated robotics hardware and software along with access to Cloud infrastructure. As mentioned in [20], the robots maintain their independent operating capabilities and rely on the Cloud for accomplishing complex tasks, such as big data analytics, collective learning, crowd-sourcing, etc. Following these principles, [31] proposes a Cloud-based framework wherein industrial robots are remotely configured so as to enable an ubiquitous manufacturing environment. An example of Cloud-based industrial manufacturing is presented in [32], where the planning of the robotics tasks is distributed and executed across a high-speed wide-area network. In [33], the proposed simultaneous localization and mapping solution uses the Cloud infrastructure to o oad the heavy computational tasks and large data sets from the robots. In [34], the Cloud infrastructure is used by the robotics system to consume context information (e.g., cognitive Industrial Internet of Things) as a mean to improve the production e ciency. Finally, [35] proposes a distributed cooperative communication and link prediction framework to cope with the network issues in Cloud Robotics. However, such framework requires pre-knowledge of the link quality in the case of robot mobility.

A streamlined provisioning of robotics software components is seen as a necessity to cope with the rapidly emerging of new robotic services. To that end, new open-source platforms are arising to simplify the software development for di erent robotic hardware. The most widespread framework nowadays is Robot Operating System (ROS)[2], which provides a meta-operating environment for developing and testing multi-vendor robotics software. In ROS, each software component is called ROS node. Moreover, ROS provides a publish-subscribe messaging framework via a specific node, namely ROS master. By connecting to the ROS master, ROS nodes can register and locate each other. Once registered, nodes can exchange data via configurable topics in a peer-to-peer fashion.

In our set-up (see Fig. 2.1), the robotics subsystem is implemented as various ROS components distributed across the robot itself and the Edge data center. The robot is

2

equipped with motored-wheels and odometry sensors[3] (e.g., motor encoders). The ROS components running on the robots are essentially drivers that are in charge of (*i*) reading data from the sensors (e.g., odometry) and send them to the brain, and (*ii*) executing the driving instructions received from the brain. The robot brain acts as a ROS master and it is also in charge of driving the robot based on the available information. The communication between the robot and the brain crosses over a Wi-Fi link and the wired network connecting to the Edge data center. In accordance with the Edge computing concept, a wireless information service is available locally at the Edge data center. This is consumed by the ROS node controlling the movement to adapt the robot driving. The details regarding the wireless information service are reported in the following paragraphs.

2.3.2. MEC subsystem

As described in [14], MEC enables the implementation of mobile edge applications as software-only entities that run on top of a virtualization infrastructure, which is located at or close to the network edge. One realization of these applications is the robot brain described above. The main MEC component is the Edge data center, which acts as *mobile edge host* and it consists of the following entities: (*i*) a Virtualization infrastructure, (*ii*) a Mobile Edge Platform (MEP), and (*iii*) Mobile Edge Applications (MEC Apps). The virtualization infrastructure provides compute, storage, and network resources for the MEC applications. The virtualization infrastructure includes a data plane for routing the traffic among applications, services, local external networks, and mobile edge platform. The configuration of the data plane is done via the Mp2 reference point. The mobile edge platform o ers an environment where the MEC applications can discover, advertise, consume and o er mobile edge services. Finally, MEC applications run on top of the virtualization infrastructure provided by the mobile edge host, and can interact with the mobile edge platform to consume and publish mobile edge services via the Mp1 reference point. How these MEC applications retrieve the data to be published as a service is left unspecified by current ETSI MEC specifications.

ETSI MEC defines a set of exemplary services. For example, the Radio Network Information service (RNIS) [15] provides radio network-related information, such as up-to-date radio network conditions, measurement and statistics information related to the user plane, and information related to users served by the radio nodes. While the original RNIS service was designed for 3GPP networks, a new service is being defined to cover also Wi-Fi networks [16]. Another example is given by the location service [17] which provides location-related information about the users (e.g., all of them or a subset) currently served by the radio nodes. The location information can be geolocation, Cell ID, etc. Finally, the Bandwidth Manager service [18] allows the allocation of bandwidth to certain tra c routed to and from MEC applications and the prioritization of certain tra c. Additional services can be then defined upon necessity based on the same MEC

[3] Odometry is the use of data from motion sensors to estimate changes in position over time.

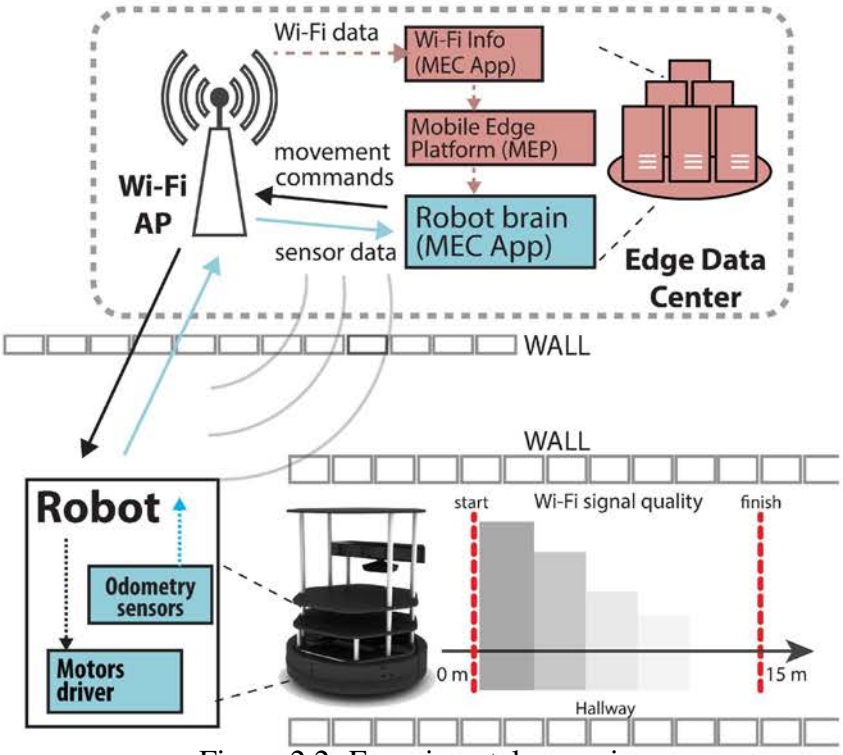


Figure 2.2: Experimental scenario

framework.

While the MEC framework can serve multiple access technologies (e.g., 4G, 5G, Wi-Fi, etc.), for the sake of our experimentation we focus on the Wi-Fi access. To that end, we developed a MEC service providing Wi-Fi information regarding the clients connected to the system. This is achieved by creating a MEC application (shown in red as Wi-Fi Info (MEC App) in Fig. 2.1) that gathers Wi-Fi information from the radio nodes and exposes it via the mobile edge platform to other MEC applications (e.g., the Robot brain (MEC App) in Fig. 2.1). The Wi-Fi network information service hence provides for each connected client (e.g., the robot) data on the signal level, transmission and reception bit rates, number of retransmission and packet losses at data link level, and number of successfully transmitted received bytes and packets. Moreover, link layer configuration is also provided: wireless channel, beacon interval, preamble and slot time (i.e., short long), QoS support and authorization authentication status. The information is then published in JSON format and can be accessed by MEC applications (e.g., the robot brain) through HTTP requests.

2.3.3. Experimental methodology

This section describes the experimental set-up and the experiments we performed to explore the benefits and performance of the Edge robotics system described in Sec. 2.3. Particularly, Sec. 2.3.3 describes the experiments performed, while Sec. 2.3.3 evaluates the relevant factors a ecting the robot performance while being controlled from the Edge

over a Wi-Fi link.

Experiments description

To evaluate the Edge robotics scenario, we have built an experimental environment in the 5TONIC [30] laboratory. In such environment, we have deployed all the components shown in Fig. 2.1. The goal of the experimental test-bed is to show how the Edge controlled robotics paradigm improves current Cloud robotics techniques towards the industry demands of high speed and high precision in robotics applications. To that end, we have designed the experiment shown in Fig. 2.2 and described in the following.

For the mobile robot, we used the ROS-compatible Kobuki[4] robotics platform. The mobile robot maximum speed is 0.75 m s, while its minimum speed is 0.1 m s. The sampling frequency for reading the odometry sensor data from the robot's wheels is 16.6 Hz (i.e., odometry sensor data is refreshed every 60 ms). When driving at full-speed (0.75 m s), the robot covers a distance of 4.5 cm in 60 ms. This results in a precision of 4.5 cm in the robot driving at full-speed since odometry sensor data can not be updated with a frequency higher than 16.6 Hz. In the case of minimum speed (0.1 ms), the precision is 0.6 cm. It is worth highlighting that the sampling frequency value is an hardware parameter of our robot. Di erent robotics platforms may o er higher sampling frequency and consequently better precision.

The mobile robot is controlled in a closed-loop by the Robot brain application. The closed-loop starts with the Robot brain (running in the Edge data center) sending movement commands to the Motors drivers (running on the robot) using ROS messages, published in a specific topic devoted to movement commands. The movement command consists of a tuple (speed, distance), where the *speed* parameter presents the velocity that the robot should maintain while driving, and the *distance* parameter represents the distance that should be reached upon receiving the movement command. Therefore, the *distance* parameter presents the movement granularity instead of the final driving destination. Upon receiving a movement parameter through the wireless link, the Motors driver initiates the movement in the robot's wheels. The movement is uninterrupted for a length equal to the received *distance* parameter with constant velocity equal to the received *speed* parameter. The loop is then closed by the robot continuously sending-back the odometry sensor data to the Robot brain application in the Edge data center. The brain analyzes and combines the odometry data together with the Wi-Fi information provided via a MEC service by the Wi-Fi MEC application. The result of the brain algorithm is a new (speed, distance) tuple, which will serve as input to the next turn of the closed-loop.

The experiment runs are performed in a closed and straight hallway (3m wide, 30m long) at 5TONIC laboratory. Each run consists of the Robot brain driving the robot on a straight line for 15 m as shown in Fig. 2.2. The starting position of the robot is placed in

[4]http: kobuki.yujinrobot.com

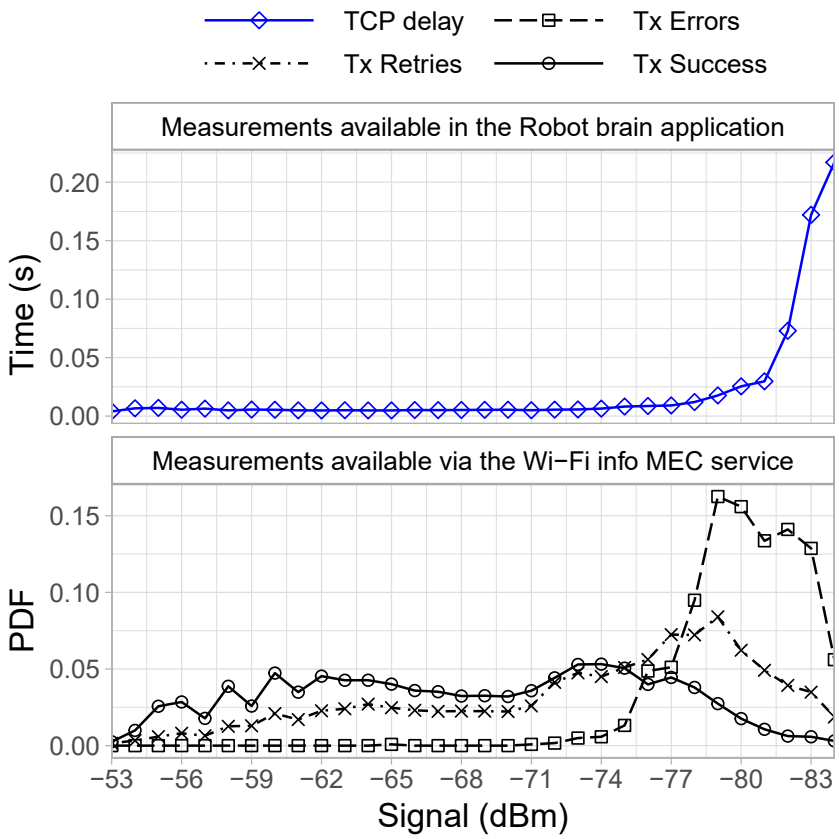


Figure 2.3: Signal and delay characterization

the middle of the hallway approximately 7 meters away from the Wi-Fi AP having a thin o ce wall (approximately 15 cm) separating the two. Then, the robot accelerates from the starting position to the target velocity (e.g., min, max, etc.) and it drives in accordance with the closed-loop mechanism. After having traveled for 15 m, the robot stops. During the driving, an additional thicker wall (approximately 25 30 cm) separates the robot from the Wi-Fi AP. At the end of the driving, the robot is approximately 22 m away from the Wi-Fi AP.

Delay and Signal characterization

This section aims at characterizing how the Wi-Fi signal quality impacts the delay in controlling the robot as perceived by the Robot brain. Indeed, the publish-consume mechanism for exchanging data between the Robot brain and the ROS components is based on TCP. This means that any transmission failure occurring on the Wi-Fi channel (Layer 2) will trigger a retransmission at TCP level (Layer 4), thus introducing an undesired delay in the closed-loop mechanism. Indeed, additional delays in the delivering of the odometry sensor data result in longer reaction times in the Robot brain. Similarly, additional delays in the delivering of the movement instructions degrade the smoothness and precision of the driving. Such characterization is therefore necessary in order to adapt the closed-loop

to also consider the Wi-Fi signal. To that end, we performed 10 experiment runs driving the robot at minimum speed (0.1 m s) and 10 experiment runs at maximum speed (0.75 m s). All the edge robotics system components are synchronized and share the same time reference for accurate measurements. Throughout the duration of the experiment, we recorded in the Robot brain the Wi-Fi information obtained via the Wi-Fi information MEC service, while on the robot itself we measured the delay in receiving the movement instructions.

The obtained data from both experiments is analyzed and aggregated to generate the results presented in Fig. 2.3. Note that the results shown here are specific for our test-bed, and therefore can only be used as a particular realization that we use later to validate and evaluate the benefits of Edge robotics. In overall, Fig. 2.3 characterizes the quality of the Wi-Fi channel covering our experimental area. Regarding the measurements available via the Wi-Fi information MEC service, the *MEC: Tx Success* line shows the probability density function (PDF) of all the downstream frames successfully transmitted (from the access point to the robot) over the measured signal level in dBm. Similarly, *MEC: Tx Retries* shows the probability density function of the downstream frames retransmissions. It is worth highlighting that Wi-Fi employs an automatic retransmission mechanism in case of packet transmission error, where frames are retransmitted up to 7 times[5], and if none of the retransmissions succeeds, a frame loss occurs. *MEC: Tx Error* shows the PDF of the failed transmissions.

It can be seen that for high dBm signal values (i.e., good signal level) the probability of successful frame transmission maintains a di erence proportional with respect to the number of retransmitted frames. This is due to the fact that frame retransmissions constantly occur in Wi-Fi networks because of its best-e ort design principle. For lower signal strengths (below -71 dBm), the probability of having a failed transmission increases. Such probability becomes drastically higher than the probability of successful transmission at signal levels lower than -77 dBm (it is actually evident that below -80 dBm it is very hard to have a successful transmission). TCP delay measurements (shown in the top graph of Fig. 2.3) confirm this, with values as high as hundreds of milliseconds.

2.3.4. Adaptive speed control algorithm

Based on the results provided in the previous section (Sec. 2.3.3), next we present the design of a control algorithm which is able to adapt the robot driving speed based on the Wi-Fi information service. The aim of the algorithm is to obtain a displacement accuracy similar to the one obtained while driving at the lowest speed, while reaching the target destination faster. Through this algorithm we showcase the benefits of consuming context information for controlling the robot, nonetheless, we acknowledge that more advanced and optimal algorithms than the one proposed in this section can be eventually designed.

[5]The maximum amount of retransmissions is configurable. 7 is a common default value.

The design approach followed for the proposed algorithm is tailored to the experimental evaluation performed in Sec. 2.3.5.

```
1: procedure ComputeRobotSpeed
2:     info    GetCurrentWiFiInfo();
3:     bu er    bu  er.removeOldestWiFiInfo();
4:     bu er    bu  er.add(info);
5:     signalLevel    bu  er.average();
6:     if signalLevel >   71 dBm then speed    0.75;
7:     else if signalLevel <   81 dBm then speed    0.1;
8:     else speed    (signalLevel   81 dBm) 10 dBm    0.1;
```

Algorithm 1: Adaptive control speed algorithm

During the experiments described in Sec. 2.3.3, we collected the information on the Wi-Fi signal every 10 ms. We observed that the Wi-Fi signal level presents significant oscillations in case of averaging it over a short time window (e.g., 50 ms). That is, two subsequent average measurements may report considerably di erent Wi-Fi signal levels. On the contrary, if we take a longer time window (e.g., 500 ms), the oscillations between subsequent average measurements are substantially reduced and the Wi-Fi signal varies in a smoother way. Based on this finding, the control algorithm will use the Wi-Fi signal level obtained by averaging it over a fixed time frame. Given the robot's speed bound between 0.1 m s and 0.75 m s, a time frame of 500 ms is considered to be a reasonable value. The computed Wi-Fi signal is then combined with the robot's odometry sensor data for adapting the robot's speed.

Alg. 1 shows the pseudo-code of the control algorithm. The Robot brain, in real-time, extracts the current signal level from the Wi-Fi MEC information service, stores it in a circular bu er and computes the moving average of the Wi-Fi signal level. For each movement command, the adaptive speed and the adaptive distance are re-calculated. In Sec. 2.3.3 we observed that packet retransmissions and failures start increasing for signal values below -71 dBm, hitting their maximum between -79 and -81 dBm. Based on this observation, the control algorithm adapts the driving robot's speed to the maximum (0.75 m s) for an average Wi-Fi signal level higher than -71 dBm. On the opposite end, the minimum robot speed (0.1 m s) is selected for an average Wi-Fi signal level equal or lower than -81 dBm. Between -71 dBm and -81 dBm, the control algorithm linearly adapts the robot's speed to the Wi-Fi signal level (e.g., 0.425 m s with -76 dBm).

2.3.5. Experimental evaluation

This section evaluates the adaptive speed control algorithm proposed in Sec. 2.3.4 and compares it with scenarios not making use of any context information. The following three scenarios are evaluated: (*i*) the robot drives at minimum speed (0.1 m s), (*ii*) the

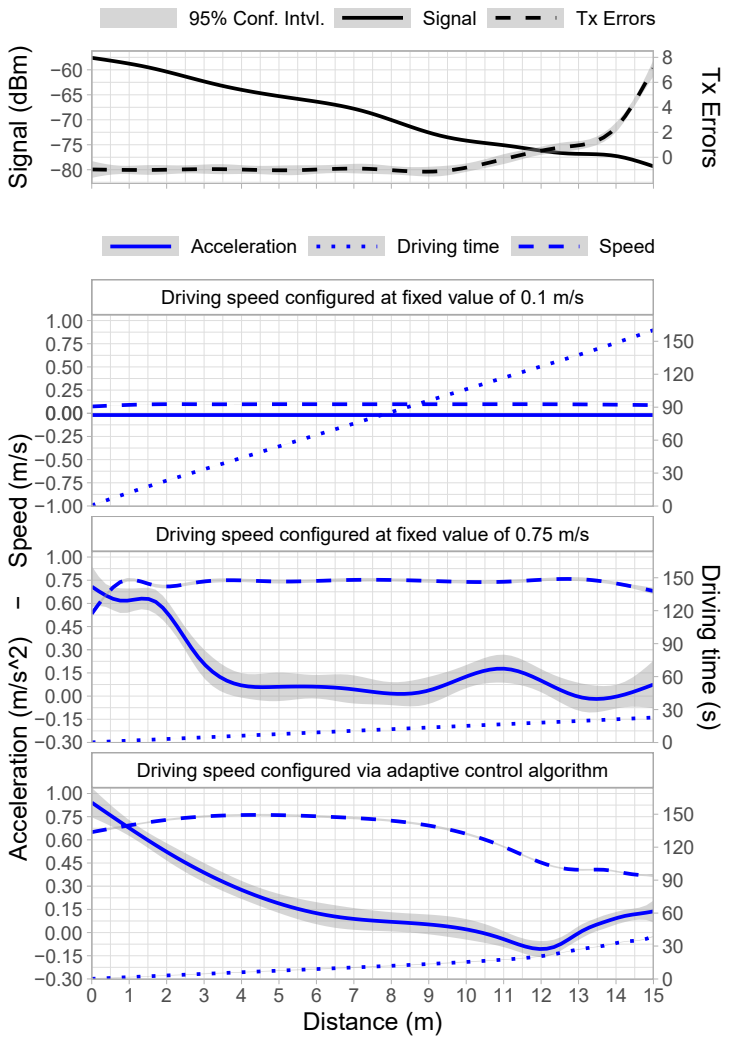


Figure 2.4: Speed, acceleration, and driving time

robot drives at maximum speed (0.75 m s), and (*iii*) the robot uses our control algorithm to drive at adaptive speed.

Following the experimental methodology described in Sec. 2.3.3, we performed 10 experiment runs for each scenario (minimum speed, maximum speed, adaptive speed). In addition to the Wi-Fi information recorded in the Robot brain, we record the odometry sensor data directly in the robot itself. This is because the data from the odometry sensors is not timestamped, and sending it over the Wi-Fi channel would not be suitable for measuring the speed and acceleration experienced by the robot (due to risk of transmission failures over Wi-Fi).

The measured data is aggregated and analyzed to produce the results on Fig. 2.4. The figure has four di erent graphs. On each graph the x-axis is the distance traveled during the experiment, from the start (0 m) to the end (15 m). The first subgraph from the top presents the Wi-Fi signal level (y-axis on the left) and the transmission errors over the robot driving path (y-axis on the right). As it can be noticed, there is a significant decay on the Wi-Fi signal quality in the last 5 meters of the driving path reflected by an

exponential increase of the transmission errors. The remaining three graphs of Fig. 2.4 present – for each evaluated scenario – the speed, the acceleration, and the driving time as measured via the odometry sensor data. Despite the acceleration and the speed having di erent units (m s and m s^2, respectively), they share the same y-axis on the left since they present the same range of values. The y-axis on the right represents the elapsed driving time since the start of the experiment run.

In the minimum speed experiment, the robot speed is set constant to 0.1 m s from the start to the end. Similarly, the acceleration presents a constant value in the order of few cm/s^2. Driving such a low speed results in a smooth run that is not a ected by the degradation of the Wi-Fi channel in the last segment of the path, since the slowness of the movement allows for more time to recover from possible transmission errors and retransmissions. As a drawback, the robot requires 160 seconds to complete each experiment run. On contrary, the maximum speed experiment is the one requiring less time (27 seconds).The impact of the decreasing Wi-Fi signal quality can be seen in the acceleration curve (notably in the last 5 meters of the path) where the acceleration fluctuates due to increased packet delay or delayed reaction, resulting in a stop-drive e ect of frequent braking and spurring acceleration to full-speed. E ect of the stop-drive behavior, the driving direction is deviating from the straight driving path.

The bottom graph shows the motion behavior in the case of using the proposed adaptive speed control algorithm. A first observation is that the acceleration and deceleration in this case is smoother. At start, the robot accelerates to full-speed, since the received signal level is in the safe zone above -71 dBm. After crossing the -71 dBm threshold, the robot speed is linearly reduced following the decrease of the Wi-Fi signal strength, reaching the end of the path driving at minimum velocity. Regarding the driving time, the robot reaches the finish line 10 seconds later than in the maximum speed experiment. Nonetheless, it is still 120 seconds faster than the minimum speed experiment while performing a smooth ride. As concluding remarks, the results show that there is a trade-o between speed and smooth movement of the robot. By adapting the velocity of the robot with information on the quality of the Wi-Fi channel, the robot is able to move with maximum speed where the Wi-Fi signal channel is good and smoothly lowers the speed in the areas of weak Wi-Fi signal coverage, thus canceling any stop-drive e ect.

2.3.6. Remarks on Edge robotics in MEC

One of the key di erentiating features of Edge computing is the possibility for applications running at the Edge to consume context information about the network. This can be used to optimize the robotics systems operations in ways otherwise impossible in the Cloud. Following the Edge computing concept, we have designed an Edge robotics system blending together the Robot Operating System (ROS) – which o ers a common development framework for robotics applications – and the ETSI MEC architecture, which defines a common framework for Edge computing. An experimental environment is de-

ployed in the 5TONIC laboratory where one mobile robot is employed. We first perform a set of experiments to characterize the relation between the robot control delay and the Wi-Fi signal strength. The resulting characterization has been used as a baseline for designing, implementing and experimentally evaluating a control algorithm which consumes context information about the Wi-Fi signal and adapts the robot's speed for a smoother driving. Our experimental results show that adapting the robot's speed based on the Wi-Fi signal provided by the MEC information service can e ectively produce a smoother driving at high speeds. This improvement allows the robot to operate faster compared to the case of not consuming any context information.

2.4. Integration at architectural level

Both ETSI NFV and ETSI MEC have common characteristics that can be drawn from Section 1.2 and Section 1.3. In an attempt to align and harmonize the two ISGs, ETSI MEC published a group report, namely MEC 017 [13], with goal of studying the deployment of MEC in an NFV environment. It is worth highlighting that this report represents just the first attempt from the industry to analyze the problem of integrating MEC and NFV technologies, and does not even aim at proposing a solution or set of solutions, but rather start identifying the main issues that would deserve additional work. The rationale of this exercise being done in ETSI MEC (and not in ETSI NFV) lies in the fact that the main focus of ETSI MEC is on the MEC platform MEC services and not on the virtualization infrastructure per se, which is instead the main focus of ETSI NFV. Therefore, ETSI MEC proposed a mapping between the MEC components and the NFV framework resulting in the MEC reference architecture illustrated in Figure 2.5, where the NFV components reference points are highlighted in red while the MEC components reference points are highlighted in blue. Moreover, Figure 2.5 highlights the components that are virtualized.

The assumptions for deploying MEC in NFV are:

1. The MEC platform is deployed as a Virtual Network Function (VNF). For that purpose, the procedures defined by ETSI NFV are used;
2. The MEC applications behave as VNFs for the rest of ETSI NFV Management and Orchestration (MANO) components. This allows re-use of ETSI NFV MANO functionality;
3. The virtualization infrastructure is deployed as a Network Function Virtualization Infrastructure (NFVI) and its virtualized resources are managed by the ETSI NFV defined Virtualized Infrastructure Manager (VIM) (part of the ETSI NFV MANO).

In the MEC architecture, the MEC host contains an instance of a virtualization infrastructure and runs an instance of the MEC platform. When integrating the MEC architecture into NFV, the concept of MEC host becomes obsolete and it is replaced by

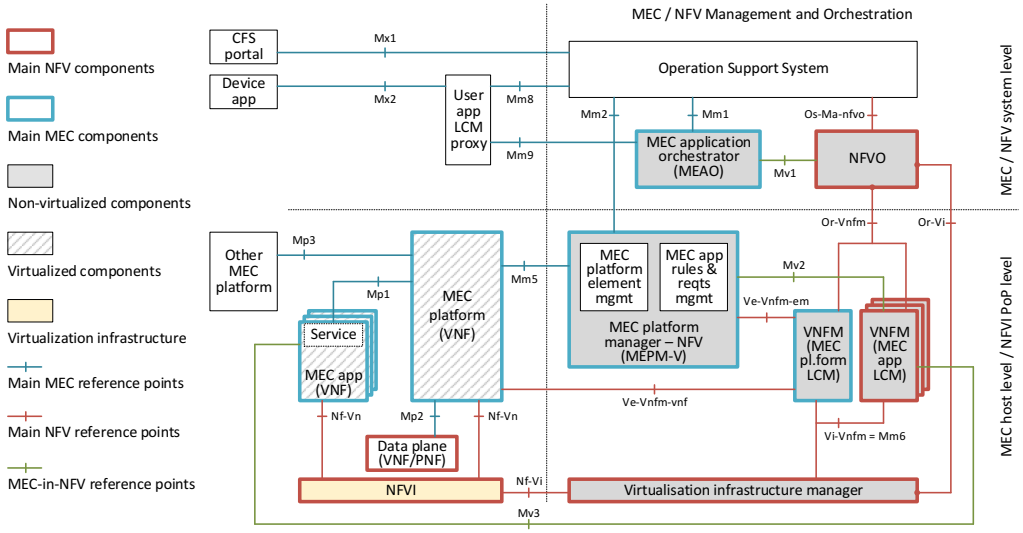


Figure 2.5: MEC reference architecture in a NFV environment as proposed in ETSI GR MEC 017 [13]

the concepts of NFVI-Point of Presence (PoP) (i.e., a data center) and zone (i.e., a set of co-located and well-connected physical resources within a NFVI-PoP). Moreover, the MEC platform manager as defined in [14] is transformed into a *Mobile Edge Platform Manager - NFV (MEPM-V)* that delegates the Life Cycle Management (LCM) part to one or more Virtual Network Function Managers (VNFMs). Similarly, the MEC orchestrator is transformed into a *Mobile Edge Application Orchestrator (MEAO)* that uses the Network Function Virtualization Orchestrator (NFVO) for resource orchestration, and for orchestration of the set of MEC application VNFs as one or more NFV NSs.

While most of the reference points defined in ETSI NFV and ETSI MEC do not require any changes, the ETSI MEC Mm3 reference point (see Section 1.3) needs to be extended and become *Mm3** to cater for the split between MEPM-V and VNFM for properly managing the MEC applications LCM. Likewise, the following new reference points are introduced between elements of the ETSI MEC and ETSI NFV architectures to support the management of MEC application VNFs:

Mv1: It connects the MEAO and the NFVO to allow the MEAO to invoke operations towards the NFVO to manage MEC application VNFs;

Mv2: It connects the VNFM that performs the LCM of the MEC application VNFs with the MEPM-V to allow LCM related notifications to be exchanged between these entities;

Mv3: It connects the VNFM with the MEC application VNF instance, to allow the exchange of messages, e.g., related to MEC application LCM or initial deployment-specific configuration.

Finally, when MEC is deployed in a NFV environment, the data plane can be realized in two di erent ways. In the first option, the data plane is realized as a VNF which is

integrated in the network service (NS) containing the MEC application VNFs. By doing so, Mp2 is kept as a MEC-internal reference point and it is agnostic to the way MEC is deployed. In the second option, the ETSI NFV MANO functionalities for configuring the data plane are used by the ETSI MEC components. That is, the MEC platform does not control the network configuration directly via Mp2 but rather requests it to the MEPM-V, which in turn requests the MEAO. When receiving such a request, the MEAO contacts the NFVO to update the network configuration accordingly. Therefore, there is no dedicated VNF implementing the data plane as in the first option, making the Mp2 reference point unnecessary.

2.4.1. Integration issues

In the previous section we went through how MEC and NFV can be integrated at architectural level, starting from what the ETSI MEC ISG reported in [13]. In addition to the proposed integrated architecture, [13] identifies 14 key issues of di erent nature, proposing solutions for some, while leaving others for future study.

Next, we provide a summarized view of these issues, classifying them into three categories: *architectural*, *workflow* and *communication* issues. This initial classification is intended to serve as baseline for a detailed analysis of the issues and gaps that exist today for the integration of MEC and NFV, which we perform in Section 2.5.3 together with our solution proposal.

Architectural (MEC as NFV) issues

The parallel development of ETSI MEC and NFV has yield to two separated architectures with multiple duplicated functions played by MEC and NFV components. Such duplication is due to the need in ETSI MEC to orchestrate applications, which were developed without considering the parallel development of NFV technology. Moreover, although functionally the ETSI MEC and ETSI NFV architectures look very similar, they are based on slightly di erent assumptions which prevent a straightforward integration at architectural level. The main issue is related to the instantiation process, as the approaches followed by NFV and MEC di er. Considering the integration of the two architectures some responsibilities are moved from one component (i.e., MEAO) to another (i.e., NFVO). This creates the following issues:

MEC NFV descriptors partial compatibility;

Mapping of MEC VNFs to network services;

Connecting a MEC platform network service with a MEC application network service;

Mapping of the concept of MEC host to NFV.

Starting with the descriptors compatibility, NFV uses a descriptor defining the information necessary to instantiate a VNF (VNF Descriptor, VNFD) while MEC uses one defining the information needed to instantiate an Application (Application Descriptor, AppD). Although the VNFD and the AppD present some similarities, the information in the AppD is not enough to instantiate a MEC Application in the form of a VNF. The information contained in an AppD focuses on the requirements of the infrastructure and the service availability for the MEC Application, while the information included in a VNFD is not tightly related to the infrastructure setup (e.g., no strict location constraints), but more explanatory of the interconnection between internal and external components of a VNF. The mapping of the identifier data fields from an AppD to a VNFD is straightforward process, however, aligning network requirements and (part of) life-cycle management procedures from an AppD to a VNFD is not that simple due to some mismatches. For example, the AppD can only model a single virtual compute resource for a MEC Application, whereas the VNFD can define templates for multiple virtual compute resources per VNF to support scalability. Considering the above issues, we believe that the AppD and the VNFD should be integrated, having both descriptors the semantics to express each other behavior, enabling a loss-less translation between both descriptors.

However, the descriptor integration is not just a mere field matching exercise since the translation of an AppD to a VNFD implies a modification of the step-by-step on-boarding process of a MEC Application VNFD, which must consider the MEC specific orchestration entities (such as the MEAO). In Section 2.5.3 an exemplary mapping of AppD to VNFD is presented.

After an AppD is mapped translated to a VNFD and then packaged into a VNF package, a ME app should be part of a network service in order to be instantiated in an NFV-MEC environment. This is the source of the second of the issues identified before (mapping of MEC VNFs to network services), which raises the concern of how to create a MEC Application VNF as part of a network service (NS). Considering the processing of an already on-boarded VNFD at instantiation time, in order to make the MEC application part of a network service that can be instantiated from scratch or that is already instantiated, the Mobile Edge Application Orchestrator (MEAO) and the NFVO have to be perfectly coordinated.

The main function of the MEAO is to orchestrate MEC application VNFs as part of a network service. In that sense, the MEAO takes care of assuring the presence of a MEC platform for each MEC application VNF that is instantiated. The MEC platform is assumed to be a VNF in the NFV-MEC environment [13].

Since every MEC application requires of a MEC platform, and both are VNFs, they can be orchestrated as part of an NS. The MEAO is responsible for assuring the correct instantiation of a MEC Platform VNF to any other MEC Application VNF. The connection between MEC Platform VNF and any MEC Application VNF can be established as a single composite Network service or as two individual Network services that are con-

catenated. The main difference between the composite solution and the concatenation approach is that in the composition case for each instantiation of a new MEC Application VNF, the general composite Network Service Descriptor (NSD) needs to be modified and the new MEC application VNF instance should be instantiated through a request for modification of already active network service instance. Whereas in the concatenation case, the MEC Application VNF is instantiated as an independent Network service and connected directly to the Service Access Point (SAP). In the concatenation case, the life-cycle management is more complex with a benefit that each instantiated VNF is non-dependent of the status of the rest inter-connected VNF instances.

In the initial MEC architecture, the MEC platform is instantiated on a specific host. In the NFV-MEC environment, the concept of a MEC host does not exist. This yields to the fourth of the issues identified at the beginning of this section. By eliminating the concept of MEC host, the issue is how to arrange nearby placement of the MEC platform and the ME applications as part of a single network Service. Since both the MEC platform and MEC applications are deployed as VNFs, determining the placement of the MEC VNFs over a virtualized infrastructure is highly challenging.

Workflow issues

We now focus on the potential problems of the procedures such as: on-boarding, instantiation, modification (e.g., mobility) and termination of the MEC Applications in the NFV-MEC environment. As for the architectural issues, the problems arise with the presence of both the NFV and MEC orchestrators: NFVO and MEAO. Both orchestrators should assess an on-boarding VNF package of a MEC Application. In an NFV MANO environment, the NFVO is in charge of the NSDs and the VNF packages on-boarding procedures using the NFV Interfaces and Architecture (NFV-IFA) 013 specifications [36]. In an NFV-MEC environment, each on-boarded VNF package that contains MEC-related files needs to be processed by the MEAO as well. The main question is: which module should take care of the VNF package on-boarding? the NFVO or the MEAO?

A clean instantiation workflow demands instantiation of a NS that contains a MEC Application VNF at the start and applying the external MEC-specific features (such as modifying traffic rules, DNS rules, required provided MEC services, etc.) in the final phase. Ideally the roles are equally split: initially the instantiation of NSs is orchestrated by the NFVO, and then it is handed over to the MEAO. If the NFVO handles the on-boarding of VNF packages, it sends a request to the MEAO to analyze the on-boarding VNF package and generate NSDs for instantiation of a MEC application VNF as part of a NS. It might require significant changes on the interfaces and in the internal workflow of the NFVO and MEAO. However, if the MEAO is the first to process on-boarded VNF packages and to generate NSDs, it orchestrates the whole instantiation procedure by requesting orchestration of NFV-related features (e.g., the NS instantiation) to the NFVO and applying the MEC-related features at the final phase. This would require applying

additional features in the MEAO and less interventions in the NFVO. On the other hand, the termination has a reversed execution sequence from the instantiation procedure. First, the MEC-related features are removed and then the MEC Application VNF being part of a NS is terminated. To sum up, all procedures (on-boarding, instantiation and termination) present a di erent set of issues, that may occur depending on which module has the main orchestration role, the NFVO or the MEAO.

The mobility feature has existed only in the MEC environment. The absence of a MEC host paradigm and non-existent VNF mobility in a NFV-only environment open a range of issues that need to be addressed. Enabling mobility features for VNFs is a completely open question.

Communication issues

Here we exemplify the potential issues on the newly introduced interfaces in the NFV-MEC architecture. With the convergence of both MEC and NFV, the interfaces that connect some of the modules may produce errors. We mainly focus on the set of interfaces:

Mv1 - interface between the NFVO and MEAO;

Mv2 - interface between the VNFM (Life-Cycle Management for MEC Apps) and MEPM-V;

Mv3 - interface between the VNFM (LCM for MEC Apps) and MEC App VNFs;

Mp2 - interface between the MEC Platform VNF and the Data plane.

The Mv1 interface is used by the MEAO to communicate with the NFVO for the deployment of a MEC application VNF as a NS along with the MEC Platform VNF. The Mv1 interface is used in all procedures with MEC application VNFs (on-boarding, instantiation, termination and modification). In the case of the NFVO being the main orchestrator, the Mv1 interface should support the demanded additional features where (*i*) the NFVO asks the MEAO to analyze an on-boarded VNF package for MEC-realated features; (*ii*) the NFVO requests the MEAO to provide continuous feedback for the duration of the instantiation of a MEC application or MEC platform VNF as part of a NS.

If the MEAO has the main orchestrator role, the Mv1 interface would be an extension to the NFV-IFA 013 [36]. The MEAO generates NSDs that would fit the concatenation or composition of a NS that contains the MEC application VNF coupled with a MEC platform VNF. Upon generation of all NSDs, the MEAO requests on-boarding, instantiation termination and modification of an NS on the Mv1 interface. The NFVO in that case is transparent to the presence of the MEC-related features and treats all requests the same way as for regular NSs.

The Mv2 interface is placed between MEPM-V and VNFM modules. Both modules are derived originally from the MEPM of the MEC architecture, where the MEPM-V is

responsible for the MEC-specific part and the VNFM responsible for Life-Cycle Management (LCM) of the MEC Application VNFs. In order the MEPM-V to be aware of the LCM operations performed by the VNFM, the Mv2 interface allows to send direct requests, such as (*i*) requests for performance monitoring and fault information related to a specific MEC Application VNF instance; (*ii*) subscribe for notifications or query for life-cycle information of a MEC Application VNF. The NFV IFA 008 [37] specification can be used for implementation of the Mv2 interface, but not all the specified operations are allowed. For example, invoking a instantiation of a MEC application VNF towards the VNFM can produce an error. Thus the implementation of the Mv2 interface must be synchronized to meet the requirements while taking care not to introduce errors.

While the Mv2 interface is used for extracting LCM information, the Mv3 interface is used to execute the LCM operations towards the deployed MEC Application VNFs and to allow MEC Applications to communicate directly to the VNFM. A "day-zero configuration" feature is enabled through the Mv3, which is a provision procedure that allows the VNFM to provide pre-boot configuration parameters of the MEC Application VNF instance via the VIM. It remains an open question the usage of Mv3 in an opposite direction or whether the MEC application VNF instance should invoke healing or scaling operations towards the VNFM.

The Mp2 interface, the interface between MEC Platform VNF and data plane, is the most complex for implementation. A possible implementation is via an indirect workflow: requests for data plane operations are sent from the MEC platform VNF towards the MEPM-V (via Mm5) which, acting as a proxy, will redirect them (via Mm3*) to the MEAO and the NFVO (via Mv1). Setting up this workflow demands invasive adjustments in each module and its internal workflows. Another option is to setup the data plane as an independent VNF. In that way, the implementation of the Mp2 interface would be dynamic and set up as a virtual link between two VNFs. A deeper analysis of these options is provided in the next section.

2.5. Nuts and bolts: NFV MEC for edge robotics

While the previous section analyzed the integration of NFV and MEC, describing the main issues of such integration and outlining some of the possible solutions, this section takes a deep dive into the issues and solutions by considering a specific use case and dwelling on all the relevant considerations.

2.5.1. Major role: MEAO vs. NFVO

As pointed out in MEC-in-NFV [13], either the NFVO (Option 1, in the following) or the MEAO (Option 2, in the following) can play the role of the master module that oversees the procedures of on-boarding, instantiation, life-cycle management, termination and

migration of MEC Applications. Therefore, two di erent approaches for the NFV MEC integration can be followed, each approach with its pros and cons. To compare both approaches, we next elaborate on an analysis based on three key points or metrics: MEC awareness, Application Program Interface (API) call requests and states. As conclusion of this analysis, we propose the MEAO to be in charge of the orchestration procedures.

MEC Awareness: The first metric considered is the need to modify current implemented behaviours to account for the presence of MEC in the platform. In both approaches, the NFVO is aware of the existence of the MEAO through the Mv1 interface that interconnects them. However, Options 1 and 2 di er in the degree of MEC awareness needed:

> Option 1: In the case that the NFVO is the master of all procedures for Network Services and MEC Applications, the NFVO has di erent degrees of MEC awareness. If the user (the MEC Application developer) uploads the MEC package, the NFVO has to be aware that it is a MEC package and redirect the package to the MEAO, so it can transform the MEC package into a VNF package. Then the MEAO would on-board the newly created VNF package. Even in the case that the MEC developer performs itself the transformation from MEC to VNF package, it has to obtain MEC related information via the NFVO. That means that the NFVO must be aware of MEC related requests and redirect them on the Mv1 interface towards the MEAO.
>
> Option 2: In the case that the MEAO is in charge of performing all procedures, the NFVO is oblivious to the existence of the MEC entities and constructs, such as MEC Packages, MEC Platform or MEC Applications. MEC packages are directly on-boarded to the MEAO where the transformation is performed according to the availability of MEC Platform VNFs. The generated VNF package is later on-boarded on the NFVO by the MEAO and included in an already existing NS that contains a chosen MEC Platform VNF by a simple NS update call towards the NFVO.

To sum up, in the case of NFVO having the master role (Option 1), the NFVO itself needs to be upgraded to satisfy the MEC related procedures, whereas in the case of MEAO being in charge (Option 2), the existence of the MEC related procedures is transparent to the NFVO. Note that the MEAO needs to be extended to understand the NFV procedures in any case. As conclusion, Option 2 is preferred since it reduces the amount of modifications to the current infrastructure.

Number of API calls: The second metric used in the comparison is the *number of API calls* that needs to be triggered on each MEC related procedure (summarized in Table 2.2). Considering the di erent possible master roles, we can observe that for Option 1 (NFVO as master):

> On-boarding procedure: there are a total of 6 API calls.

Table 2.2: MEAO vs. NFVO: Number of API calls per procedure

Procedure	MEAO major role	NFVO major role
On-boarding	3 calls per MEC package	6 calls per MEC package
Instantiation	10 calls per MEC App VNF	11 calls per MEC App VNF
LCM	2 calls per MEC App VNF	2 calls per MEC App VNF
Termination	7 calls per MEC App VNF	8 calls per MEC App VNF

Instantiation process: there are a total of 11 API calls.

Termination process: there are a total of 8 API calls.

While for Option 2 (MEAO as master):

On-boarding procedure: there are a total of 3 API calls.

Instantiation process: there area a total of 10 API calls.

Termination process: there are a total of 7 API calls.

Regarding the rest of life-cycle management operations, there is no di erence in terms of number of API calls.

Summarizing, the case of MEAO playing the major orchestration role involves significantly less API calls compared to the NFVO being the one playing the major role, therefore Option 2 is preferred.

Number of states: The last metric analyzed is the *number of states* that each module needs to be aware when performing all procedures. In general, the total number of states that the whole system needs to hold for each MEC Application remains the same in both cases, but the level of complexity needed in both approaches is not the same. For Option 1, when the NFVO plays the master role, all state is held by the NFVO, preventing the MEAO of controlling the state of the MEC Applications and Platform. Therefore, to implement Option 1, both the MEAO and the NFVO need to be refactored to transfer the MEAO capability of holding the states for each MEC Application or MEC Platform to the NFVO. For Option 2 (MEAO is in charge), the NFVO does not need to hold any MEC related states other than the usual tracking of the status related to NSs and VNFs.

From our point of view, although the total number of states remains the same, we argue that the NFVO performing the central orchestration role would mean that the workflow of each NFVO implementation should be significantly changed, as well as, the workflow of the MEAO due to transferring the functionality of recording states for each MEC Application Service Platform to the NFVO. As conclusion, Option 2 is also preferred for this metric.

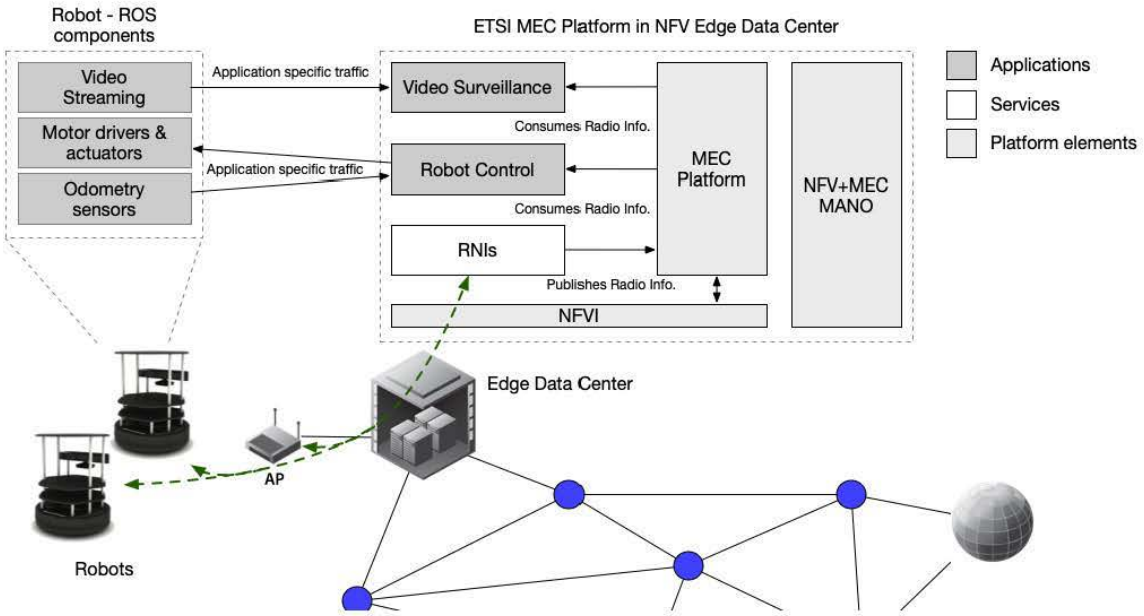


Figure 2.6: Edge Robotics scenario

As a consequence of the three mentioned aspects, our view is that the joint-implementation should have the MEAO as the major orchestration entity, being the first module to receive requests from the OSS and orchestrate all NFV-MEC procedures, so Option 2 is preferred. Although this approach places the orchestration burden mainly in the MEAO, it demands little updates in the NFVO module and have more natural evolution of joining the MEC in NFV as a future implementation.

Stepping forward with the approach of having the MEAO as major orchestration entity, a detailed explanation of the procedures for Edge Robotics MEC Application is presented in the next sections. First, the description of the Edge Robotics scenario is presented, followed by the detailed description of all the procedures to establish a Robot Control MEC Application on top of MEC in an NFV platform.

2.5.2. Edge robotics: Scenario setup

The Edge Robotics scenario envisages a fleet of mobile robots remotely controlled and coordinated to perform di erent tasks in a multi-access indoor or outdoor environment. An example of this environment would be a shopping mall, where robots are used to move goods and perform cleaning tasks. A similar scenario setup is used in [38]. Figure 5.14 illustrates the main components and services of the Edge Robotics platform. In this scenario, the robots act only as sensors and actuators, that is, all the robotics intelligence controlling the navigation, driving speed, driving direction, video surveillance, angle of sight, etc., is executed in a MEC server located at the Edge of the network. Therefore, the robots embed only minimal applications for (*i*) executing driving commands (e.g., drive forward, turn right, increase speed, etc.,) received from the MEC application in charge of the control of the robot, and (*ii*) sending application specific data towards the MEC

applications, such as encoding and streaming video from the on-board camera to a Video Surveillance application. An important component, and key feature of the MEC platform used on this application, is the radio information service used to gain knowledge of the status of the radio connectivity. This information may be used for closed control-loop between the robots forming the fleet and the MEC applications. The performance of the overall system depends on the quality of the radio channel which can be a limiting factor in terms of maximum number of robots, cameras, or video stream quality in certain areas.

To react upon radio connectivity variance, the Edge Robotics use case leverages on the Radio-Network Information (RNI) service provided by the MEC platform which offers context information about the robots connectivity. Regardless of the radio access technology (e.g., LTE or Wi-Fi), the RNI service monitors the connectivity and reports real-time information about the signal strength, MAC layer parameters, packet loss, etc., for each robot (UE or STA). This context information is then consumed by two exemplary robotics MEC applications for this specific use case: (*i*) Robot Control application and (*ii*) Video Surveillance application. The Robot Control application implements all the logic for coordination, navigation, and control of the movement of the robots in the physical environment. The precision of the control is determined by the quality of the radio signal, therefore the RNI can be used to optimize the control of the robots accounting for variance in the signal level. Clearly, this application takes benefit of the close proximity and on the contextual (RNI) information available at the MEC platform. The Video Surveillance application controls the on-board cameras installed on the robots, collects the video streams, and cooperates with the Robot Control application on navigating the robots for better Video Surveillance of certain areas of interest. Using the real-time information from the RNI service, the Video Surveillance application can reduce or increase the up streaming quality (e.g., changing encoding, frame-rate, etc.,) improving the video quality in order to meet certain application specific constraints. In [28] the potential requirements for robots used as automated guided vehicles (AGV) are presented.

In addition, this use case makes explicitly visible the new business roles that the use of MEC technology enables. Let's take a shopping mall as a potential deployment scenario (to be further described next). In this case, the owner of the infrastructure (e.g., the micro-datacenter at the Edge) might be a Shopping Mall owner, or even a third party deploying infrastructure and managing it for the Shopping Mall owners. The Shopping Mall requires a service, which can be delivered through robots, such as cleaning, Video Surveillance or transport of goods. This service is provided by a third party robotics application provider, which delivers its applications in the infrastructure located at the Shopping Mall. To enable this future scenario, the underlying infrastructure of the access network and the network's edge should be virtualized. Enabling multiple providers to leverage a common infrastructure to deploy services. A virtualized underlying infrastructure is ready to adopt the MEC in NFV solution. The co-existence of multiple virtualized applications allows the users to easily extend the exemplary scenario by using di erent services or deploying di erent applications on the MEC platform.

2.5.3. Edge Robotics in NFV-MEC: issues and solutions

In this section we explore: 1) the initial setup of the NFV-MEC environment; the procedures of 2) on-boarding; 3) instantiation; 4) life-cycle management (LCM); and 5) termination of an exemplary MEC Application. From the Edge Robotics scenario, the observed procedures apply to both the Robot Control and Video Surveillance MEC Applications.

We first present the initial setup of the NFV-MEC environment on top of which the MEC application VNFs are deployed.

The main focus is on the step-by-step deployment procedures that are presented in the subsequent sections, explaining in detail the workflow of message exchange for each procedure. Note that the sequencing of the message exchange in the workflows is continuous and globally determined, meaning that the on-boarding procedure starts with number 1), but the termination procedure starts with sequence number 30).

Initial setup

Throughout the description of the workflow procedures we refer to the Edge Robotics scenario. In that sense, the following assumptions are made regarding the scenario setup shown in Figure 5.14 and deployed in a shopping mall:

The robots have already been configured and ready to be used.

Two embedded software features are up and running on the robots: (*i*) motor drivers & actuators, and (*ii*) odometry sensors, together with the communication protocol stack, as explained in section 2.5.2. Both embedded features can be developed in various ways (e.g., as Robot Operating System (ROS) applications, firmware software, etc.) and they are out of scope for this work.

The access network (e.g., Wifi or LTE) network has already been configured to cover the whole shopping mall area or at least the operating area of the robots. The access network is connected to the underlying virtualized infrastructure (e.g., small data center). It is assumed that the requirements for both the underlying infrastructure and the access network are satisfied, such as the low latency (e.g., lower than 40ms, as defined in Fig. 2.7) and enough computational power to track robots' location and issue movement commands in real-time. The choice of the access network technology mainly depends on the network coverage in the shopping mall as well as the robots' capability to access the network.

It is assumed that the virtualized infrastructure contains the NFV-MEC environment and it is able to run MEC application VNF instances on top of it. The virtualized infrastructure can be owned by an arbitrary operator or the shopping mall itself.

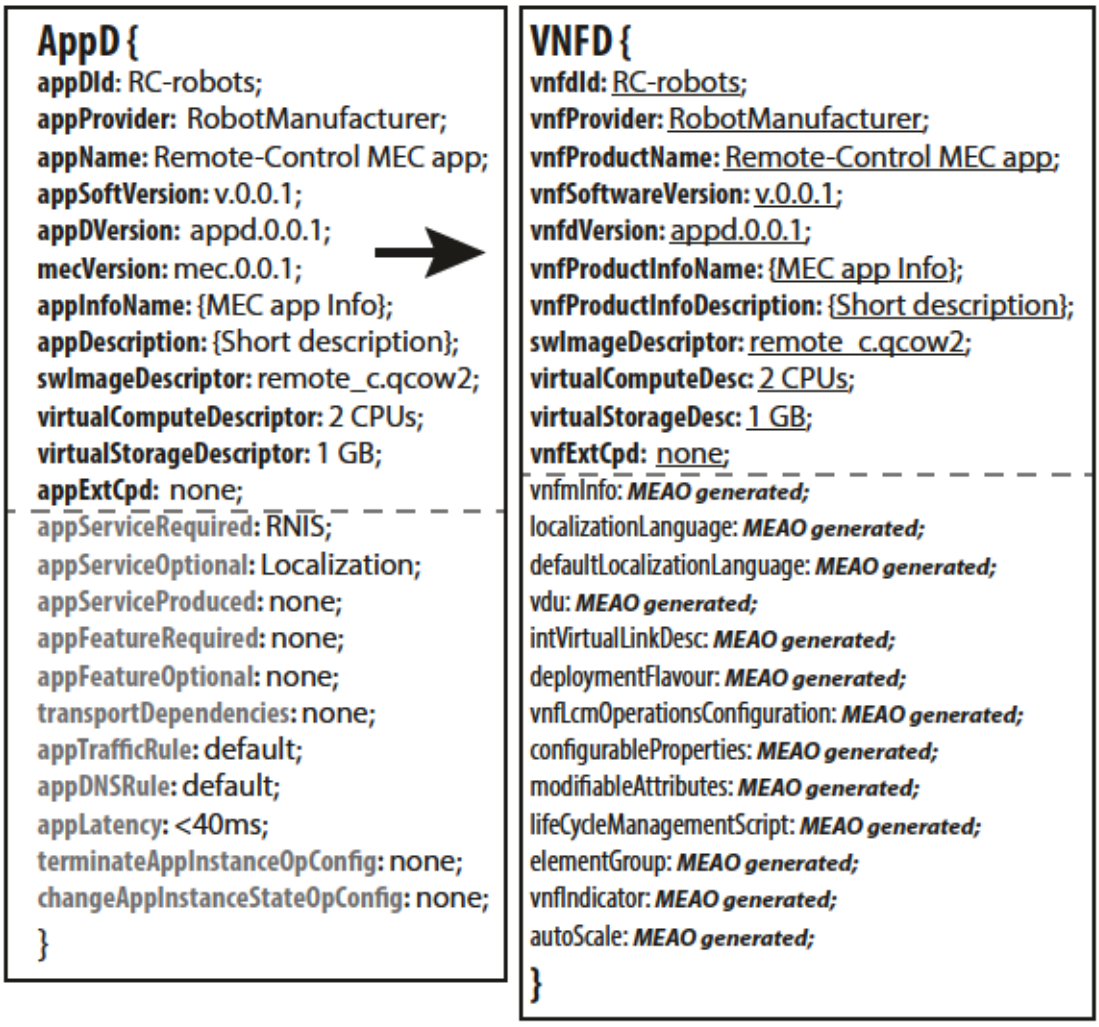


Figure 2.7: Mapping of AppD to VNFD.

- The MEC applications that offer Robot Control or Video Surveillance services can be provided by a robotic manufacturer (that provide the robots) or a third-party entity.

In this scenario, the following stakeholders relationships are assumed: the operator is providing the virtualized infrastructure, the shopping mall owns the access network, and the robot manufacturer provides the robots and the MEC applications (Robot Control and Video Surveillance).

The overall workflow relies on the following assumptions:

- The MEC Platform VNF descriptor (VNFD) is already on-boarded.
- An Network Service Descriptor (NSD) that contains the MEC platform VNFD is generated by the MEAO.
- The generated NSD is used by the NFVO to instantiate the NS containing the MEC platform VNF.

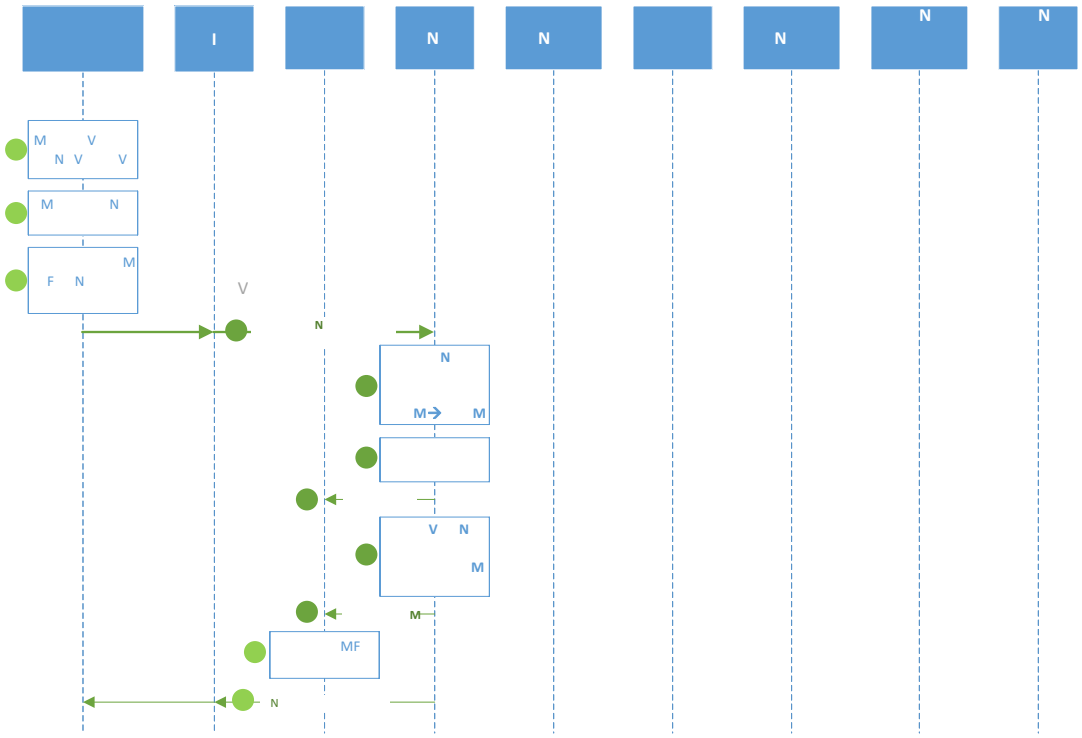


Figure 2.8: On-boarding workflow.

The NS instance is up and running.

All required MEC services are instantiated and initialized on top of the MEC Platform VNF.

After describing the general scenario setup and its assumptions, in the next sections the step-by-step message exchange per procedure are described.

Please note that each numeric bullet corresponds to a message in the corresponding workflow.

On-boarding procedure

The on-boarding procedure is the first procedure that the application provider (in this case the robot manufacturer) has to perform successfully before being able to instantiate the MEC Application on the MEC in NFV environment. The workflow exchange of messages is presented in Figure 2.8 and a step-by-step detailed description is provided below:

1. First, the robot manufacturer defines the development of the MEC applications for the Robot Control and Video Surveillance. The work in [39] proposes guidelines for the development of MEC applications. In this case, the application will use one of the defined MEC services, the RNIS.

2. Afterwards, the MEC application is developed by including all function calls towards the required MEC services in the internal app-workflow. For example, the Robot Control MEC Application is requesting information from the RNIS MEC Service regarding the radio-link quality and requests the information of the current robot position in the shopping mall area from the Localization MEC Service.

3. Once the MEC application is completed, the developer assembles the application descriptor (AppD) and the MEC App Package. AppD parameters provide a description of the MEC features to be used by the application (e.g., required MEC Services, DNS rules, tra c re-direction rules, etc.). An example of those can be found in the leftmost part of Figure 2.7. In the exemplary case of the Robot Control MEC Application, the developer will include the RNIS as required ("appServiceRequired") and Localization as optional ("appServiceOptional") MEC services (leftmost part of Figure 2.7). As a last assembling step, the manufacturer packs the AppD along with all necessary files into a MEC App Package.

4. At this point, the application is ready to be deployed in a MEC infrastructure. Therefore, a robot manufacturer may initiate the on-boarding procedure at any time. This procedure starts by sending an on-boarding request to the OSS BSS of the operator on the Mx1 Mm8 interface. The MEC App Package or a reference pointer (e.g. URL to the package) is attached to the request. The authenticity of the request is checked by the OSS BBS and it is redirected towards the MEAO on the Mm1 interface. This step in the process defines that the MEAO has the main role of orchestrating and managing the requests.

5. Upon reception of the MEC App Package, the MEAO starts the translation of the AppD to a new VNFD. To do so, the MEAO performs a one-to-one mapping of the di erent parameters of the AppD to the ones defined in the VNFD. This is a crucial step to enable seamless co-existance of the MEC functionalities in the NFV environment. As a consequence of the issues described in Section 2.4.1, the one-to-one mapping is not entirely feasible. Therefore, the MEAO performs a partial mapping, including in the VNFD only the matching parameters, while the other parameters are placed in a separate (external) file. In addition, for some specific fields of the VNFD, the MEAO must generate values to generate a complete VNFD. An example of mapping the Robot Control MEC AppD to the Robot Control MEC VNFD is presented in Figure 2.7.

6. After the VNFD is finalized, the MEAO generates a VNF package, which contains the VNFD along with all external files, indexed by a manifest file. The external files consists of all files included in the MEC App package plus the file with unmapped AppD parameters, generated in the mapping process. The generation of the VNF package is an important process which might require a lot of MEAO processing.

7. The MEAO sends an on-boarding request, including the VNF package, to the NFVO

on the Mv1 interface. The NFVO is not aware of any MEC related information and processes the request as a normal VNF on-boarding request.

8. Simultaneously, the MEAO performs a check on the requirements imposed by the AppD parameters (e.g., required MEC Services, tra c rules, DNS rules, etc.). Since the MEAO knows all up-and-running MEC Platform NS instances, it can make the selection of the MEC Platform to be used by the newly on-boarded MEC Application, based on the requirements described in the AppD. Note that the selection of a MEC Platform VNF is mainly determined if the required MEC Services are already present and initialized. Once the MEC Platform to be used has been selected, the MEAO generates a new MEC App Network Service Descriptor (NSD), that contains the new MEC App VNFD (generated in step 5) and the definition of the di erent Virtual Links (VLs) to connect the MEC App NS with the chosen MEC Platform NS instance (at instantiation time).

9. The generated MEC App NSD is on-boarded to the NFVO on the Mv1 interface and similarly processed by the NFVO, as in step 5.

10. The NFVO stores the NSD and VNF package in an internal database.

11. The MEAO concludes the MEC Application on-boarding procedure by sending a confirmation of successful MEC package on-boarding via the operator's OSS BSS (on the Mm1 interface) back to the Robot manufacturer (on the Mx1 Mm8 interface).

Note that in this example, it is assumed that the MEAO chooses the MEC platform VNF to associate the MEC application before the actual instantiation of the application. This may cause some problems if the resources available for the MEC platform change between the time the MEC Application is on-boarded and the time of its instantiation. An alternative workflow (not presented in this work) performs steps 8-11 on instantiation time.

Instantiation procedure

At this point of the workflow, the application has already been on-boarded and is ready to be instantiated, or it is not yet running. In Figure 2.9, the following steps are taken in order to instantiate and boot up the application:

12. The Robot manufacturer requests the instantiation of the MEC application through the OSS BSS (interface Mx1 Mm8) via the Mm1 interface to the MEAO.

13. The MEAO requests the NFVO to generate an NS identifier for the previously on-boarded NSD (as defined in [36], operation *CreateNsIdentifierRequest(NSD)*). The NFVO generates a new identifier (*NSid*) and returns it back to the MEAO.

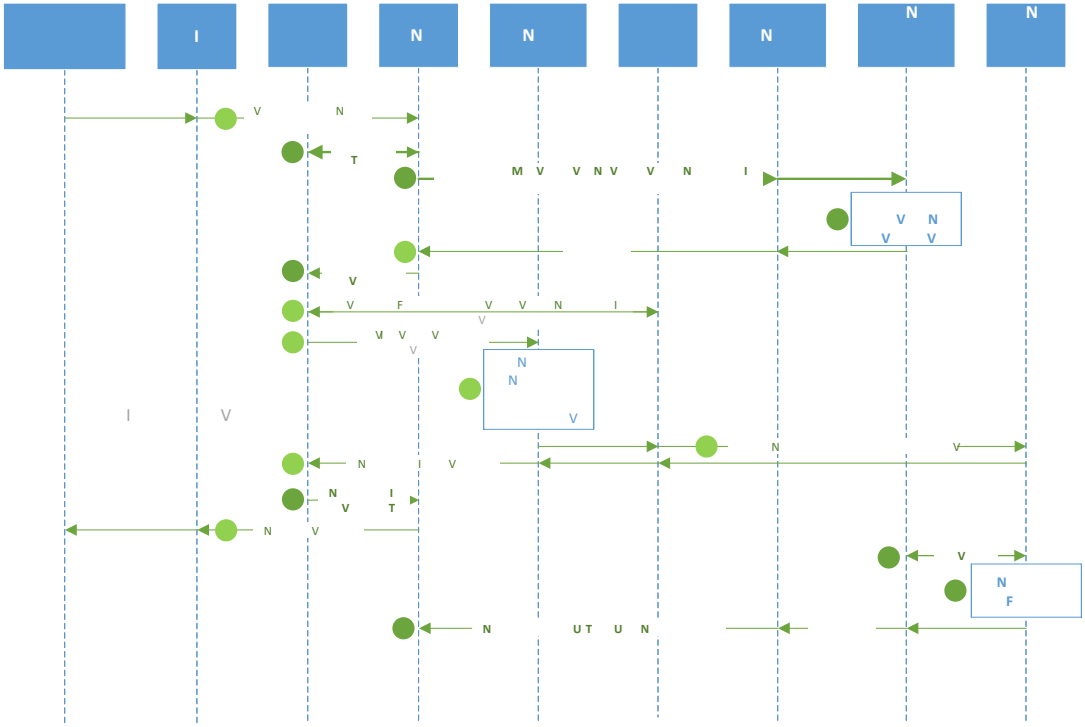


Figure 2.9: Instantiation workflow.

14. Based on the information of the descriptor, the MEAO instructs the Mobile Edge Platform Manager - VNF (MEPM-V) to configure the MEC Platform to include all the requirements for the application, such as tra c redirections, DNS rules or to configure the required MEC services. The MEPM-V redirects the request to the MEC Platform VNF on the Mm5 interface.

15. The MEC Platform VNF analyzes the information from the request, applies all the tra c rules and configures the access for the required MEC Services. For example, the Robot Control MEC application needs that the platform supports the RNIS MEC Service. It is important step to prepare the MEC Platform to accept association of the new MEC Application and inter-connected all necessary MEC Services.

16. The MEC Platform VNF sends confirmation of the applied configurations and rules to the MEPM-V on the Mm5 interface, which are redirected to the MEAO through the Mm3* interface between the MEPM-V and MEAO.

17. The MEAO generates an instantiation request to the NFVO (on the Mv1 interface), using the previously generated *NSid* (as defined in [36], operation *InstantiateNsRequest(NSid)*). The NFVO accepts the request and checks if it is feasible considering current resources availability.

18. In case the application instantiation is feasible, the NFVO requests the VIM to reserve the computational resources needed by the application. Note that according to the specified location constraint in the NSD, the resource reservation is executed

at the same NFVI-PoP where the MEC platform VNF instance resides. Once the computational resources are reserved on the specific NFVI-PoP, the NFVO requests for the allocation of networking resources in order to enable network connectivity between the MEC application VNF NS instance and the MEC platform VNF NS instance. The operations are part of a common ETSI NFVO MANO instantiation workflow procedure [40]. After the computational resources are reserved and the networking connectivity is up and running, the NFVO requests the VIM to allocate the components of the MEC Application VNF. Since the MEC Application NS is a single VNF, there is no need to di erentiate between the VNF and NS for this example. The VIM deploys the software images (e.g., Virtual Machines, VMs) and connects them to the networking fabric. Finally, the VIM notifies the NFVO for the instantiated VNF components.

19. The NFVO notifies the VNFM (MEC App LCM) of the successful allocation of the resources (VMs). In addition, it forwards the VNFD to the NFVO complemented with some extra information regarding the MEC Platform used, such the IP address to be used by the application to connect to the platform.

20. The VNFM (MEC App LCM) extracts all necessary MEC App VNF deployment parameters from the VNFD, and creates a set of configuration parameters, including MEC Platform specific information, to configure the MEC App VNF.

21. The VNFM (MEC App LCM) performs the provision of the MEC App VNF configuration via the VIM. Once applied, the MEC App VNF NS is started.

22. The successful completion of the MEC Application VNF instantiation is acknowledged by the VNFM (MEC App LCM) to the NFVO.

23. The NFVO confirms the instantiation of the MEC Application VNF NS to the MEAO on the Mv1 interface. This step concludes the instantiation of the Robot Control MEC Application as a VNF orchestrated by the NFVO.

24. The MEAO sends confirmation of instantiated MEC Application VNF NS to the Robot manufacturer via the OSS BSS. From this point on, the Robot Control MEC Application mainly depends of the implementation of the MEC functionalities in the MEC in NFV environment.

25. Meanwhile, at bootstrap, the Robot Control MEC Application registers to the MEC platform VNF. The Robot Control MEC Application will use the IP address of the MEC Platform VNF provisioned by the VNFM. The registration request is received by the MEC Platform VNF and the registration is confirmed. Note that all required MEC services are already up & running at the MEC Platform VNF.

26. The Robot Control MEC Application is up & running, consuming all the requested MEC Services. The Robot Control MEC Application is ready to accept connection for robot devices. When a robot is powered on, it attaches to the Wi-Fi access

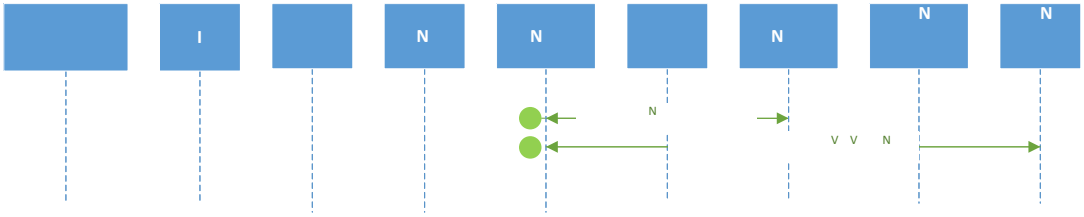


Figure 2.10: Life-cycle workflow.

network deployed in the Shopping Mall. Then a connection is established between the robot and the Robot Control MEC application, as result of the tra c redirection executed by the MEC platform. From this point on, robots continuously send sensor data towards the Robot Control MEC Application, where the data is analyzed. Along with the sensor data, the Robot Control application obtains near real-time information regarding the quality of the radio connection of each robot through the RNIS MEC Service, and obtains near real-time localization information from the Localization MEC Service.

27. The MEC Application VNF NS confirms the correct set-up to the MEC Platform VNF NS via the Mp1 interface. The MEC Platform VNF NS confirms the correct set-up to the MEPM-V via the Mm5, which in turn redirects the confirmation to the MEAO via the Mm3*.

Life-cycle management (LCM) procedure

Once the Robot Control MEC Application is up & running, it performs its activity while being monitored for performance issues by the Orchestration system (specifically by the MEPM-V). A couple of messages are used for maintaining the life-cycle management of the application shown in Figure 2.10.

28. The MEPM-V polls for monitoring information to the VNFM (MEC App LCM) via the Mv2 interface.

29. The monitoring information for the Robot Control MEC Application is periodically updated by the VNFM (MEC App LCM) through periodically polling the Robot Control MEC Application via the Nf-Vn interface.

Termination procedure

At some point of time, the Robot Control MEC Application is no longer be needed. The termination process involves MEC and NFV specific signaling as shown on Figure 2.11 and defined in the following steps:

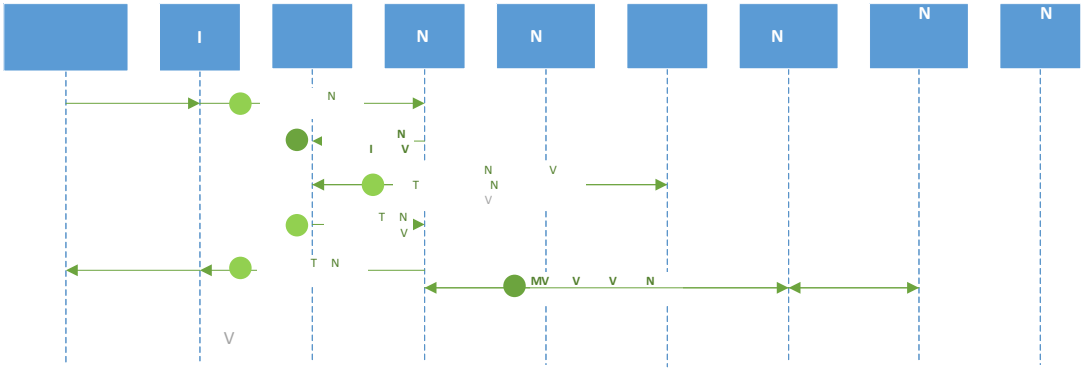


Figure 2.11: Termination workflow.

30. To terminate the Robot Control MEC Application VNF NS, the Robot manufacturer issues a termination request via the OSS BSS (using the Mx1 Mm8 interface) to the MEAO (using the Mm1 interface).

31. The MEAO forwards the termination request to the NFVO.

32. The NFVO using the common ETSI NFV MANO termination workflow procedures (described in [40]) that instructs the VIM to de-allocate all the resources used by the Robot Control MEC Application VNF NS. The VIM de-allocates the resources assigned to the Robot Control MEC application VNF NS and confirms the operation to the NFVO.

33. The NFVO confirms the termination of the Robot Control MEC Application VNF NS instance.

34. The MEAO acknowledges the termination procedure to the Robot manufacturer via the OSS BSS.

35. The MEAO instructs the MEC platform VNF to disable the provision of MEC services and all tra c DNS rules configurations. This is done by sending a request on the Mm3* interface to the MEPM-V which is redirected to the MEC platform VNF NS instance. The MEC Platform un-registers the Robot Control MEC Application and acknowledges the completion of the procedure to the MEAO (Mm3*) via the MEPM-V (Mm5).

2.5.4. Edge robotics in NFV-MEC: experimental considerations

This article departed from the idea that integrating ETSI MEC in an NFV platform brings benefits that can not be achieved by any other mean. From the application developer point of view (following the same rationale as the rest of this work), the benefits of using an ETSI MEC platform can be summarised as: *i)* experiencing a lower end to end delay due to the presence of computing power that can be used to host the application in the vicinity

of the user, and *ii)* the possibility of using contextual information to improve the user experience, information that cannot be obtained or that looses its validity if the processing is done far away from the user. In the same way, the use of a virtualized platform to host the MEC platform benefits from the NFV concept by enabling: *i)* the possibility to scale up or down the resources available for the application based on its current needs, *ii)* the possibility of moving or migrating the application to the best location to serve the users (maybe at some point the application is needed in a per user basis and in another it is better to locate it slightly further away for aggregation purposes), and *iii)* the use of a common white box platform that can host a heterogeneity of applications under a common framework.

In order to provide an experimental proof of these benefits, the authors performed a validation of the Edge Robotics use case in a prior work [38]. This work showcases and measures, under real conditions, the benefits of deploying a Robotics application in a MEC-like platform on top of virtualized resources, as well as provides some hints on the performance factors involved in the edge robotics use case. [38] tackles the above benefits, showing how the use of contextual information can be used to improve driving performance of a robot. The experimental setup is similar to the described use-case in this work. The scenario is deployed on virtualized infrastructure, and makes use of a MEC Application for Robot movement control and a RNIS MEC Service for extracting context data regarding the radio connectivity of the robots. The goal is to make robots go as fast as they possibly can based on their radio conditions. Obtained results shows that the use of the context information through the RNIS MEC Service improved significantly the driving performance of the robot. The control-loop algorithm of the MEC Application adjusts the speed of the robot so it adapts to the the radio quality of the link. Hence in case the robot entered in an area of low radio coverage, the speed would smoothly drop from the maximum of 0.75 m s to the minimum of 0.1 m s. In the event of no context information provided by the RNIS MEC service, the robot starts to perform drive-stop movements that changed the trajectory and driving smoothness, which for a real environment (such as factories, labs, etc.) questions the usability of the robot itself. In addition, the smooth control of the robot was achieved thanks to the reduced latency provided by the MEC platform. The control software has been developed using standard Linux tools and deployed in a set of virtual machines and containers running in heterogeneous hardware, therefore proving that the use of a virtualization approach was useful. Note that this experimental work also showed that the signaling required to enable the use of MEC does not impose critical requirements, as it mostly involves pre-provision mechanisms or exchanges that can be made in a preparation-reaction fashion (e.g., the migration of MEC functions can be triggered in advance).

This previous work did not considered a full NFV platform, but relied on a simpler virtualization platform, which could be extended to become a full NFV platform. However, while conducting these experiments, we discovered the di erent caveats and holes in the current MEC NFV integration specifications, yielding to the work presented in this

article. By applying the proposed MEC-in-NFV integration following the workflows proposed in this work, the experiment could easily evolve into a complete platform where new MEC Application and or MEC Services can be instantiated. For example, the experimental system can be upgraded with the Video Surveillance MEC Application and Localization MEC Service. The combination of the Localization MEC Service and the RNIS MEC Service can produce a radio heat map of the experimental area and assist the MEAO for performing the migration of MEC Application VNFs.

2.6. Comparison with previous work

In this section, we identify other existing e orts, and compare our work with them. Here we point out how our contribution impacts and extends the state of the art, as well as analyze the di erences with existing works. We also try to identify the gaps of existing works, where the application of our proposed signaling flows would evolve the proposed approaches into integrated MEC in NFV compatible solutions. We refer to the experimental considerations summarized in Sec. 2.5.4 for an overview of the performance results that our approach could enable. Note that the experimental results reported in [38] (evaluated in Sec. 2.5.4) can not directly be compared with other works, due to di erent use-case scenarios thus di erent requirements. According to the ETSI technical report [28] the results obtained in [38] show that the conducted experiments satisfy the proposed potential requirements for mobile robots. While the approach evaluated in [38] does not follow the integrated workflows proposed in this article, we argue that the results would be similar, with the additional advantage of providing a MEC and NFV compliant solution.

In [13], which presents the initial idea of joint MEC in NFV architecture, most of the key issues are listed and provided with some *possible* solutions. We have extended this work by categorizing the issues (Table 2.3) and going deeper into the solution space by taking a specific use-case and explaining the step-by-step workflow procedures. Following this approach, we have been able to identify an additional issue that was not foreseen by the time [13] was published. For some of the cases where multiple approaches were proposed, we have favored one solution over the others, and gone further in terms of detailing it, based on our implementation and use-case experience.

In the work presented in [41], the authors similarly propose a deployment of a specific use-case (immersive video) on an integrated MEC-in-NFV platform. The integration of the MEC in NFV environment is broadly described with small e ort in evaluation of the issues described in [13]. The experimental results evaluate the performance enhancement through the use of MEC Application running close to the end users over NFV Infrastructure. The improvement is clearly noticeable through the upload improvement from order of tens of minutes to order of seconds. By using GPU acceleration, the application significantly improves the performance of transcoding several parallel media streaming sessions. In our view, modifying the workflow procedures can even more increase the e ciency of the system. Although the used MEC-in-NFV platform is a modified proto-

Table 2.3: Issues description

Issues	Description	MEC 017	MEC 017 solution	Solution provided in workflow
Conceptual	Mapping of MEC app VNFs to NSs	#1	Multiple	X
	Usage of NFV NS	#2	Multiple	X
	AppD vs. VNFD for MEC App VNFs	#6	Partial	X
	VNF Package vs. MEC application package	#7	Multiple	X
	Comparison of AppD and VNFD data structures	#10	Multiple	X
	NFV construct that corresponds to MEC Host	#11	None Partial	
Workflow	VNF package on-boarding	#8	Multiple	
	MEC package on-boarding	-	-	X
	MEC App VNF Instance Relocation	#12	None	
	MEC App instance instantiation	#13	Multiple	X
	MEC App instance termination	#14	Multiple	X
Communication	Communication between MEAO and NFVO via Mv1	#3	Multiple	X
	Communication between VNFM and MEPM-V via Mv2	#4	Partial	X
	Communication between VNFM and ME app instance via Mv3	#5	Partial	X
	Managing tra c redirection	#9	None Partial	X

type version of the described architecture in [13], some relevant points related to both the on-boarding and the instantiation procedures are missing. In the on-boarding procedure, it is assumed that the MEC app developer would develop the application and generate a set of descriptors accordingly. Both AppD and VNFD are packed in a MAP (MEC Application Package). This is on-boarded together with generated NSD onto the MEC-in-NFV platform. The MEAO is in charge of the analyzing of the on-boarded files (MAP NSD), which basically separates the AppD from the VNF package, stores the bindings of the AppD, VNFD and NSD; and then on-boards the VNF package plus the NSD to the NFVO. In the instantiation procedure, the dependency between the MEC Platform VNF and the MEC App VNF is out of scope. The assumption is that the MEC App VNF would be included as part of the NS (defined by the on-boarded NSD). Once the NS is instantiated, the connection to a already deployed MEC Platform NS instance would be handled by the NFVO. Although the main focus of the work is not to tackle most of the problems, it manages to describe the coupling between a MEC Platform VNF and the data plane into a single VNF (SGWLBO). This approach is extension of the optional deployment of MEC in 4G [42], which is suitable for applying the tra c and the DNS rules for the proposed (video streaming) use case.

The work done in [43] showcases the adaptation of MEC in NFV architecture as a key driver for expansion of Vehicle-to-Everything (V2X) applications and its development. The work lists several emerging challenges that need to be overcome for successful adaptation for the V2X Applications. With the tutorial perspective of our work, we contribute towards tackling one of the major emerging challenges in [43]: the on-boarding and running MEC applications provided by developers not aware of NFV procedures.

A di erent realization of the MEC in NFV environment is proposed in [44]. The authors, in similar manner as our work, propose a parallel extension of the ETSI NFV MANO platform towards a new enhanced MANO platform that can support MEC Applications without additional hardware or coordination overhead. Additional feasibility analysis is provided for several use-cases (i.e., migration of gaming applications, user mobility). The proposed architecture is not inline with [13], despite proposing the MEC

Platform to be realized as VNF, other elements such as the Mobile Edge Orchestrator is proposed to devolve into the NFVO and maintain only application orchestration capabilities into a new orchestrator - Application Functions Virtualization Orchestrator (AFVO). Hence the workflow procedures would be incompatible with this work or with the proposed integration in [13].

Similarly, the authors in [45] propose a novel approach of designing flexible and independent framework for MEC applications that collaborates with NFV frameworks. The work proposes a novel architecture design with introduction of similar concepts of MEC Applications realized as NSs or VNFs. The work envisions the novel Automated Provisioning Framework for MEC (APMEC) framework on top of existing NFV environment as an extension that is able to communicate directly to multiple VIMs and coordinate in parallel with existing MANO platforms. The procedures aim to provide MEC Applications as part of NS by breaking a received NS onto MEC and NFV parts and deploying each part in parallel, using existing MANO platforms or the APMEC framework. However, the descriptors used are not defined and used only by the APMEC framework. A MEC Platform that provides MEC Services towards MEC Applications is non-existing concept in the APMEC framework. In [46] is showcased the parallel orchestration using the implementation of the APMEC framework.

Another framework for parallel orchestration of NFV and MEC is envisioned in [47]. The focus of the work is mainly orchestration of multi-domains than integration of MEC in NFV. The proposed framework does not envision any changes to the NFV and MEC architectures, but the focus is more how to implement an integrated solution for simultaneous orchestration.

In [48], the authors propose specific use-case driven orchestrator module that would bridge the gap between orchestration in NFV and MEC environment. The work proposes some ideas how to overcome the orchestration gaps between the NFV and MEC environment without focusing on the main challenges, suggesting that the new module is adaptable to a MEC in NFV environment.

The implementation of a MEC Radio Network Information Service (RNIS) aimed for a MEC-in-NFV environment is explored in [49]. That work presents how a standard-compliant RNIS can be realized on top of virtualized infrastructure and compares di erent message brokers that enable the RNIS MEC Service to provide information towards MEC Applications. The work does not dive into the implementation details of a MEC-in-NFV environment. Similarly, in [50] the focus is on proposing a VNF placement scheme for the purpose of optimizing the latency and reliability of MEC Applications deployed as VNFs in MEC-in-NFV environment, but not in examination of the MEC-in-NFV environment itself.

2.7. Final remarks on the integration of MEC in NFV and future work

Virtualization, and more specifically, its application to bringing resources to the edge of the network, is one of the key technologies in 5G and future beyond-5G networks. The combined use of MEC and NFV enables a virtualization layer providing the features needed for fine grade customization of the networks. This customization is key for the flexibility demanded from vertical industries.

Despite of the clear inter-relation of MEC and NFV technologies, they have been evolving with isolated, and quite parallel, tracks until very recently. This separate evolution has created integration issues that need to be addressed, as early identified in [13]. We have performed an architectural analysis identifying all possible MEC-NFV integration issues, including feasible approaches to address each of them. In order to better scope this work, we have adopted the specific use case of Edge Robotics to dive into the detailed workflows and mechanisms to combine MEC and NFV. Besides, while doing so we have tried to put the focus point into application developers, as they are the final users of combined MEC-NFV deployments.

A key aspect of the conducted analysis of the combination of MEC and NFV technologies is which logical entity has to be in control of all the orchestration procedures (on-boarding, instantiation, general life-cycle management and termination). Based on an detailed analysis of the following aspects: MEC awareness, number and type of API call requests and states, we have concluded that the MEAO is better suited to be in control of the orchestration. Detailed workflows for the case of Edge Robotics are provided to support this design decision.

Compared with previous related work looking at integration aspects of MEC and NFV, our work is not just limited to a high level analysis of the issues appearing when combining the technologies, but it goes into actual design and validation using a specific use case. Future work includes conducting a proof-of-concept with a prototype of the edge robotics use case. Additionally, we will also explore the issues connected to the the data plane and applying various tra c rules via the MEC Platform VNF (e.g., tra c redirection, DNS rules, etc.).

3. FEDERATION

This chapter provides the definition of the federation concept. The main focus is on the definition of the concept itself and step-by-step guidance of its realization based on di erent environments. The defined concept in this chapter is later used as the base ground for the experimental work done in Chapters 4, 5, and 6.

3.1. Definition of federation

Federation is a mechanism for integrating multiple administrative domains at di erent granularity into a unified open platform where the federated resources and services can trust each other at a certain degree. An administrative domain is a collection of network services and resources operated by a single organization. The administrative domain is viewed as complete entity and its internal structure is hidden or unimportant from outside. The resources and services inside the administrative domain operate with high degree of mutual trust among themselves, but the interaction with other administrative domains - is subject to stringent trustworthiness constraints, with a default high level of alert. The federation is formed in order to increase the degree of trust among di erent administrative domains with a goal of better interoperability of services and resources. Embodiment of a service business-level agreement or partnership between two administrative domains is a federation of trust [51].

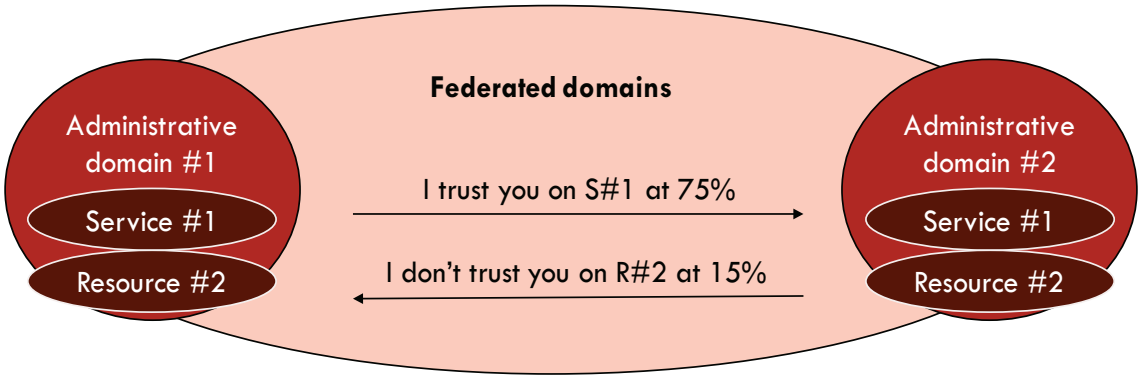


Figure 3.1: High level view on federation

3.1.1. Federation Levels - Consumer and Provider domains

Each agreement defined between two administrative domains have di erent terms and conditions that both should follow to maintain their partnership or federation relationship. The federation procedure is dependent on the setup scenario or the circumstances that demand multiple administrative domains to enable federation among themselves. In the fed-

eration process a domain can play two roles: consumer and provider. Consumer role has the administrative domain that requests federation of services resources, or in other words services resources from external domain to be included as part of its domain services. The provider role is when the administrative domain provides set of services resources to an external (consumer) domain under certain conditions. In each federation scenario there are at least a single consumer domain and a single or multiple provider domains. Administrative domains that have the underlying infrastructure in a near proximity (e.g. same geo-location, co-exist in mutual coverage area, etc.,) are keener to employ federation than administrative domains that are distant (e.g., domain in separate countries).

A single administrative domain can have multiple di erent peer-to-peer agreements with other administrative domains. Each agreement, according to agreed terms and conditions, can belong to a di erent category of federation relationship or di erent federation level. The federation levels indicate the mutual degree of trust among administrative domains, e.g., they can be defined as bronze, silver, gold, platinum, etc. In that sense, if an administrative domain has low-level of trust or intentions to share only limited resources and or services with other administrative domain, both would agree to terms and conditions of a bronze federation level. On the contrary, if two administrative domains establish high degree of mutual trust and share more transparent view on their resources and or services, the federation would belong to a platinum federation level. For higher federation levels, significantly broader options and specific information parameters about resources and or services are exchanged between the administrative domains. For the lower levels, the o ering of resources and or services is limited and information for parameters is more abstract and descriptive.

3.1.2. Service federation

Service federation is the overall process of deploying nested NFV Network Service (NFV-NSs) in a peering administrative domain and stitching them (local and remote) to make an E2E composite NFV-NS. The administrative domain that requests services is referred to as consumer domain while the administrative domain capable of providing services is a provider domain. A requirement of federation scenarios is that domains must have business service level agreements, in place. In turn, there are implicit technical implications, like setting peer-to-peer interconnection among themselves and exchanging their respective NFV-NS catalogues. Eventually, federation agreements should maximize the administrative domains' profit by extending the local domain service o ering and by avoiding the rejection of NFV-NS deployment requests by requesting services to other domains, hence increasing computing networking resource availability and service footprint.

3.1.3. Resource federation

Resource federation is the overall process of consuming providing computing and network resources from to external federated domains. In this case, the decision point is also placed in NFV Orchestrator module. After that decision, a consumer NFV Orchestrator requests resources to a provider NFV Orchestrator. Once granted, the consumer 5GT Service Orchestrator (5GT-SO) takes control over provider's resources. The resource federation process contains the establishment of business service level agreements and the allocation process of providing consuming resources. Based on the agreed terms and conditions, the consumer domain controls and consumes the provided resources while the provider domain charges for their usage.

3.2. Federation characterization

Federation is indeed a broad concept, and as such, has been already tackled in existing works [52]–[59], which consider di erent aspects that we represent in Table 3.1. By doing so, we try to spell out what might be involved in a federation process and what variables could be in place. The first column of Table 3.1 presents the main federation characteristics as found in relevant previous works, along with possible options for each characteristic in the second column (briefly described in the third column).

As mentioned earlier, in a federation scenario, an administrative domain can be either a consumer or a provider. Depending on what is being federated, we refer to service or resource federation. In service federation [52], [54], [56], the provider domain deploys the service and provides the required connectivity for the consumer to use (i.e., *consume*) the federated service. Upon success, the provider domain is used as a proxy for the consumer domain to perform life-cycle management operations over the federated service. In resource federation [55], the provider domain leases the control and management of the federated (virtualized) resources.

A key aspect of federation is how dynamic the environment is. There are two main scenarios that can be considered: (*i*) pre-established & static, and (*ii*) open & dynamic. Prior to any federation procedure, the administrative domains need to define the relationships among themselves in each case they interact as provider and or consumer roles. The relationships are set on business level in terms of trust policies. These agreements can be statically set in advance (e.g., long time before any federation interaction) or they can be dynamically set, minutes range before any federation procedure. The static agreements or pre-established are useful for administrative domains that would expect frequent interaction among themselves, usually neighbouring administrative domains. The agreements set up all the terms for both consumer and provider roles, the pricing models, the trust policies, the security level among the administrative domains. For instance, in a cooperative neighbouring interaction, the terms and policies for general resource federation can be set in manner that is better for the provider, while for particular use-cases a di er-

Table 3.1: Federation characteristics

Federation	Options	Description
Domain Roles	Consumer	Domain consuming federation
	Provider	Domain providing federation
What is federated?	Services	Service extension or new service (resource agnostic);
	Resources	Specific amount of virtualized resources;
Environment dynamicity	Pre-established & Static	Business agreements with known members (e.g., Service Level Agreements (SLAs)); Advertisement discovery of federation capabilities;
	Open federation & Dynamic	Rapidly changing and unknown members; Requires higher security degree; Announcement negotiation for federation;
Interconnection framework	Decentralized peering	Peer-to-Peer connection with each agreed administrative domain. Individual connections contain independent rules of interaction.
	Centralized	Single central entity manages the interaction and applies rules for all involved administrative domains
	Decentralized distributed	Peer-to-peer framework using consensus protocol for applying rules for all connections, maintain trust, security, or node failure.
Layer communication	Single	Orchestrator-to-orchestrator (same entities)
	Cross-layer	Cross-layer interaction;
Federation deployment	One-to-one	Requesting federated service or resource from a single domain
	One-to-many	Requesting federated services or resources from multiple domains; Simultaneous deploy & chaining

ent set of terms and usage polices can be favourable for consumer. These agreements in pre-agreed federation are usually long-term agreements with fixed pricing (subscription based), but any length or pricing can be applied.

Dynamic or open federation relationships are set on-line, minutes prior to establishing any federation of resources or services. These agreements usually define roles in a particular use-case. They contain similar terms and policies; however, they are mostly short-term with dynamic pricing policies. The open federation is usually competitive following an auction model of reserving resources. Moreover, as in an open federation, the administrative domain decides dynamically whether to join or leave an existing federation. The administrative domain does not need to make decisions at predetermined time, so the duration of its federation membership is not fixed. Federation in this case is dynamically formed in a distributed, bottom-up manner. An open framework has to o er secure

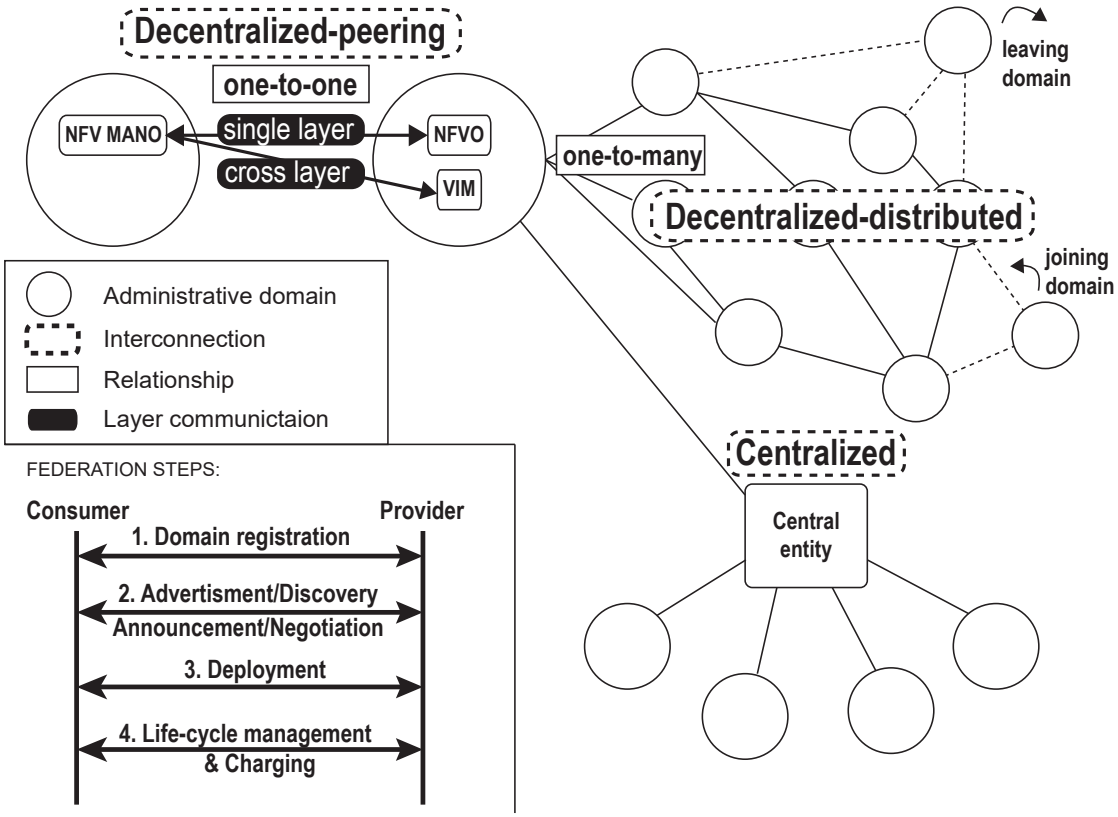


Figure 3.2: Federation classification and steps

Table 3.2: Comparison between dynamic and static federation

Feature	Static Federation	Dynamic Federation
Negotiation frequency	Low	High
Changing terms approach	Central; Peer-to-peer; top-down	Autonomous; Distributed
Stability	High	Low

and trusty processes to enable domains to reach short-term agreements shortly before a service or resource is deployed, in a form of *dynamic SLAs* [60].

For particular use-cases, the static approach would have pre-determined roles and amount of resources that each provider domain provides to the consumer domain. The time to request, reserve and use federated resources is shorter than in the open-federation manner. Moreover, administrative domains form a federation based on a (long-term or short-term) agreement so that their membership remains unchanged for an extended period of time. Also, mutual agreements are required for any membership change to an existing federation. Federation in this case can be formed by a central entity in an o ine, top-down manner.

Also related to the dynamicity of the environment, another key federation characteristic is the nature of the interconnection between the involved domains. As shown in Table 3.1, we can di erentiate among: (*i*) centralized, (*ii*) decentralized-peering, and (*iii*)

decentralized-distributed options. For static federation, either centralized or decentralized-peering approaches are considered as most appropriate. In a decentralized-peering scenario, domains establish independent peer-to-peer connections to every administrative domain with an existing agreement. The maintained policies linearly increase with every new federation agreement, which would make this hard to handle and scale in dynamic environments. A centralized framework provides a single central entity to oversee and manage the federation among all involved administrative domains. A hybrid option is a decentralized-distributed framework, that relies on a consensus protocol to apply common rules and policies of interaction in a peer-to-peer network. Any domain joining or leaving does not alter the ratified behavior. Figure 3.2 illustrates the di erent federation options and federation steps. In Section 5.5, we describe how Blockchain technology can be applied as a decentralized-distributed network solution.

Regardless of the interconnection framework in use, administrative domains are composed of di erent layers (e.g., of di erent resources) and therefore communications may happen at di erent layers through di erent interfaces [56]. Single-layer federation connections are established between equivalent entities (e.g., orchestrator-to-orchestrator, shown on Fig. 3.2), laying on the same architectural layer using east west interfaces. Hierarchical federation connections are cross-layer connections established between entities on di erent layers using di erent interfaces –northbound or southbound.

Last, but not least, we can also characterize how many domains are involved in a given federation instance. This is what we refer to as "federation deployment", and we consider two possible cases: one-to-one and one-to-many. A one-to-one federation deployment occurs when a consumer domain generates a federation request for a single provider domain to provide the required service or resource. In [57], this is referred to as the Resource Manager Role. In a one-to-many federation deployment, the consumer domain simultaneously requests several services to be provided by multiple provider domains. The consumer domain later orchestrates complex chaining of the federated services and or resources from the diverse provider domains [52]. Similarly, in [57], this is referred to as Aggregator or Hub role.

3.2.1. Procedures involved in federation

Although the ETSI NFV has defined the di erent orchestration and lifecycle management procedures (e.g., instantiation, termination, scaling, etc.), and has even explored di erent architecture options to support the collaboration among multiple federation domains, there is not yet a clear standardized set of procedures defined. It is critical to preserve interoperability between domains, thus minimizing the extent to which existing mechanisms (e.g., ETSI NFV MANO procedures) would need to be modified. We next summarize the high-level sequential steps that are involved in a generic service resource federation process [55], [56]:

1. **Domain registration.** This is required regardless of the dynamicity of the environment. In a static federation, administrative domains establish peer-to-peer individual connections to every contracted administrative domain. In an open federation, the registration, although open to new administrative domains, might require a voting consensus to allow transparent decisions (of new members acceptance, governance, etc.). Security mechanisms and integrity checks are fundamental.

2. (a) **Advertisement Discovery.** In static federation environments, depending on the pre-established agreement, peering administrative domains periodically exchange or *advertise* information on available services resources. For a large number of connections, an administrative domain uses polling or *discovery* instead of advertising. Based on the exchanged shared information, a consumer domain is capable to generate a global view of available services or resources for federation, helping it to take a better federation decision.

   (b) **Announcement Negotiation.** In open and dynamic federation environments, domains re-negotiate federation terms repeatedly. Di erent negotiation techniques are available: *bilateral*, *match-matching* or *autonomous* [61]. Match-matching and autonomous are more suitable for a centralized entity. A consumer requests federation by specifying a range of terms. The central entity matches potential provider domains that strictly match terms - *match-matching*, or by close-to-full fulfillment - *autonomous*. Upon a matched domain, both provider and consumer domains receive the connectivity details. In the *bilateral* case, more suitable for a decentralized-distributed interconnection scheme, the consumer domain broadcasts an *announcement* request for federation. Potential provider domains engage in a reverse-auction fashion by replying with bidding o ers. The final decision is made by the consumer domain using internal policy criteria. The selected winning provider domain proceeds into fulfillment of the federation request.

3. **Deployment.** In this step, the "winning" provider proceeds with the deployment of a federated service or resource. Upon successful deployment, both, the consumer domain and the provider domain, establish data plane connectivity and inclusion of the federated service resource for the intended purpose. In an NFV MANO environment, these steps are repeated for every service extending scaling healing operation [54]. Finally, at the end of this stage, federated resources services commence running as an integral part of the consumer domain.

4. **Life-cycle management charging.** Once the federated resource service is embedded and running, the consumer domain manages its life-cycle using the provider domain orchestrator as a proxy in a single layer communication scheme. In a hierarchical communication scheme, the control plane goes through the north southbound interfaces (e.g., cross-layer on Fig. 3.2). Both domains monitor the federated usage and calculate the fee according to the established agreement. Note that the provider

domain has a "kill switch" or the ability to terminate the federation at any point in time [55]. As opposed to static environments, establishing a monitoring and charging process can be quite challenging in dynamic environments.

3.3. Federation in 5G-TRANSFORMER (5GT) project

3.3.1. Service and resource federation in 5GT

This section introduces the high-level architecture of the 5G-TRANSFORMER (5GT) system, with a focus of the federation as an essential feature of the 5GT framework. The federation feature allows 5GT service providers to deploy and manage services in external domains through service federation and to obtain and control computational network resources from other domains through resource federation. The 5GT platform is more suitable for static federation.

3.3.2. Baseline 5GT architecture

The 5GT architecture is built with the goal of providing a platform with flexible and dynamic management features to serve the needs of multiple and heterogeneous services coming from di erent vertical industries. Such services can be concurrently instantiated over a shared infrastructure that combines multiple heterogeneous types of resources in terms of computing, storage and networking. The 5GT architecture consists of four main building blocks, namely 5GT Vertical Slicer (5GT-VS), 5GT-SO, 5GT Mobile Transport and Computing Platform (5GT-MTP), and 5GT Monitoring (5GT-MON). Two domains are represented in Fig.3.3, each having a full stack consisting of these four blocks (5GT-MON not represented for the sake of simplicity).

The 5GT-VS is the entry point for vertical industries to access the 5GT platform. It provides a web portal and an Application Programming Interface (API) for (vertical) end-users, simplifying the process of requesting vertical services. The 5GT-VS exposes a catalogue of vertical services o ered to vertical end-users, which are customized by the vertical users by setting parameters to mach their service requirements. The 5GT-VS translates from business-oriented vertical service requests into slice requests, which are eventually mapped to NFV-NSs. In turn, these NFV-NSs are requested by the 5GT-VS to the 5GT-SO.

The 5GT-SO manages the E2E orchestration and the lifecycle of the NFV-NSs. They are deployed by matching their requirements with the resource availability in the 5GT-MTP [62] through placement algorithms. The 5GT-SO contains two types of orchestrators: (*i*) the NFVO Network Service Orchestrator (NFVO-NSO) and (*ii*) the NFVO Resource Orchestrator (NFVO-RO). Both orchestrators embody the functionalities of a typical NFV orchestrator (NFVO) [63]. Using the 5GT architecture, NFV-NSs can be deployed over single or multiple Administrative Domain (AD). For deployment over single (local) do-

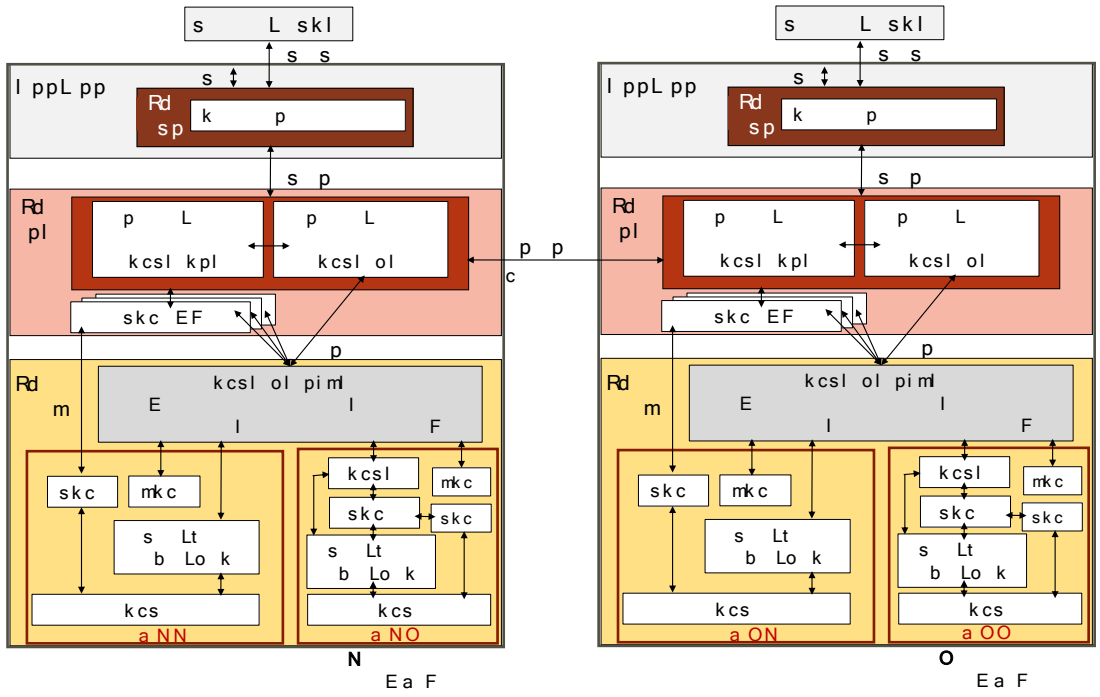


Figure 3.3: 5GT Architecture

main the 5GT-SO uses the So-Mtp interface towards the local domain 5GT-MTP. For multi-domain deployment and management, referred to as federation, the 5GT-SO establishes peer-to-peer inter-connections with external ADs via the So-So interface [64] [65].

The 5G-MTP is responsible for orchestrating heterogeneous resources (computing, network, storage) available in each domain, exposing them to the 5GT-SO and serving its resource allocation requests to instantiate VNFs and to manage the connections of the underlying transport network. The 5G-MTP embeds plug-ins for interaction with the managers of each subset of resources : for (*i*) compute resources, the the Virtual Infrastructure Manager (VIM) or for (*ii*) inter-PoP networking connectivity, the WAN Infrastructure Manager (WIM). The 5GT-MTP generates an abstracted resource view towards the 5GT-SO via the So-Mtp interface according to policies in place.

The instantiation of NFV-NS follows a top-down workflow from the 5GT-VS down to the 5GT-SO and to the 5GT-MTP and underlying physical infrastructure. Note that a NFV-NS can be a single compact NFV-NS (containing multiple VNFs) or a modular composition of multiple (nested) NFV-NSs, referred to as composite end-to-end NFV-NS.

In the case of single domain instantiation, the important decision point is the placement of VNFs over the available resources. The Placement Algorithm, part of the 5GT-SO, generates placement decision for all the VNFs that composed the NFV-NS being instantiated [66]. The decision is based on the abstracted resource view provided by the 5GT-MTP and the NFV-NS requirements coming in the instantiation request from the 5GT-VS. In case of a federation scenario (multiple ADs), the number of deployment

options significantly increases.

3.3.3. Service federation in 5GT

In the 5GT framework, service federation applies to composite NFV-NS. A request for instantiation of a composite NFV-NS is sent from the 5GT-VS to the 5GT-SO. Each nested NFV-NS (of the composite) has a specific NFV-NS descriptor stored in the 5GT-SO repository which contains the service information for the specific nested NFV-NS. This information contains the number of VNFs, the computational resources used (i.e., CPU, memory, storage), the VNFs topology and requirements (i.e., bandwidth, latency requirements, service access points, etc.). The 5GT-SO decides for each nested NFV-NS whether to instantiate it locally (using the local 5GT-MTP) or to request service federation from a provider domain 5GT-SO via the So-So interface.

The service federation, as a concept of deploying a nested NFV-NS of a composite NFV-NS in an external domain or of providing a nested NFV-NS to an external domain, is a broad concept. It appears in two forms: service delegation service federation. Both procedures are orchestrated by the Composite Network Service Orchestrator (NSO) module of the 5GT-SO. Service delegation is the process of delegating the request for a service instantiation to an external (federated) domain. Generally, when a 5GT-VS issues a request for instantiation of a single NFV-NS (not composite), the request is received by a 5GT-SO (Composite NSO) on the Northbound Interface (NBI). The Composite NSO analyses the request and decides to delegate to an external peering domain. The Composite NSO simply redirects the request on the Eastbound Westbound Interface (E WBI) (So-So-LCM). The peering 5GT-SO (Composite NSO) accepts the request and instantiates the single NFV-NS. Note that federation of composite NFVNS or further decomposition by the provider domain Composite NSO is not foreseen. More information regarding decomposition of NFV-NSs can be found in Section 6.6.5.2. Upon instantiation of the single NFV-NS, it sends back all the information to the consumer 5GT-SO (Composite NSO). The consumer Composite NSO redirects the response back to the 5GT-VS on the NBI. The local consumer 5GT-SO role is to act as a proxy between the 5GT-VS and the federated provider 5GT-SO. The 5GT-VS is not aware of the setup, hence service delegation is transparent for the 5GT-VS. The lifecycle management is handled by the federated provider 5GT-SO. The monitoring information is delivered to the 5GT-VS using the local consumer 5GT-SO as proxy as well. Federation of services in 5GT-SO follows a similar approach as in ETSI Group Specification (GS) NFV Infrastructure and Architecture Working Group (NFV-IFA) 028 [28]. Federation of services is used for composite NFV-NSs or single NFV-NSs that are decomposed by the NS decomposition algorithms. The composite NFV-NS has defined a set of nested NFV-NSs that by instantiating each of them and stitching them together form the composite End-to-end NFV-NS. On the other hand, the NS decomposition algorithm runs at instantiation time and generates a set of nested NFVNS out of a single NFV-NS. In both cases, the Composite NSO decides for

each of the nested NFV-NS whether to deploy it locally or in a peering federated domain. The process of instantiating part of a composite NFV-NS (one or more nested NFV-NSs) in a federated domain and another part in a local domain in order to form the end-to-end NFV-NS is defined as service federation. The main di erence with respect to service delegation is that the consumer Composite NSO has an active role in orchestrating the composite End-to-end NFV-NS. The 5GT-SO receives a request for instantiation of a composite NFV-NS single NFV-NS from 5GT-VS on the NBI. In the case of single NFV-NS, the Composite NSO first makes decomposition of the requested NFV-NS into several nested NFV-NSs using the NS decomposition algorithm module. More on the decomposition are described in Section 6.6.5.2. Afterwards, the Composite NSO decides where to instantiate each of the nested NFV-NS, in the local domain or in a federated domain using service federation. The reasons can be diverse, such as: lack of resources provided by the local 5GT-MTP, di erent target location that is not supported by the administrative footprint, extension of services, unsupported NFV-NSs, lack of on-boarded NSDs for the nested NFV-NS, etc. Upon the decision for service federation, the Composite NSO queries a peering administrative domain for availability of providing the requested nested NFV-NS. (Alternatively, the set of o ered services could be pre-negotiated o ine among administrative domains.) Note that the information of availability is previously stored in the local Catalogue Database (DB) and the network connections to all peering 5GTSOs are already established. Optionally, for the dynamic decomposition case, the consumer Composite NSO can provide an NSD to the peering (provider) 5GT-SO for the requested NFV-NS for federation (e.g., for more specific deployment flavor). This optional case is for further study. The provider Composite NSO checks confirms the availability and starts with instantiation of the federated NFV-NS. The instantiation procedure in the provider domain is similar to instantiating local nested NFV-NS (through the Constituent NSO, RO-OE, RO-EE, etc.). However, the process is marked as "federated NFV-NS" at starting time. Once the provider domain instantiates the federated nested NFV-NS, the provider Composite NSO confirms the instantiation to the consumer Composite NSO. The Composite NSO checks instantiation of both local nested NFV-NSs and federated NFV-NSs before proceeding with the stitching procedure. In the stitching procedure the Composite NSO instructs the Composite RO to exchange information and setup interconnections with the provider domain Composite RO on the E WBI (So-So-RM). Once all the interconnections are set locally and with the provider domain, the composite End-to-end NFV-NS is up and running. The implementation workflow of establishing service federation is covered in Section 6.7.7. As in ETSI GS NFV-IFA 028 [28], the consumer Composite NSO is not aware of the resources and VNFs that the provider Composite NSO is using to provide the federated NFV-NS. The consumer 5GT-SO NFVO-NSO is using the So-So-LCM to send requests for lifecycle operations of the provided NFV network service. The lifecycle operations are performed by the provider 5GT-SO NFVO-NSO. The So-So-MON allows exchange of limited information consisting of performance indicators and fault alarms. According to the already negotiated terms and conditions, some performance indicators may be hidden from the consumer domain. The 5GT-SO NFVO-NSO is using the So-So-

CAT reference point to exchange catalogue updates, querying for NSDs or on-boarding MEC AppDs.

3.3.4. Resource federation in 5GT

There is another federation scenario in which our algorithm can be used, that is, resource federation. Eventually, local domain resources of the 5GT framework may fail or they may be exhausted by running services. In this case, the 5GT-SO uses resource federation to request resources from a provider domain and to be able to use them as if they were local.

Resource federation is the overall process of consuming providing computing and network resources from to external federated domains. In this case, the decision point is also placed in the 5GT-SO module. After that decision, a consumer 5GT-SO requests resources to a provider 5GT-SO. Once granted, the consumer 5GT-SO takes control over provider's resources. Based on the agreed terms and conditions, the consumer domain controls and consumes the provided resources while the provider domain charges for their usage.

3.4. Federation of resources concept in the dynamic edge - 5G Coral

In the thesis provides description on a conceptual work for federation specified for dynamic environments. Similar to the static NFV environments described in the previous section, it is described the step-by-step concept of enabling federation in a volatile environment where networking and computing resources are constrained with fluctuating performances

3.4.1. 5G-Coral architecture

Before diving into the federation concept, first we briefly go through the 5G-CORAL project system architecture. Figure 3.4 shows the 5G-CORAL system architecture with the two main components: Edge and Fog computing System (EFS) and Orchestration and Control System (OCS). The EFS is a logical system subsuming Edge and Fog resources that belong to a single administrative domain. An EFS provides service platforms, functions, and applications on top of available resources, and may interact with other EFS domains.

The OCS is a logical system in charge of composing, controlling, managing, orchestrating, and federating one or more EFS(s). An OCS comprises VIMs, EFS managers, and EFS orchestrators. An OCS may interact with OCSs of other administrative domains. The OCS components, which are shown from bottom to top in Figure 3.4, are:

VIM contains the functionalities used to control and manage the interaction of the

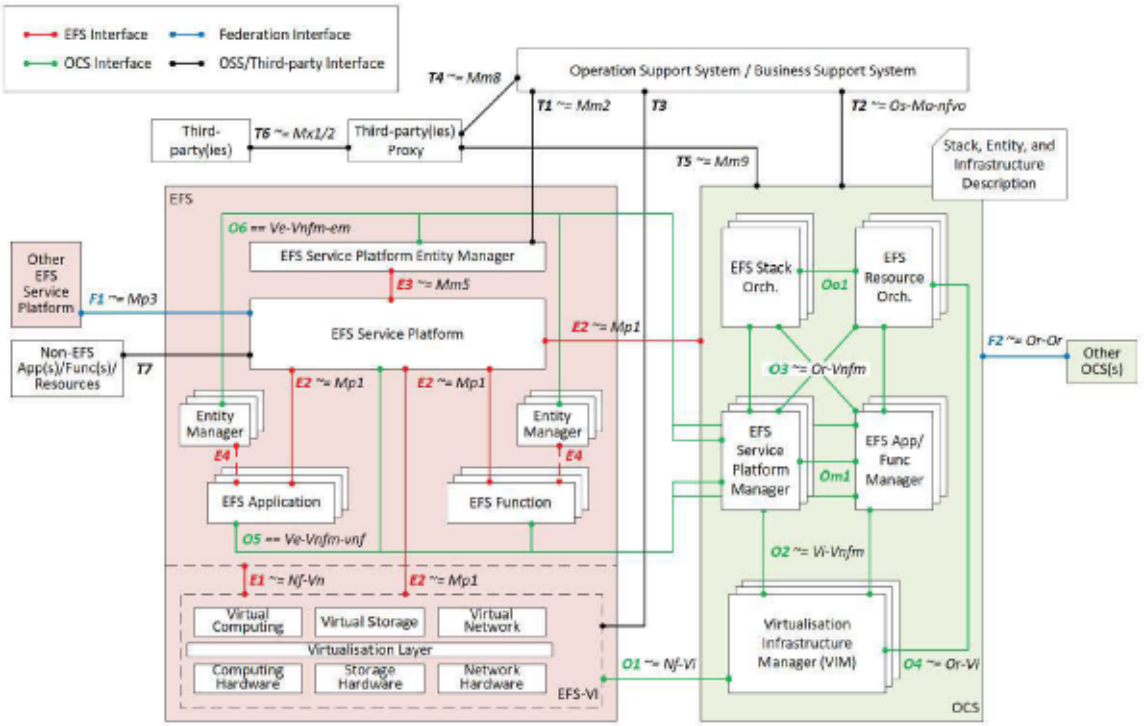


Figure 3.4: 5G Coral Architecture

service platforms, virtual functions, and applications with the constituent edge and fog resources;

- An EFS Manager has responsibility for the life-cycle management (e.g. instantiation, update, scaling and termination) of the service platforms, virtual functions, and applications in the EFS;

- An EFS Orchestrator oversees the orchestration and management of edge and fog resources and composing the EFS. An EFS Orchestrator is split into an EFS Resource Orchestrator and an EFS Stack Orchestrator. The EFS Resource Orchestrator supports accessing the edge and fog resources in an abstracted manner independently of any VIM. While the EFS Stack Orchestrator is responsible for the EFS Stack life-cycle management operations (e.g. instantiation, update, query, scaling and termination);

- An EFS Stack can be viewed architecturally as a forwarding graph of functions and/or application interconnected by supporting edge and fog resources and/or service platforms. An EFS Stack extends the ETSI NFV Network Services by also considering interconnections with applications and service platforms;

- An EFS Stack Descriptor extends the ETSI NFV Network Service Descriptor by also considering applications and service platforms in addition to network functions. It describes the requirements and interconnections of one or more EFS Functions and EFS Applications between them or with the EFS Service Platform;

- An EFS Entity Descriptor extends and combines ETSI NFV VNF and ETSI MEC

App descriptors to uniformly describe the various characteristics of EFS Functions, EFS Applications, and EFS Service Platform. EFS Entity Descriptors are referenced and included into an EFS Stack Descriptor.

3.4.2. Motivation for federation in 5G-CORAL

Each administrative domain is composed of set of computing storage networking devices that shape the underlying infrastructure of a single administrative domain. As mentioned in [67], multiple administrative domains may exist in a same service area. Considering the 5G-CORAL environment, the underlying infrastructures of multiple administrative domains are in constant adjacency. The nearness of various technologies opens a spectrum of possibilities for deployment of di erent EFS services applications that rely on multiple underlying infrastructures. By cooperation among administrative domains and losing the strict boundaries, the inclusion of external resources is feasible. The process of adopting external resources provided by another peering provider domain for the goal of deploying an EFS service application is called federation of resources.

How an administrative domain would benefit from a federation of resources? In 5G-CORAL environment, each administrative domain has its own underlying infrastructure as EFS resources. The quantity of the set of EFS resources varies from large to a set of few EFS resource per administrative domain. In both cases, large or few amount EFS resources, each underlying infrastructure is limited. The limitation can be in terms of capacity, lack of certain technology, user accessibility, etc. In order to expand the limitation without extending the CAPEX and or OPEX, the administrative domains can use federation feature. The federation as concept allows the administrative domains to maintain the service level without service interruption and high expenses. Depending on the inter-domain interactions, the global welfare of the administrative domains may increase with adoption of federation feature. In environment close to the edge of the network where the infrastructure resources are volatile, through the use of resource federation, the stability can be increased.

In order to enable the federation of resources through 5G-CORAL platform, the whole process of federation goes through several steps. First, it is mandatory to identify all the stakeholders actors that are part of a certain use case scenario. A proper model of interaction between all the involved parties or stakeholders has to be established. Finally, the process of resource federation implemented by setting up how EFS resources interact and establish multi-domain connections between each other using the 5G-CORAL system.

3.4.3. Federation interaction model

Once the federation between 5G-CORAL administrative domains is defined as static or pre-established. The next step is to define the interaction between the 5G-CORAL platform at each domain. The interaction between the administrative domains can be on hi-

erarchical or peer-to-peer level. The approach of the 5G-CORAL is to apply peer-to-peer cooperative model of interaction. There are three cooperative models for EFS resource federation [67]:

Trust model

Loan model

Concession model

The loan model is preferable for the open federation, while the concession model for the non-volatile resources and the trust model is well suited for long-term inter-domain relationships. For these reasons and since the static method is adapted in 5G-CORAL, the trust cooperative peer-to-peer model is most suitable at this point. In this static model the pricing can be fixed or posted-scheme that goes through subscription-based charging scheme (monthly or yearly based) [67]. Additional to the defined federation model, each administrative domain may introduce sub-models for specific use-cases that needs to be translated to well-defined SLAs. Moreover, the specific use-case would be seen as a case where di erent SLA agreements providing better conditions is in place instead of the agreement for a general federation. For example, for a certain administrative domain that provides specific set of services over Wi-Fi access, it may set up specific SLA agreements with neighbouring domains over their Wi-Fi radio resources.

3.4.4. Inter-domain connection (F2 interface)

Next, an administrative domain establishes links to all federated domains on the OCS level via the F2 interface. For example, if administrative domain A has established federation agreements with administrative domain B and administrative domain C then there will be two links on the F2 interface, one from OCS A towards OCS B and another one from OCS A towards OCS C. The F2 interface is an interface for inter-connection of peer-to-peer OCS platforms residing in di erent administrative domain. The document focuses on the resource federation, hence the communication through the F2 interface would be mainly towards the federation of resources related operations. Having that in mind, the communication on F2 interface is between EFS Resource Orchestration modules.

The EFS Resource Orchestrator module supports accessing the edge and Fog resources in an abstracted manner independently of any VIMs, as well as governance of service platform function application instances sharing resources in the EFS [67]. In the federation (SLA) agreements the administrative domains share the endpoints (e.g., IP addresses, URL, etc.,) of their EFS Resource Orchestrators. The endpoints are used to enable communication through the F2 interface. The communication on the F2 interface is composed of three phases: advertisement phase, instantiation phase, and termination phase (shown on Figure 3.5).

To successfully perform the federation, EFS Resource Orchestrators (ROs) belonging to different domains will communicate via interface F2 to execute a federation message exchange. Within the message exchange, the consumer domain EFS RO has to start the procedure, and the provider EFS RO will suggest a feasible node to be federated (advertisement/discovery phase). Then, the consumer EFS RO will accept or decline the offered resource (negotiation phase), answering to the provider EFS RO. The EFS RO should interact with the VIM and the EFS Application/Function Manager to complete the process (instantiation phase).

Advertisement/negotiation phase. The advertisement phase or negotiation phase is when all inter-connected administrative domains request/offer set of EFS resources. An administrative domain as a consumer role requests federating resources from other provider domains, whereas an administrative domain as a provider role offers available resources for federation and negotiate over their usage (e.g., duration, pricing, etc.). The providers of federated resources can periodically update their capabilities or reply the offered resources per request. The periodic update of currently available resources for federation would enable all peering administrative domains to have updated global view and rapidly decide for the optimal resources. However, the mobility and volatility of the 5G-CORAL resources demand frequent message exchange on the F2 interfaces, which due to delays or traffic congestions may produce inaccurate updates of the global view. To overcome this issue, the provider EFS RO advertises the available resources for federation only upon received request from a (potential) consumer domain. The request/advertise approach would allow each administrative domain to apply policies and prioritize requests. For example, domain B may respond to a request arrived from a highly ranked domain A and not respond to a request from lower ranked domain C, in case that both requests arrived at the same time at domain B. In this way, by applying the policies, the signalling overhead is significantly reduced.

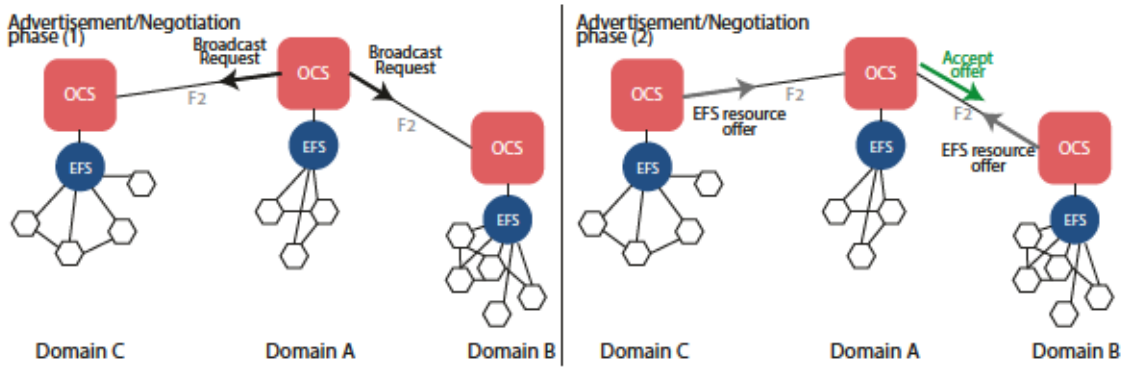


Figure 3.5: OCS Federation interaction – Advertisement/Negotiation phase

Once the consumer domain has the need of adapting federated resources, the constituent EFS RO prepares a request for federation. The request is multi-casted towards the peering administrative domains according to the demands needed (e.g. geo-location of

the resource). For example, as Figure 3.5 shows, domain A broadcast requests to neighbouring domains (domain B and domain C). The potential provider domains (B and C) generate their o ers advertisements of available resources for federation and respond to the request. The consumer domain A accumulates the responses for a certain time (e.g., once a timeout for received o ers expires) and then ranks the received advertisements. As shown on Figure 3.5, the consumer domain A chooses the optimal set of resources (from domain B) and the EFS RO sends reservation requests (Accept o er) to the chosen provider domain B. The chosen providers confirm the reservation request and that is the last message exchange for the advertisement negotiation phase.

During the negotiation phase, parties should take into account the federation stability, which could be a ected by at least two factors. First, mobility and volatility of EFS resource may later invalidate the usability of federated resources that have been o ered. Second, the provider domain may unilaterally retract federated resources that have been o ered to some consumer domain and provide another consumer domain with the retract resources as a means to earn more profit. Generally speaking, if a participant can earn more profit by leaving a federation, the federation will fall apart; if a group of participants can all earn more profits by leaving a federation and forming another one, the federation will fall apart. This scenario may not be avoided if administrative domains earn their own profits individually, as in the case of peer-to-peer federation model. However, if all participants share the total profit in the federation (a group federation), instability of federation can be avoided by an appropriate allocation of federation profits to members.

Federation instantiation phase. The instantiation phase begins when the provider EFS RO confirms incoming request for reservation of available resources. Then the EFS RO sends reservation request to the VIM on the O4 interface. From the three planes (management, control and data plane), only the management plane is not federated. The provider domain keeps the EFS resource attached to the local management plane. The VIM reconfigures the control and data plane of the resources that are being reserved. Once both planes are reset to idle, the operation is confirmed from the VIM to the EFS RO. In order to connect the reserved resources with the consumer domain, the EFS RO issues request to the EFS Application Function Manager to instantiate tunnelling function (e.g., SDN-WAN function) on top of the reserved resources (see Figure 3.6). The tunnelling (SDN-Wide-area Network (WAN)) function is instantiated in order to create secure tunnel and grant orchestration privilege to the consumer (external) domain over the control and data plane of the reserved resources. Note that the management plane of the reserved resources would remain orchestrated by the constituent EFS RO and VIM for the whole duration of the federation process.

Upon instantiation of the tunnelling (SDN-WAN) function, the EFS Application Function Manager exchanges security parameters (e.g., security keys) or provides the ID and the IP address of the tunnelling (SDN-WAN) function to theEFS RO. The EFS RO provides this set of information (ID and IP address) on the F2 interface along with a confirmation that

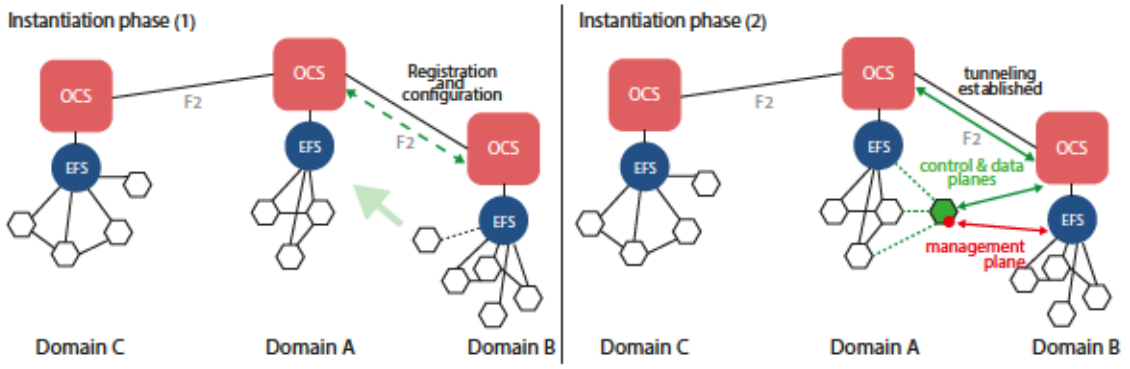


Figure 3.6: OCS Federation interaction – Instantiation phase

the reserved resource is ready to be federated by the consumer domain. The consumer EFS RO receives the information and instructs already instantiated consumer SDN-WAN function to establish the tunnel. After the tunnel is established, the resources are federated and ready to be used by the consumer domain. The consumer EFS RO sends confirmation to the provider EFS RO and the charging process is initiated.

Federation termination phase. When the consumer domain wants to terminate the federation of the resources, the consumer domain sends termination request to the provider EFS RO on F2. The provider EFS RO initiates termination of the SDN-WAN function to the local EFS Application/Function Manager. Once this operation is done, the provider EFS RO sends reconfiguration request to the VIM. Both (control and data) planes are reconfigured to retrieve the reserved resources and make them available in the local domain. The VIM notifies the provider EFS RO for concluded reconfiguration and the provider EFS RO stops the charging and/or accounting process. The provider EFS RO notifies the consumer EFS RO that the federation has terminated successfully and optionally provides the charging information.

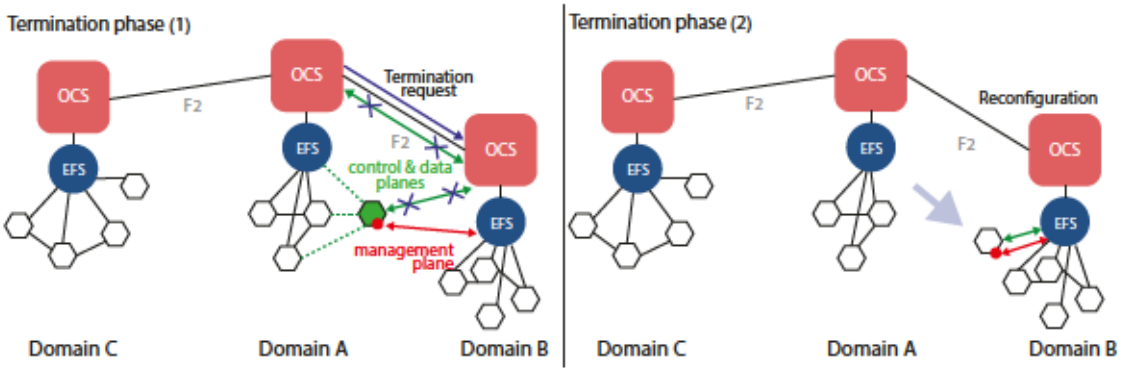


Figure 3.7: OCS Federation interaction – Termination phase

During the termination phase parties should take into account that a federated EFS resource is stable if it can be used for an extended time so both the provider and consumer domains can benefit from it. Instability of federated EFS resource incurs high signalling costs without real benefits. There are several reasons for a federated EFS resource to be

unstable. One occurs to mobile EFS nodes (fog nodes). If a fog node is a part of the EFS resource of a provider domain, o ering it to a consumer domain may risk the possibility of losing connection with it possibly due to its movement.

3.5. Remarks on the federation concept

In this chapter we presented the general definition and concept behind federation of NFV-NS and resources. We performed a classification of the federation concept based on di erent federation characteristics. Afterwards we present the general step-by-step procedures that might be in the federation of NFV-NS or resources. The concept of federation is introduced in two projects: 5GT and 5G-Coral. We have briefly went through their system architecture and the realization of the federation using the developed platforms.

In the next chapters, we re-use the same concept to showcase federation of NFV-NSs for static environments using the 5GT platform (Chapter 4) as well as for dynamic environments using Blockchain technology (Chapter 5).

4. FEDERATION IN NFV ENVIRONMENT

In this chapter, the static federation is presented through a vertical service deployment. A mission-critical service for eHealth, a multi-domain end-to-end service, shows how novel technologies (e.g., AR) combined with the federation feature and edge MEC deployment capabilities can potentially increase the life-savings by emergency services.

4.1. Introduction to eHealth in 5G

The new generation of mobile communications, 5G, is expected to bring significant improvements on many fronts: enhanced mobile broadband experiences to the end-user, ultra-reliable extremely low latencies to enable industry automation, autonomous driving, and massive machine-type communications, which will make the wireless Massive Internet of Things (MIoT) a reality. But in addition to these highlight-worth well-known use cases, there are many application areas that could benefit from 5G and associated technologies. One key example, with a clear and direct impact on society at large, is emergency services and healthcare.

Nowadays, emergency services depend on human intervention. A witness in the vicinity of the emergency will, luckily, start the emergency procedure described as follows: (*i*) the witness calls the emergency number (112 in Spain) and explains the situation and the location (this explanation is subjective and prone to errors based on the background of the witness, since there is no available data of the patient's condition, also the location is subjective and very often referred to geographical items di cult to locate, e.g., next to the bakery), (*ii*) the operator at the 112 call center assesses the situation and decides which is the most suitable emergency response team, and, (*iii*) the emergency team is deployed.

In the city of Madrid, this procedure takes around 4 minutes, while the time an ambulance takes to reach the location is estimated to be around 8 minutes (depending on distance and tra c)[6]. By analyzing the data provided by the Emergency services of Madrid, we realized that considerable improvements can be achieved in improving the e ciency of the emergency service by employing automatized detection systems.

The automatic detection of emergencies is an incipient business that will increase in the next years thanks to the new capacities in terms of massive connectivity of devices with low power consumption brought by 5G. The increasing trend on the use of smart wearable devices, together with the great advance in terms of portable medical monitoring and sensing, can be used to enable continuous monitoring of health parameters (e.g., heart rate, sugar blood level, blood pressure), and thus detect, and even predict, potential health

[6]

issues in a personalized way.

Additionally, the introduction of 5G brings more opportunities to improve the quality of care of the emergency team. New AR technologies allow for better treatments on-site, as well as to enable remote support from other medical teams, which may reduce the cost of the service (reduced emergency teams supported remotely). AR requires significant computing resources that cannot be pushed to the cloud, due to strict latency requirements. However, 5G capabilities in terms of low latency, rapid edge deployment, and high reliability can be used to make possible the use of AR services when treating an emergency situation. The 5G network also enables providing such service globally through service federation. In this way, Emergency Systems that are customers of a given 5G service provider can track and respond to emergencies of their patients everywhere by using resources of other providers.

With the goal of realizing an improved and automatized emergency service, this chapter proposes a design and realization of a 5G personalized health emergency system through the use of federation. It describes a real-life experience of the deployment of a system capable of detecting and responding to emergency situations in an automatic and personalized way while enriching the tools at hand of the emergency team by enabling the use of AR services at the location of the emergency. Thanks to the dynamic network reconfiguration and deploying NFV-NS in an external domain using federation. In short, the system is capable of patients' live-monitoring so that when an accident occurs (e.g., irregular heartbeat, which is a sign of a possible cardiac arrest), the system triggers an alarm to send an emergency team to the emergency location while the network is reconfigured with the deployment of a new network service in order to support the emergency team with AR services on-site. The presented scenario has been completely implemented and it has been validated by the emergency services of the Madrid Municipality. As further validated, the time needed to detect and process the emergency, select the best team and send it to the right location can be mostly eliminated by enabling an automatic and personalized emergency detection system, which also removes the need for the witness. This reduction translates into an increase in the patient's chances of surviving.

4.2. Related Work

5G will allow new kinds of health care services that are not feasible with the current capabilities of network technologies. In [68], the authors summarized the application of health care enabled by the capabilities of new mobile network technology. They divided them into mainly four categories, namely online consultation, online health monitoring, remote diagnosis, and mobile robot surgery (part of this work falls under the online health monitoring category). One of the challenges identified for this category was the high density of devices required to monitor a large part of the population. 5G is able to cope with the increasing number of chronic patients being monitored, as it introduces higher connection density, higher bandwidth, and lower latency with respect to the previous

network generation (i.e., 4G). In [69], the authors demonstrate the need for a 5G network (compared to the 4G network) to guarantee high e ciency and fast responses in a health monitoring scenario.

Di erent technologies combined with 5G networks aim to improve health services. The works in [70], [71] focus on applying deep learning and AI to enhance the performance in heterogeneous networks, also tackling how to enhance health services in [72]. Other works [73], [74] o er solutions to improve the security in healthcare systems, while some of them focus on multiple heterogeneous networks settings [75].

Edge computing is a key part of the 5G concept. The possibility of placing computational resources closer to the user contributes to providing the low latency required by health care applications while opening the door to new patient-centric applications. Both [76] and [77] make use of the concept of edge to provide patient care. In [76], a remote patient monitoring system makes use of edge resources to reduce the bandwidth needs of the telemetry system used to monitor the patients. using a variety of sensors and cameras. Also, it shows how using the edge ensures the real-time constraints of webRTC. In [77], the authors present a framework to assess voice disorders through deep learning processing at the edge. In our work, we use the edge for two di erentiated functions: *i)* deploying a network service implementing an AR supporting system, and *ii)* deploying a virtual local breakout point.

Recent researches show that AR has huge potential for applicability in the healthcare system, comprising user-environment interfaces, telemedicine, and education [78]. A key example is the possibility to show relevant patient health records on the head-mounted AR device without losing focus on the patient. In [79], authors developed a smart AR application that supports healthcare professionals with procedure documentation and patient information during wound treatments. In addition, they evaluated the interest of their work among healthcare professionals, who showed to prefer AR-based documentation systems with respect to the current documentation procedures (i.e., books, tablets, or smartphones). However, they only considered scenarios in which the AR application runs locally on the AR device.

Reference [80] evaluates the use of a particular AR device to assess its performance in a disaster scenario. Similar to the previous case, the AR application runs locally in the AR device, which additionally degrades the battery lifetime. In the study [81], AR devices are used to triage patients in a disaster event. In the study, the AR device operates in two modes (*i*) algorithm-assisted and (*ii*) with telemedical support from a remote professional. The results showed 90% of accurate triage. The study emphasizes the (WiFi) connectivity limitations, especially in the telemedical support case, and the low battery durability of the AR devices.

4.3. The scenario: 5G personalized health emergency system

In this section, we identify several areas where emergency services can be significantly improved by the use of new communication technologies enabled by next-generation mobile networks. On top of that, we present an eHealth emergency solution scenario, which we implemented as described in Section 4.4.

4.3.1. eHealth improvements for saving lives

eHealth is defined as the delivery of health services by means of information and communication technologies (ICT). The goal is to improve the information flow between the actors involved (e.g., patients, paramedics, hospitals, doctors, surgeons), supported by ICT. Mobile health, which is a component of eHealth, is defined as a medical health practice supported by wireless devices, including wearable medical devices, patient monitoring devices, and personal assistants [82]. Adults are becoming more concerned and take measures for continuous health monitoring by investing in mobile health devices [83]. With the clear goal in mind of exploring how ICT can enhance the emergency response services, we have group di erent improvement opportunities as follows:

Emergency response time and real-time data In addition to preventing cardiac arrests through constant monitoring, reducing an emergency team's response time and being able to reach the exact patient's location clearly contributes to saving more lives. For example, the reduction of the emergency response time significantly lowers the door-to-balloon time[7]. According to [84], the guidelines suggest a door-to-balloon time of fewer than 90 minutes. The study points out few e ective strategies in reducing the door-to-balloon time. Some of them are (*i*) having a single call to a central operator, and (*ii*) providing real-time data feedback to emergency and catheterization laboratory prior to arrival. A similar study [85] analogously concludes that the major delays are due to reaching the patient and moving her to the closest hospital.

Connectivity, durability and performance As mentioned, technologies such as AR, have been proven to improve emergency services [81]. However, the main requirements to make AR useful in emergency scenarios are: (*i*) to have a stable and high-bandwidth wireless connectivity, and (*ii*) long battery duration. Various works [86], [87] suggest that computational o oading at the Edge can reduce energy consumption by 90%, while improving the overall performance of the devices. On top of that, Ultra-Low Latency Communication (URLLC) slices in 5G networks are envisioned to address the connectivity requirements for AR [88], [89].

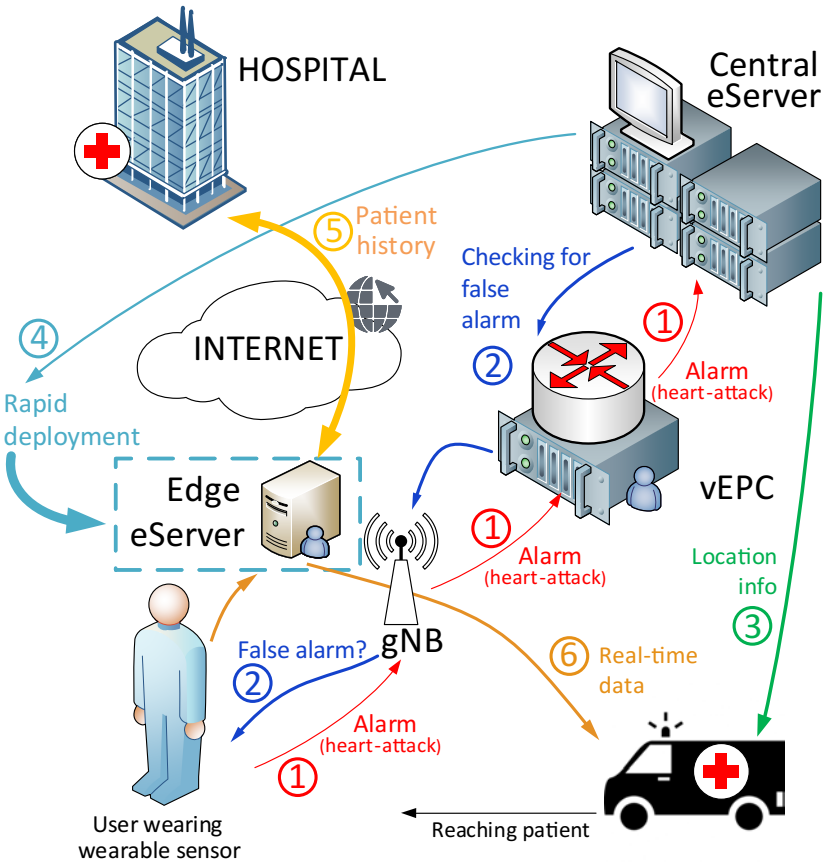


Figure 4.1: eHealth scenario

4.3.2. Scenario design

Taking as a starting point the improvement considerations described before, plus the constraints imposed by the way emergency response teams operate and how cellular networks work, we arrived at the scenario shown in Figure 4.1. The ultimate goal, identified as critical by the Madrid emergency response team, is to develop a fully automatic and personalized emergency response system.

To reach the goal, we set a simple scenario for which we designed the fully automatized system (explained in detail later). The scenario takes into account a continuously monitored patient for its vital signs (e.g., heart rate). Once the patient heart rate is abnormally low, an alarm is triggered which can be disregarded as a false alarm by the patient. If the patient does not mark the alarm as false within a short pre-configured amount of time, the system automatically dispatches an emergency team to the patient's location. In order to support the dispatched emergency team, the system deploys an AR system close to the patient's location.

To provide intensive health monitoring capabilities to an increasing part of the popu-

[7]The time period between the moment a patient with a possible acute heart-attack enters an Emergency Room and he she undergoes balloon angioplasty surgery.

lation, we rely on the 5G massive connectivity properties.

A smart wearable (e.g., a smartwatch) is used to monitor the health of a person. This device is able to detect potential health issues, such as low blood sugar incidents or, as in our testing case, a heart-attack. Although 5G will support direct communication of these wearable devices to a central cloud through a low power communication, for this early stage of 5G deployment, we can assume the wearable is connected to a mobile phone application which periodically reports the health status and the patient's location to a central cloud (Central eServer, step 1 in Figure 4.1). At the functional level, there is not much di erence between both solutions as far as the concepts presented in this article are concerned. If the monitored data reveals a potential (predicted) or actual health issue (e.g., heart rate down to zero), the Central eServer issues an alarm to the user mobile smartwatch or mobile phone (to check if it is a false alarm, step 2) while continuing the processing of the emergency event. This involves analyzing the health issue and the medical records of the person, deciding which might be the disease, and selecting the most appropriate team to deploy, considering both the time required to reach the location and skills that best address the emergency (e.g., a quick intervention medical vehicle, a regular ambulance, or the combined deployment of a firefighters team). The Central eServer automatically dispatches the selected emergency team (this can be canceled by the user at any time if the person notifies that it was a false alarm) to the location of the person (step 3). In this specific example, our system is able to deploy an AR service – for use of the emergency team – to improve the quality of care, by displaying geolocation and health information from the patient.

To provide high-performance, stable and durable AR service, the system requests the deployment of an emergency Edge eServer closer to the emergency location (step 4). This Edge eServer hosts the AR service helping the emergency team once deployed at the emergency location and may include patient health data used by the emergency team (step 5). This edge service is automatically deployed in a matter of a few minutes (with current technologies), while the emergency team reaches the indicated location. The deployed edge application establishes a connection to the emergency team and guides towards the location of the user by streaming an AR-marked pathway to the doctor's AR headset (step 6). The edge application also obtains the user's health records and live-streams them on the doctor's AR headset together with real-time sensor data from the user's wearable. The AR headset is also used to live stream video to a remote medical team that can provide specialized support (if needed). Thanks to this, the paramedics' team can significantly increase their e ciency (e.g., faster triage and provide real-time feedback to the hospital), thus lowering the door-to-balloon time and increasing the probability of saving people's lives.

The use of AR technology in emergency scenarios has been already proposed, but it is now thanks to 5G that it becomes feasible to actually consider its wide use by emergency response teams since AR requires very low latency between the AR device and the AR server. As it will be proved later in this article, previous mobile networks (i.e., 4G) do

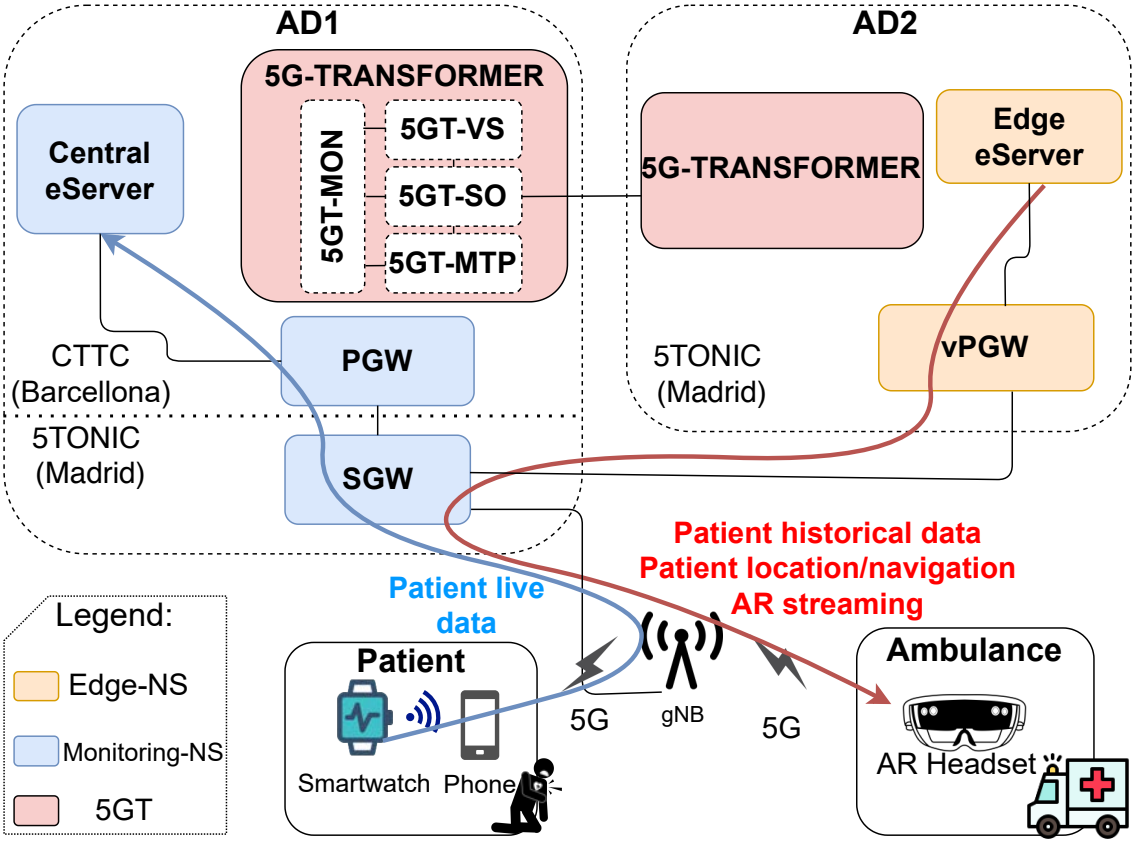


Figure 4.2: eHealth system deployment

not provide a latency low enough to guarantee good AR experiences. Our 5G-based solution is capable of dynamically instantiating an Edge eServer close to the location of the emergency and adapting the mobile network infrastructure to provide a low latency path to the newly deployed Edge eServer. In order to achieve this, a network service federation from di erent operators might be needed to satisfy the requirements of the AR service dedicated to the emergency case.

4.4. The solution: 5G-enabled personalized health emergency service

This section describes the technical solution enabling the scenario described before. First, we provide an overview of the used 5G vertical service orchestration platform. Then, a detailed description of the developed emergency and AR applications is provided.

4.4.1. Orchestrating Network Services in 5G networks: 5G-TRANSFORMER

As mentioned before (in Chapter 3), the 5GT architecture allows the deployment of network services spanning multiple ADs [90], known as Network Service Federation (NSF). This is possible thanks to the capabilities of the 5GT-SO to orchestrate composite NFV-NSs (composed of multiple nested NFV-NSs). The Network Service Federation (NSF)

feature is essential for the deployment of the 5G personalized ehealth emergency system when and where needed, as shown in [91]. Let us just consider a simple example: the emergency services of a city municipality have a contract with a 5G operator to provide the patient's monitoring and edge emergency NFV-NSs and the communication services used by all the emergency teams. This operator deploys with the 5GT platform most of its core network components and the monitoring NFV-NS (Monitoring-NS in Figure 4.2) in a remote cloud location because of operational reasons, e.g., not demanding latency constraints (AD1 in Figure 4.2). In case of an emergency, the Central eServer requests to the operator (by means of a query to the 5GT-VS) the instantiation of an edge emergency service (Edge-NS in Figure 4.2) connected to the monitoring NFV-NS close enough to the emergency location. A placement algorithm (in the 5GT-SO) [92]–[94] is in charge of deciding the placement of the Edge eServer based on (*i*) location constraints, (*ii*) information regarding the availability of local computation resources, and (*iii*) latency constraints of the AR application. The placement algorithm only computes the ideal Edge eServer location over the operator's local resources. However, there might be situations in which the operator does not have the infrastructure available in the proximity of the emergency location, requiring the use of infrastructure from a di erent operator o ering the edge emergency service (AD2 in Figure 4.2).

Hence, excluding the NSF feature, an operator to implement our proposed solution would (*i*) require dedicated infrastructure, (*ii*) incur additional costs, and (*iii*) su er from long implementation times. For example, a non-NSF implementation over the Madrid Municipality [93] (area of 8000 km^2) would require a dedicated infrastructure to satisfy the stringent AR latency requirements. That implies additional deployment & maintenance costs as well as longer implementation time.

4.4.2. Health monitoring and AR applications

In addition to the network services and their orchestration logic, we developed the three applications needed for the 5G-enabled personalized health emergency service: (*i*) the monitoring application providing the heart rate measurements and location of users, (*ii*) the server application processing the monitored information, deciding if an emergency team has to be deployed and selecting the best one considering di erent information (e.g., time to reach the emergency based on the location of available ambulances), and (*iii*) the AR service, required to compute and stream the information reproduced on the AR headset (guidance to the physical location of the person, collection and representation of the relevant medical information at each moment, and video streaming to a remote medical team to better excel the patient's triage).

The monitoring application is based on a smartwatch streaming heart rate data continuously to a 5G smartphone via ANT (Bluetooth could also be used). The smartphone is connected through 5G NSA to the 5GT system and continuously sends new data to the Central eServer (see Figure 4.1).

The Central eServer itself is continuously checking the state of the patient based on the information received, detecting (or even predicting) when an emergency occurs, and contacting the person (to detect potential false alarms). A false alarm occurrence is discussed in Section 4.6. The steps performed by the Central eServer while attending the emergency are: (*i*) contacting the closest ambulance using the legacy emergency location system of the Madrid municipality (based on a Global Positioning System (GPS) Fleet Navigation API), and (*ii*) triggering the instantiation of the edge emergency network service providing networking and computational resources required to support the emergency team upon their arrival to the patient's location.

The Edge eServer provides remote rendered AR VR video flow streamed through a 5G smartphone to the AR headset carried by the emergency team (we use Microsoft Hololens v1). The reason for this setup is the current lack in the market of AR headsets with 5G modems. Once the ambulance arrives at the emergency location, the Edge eServer starts streaming guidance information to the Hololens, indicating directions to reach the patient location. When the team reaches the patient, medical information is displayed on the Hololens. This information is selected based on its temporal relevance and availability (e.g., results from historical blood tests) aimed at facilitating the decision flow of the medical team. This leads to more organized patient transportation along with a feedback video streamed from the Hololens, thus enabling real-time remote support from other remote medical teams or specialists in nearby hospitals.

The monitoring application is implemented using Android studio, using ANT API to gather information from the wearable and REST services to request functions from and to push data to the server. The Central eServer runs behind Apache HTTP server and it consists of a set of REST APIs, functions developed in PHP and GPS Navigator (Tomtom) APIs are used to find and contact the closest ambulance to the patient location.

The AR application is developed using Unity 2019 and built as a Universal Windows Platform application, it receives patient and paramedics position using the GPS location of the 5G smartphones. The streaming from the Edge eServer to the Hololens is implemented with the Holographic Remoting API provided by Windows Mixed Reality Toolkit (MRTK) [95]. The Hololens receives the stream using the MRTK native application, Holographic Remoting Player. The AR navigation was implemented using Mapbox [96]. Note that we did not implement any additional algorithms to assist the triage, as mentioned in [81]. The Hololens itself is able to capture uplink real-time video stream, while a hospital team is able to push feedback augmented information in the Hololens, assisting the emergency medic in real-time.

4.5. Validation results

This section describes the experiments performed to assess the validity of our design and its usefulness for the emergency system of Madrid. To demonstrate the feasibility of

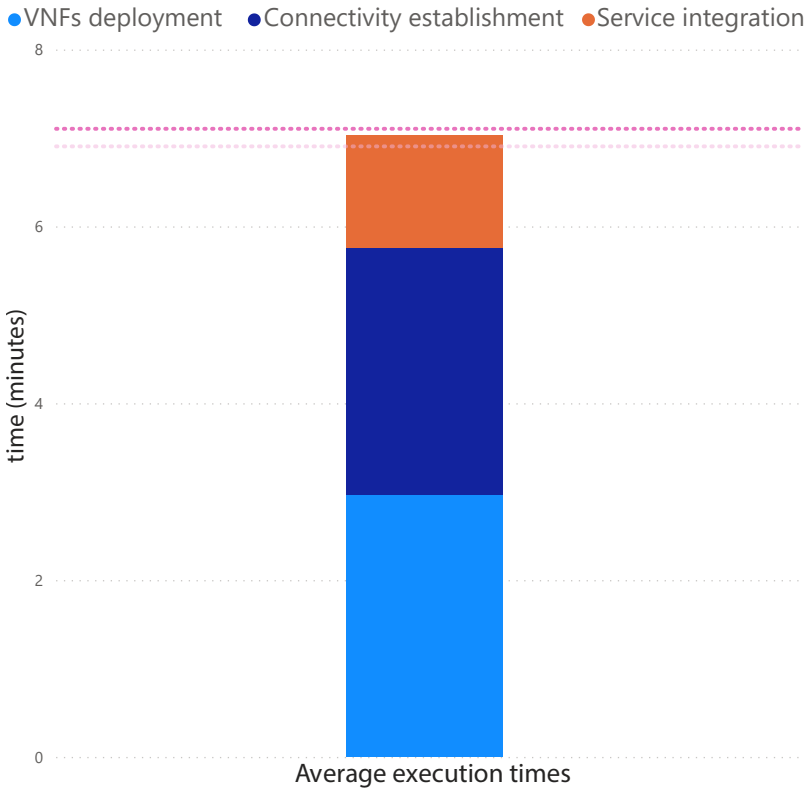


Figure 4.3: Service federation deployment time

the system, we deployed an end-to-end system as described in Figure 4.2, including a smartphone (Samsung S10) connected to a cellular 5G NSA network (provided by Ericsson BB630 baseband and Advance Antenna System AIR 6488) shown in Figure 4.5, the virtualized core network modules (implemented using the OpenEPC framework), a multi-domain 5G orchestration system (using the 5GT stack with one provider domain using Cloudify[8] and the other one OpenSource MANO[9] as coreMANO platforms) and the di erent applications required (both at the end-user and server sides). Validation was done by demonstrations drills involving real emergency response teams with ambulances, medical sta and firefighters.

The location of the emergency is the Institute IMDEA Networks, host of the 5TONIC lab. The network functions of the di erent involved network services were deployed at 5TONIC and also at Centre Tecnològic Telecomunicacions Catalunya (CTTC) premises in Barcelona, allowing us to resemble a scenario of a mobile operator providing services in Madrid, but having some of their core network entities in Barcelona (with a geographical distance of around 650 Kms). Following Figure 4.2, the RAN is deployed in 5TONIC, while the core network and eHealth Central eServer are deployed at CTTC. This geo-

[8]https: cloudify.co
[9]https: osm.etsi.org

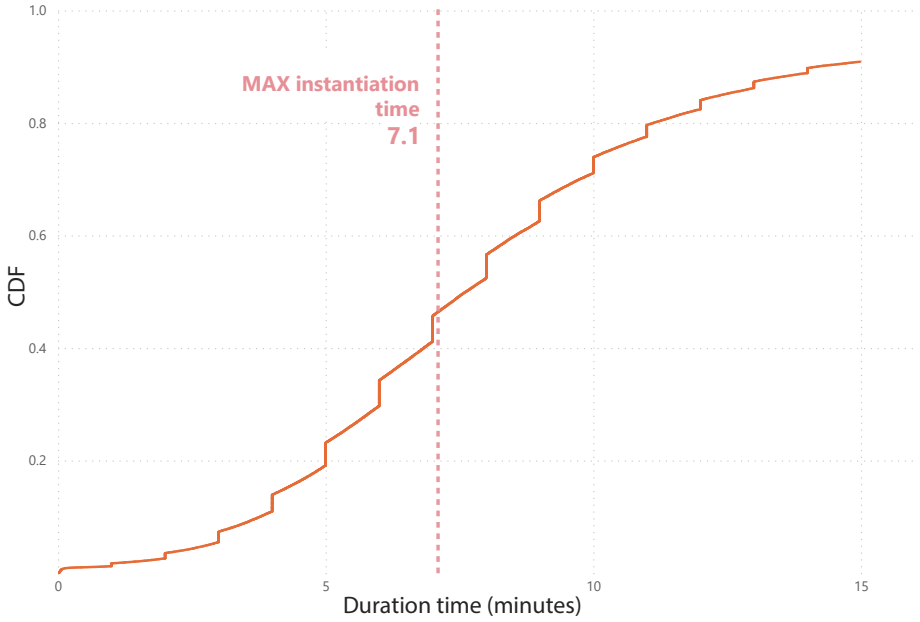


Figure 4.4: CDF of the duration from the moment that an emergency team accepts an emergency to the moment it reaches its location

graphical distance accounts for 10-15ms measured one-way delay in the communications between a UE and the Central eServer, which we will prove to be critical in order to deploy AR services.

The first step in our experimentation was to evaluate the service deployment time upon an emergency occurrence. More specifically the time it takes for the deployment of the Edge eServer at 5TONIC premises using the NSF. The bar chart in Figure 4.3 sums up the average time of all phases included in the Edge eServer deployment: (*i*) VNFs deployment, (*ii*) Connectivity establishment, and (*iii*) Service integration. The summed average deployment time is 7 minutes for a set of 10 deployments that we performed in 5TONIC.

To further evaluate the feasibility and usefulness of our design, we analyzed the duration of every emergency operation that occurred in the Madrid Municipality from 01 01 2019 to 31 12 2019. The dataset has been exclusively provided by the Emergency Services of the Madrid Municipality, excluding any patients' private information. The graph in Figure 4.4 represents the CDF of the duration of every emergency. The duration time for each emergency is measured from the moment the emergency team unit accepts the operation until it reaches the emergency location. Given the maximum deployment time of 7.1 minutes, there is a probability over 0.55 that the AR application is deployed and ready to be used by an emergency team upon arrival at an emergency site.

As mentioned before, the emergency average response time in Madrid is around 12 minutes, including around 4 minutes to issue the alert (receive the alarm and allocate the

Figure 4.5: 5G NSA radio in the 5TONIC lab

appropriate medical resources). The remaining portion is the time required to achieve the patient location shown with the CDF in Figure 4.4. In this context, the automatic detection of the emergency reduces the first 4 minutes almost to zero. Also, by the time the ambulance arrives at the emergency location, the Edge eServer will be up and running. This highly improves the response time which results in increasing the number of lives saved and reducing the number of side e ects of a stroke.

The next step in our experiment was to evaluate the delay in the connection between the Hololens device and the server performing the AR computation, considering 4G and 5G technologies and the availability of local computing resources through federation. Note that if 5TONIC (local) computing resources are available for federation, the AR service can be instantiated in the Edge eServer at the 5TONIC site. In other cases, the AR service can not be deployed close to the emergency location, and therefore the AR minimal latency requirements are not met, e.g., if the AR service is deployed in CTTC central location. Table 4.1 summarizes the average one-way delay (OWD) measurements in the di erent configurations. From the obtained results, it is clear that in order to achieve an optimal AR service we need a 5G network connectivity with the AR application deployed using federated service close to the emergency location. In the case of 4G with local federated service and 5G without federated local service, the measured latency is too close

Table 4.1: Average OWD of each scenario.

Technology	Latency
4G without federated local service	30 ms
4G with federated local service	18 ms
5G without federated local service	13 ms
5G with federated local service	5 ms
Wifi	2 ms

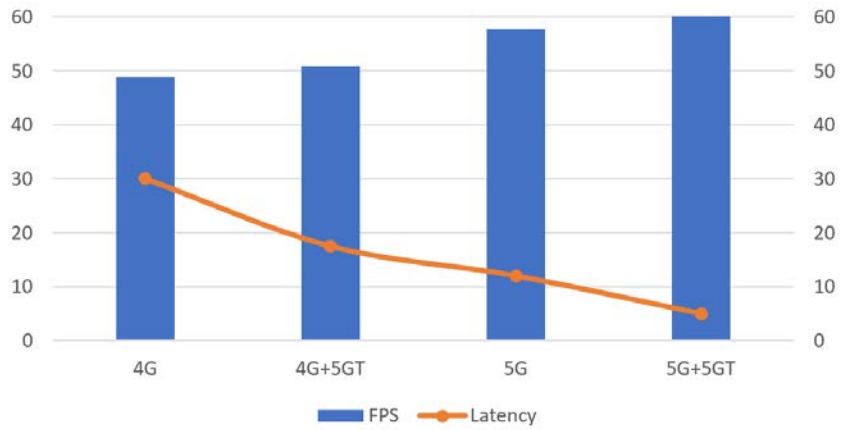


Figure 4.6: Average FPS achieved in the di erent scenarios.

to the minimal AR requirements.

For that reason, the final step in our experimentation has been to quantitatively assess the Quality of Experience (QoE) of the AR user, which is extremely sensitive to delay. To do so, we measured the average FPS achieved at the Hololens on the di erent streaming settings. For example, X frames per second are streamed from the AR application to the Hololens, however, due to latency and packet loss, only Y X can be reproduced into the Hololens goggles. To capture the e ects of latency on the FPS, we implemented an application module, using the diagnostic tool of MRTK, providing statistics about the actual frame rate sampled every half second. In this way, it was possible to analyze the average FPS achieved in the di erent streaming settings. Also, considering the caching, the frame prediction, and optimization of the Hololens, we performed our tests for a short time while moving into the AR world. In this way, it was possible to appreciate the real FPS experienced by the user. Obtained results are shown in Figure 4.6, where we denote the availability of local infrastructure for federated service with the label 5GT. It can be concluded that 5G is a clear must to have to achieve the best performance (being 60 FPS is the maximum frame-rate achievable by currently available AR devices).

Although the di erence between 50 FPS and 60 FPS may seem insignificant for the reader, it is important to highlight that FPSs are critical for AR applications. Any difference in the FPSs makes a huge di erence in the experience of the user since head's

Figure 4.7: The object misplacement in the 4G without local edge scenario (top-left), a correct object placement in the 5G with local edge scenario (top-right), sanitary sta wearing the Hololens (bottom-left), patient health report shown on the Hololens (bottom-right).

movement tracking introduces a latency which is clearly visible for FPSs below 60.

Not only frame rate is a ected by latency, but another problem that may occur is also object misplacement. Indeed, more than 10ms of delay leads to object misplacement of at least three degrees [97] in mobile AR. Figure 4.7 highlights the misplacement of 3D objects in the real world experience in the 4G scenario, compared to the correct position of the objects in the 5G one. The arrival point indicator is moved by various degrees from the original position in the test scenario using 4G without local edge (top-left), while in the best scenario, using 5G with a local edge, the indicator is placed correctly in the real world (top-right). Thus, enforcing the need for the proposed solution for mobile AR applications.

4.6. Lesson Learned

This section lists the lessons learned during the implementation, integration, and deployment of the end-to-end eHealth system and network service development. We divided

them by application-related and network-related.

4.6.1. Application-Related Lessons

Do not always trust your phone's Geo-location. The GPS location of smartphones has very high variability. In our work, an average of over ten samples of the coordinates was used to stabilize it. Hopefully, 5G will bring a more e cient localization mechanism.

Always synchronize the orientation. The AR headset used in this work often lost its orientation when operating in dark environments. As consequence objects are misplaced in the real world by various degrees. There is a need for a fast and continuous synchronization mechanism of orientation tracking at the application level.

Choose carefully your Graphics Processing Unit (GPU). The remote rendering of the AR scene needs a server (Edge eServer) with powerful GPUs to stream the AR experience to the AR headset in real-time. In order to dynamically instantiate the Edge eServer, the GPU must be virtualized, a feature not supported by every GPU (including recent ones).

Need for AR feedback on AR stream reception. We experienced that in some cases, when the latency is too high (over 100ms) and the bandwidth is too low (less than 8Mbps) the device does not get any AR input, without the server getting any notification error. This needs to be improved to make the system more reliable.

False-alarms. The occurrence of false alarms is a common and well-studied problem. However, there is no clear solution to approach it [98]. According to [99] almost 25% of the health emergency calls are false alarms, which avoiding them can produce immense savings. There are some models that can reduce these unnecessary calls [100], however, the authors warn that focusing on minimizing the false alarms can lead to an increase of more severe outcomes, even deaths. In our view, employing a limited timer to signal a false alarm (so once it expires, the emergency team is dispatched) could be su cient to lower the number of false alarms without increasing the patients' risk.

4.6.2. Network-Related Lessons

Some VNFs require function-specific management and platform adaptations. To exploit the capabilities of the 5GT platform, network services and VNFs deployed on top of the 5GT infrastructure might need to be adapted. Although this goes a bit against the virtualization principle of services and functions to be platform agnostic, as of today many virtualized functions (such as OpenEPC) have been designed with

some platform assumptions in mind. In our tests, we had to deal with some specific VNF configurations in order to be able to instantiate and manage services over the 5GT platform.

VNFs are not just virtual machine images. As an example, the EPC software used imposes a rigid IP and MAC addressing scheme, hindering its deployment in generic scenarios. This may prevent the use of this EPC stack in interoperable scenarios, where VNFs may be provided by di erent VNF vendors or pose di culties to interact with physical equipment Physical Network Functions (PNFs), like the RAN component. Additionally, the design constraints of the associated VNFs required the introduction of ad-hoc operations to e ectively enable the connection among the PNFs and VNFs, because some interactions could not be captured in the associated Network Service Descriptors (NSDs).

NSF helps in ubiquitous emergency handling. As already mentioned in section 4.4.1, the NSF feature is an instrumental feature to enable the AR low latency requirements, and it provides an extended (emergency) service geo-coverage while omitting the need of exclusive resources. This is proven through the validation results we obtained in section 4.5.

4.7. Remarks for eHealth scenario and federation in NFV static environment

Nowadays, it is quite normal to find wearable devices capable of tracking sleep patterns, monitoring the heart rate, measuring the number of steps, or even perform an electrocardiogram. People with chronic diseases are taking these measurement devices to the next level, with patches able to measure glucose levels, connected insulin pumps, or wearable blood pressure meters. These devices will be complemented and augmented in 5G, one of its main characteristics being the focus on massive machine-type communications. Therefore, we expect all these devices to be connected to eHealth services, provided by public or private companies, which will perform continuous monitoring in order to detect anomalies and act upon them.

We have departed from this assumption and implemented a real use case showcasing the impact of 5G in the emergency service of Madrid. The deployed service aims at improving the quality of care in two ways: (*i*) reducing the time required to detect an emergency, while removing the variable of a human witness, (*ii*) providing location and health-related data to the emergency team for increased triage e ciency through augmented reality deployed in the field, and deliver real-time data back to hospitals to ameliorate surgery preparations.

These two services leverage the new capabilities of 5G, including not only the high-bandwidth and low-latency connectivity, but also the orchestration, federation, and dynamic instantiation of virtual functions at the edge of the network. The federation in

static environment is convenient and useful for mission-critical services that depend on the service deployment times.

The use case was tested and validated by a real emergency team, showing that decreasing the response time by 30% is possible. Since every second is relevant when responding to emergencies, we believe the designed use case showcases and exemplifies the future evolution of emergency services.

5. FEDERATION IN DYNAMIC ENVIRONMENTS USING BLOCKCHAIN

5.1. Motivation

In the previous chapter, the federation in static environment is described paying special attention to how to extend the NFV MANO stack to enable federation of services and resources. While there minor di erences in the definition of federation for static and dynamic environments, a common assumption is a presence of a pre-established agreement among service providers. Settling an agreement is time-consuming and suitable only for static environment.

The goal in this chapter is to present how Blockchain technology can complement NFV MANO service providers to accomplish federation in dynamic scenarios.

First, this chapter provides description of the DLT, specifically Blockchain: how it works, the consensus mechanisms, and how is used by the vertical industries. Later, we describe how the Blockchain is applied for federation in dynamic scenarios to (*i*) enable NFV-NS federation and healing; (*ii*) and in real-case scenario providing federation in Edge Robotics.

5.2. DLT and Blockchain

What is a DLT? A distributed ledger is a type of distributed database that by default assumes presence of malicious nodes. The DLT enables the realization of distributed ledgers through a shared consensus mechanism to establish immutable records of transactions despite failures [101].

What is Blockchain? Blockchain is a DLT realization that enables creation of cryptographically linked and chronologically ordered blocks, containing a certain number of transactions. Bitcoin is the first Blockchain, designed as a public, immutable, append-only, distributed ledger.

Blockchain is regarded as a disruptive powerful technology that has potential to radically reshape the society and the world economy through decentralized governing structures [102], [103]. The Blockchain idea is captivating because for the first time in human history people from distant locations can securely transact within a massive peer-to-peer network with decentralized distributed management (i.e., no central authority).

According to [104]–[106], Blockchain is going to be the driving force for the next generation of Internet (i.e. 5G and 6G) and network slicing is fundamental part of it. To

fully elaborate the Blockchain as a DLT integration with network slicing in later sections, this section first presents the Blockchain's history, fundamentals, taxonomy, consensus mechanisms. Later, we unfold the application smart contracts and the Distributed Applications (DApp) paradigm. Finally, we go through the leading openly available platforms.

5.2.1. History of Blockchain: An Overview

In 2009, after the Financial Crisis of 2008 [107], Satoshi Nakamoto published the Bitcoin paper [108]. Despite the initial idea of creating an open source peer-to-peer electronic cash system that would avoid double-spending attacks, the outcome produced a disruptive technology [109]. Satoshi Nakamoto combined encryption and distributed computing in a unique way to assist a network of computers in collaborating towards maintaining a shared and secured database. Nakamoto generated the genesis block and mined the initial bitcoins, giving birth to the cryptocurrency era. Satoshi Nakamoto is a pseudonym for the person or group of people that design and built the Bitcoin. The identity of Satoshi is a mystery to date [110]–[112].

Bitcoin's popularity began to increase in 2011. Soon, technologists realized that Blockchains could be used to track other things besides money. In 2013, 19-year-old *Vitalik Buterin* proposed Ethereum. The idea of smart contracts was initially introduced by Nick Szabo [113]. This marks a new milestone in the evolution of Blockchain technology, often referred to as Blockchain 2.0 [114].

5.2.2. Fundamentals of Blockchain and its Working Principle

The key strengths of Blockchain are founded on its verifiability and tamper-proofness. To understand how Blockchain achieves its key characteristics, in this section we describe its building blocks and how the Blockchain works.

Blockchain building blocks

The main components to implement a Blockchain are:

Peer-to-peer network: A Blockchain is constituted by *Blockchain nodes* that are inter-connected in a peer-to-peer network. When a new Blockchain node is setup and initiated, first connects to the peer-to-peer network, and once it has established a connection to at least one node, it starts the syncing process. This consists of downloading all the blocks of the Blockchain, till the latest block. Once a node is in full-sync, it can actively participate in the Blockchain.
The Peer-to-Peer network is critical for Blockchain technology, as a base layer (similar to IP layer for Internet). In a *centralized* system, there is a high risk of single-point failures (SPOF) or denial of service cyber-attacks [115]. In a Blockchain in-

stead there is no central authority to set the rules making it a *decentralized* network. Information is continuously recorded in append-only fashion, and an identical copy is transferred and stored between the nodes.

Blockchain address: Each user of the Blockchain needs a unique Blockchain address. A Blockchain address is a password protected and has asymmetric keys (private and public key-pair). Users issue and authorize transactions by signing them with the private key. The public key is used for receiving transactions. More precisely, the Blockchain address represents a hash (SHA-256) of the public key. In Bitcoin, a pay-to-public-key-hash (P2PKH) script is used, where the Bitcoin address is a unique 27-34 alphanumeric characters long hash identifier [116].

Transaction: Every transaction is a new and unique record exchanging value or data between two Blockchain addresses or entities. It has an origin and recipient Blockchain address. The issued transaction is added to a pool of unconfirmed transactions - a collection of signed transactions ready to be added in a block [117].

Block: A block is a structured collection of multiple transactions. Each block contains a block header and a list of transactions. The block header contains: (*i*) a hash of the previous block, (*ii*) a hash of all listed transactions in the block, (*iii*) a nonce, (*iv*) a timestamp, (*v*) the di culty, as explained in detail below. The list of transactions in a new block is populated from the pool of unconfirmed transactions. The miner is in charge of the process of block creation, and blocks are appended to the Blockchain after consensus is achieved, as it will be better described later. It is important to note that participants can explore the Blockchain data transactions back in time to the genesis block (Block 0) thanks to the hash of the previous block. In this way each block points back to the preceding block creating a chain of blocks.

Consensus mechanism: To append a new generated block to a Blockchain a miner needs to follow a consensus mechanism. This is a key procedure that enables immutability, security, and integrity to a Blockchain. A consensus mechanism includes a diversity of advanced cryptographic techniques and mathematical models that define a strict procedure for (*i*) generating the necessary block headers, and (*ii*) validating the new block. The consensus mechanism is run by all (peer-to-peer) nodes participating in a Blockchain network [118]. Satoshi Nakamoto proposed a Proof-of-Work (PoW) consensus mechanism to regulate nodes participants in Bitcoin [108]. The consensus algorithms dictate the overall performance of a Blockchain.

Hashing and hash functions: A hash function takes any (data) input and produces a finite output of a specific size. The process of applying a hash function to data is called *hashing*, and the output of a hash function is called a *hash*. The essential feature of a particular hash function is the size of the output it produces. Essential for preserving structured, manageable and secure Blockchain data is through a

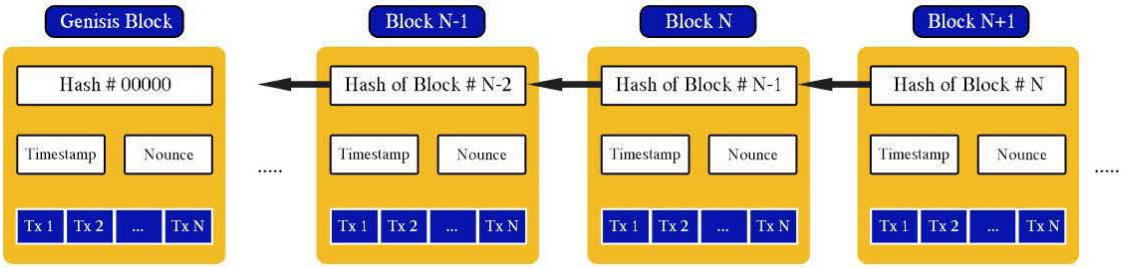


Figure 5.1: Blocks are chained together using the previous block's hash to form a Blockchain

hash algorithm with a data structure known as a *Merkle Tree*. This is a method to structure data that enables a large body of information to be verified accurately and e ciently [119].

Timestamp: Each block in Blockchain is timestamped. Timestamps prove chronological order of blocks and transactions, representing the time of each recorded transaction. These tamper-proof timestamps serve as a notary service that prevent occurrence of double-spending transactions [120].

Nonce: It is the number that a miner node has to *guess* in order to successfully *mine* a block. It is mainly used in a PoW-based Blockchains, such as Bitcoin. A nonce is an arbitrary whole number, which is 4 bytes field. The combined hash of the desired Nonce and the block header of a new block should produce a result with leading zeros, depending on the di culty. For example, if the di culty is 1, the combined hash *(block header nonce)* should produce a result of single zero leading hash (0x0...). In case that the di culty was 2, the combined hash should be double zero leading hash (0x00...), and so on. Thus this result is easy verifiable by the rest of miners, running the consensus algorithm. The found hash is added to the hashed block [120].

Smart contracts: At the most basic level, smart contracts are programs that run independently on top of a Blockchain. They have been introduced by Nick Szabo [113] and contain immutable deterministic code, the creator's Blockchain address and cannot be modified by anybody, not even by their creator. The benefits of smart contracts are most apparent in business collaborations, in which they are typically used to enforce some agreement so that all participants can be sure of the outcome without any intermediary's involvement [121]. This concept is essential for designing frameworks or distributed applications.

How Blockchain works

Since we introduced the basic building blocks, in the rest of the section we focus on how a Blockchain generates a new block, and how the new block is appended or *mined* .

How blocks are created Figure 5.1 shows how Blocks are chained together and the information they contain. The figure represents a chain of three blocks.The first block is di erent as it can not contain the previous block's hash, and is called the *Genesis block*. Every Blockchain is instantiated or starts with a genesis block. A genesis block is created or mined by a single node, usually the node of the Blockchain's creator.

Once a genesis block is created, all nodes of the Blockchain start to *compete* for a block creation. The rules of the competition are defined by the consensus mechanism. A Bitcoin block creation, can be summarized as follows:

A node collects limited number of transactions from the pool of (pending) transactions

A node populates all the necessary block headers, especially the hash pointer to a previous block and the hash of all included transactions (or the Merkle root).

A node competes to win the consensus. If it wins, the generated block is appended to the Blockchain. In case it does not win the consensus, the transactions are released (or unlocked) back into the pending transactions pool.

Tampering the information in the second or any of the following blocks (in Figure 5.1), modifies the resulting hash. As a consequence, there would be no match in the following blocks, making all the subsequent blocks invalid. As a result, all nodes in the Blockchain can not validate the modified block and discard it. An attacker can only succeed if it controls at least 51% of nodes in the Blockchain network.

The data that is stored inside a block depends on the type of Blockchain. For instance, in Bitcoin, a transaction contains: *Sender A* sends bitcoins to *Receiver B*. Hence the transaction data consists of information regarding the sender, the receiver, and the amount of transferred bitcoins (tokens). Note that Bitcoin-capitalized refers to the first Blockchain technology created by Satoshi Nakamoto [108]. While bitcoin-lowercase refers to the token or (cryptocurrency) used to transfer di erent amounts between users.

The continuous creation of new blocks in Bitcoin using the PoW consensus mechanism is called *mining*.

How mining works: The active nodes in a Blockchain such as Bitcoin are referred as *miners*. They are accountants which record every transaction to the Blockchain. Mining involves creating a hash of a block of transactions that can not be easily forged, protecting

the entire Blockchain's integrity without the need for a central system [122]. From a high-level (user) perspective, the concept is simple; a proof of payment is essential if a person wants the payment to be valid. The miners are the ones who keep the record of all the payments. Mining is typically done on a dedicated computer [123], as it requires a fast CPU and higher electricity usage, and more heat generated than typical computer operations [122].

To *mine* a block, the miner collects a batch of transactions, creates a block and generates all block headers, as mentioned previously. The last step for the miner is to guess or find the proper nonce. The *mining* process is a simple brute-force generation of random nonce. The right nonce hashed with the block header hash should produce a result with a specific number of leading zeros. The *mining di culty* or the number of expected leading zeros is modified by the consensus algorithm. In this way the consensus algorithm can control the block creation time when new powerful computing devices are joining the Blockchain network as miners. For example, in Bitcoin the block creation time is around 10 minutes, and in Ethereum is around 13 seconds [124].

Once the miner brute-forced a proper nonce, records it in the block header and broadcasts the block on the Blockchain network. Note that multiple miners may generate a block at the same time, but only a single block is elected as the winning block that is appended to the Blockchain. The winning block is the block that is first validated by at least 51% of the miners nodes in the Blockchain network [108].

The miner that *mined* the winning block is awarded with bitcoins to the miner's coinbase address. The amount of bitcoins or the *mining reward* depends on the block height. The mining reward is reduced by half every 210 000 blocks. For example, on 11th of May 2020 for the 629 999 block, the miner received 12.5 bitcoins, whereas for the next block (630 000), the miner received 6.25 bitcoins. The reduction of mining reward for Bitcoin is known as *bitcoin halving* [125]. According to calculations, it is expected miners to receive rewards up until year 2140 [125].

5.2.3. Taxonomy of Blockchain

Di erent types of Blockchain are available. We focus on the three major types: (i) public, (ii) private, and (iii) consortium. We take a closer look at each of them, discussing their features and mapping them on Table 5.1.

Public Permissionless Blockchain

Public Blockchains are highly decentralized, are accessible to everyone and rely on active network nodes. The first Blockchain in the form of Bitcoin, created in 2009 by Satoshi Nakamoto [109], it is a public Permissionless Blockchain. Facilitating auditability is one of the benefits of using Blockchain technology and permissionless Blockchain allows

public auditability. Nowadays, most public Blockchains run PoW consensus mechanism to maintain trust, immutability and security. To encourage users in participating as active nodes (e.g., miners in Bitcoin or Ethereum), the network rewards block creators with a finite amount of tokens (e.g., bitcoins, ethers) for each block created.

An utterly public Blockchain with open-source community models is designed to leverage expertise from many diverse people worldwide and use a broad-ranging user base to have supreme decentralization. Public Blockchains are criticized for the vast amount of computational power required to support a distributed ledger at a massive scale. Other concerns are associated to the transaction approval frequency and to the confirmation delay [126]. The performance of other consensus than PoW, like Delegated Proof-of-Stake (DPoS) or Proof-of-Staked Authority (PoSA), running on public Blockchains is significantly higher. For example, they produce 1 block every second, compared to 1 block every 10 minutes [127] provided by PoW.

Private Permissioned Blockchain

Private Blockchain or permissioned Blockchains are only accessible by a limited number of admitted participants as it follows a partial decentralization technique. A private Blockchain has a organization entity (e.g., the Blockchain creator or several members) which manages the Blockchain. Every new user requires an access invitation issued by the governmental entity. Frequently, enterprises or companies deploy private permissioned Blockchains. In this way they are able to define specific access and operating constraints to the user, making the auditability restricted. Enterprises or companies using private Blockchain can keep the autonomy limited. Additionally, the private Blockchains come with the possibility of immutability. Implicitly, these systems are not highly centralized, and often employ less computational demanding consensus mechanism (e.g., Proof-of-Stake), allowing for higher transaction throughput or more frequent block creation [128], [129], which leads to better performance compared to public Blockchain. [129].

Federated Consortium

A federated or consortium Blockchain is a permissioned and group-owned system where individual autonomy is removed, and instead, permissions are vested in a group of companies or individuals. In other words, the consortium Blockchain is a system that is *semi-private* and has a controlled user group (as in a company); however, it works beyond various organizations. Moreover, consortium Blockchain vs. private Blockchain is a sweet-spot between fully open, decentralized and fully centrally-controlled systems. There is more likely to be a trusted consensus, as multiple organizations have a stake in the outcome [130]. Consortium Blockchains have restricted audibility and only selected nodes have autonomy to validate new blocks, which makes them not completely immutable. Moreover, the transaction approval frequency is shorter than that of public

Table 5.1: Taxonomy of Blockchain

PropertyType	Public [126]	Consortium [131]	Private [130]
Decentralization	Yes	Partial	No
Auditability	Public	Public and restricted	Public and restricted
Autonomy	All nodes	Selected nodes	One organization
Immutability	Nearly impossible	Possibility	Possibility
Transaction approval frequency	Long	Short	Short
Performance	Low	High	High

Blockchain and o ers a higher performance level [131].

In conclusion, federated consortium Blockchain o ers the same benefits provided by private Blockchain: productivity and privacy of transactions. However, it gives the combined advantage of separating the consolidation of power only to a single company. This realization of a Blockchain network is ideal for an organizational collaboration.

Table 5.1 summarizes the type of decentralization, suitability, autonomy, immutability, transaction approval frequency, and overall performance.

5.2.4. Consensus mechanisms in Blockchain

To achieve our goal of comparing how di erent consensus mechanisms can influence the performance of NSF, we decided to compared them over a simple scenario. The experimental scenario and evaluation are further explained. In this section we are describing the consensus mechanisms that we are going to compare. Each of these consensus mechanisms are implemented in a platform that can be deployed and used for experimentation. For better description, we are coupling the description of the consensus mechanism with each of the platforms.

Proof-of-Work (PoW) - Ethereum

Proof of Work is the fist consensus mechanism implemented in the first Blockchain implementation - Bitcoin [108]. The same consensus mechanism is used for Ethereum, the first Blockchain platform supporting smart contracts.

The PoW consists of generation and validation of a new block. The process of generation a new block is when a Blockchain node (i) collects finite number of pending transactions to form a block. The transactions are hashed to form Merkle tree, the Merkle root is added in the block header. The Merkle root along with a timestamp, hash of the previously confirmed block, transaction count and nounce are added in the block header. In order the block to be valid, a node needs to compute a hash of a nounce that would produce a SHA-256 number with a defined number of leading zeros. Producing a SHA-256 hash with leading zeros is computational intensive puzzle-solving work. The number of leading zeros represents the di culty of the consensus mechanism. This is fundamental feature that allows the Blockchain to adapt the mi*mining* di culty when nodes with extra computational capability join the Blockchain network. Once the mining node successfully gets solves the puzzle and produces valid nounce, it broadcasts the created block into the network. Rest of the nodes can easily validate the result by simple hash of the nounce and the block header to produce the resulting block hash. The validated block is appended to the Blockchain ledger, and new round of block creation starts.

The di culty of the consensus mechanism is also adjusted to maintain the block time - time it takes to append a new block in the ledger. In Bitcoin, the block time is 10 minutes while in Ethereum it is 14 seconds. On average Ethereum is producing 15 transactions per second [132].

Besides the consensus mechanism, it is important to note that Ethereum implements smart contracts (introduced by Nick Szabo [113]) on top of Ethereum Virtual Machine (EVM) [133]. The EVM is a near Turing-complete on top of which the smart contracts are executed. Smart contracts contain set of rules functions stored at specific account address in a form of a bytecode. Users use accounts to issue transactions to other users, or to smart contracts. When a user makes a message call to a smart contract, the bytecode is executed, and returns a result, changes a state, etc.

Proof-of-Authority (PoA) - Ethereum

In 2017, as a consequence of a spam attack to the Ethereuem test network - Ropsten, a new test network was deployed using Proof of Authority (PoA) consensus mechanism [134]. The PoA consensus was proposed in the EIP-225 and later implemented in the Clique proof of authority protocol [135]. The new protocol is maintaining the block structure as in PoW Ethereuem, however instead of mining nodes competing to solve a di cult puzzle, there are pre-elected authorized signer nodes that can generate new blocks at any time. Each new block is endorsed by the list of signers and the last signer node is responsible for populating the new block with transactions. The transaction reward for each new block created is shared between all the signers [136].

The Ethereum PoA permissionless test network - Kovan, has been released with the initial validators assigned to 12 independent public notaries with active commission license [137]. In our experimental scenario, we are using a private instantiation of the

Clique Ethereuem network which is explained further in details. The performance of PoA Blockchains depends on the number of signers. In private chains, the performance can reach 70 transactions per second [138].

Practical Byzantine-Fault Tolerant (PBFT) - Tendermint

The Byzantine-Fault Tolerant consensus mechanism is based on a property of a system that can resist the failures derived from the *Byzantine Generals' Problem* [139]. The main characteristic of a BFT system is the ability of continuous nominal operation even if some of the participating nodes fail or act maliciously. When applied to a Blockchain realization, it has the ability to rule out validations from malicious nodes [140].

Practical BFT aims for Blockchain with high performance (e.g., high transactional throughput, low latency, etc.), and high execution time. PBFT nodes of a permissioned Blockchain are sequentially ordered and all permitted nodes assist in attaining a consensus. The PBFT Blockchain is able to maintain the consensus if the maximum number of malicious nodes is not more than a third of all the participating Blockchain nodes. The Blockchain security increases with the increase of participating nodes.

Tendermint is an application-based Blockchain with a default Byzantine Fault-tolerant (BFT) consensus [141], [142]. Tendermint enables users to turn any deterministic application into a Blockchain application through the use of the Tendermint BFT state-machine replication. Simplified, an application (as a state-machine) needs to be adapted to use an Application BlockChain Interface (ABCI) in order to communicate any state-transitions in form of transactions to the Tendermint Blockchain. On run-time, the Tendermint BFT consensus handles the state transitions by recording them into blocks of transactions. The state transistions are then replicated in each of the Blockchain nodes that run the same application. Hence, each application would run its own Blockchain (network) making the Tendermint an application-based Blockchain.

Unlike Bitcoin, blocks in Tendermint are added through voting by validators or validator nodes. The validators depend on how they are set. This can define if the set Tendermint network would be public or private. On top of that, a Proof-of-Stake (PoS) consensus can be employed. In that case, validators are user accounts nodes that lock coins in a bond deposit transaction. In return, the validators gain voting power equal to the amount of bonded coins. In all cases, a block is validated and added to the Tendermint Blockchain when 2 3 of the voting power has signed and committed the block. Thus even if 1 3 of the validators fail, the Tendermint is still generating new blocks. Additionally users can run full-nodes or light nodes (suitable for IoT applications). A block is added in three rounds: (*i*) Proposal, (*ii*) Prevote and (*iii*) Precommit.

Tenderimint is a high performance Blockchain which can handle maximum 10^4 transactions per second [143] with an average block latency of one second.

Proof-of-Stake (PoS) - Cosmos

Proof of Stake consensus Blockchain is based on a Blockchain network of nodes that generate and validate new blocks di erently than solving a complex puzzle as a proof of work. A PoS validator can generate (mint) or validate a new block with a probability equal to the Blockchain tokens coins it holds. In PoS Blockchains the competition to generate a block is minimized compared to the PoW Blockchains. The node that generates the subsequent block is randomly chosen in a pseudo-random-selection process based on a combination of various Blockchain specific variables or processes (e.g., token staking) [144].

Blockchain nodes that compete in the block generation process need to secure and lock, a certain number of coins into the network as their stake. The size of the stake provides is linear to the probability of a node to be elected as the next-block validator to produce the subsequent block - the bigger the stake, the higher the chances [145]. PoS is considered as not fully decentralized Blockchain mechanism with high scalability, 50% fault tolerance and relatively high transaction throughput.

Cosmos is a network of many Tendermint Blockchains that are joined in a single Blockchain with a global transaction ordering [146]. Considered as an upgrade of the Tendermint with a goal of enabling inter-operability between di erent applications realized as Tenderemint Blockchains. The mechanism for enabling the inter-communication is referred as Inter-Blockchain Communication (IBC). A first public Cosmos Blockchain is the Cosmos Hub which serves as a central ledger for multiple Zones or Tendermint Blockchains. The Cosmos Hub is PoS based and it has its own cryptocurrency - Atom. Users can stake Atoms to become validators or delegate their Atoms to trusted validator in order to earn portion from transaction fees. To maintain performance, there are limited amount of validators (e.g., up to 100 in the first year). Cosmos inherits the Tendermint performance and it is useful for connecting di erent Blockchains [147] or realization of specific use-cases such as Decentralized Exchange (DEX) [148].

5.2.5. Application of DLT Blockchain in Verticals

In this section, we analyze a number of vertical industries focuses on how the Blockchain as a technology is currently used to improve their bussiness logic, to which the architectural blocks explained in the previous section are somewhat agnostic. At the end of the section we explore how some works have already integrated a Blockchain solution into a network slice. In our view, the application of network slicing with Blockchain should append and improve the current solutions. The goal is the reader to understand how specific vertical related problems can be solved with a Blockchain technology which most of them can be implemented within a vertical network slice.

Media and Entertainment: The emergence of the Blockchain technology is significantly a ecting the *media and entertainment*. The Blockchain brings novelty in the

media and entertainment eco-system. It provides added value to media publishers and content creators thus shifting the economical benefits more towards the copyright-owners (e.g., the creator can be the copyright-owner of the content, or the publisher has the full ownership) [149]. The impact is measured as disruptive and sustainable [150]. The micropayment channels [151] disrupt the configuration of the ecosystem by allowing content providers and aggregators to be bypassed and shift the power to content creators. Each art piece, song or movie is published on Blockchain-based platforms by the creators owners and directly sold to the consumers. This disruptive concept referred as one-stop shop model enhances the relationships between the content creators and the consumers. Through the application of Smart Contracts, each created content can be tokenized and its ownership fairly distributed [152]. The distribution of royalty payments is automatized and fairly distributed to each musician. An exemplary platform is Steemit - a DPoS consensus [153] Blockchain-based social network. Steemit rewards content creators with a digital currency (called "Steem") based on the popularity of their posts. The platform is based on several principles, where the most important one being that everyone who contributes to a venture should receive pro-rata ownership, payment or debt from the venture.

The protection of intellectual property is another example where the application of Blockchain enhances the copyright protection [154]. The created content can be tracked, protected from piracy, and the Blockchain allows a customized way for creators to manage sharing rights through the use of Smart Contracts [155]. The proof of ownership is recorded on-chain through time-stamping and hashing the content, so that this allows news media to prevent the spread of fake news [156]. In the gaming industry, the in-game assets are registered on public Blockchain (e.g., Bitcoin, BitCrystals [157]). Users can trade or exchange in-game assets outside the game.

AR VR: Vibehub [158] is a combination of a VR and Blockchain platform for creating virtual spaces where a variety of activities can be conducted, from marketplaces to virtual business meetings. Vibehub has 3D photo-realistic in-house holograms (Holoportation) technology that is used for body scanning of musicians and educators. These holograms can be placed in a custom VR or AR environments where users can take part of the experience.

Decentraland [159] is an open-source and a community-driven platform that simulates a virtual world where users can access with VR devices through a web browser. The Decentraland uses a distributed storage paired with Blockchain that holds all the information to recreate the virtual space in the users' devices. Decentraland users can explore the world, consume a user-generated content or create their own experiences and o er it to peering users on the platform.

Drones: In the Unmanned Aerial Vehicles (UAVs) industry, the Blockchain solves issues and challenges related to cyber-security, air-tra c control and insurance. With the

drone technology advancements, the information gathered by drone-control systems and the drones becomes an attractive target for cyber-attacks. Blockchain can then be used as a defense against the growing threat of cyber-attacks. In [160], the authors focus on the application of Hyperledger fabric to increase the security of networked swarms of UAVs. More specifically, the authors in [161] analyze the current 5G network security solutions and open issues, and propose an application of Blockchain to solve most of the security challenges.

Air tra c control is essential to prevent drones colliding with an aircraft and or other drones. The increasing number of active drones may lead to potential mid-air collisions. In this context, Blockchain has been proposed to resolve the issue through an air tra c management system based on Blockchain [162].

Delivery companies expand their operations using drones to deliver a variety of products from common food supplies, to packages, medical supplies, fresh food, as it happened for example during the COVID-19 pandemic [163]. The insurance of the drones is essential for identification of the cause in case of a drone crash. With the application of Smart contracts, the tracking of the accident as well as the movements of the drones can be registered accordingly so that insurance companies can take e cient actions in the best interest of the clients they represent [164].

Aviation: Currently, the radar-based air tra c service providers can preserve the privacy of flight plans and position of airplanes, mainly for military and corporate operations. In the US, the Federal Aviation Administration (FAA) adopted in 2020 the Automatic Dependent Surveillance Broadcast (ADS-B), which does not include the privacy features with following implications in terms of potential security issues (e.g. spoofing, denial of service, etc.). In [162], the National Aeronautics and Space Administration (NASA) proposes a Blockchain based prototype for air tra c management with the goal to mitigate the ADS-B security issues. The framework envisions the use of Hyperledger fabric, as permissioned Blockchain, which would provide a framework that includes certificate authority, use of smart contracts and high bandwidth communication channels for secure communication channels between entities (e.g. aircraft, authorized members, etc.)

Related with drones UAVs, but still part of aviation is the fact that the number of UAVs is increasing world-wide, which by default increases the communication, networking and data generated to interconnect the UAVs. The Edge computing nodes evolve into the main providers of computing and storage for the UAVs. The application of Blockchain technology can establish aviation terminal data security architecture for secure and trusty interconnection of the Edge computing nodes [165].

In [166], the authors propose to replace paper records through the use of Blockchain based distributed ledger. The work provides ideas of improving the aviation record management systems through the example of a record flow using a paper record and the advantage of the use of the Blockchain technology. These records present all the logs that are

kept regarding flights (e.g., crewmembers records, airplane maintenance records, etc.,). The authors pay attention to the potential risk of the Blockchain application through the STRIDE model - Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service and Elevation of Privilege. The analysis of the Blockchain application demonstrates a number of potential gaps that need to be addressed (e.g., authenticated trusted digital identities of the participants whose transactions are recorded in distributed ledgers).

The work in [167] emphasizes that soon half of the global aviation fleet will be leased. The authors explore if the application of Blockchain in the aviation is a desirable, feasible and viable alternative for smooth ownership change. After interviewing multiple experts of the aviation field the author presents valuable insight on the application of Blockchain as a tool for future leasing solutions in aviation.

eHealth The application of Blockchain technology to the healthcare industry has been subject of numerous reviews in the last years [168]–[172].

The maintenance of medical records using Blockchain is the most anticipated use-case [173]–[176]. The MedRec [177] is one of the early proof-of-concepts that demonstrate the usability of the Ethereum smart contracts to maintain the patients' records over the years or even for future generations. The feasibility study in [178] confirms that permissioned Blockchain can be successfully used for exchange of personal health records. However, its generalized practical use requires numerous modifications (e.g., reduction in records data size) and reduced operational cost.

The work in [179] proposes a light-weight Blockchain implementation for healthcare data management. The work uses customized Blockchain implementation where the adopted consensus approach is PBFT and the main network regulator is the Head Blockchain Manager (HBCM), which acts as a Certificate Authority (CA). The concept relies on the usage of channels, referred as canal(s), similar to the Hyperledger network. The results show at least 67% increased e ciency or speed in the ledger updates.

In [180], the authors propose VerifyMed, a proof-of-concept Blockchain platform that enables patients and medical professionals to establish trusty communication using online services. The platform is running over Ethereum where smart contracts are issued for each medical treatment, with the results showing modest operational cost for issuing and evaluation of medical treatments.

The healthcare industry is looking forward to the application of Blockchain to battle the drug counterfeit. Numerous studies evaluate the Blockchain benefit for tackling the drug counterfeit [181]–[184].

Automotive: The automotive industry, is going to be revolutionized by next generation of communication technologies [185]. Interconnected vehicles would significantly

enhance the range of services that would bring benefit for all involved players (e.g., car owners, car manufacturers, transportation companies and authorities, etc.). However, the introduction of vehicular-to-vehicular communications introduces a number of security and privacy issues [186]. The application of Blockchain has been seen as a solution to the security and privacy issues [187]–[189], as well as a solution for trustful collection of vehicle's data [190]. Specifically, to protect the trust among all involved parties, the Blockchain technology can be applied to counter fraud. Companies, like Bosch, have committed to build a framework to counter fraudulent actors which are connected to manipulation of car odometers [191], [192].

The authors in [193] present an extensive review on how the Blockchain technology is applied to the automotive use-cases. The work evaluates a di erent set of challenges for each stakeholder in the automotive industry (e.g., car owners, car dealers, insurance companies, car manufacturers, tech companies, etc.) and it reviews the most relevant Blockchain applications for the automotive industry such as: global vehicle ledger, smart manufacturing, anti-counterfeiting, peer-to-peer lending, connected services, forensics, etc.

The applications of Blockchain technology are already explored by some manufacturing companies. BMW envisions several use cases where the Blockchain can be applied to generate a digital passport of a vehicle, improved car manufacturing supply chain and transparent charging of e-cars. Similarly, the work in [194] analyzes a refueling scenario of autonomous electric vehicles. The authors focus on implementation of a smart mobility scenario through the use of Ethereum's state channels or micro-payment channels [195].

The work in [196] explores the application of Hyperledger fabric as a proof of concept to verify and record reports for vehicle-to-vehicle (V2V) messages exchanged in multiple areas. Thanks to the implementation of the Hyperledger solution, the proposed system manages to collect individual reports of received messages from each vehicle in a certain area and to join them in a single distributed ledger for all areas. To improve the authentication, trust and validation in the vehicle-to-infrastructure (V2I) or V2V communication, the work in [197] proposes a new Blockchain algorithm that uses local dynamic Blockchain for keeping local information of the events that are happening in a precise region, and a main Blockchain that keeps track of the global events. Each vehicle in a certain region is authenticated through a unique ID. If an unusual event occurs with a vehicle, the event is directly reported to the main Blockchain. Similarly, the work in [198] proposes a forensic framework to track post-accident scenarios, especially with self-driving vehicles. In case of autonomous vehicles, the work in [199] envisions firmware updates through application of Blockchain and smart-contracts.

Logistics Supply chain: From the logistics and supply chain perspective, the Blockchain technology is seen as a disruptive technology that will change the way that the industry operates. Stakeholders into the supply chain eco-system expect a major impact in in-

creased e ciency, transparency and reliability.

The authors of the work in [200] conducted a survey on social media to measure the acceptance of the Blockchain technology applied to the logistics and supply chain industry. The findings reveal that most of the companies understand the positive impact of the Blockchain over the logistics industry. However, companies are more hesitant to devote significant resources in developing Blockchain applications.

The work in [201] aims to overcome the adoption fear and to design a strategy for how to design, develop, validate and integrate a Blockchain solution in a logistic and supply chain business strategy. The authors present a case study of fresh food supply chain deployed with Hyperledger Fabric. The results show that the implementation of Blockchain solutions is highly sustainable and is completely covered by the savings. The most critical issue is that the Blockchain should be adopted by all involved actors.

The work in [202] proposes a decision framework for the logistics industry based on using a quantitative approach. The framework is applied on a large-scale logistics company where the findings suggest a range of important criteria for Blockchain applications (e.g. security, visibility and audit) and a range of feasible logistics operations where the Blockchain can be applied (e.g., transportation, materials handling, warehousing, order processing, etc.)

In the aviation industry, with the raise of aviation travelling the demand for airplane spare parts is increasing globally. Through application of Blockchain and IoT, the supply chain management teams can predict the life expectancy of the spare parts and distribute them all around the world using a distributed Blockchain-based data-driven system [203] [204].

DLT Blockchain within vertical slices: Previously are described the vertical industries where the application of DLT Blockchain is the main building block of the system. Here, we discuss the works where the vertical industries use network slicing to provide service, and on top of that, a DLT Blockchain is applied within the network slices.

In the work in [205], the authors propose the deployment of an automotive slice that supports Blockchain-based interaction among vehicles. The communication between the vehicles uses 5G infrastructure and content-centric networking (CCN) instead of traditional peer-to-peer (TCP IP) networking. With a dedicated network slice, the V2X CCN tra c is separated from the rest of the operator network and there is no need of additional infrastructure to support the CCN-based Blockchain communication. On the other hand, the main benefit of the CCN-based Blockchain is that it enables trusty communications (via CAM messages) among all the vehicles that drive on the city roads or highways.

Similarly, for the future autonomous robotics, the work in [206] presents the need for a dedicated Blockchain slice to provide on-demand MEC services or third-party applications to autonomous robots or self-driving cars. The idea is to build a framework for

autonomous robotics as a sum of four di erent MEC network slices, where the Blockchain slice is the main slice to interconnect the autonomous robots with the rest of the slices of the MEC layer. The work proposes novel applications such as the provision of real-time HD driving maps, o oading MEC services, etc. Specifically, the work proposes the use of Hyperledger for the Blockchain slice and ROS2 for the robotics infrastructure.

5.3. Realizing federation in dynamic environments through the use of Blockchain technology

In the previous chapter, the federation in static environment is described paying special attention to how to extend the MANO stack to enable federation of services and resources. While there minor di erences in the definition of federation for static and dynamic environments, a common assumption is a presence of a pre-established agreement among service providers. Settling an agreement is time-consuming and suitable only for static environment.

The goal in this chapter is to present how Blockchain technology can complement NFV MANO service providers to accomplish federation in dynamic scenarios. Several Blockchain platforms are tested to provide insights in their performance through profiling and execution time. Some of these platforms are implemented in a real case scenario – such as the on-demand deployment of virtual access points to expand remote control of Edge robots [59].

5.4. Federation challenges in a dynamic environment

Next, we summarize the main challenges posed by multi-domain federation [207], [208] in dynamic environments (Table 5.2), identifying how these challenges are tackled depending on the interconnection approach. We later elaborate (Section 5.5) and propose how Blockchain can be the basis of a solution to all these challenges, but we first focus on how this is done for the centralized and decentralized-peering types of solutions:

Admission Control. Administrative domains in an open federation are free to join or leave at any time. If a centralized interconnection is adopted, the central entity oversees the admission control, i.e., which domain is allowed to leave or join the network. If a decentralized-peering interconnection is used, the access can be completely open depending on each individual administrative domain. In both cases, the main challenge is to balance the trade-o between domain openness and preserving privacy, security, and trust. Highly secured frameworks and message exchanges may introduce higher delays or congest the federation interaction. To the contrary, an absolutely open admission may expose administrative domains to passive spoofing.

Availability. The number of participating administrative domains changes over time in a dynamic environment. With centralized approaches, it is easier to monitor who is participating, though there might be inconsistencies if there are sporadic failures, due to the single point of failure nature of a centralized approach, which might lead to the federation becoming unavailable. Decentralized-peering solutions are inherently more resilient to failures, but tracking administrative domains is more challenging and may introduce spoofing risks.

Dynamic pricing billing. Administrative domains have the incentive to increase the profit by adapting the federation price o erings, especially in a dynamic environment. A central entity, as an auctioneer, can change federation o erings and track the billing process. However, participating domains should voluntarily trust this federation control and pay for it. In the decentralized-peering scenario, the administrative domains autonomously set the price o erings. The di cult part is to quickly arrange secure agreements in the form of *dynamic SLAs*, that would establish a baseline for assuring billing. Additionally, employing mechanisms to identify other domains and securely implement a charging process is costly.

Multi-domain Quality of Service (QoS). Guaranteeing the quality of service across federating domains is quite challenging in dynamic environments. In both decentralized-peering and decentralized-distributed, the major challenges are establishing dynamic SLAs and guaranteeing unbiased monitoring data [208]. Domains often disagree on the monitored data avoiding the responsibility in case of a low QoS or SLA breach. Often, for avoiding legal disputes, third-party entities are monitoring the SLAs. While in the centralized option, a centralized entity is responsible for establishing QoS across every domain, thus controlling and monitoring the whole process.

Security privacy. With a centralized interconnection schema, the administrative domains rely on the central entity. To increase security, the central entity demands more information from every administrative domain, which comes at cost of a lower privacy per domain. On the contrary, a decentralized-peering solution may achieve higher privacy (exchanging less information with peering domains), at the cost of lower security policies employed.

5.5. Applying Blockchain to federation

Blockchain can help to overcome most of the challenges posed by federation in dynamic environments. Actually, ETSI has formed an ISG for Permissioned Distributed Ledger (PDL) which lays the foundations for the application of Blockchain in globally open telecommunication networks [209], [210].

Table 5.2: Federation challenges and how they are tackled through di erent interconnection realizations

	Interconnections		
Challenges	Centralized	Decentralized peering	Blockchain
Admission control	Central;	Open;	Distributed; Consensus voting;
Availability	High; Single point of failure;	Unknown; Fail-safe;	Balanced; Fail-safe; Incentive to participate;
Dynamic pricing & billing	Central auctioneer; Single-point control;	Autonomous; No control; No billing;	Autonomous by default; Token based billing
Multi-domain QoS	Central control & monitoring (dynamic SLAs);	None;	Smart contracts as dynamic SLAs; O -chain oracles;
Security & Privacy	High security; Low privacy;	Low security; High privacy;	High security; High privacy;

5.5.1. Benefits and drawbacks

The main benefits of applying Blockchain for federation are:

Security. The transaction data included in each block of the Blockchain is timestamped, tamper-proof and immutable. Data alteration is only feasible if at least 51% of the nodes are malicious compromised.

Verifiability, integrity, and trust. The state of the Blockchain is easily verifiable by all the members confirming an equivalent observed Blockchain state.

Smart Contracts. Programmable applications that run as independent entities (or members) on top of a Blockchain (e.g., Ethereum). These applications have deterministic and atomic functions that can embed business logic and rules as in regular contract agreements.

Balanced privacy and transparency. All transactions, state transitions, and blocks creations are transparent. Using cryptography enables private data to be encrypted and exclusive while maintaining the defined transition rules.

Third party absence. The consensus mechanism enables collaboration among unknown members in a trusty manner without a third-party authority (e.g., central entity) to guarantee the integrity of the members.

The main drawbacks are:

Adoption and complexity - the platforms (e.g., Ethereuem, Cosmos, Polkadot, Solana, etc.) are still into the development phase. Although there are indicators that companies are willing to invest in Blockchain technologies and adapt them as key strategic priorities [211].

Scalability - with the increase of the network, the transaction cost increases due to increased transaction fees [212].

Energy e ciency - the energy spent per transaction increases linearly with the network size [213].

Storage - the storage of a Blockchain can significantly increase if the Blockchain itself allows for big files to be stored on-chain [214].

5.5.2. Blockchain for dynamic and open federation

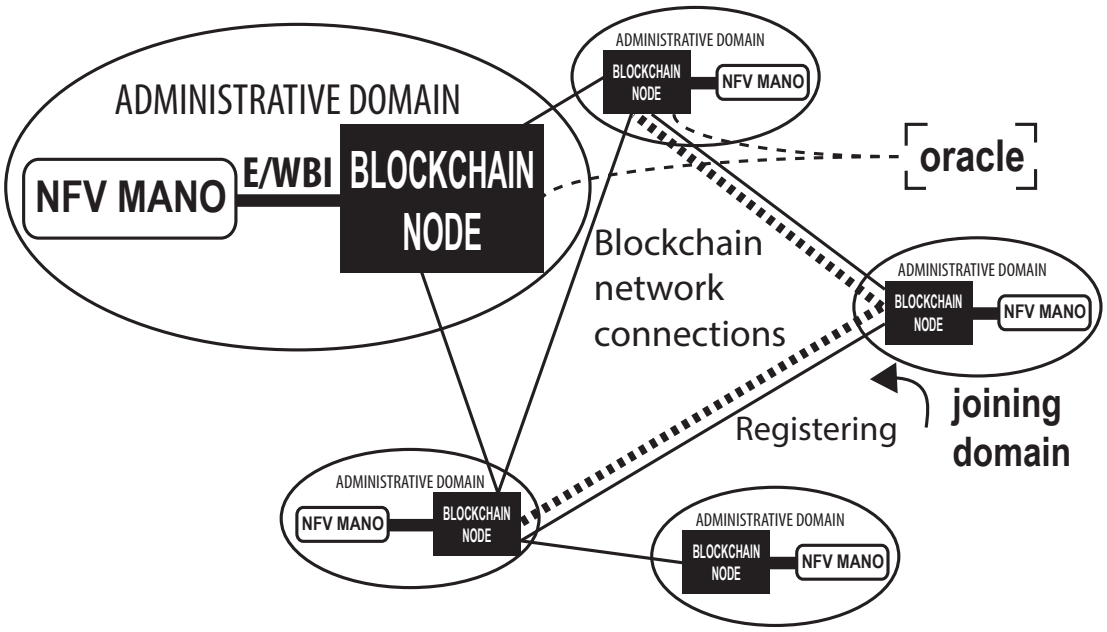


Figure 5.2: Application of Blockchain to open federation

We envision the application of Blockchain to support federation in dynamic environments, in a way complementary to the existing architectural approaches (e.g., NFV MANO frameworks). Implementation on a public permissionless Blockchain can be costly due to costly transactions, whereas the cost for implementation and maintenance of a permissioned Blockchain is low [208].

We propose the design illustrated in Fig. 5.2, to apply Blockchain to open federation. Every administrative domain should deploy a Blockchain node connected to the

East Westbound interface of an NFV Orchestrator. We argue that maintaining this decoupling enables the independent evolution of both NFV and Blockchain technologies.

How can Blockchain solve the dynamic federation challenges Just by deploying a non-customized (vanilla) version of a public permissioned Blockchain network (e.g., Ethereum, Hyperledger, Cosmos, Polkadot, and etc.), most of the challenges enumerated in Table 5.2 can be addressed.

Admission control is dependent on the Blockchain governance policy [209]. In a permissioned Blockchain, a common approach is to accept members via voting. Although domains may act maliciously and reject the entry of new members, domains typically have the incentive to increase the participants. If it is not the case, di erent Blockchain instances may operate in parallel. **Availability** is guaranteed by the incentive of each domain to maintain an active Blockchain node. Therefore, this improves the Blockchain network security (avoiding 51% attacks), and (*ii*) increases the domain's *usage budget* (e.g., gas in Ethereum). In short, the 51% attack happens when a malicious user controls 51% of a Blockchain network, thus can modify all the transactions in every block. In case that a node fails, leaves or is compromised, the Blockchain network remains active and operational as well as the domain has access via other nodes by using its unique Blockchain address.

Security and privacy are established by limiting the *usage budget* and the use of cryptography. Newly joined domains have a lower limited *usage budget* or a limited number of federation announcements being unable to spoof or spam the participating domains. Communications between domains are recorded and validated as immutable transactions on the ledger. Cryptography is used to preserve the privacy of the data in the transactions exchanged [209].

Dynamic pricing and billing, and Multi-domain QoS require implementation of dynamic SLAs and QoS monitoring. The use of Smart contracts is a promising solution towards the integration of both dynamic SLAs and QoS monitoring. Smart contracts are deterministic and independent applications that reside on the Blockchain ledger. The ETSI PDL specification [210] provides a hint of how to employ QoS through an example scenario of using Smart contracts. It envisions a marketplace of SLAs where each Smart contract presents a specific service o ering with QoS metrics. Customers, ready to deploy a service from the marketplace, need to send a payment Blockchain transaction to the specific Smart contract. A third-party entity is used (as an *oracle*) to monitor the QoS metrics and record the SLA fulfillment directly in the Smart contract. In the case that QoS is not satisfied, the Smart contract automatically sends back a Blockchain payment transaction to the customer Blockchain address with the penalty amount. Additionally, service providers as Smart contract owners can dynamically change the prices in every Smart contract, of course, prior to the customers making the deposit transaction. Similar ideas have been tackled in [60], [215]. Our Blockchain solution for federation The adaptation of the described PDL concept in a federation scenario implicates a new Smart contract creation for every new federation of services or resources. These Smart contracts

represent dynamic federation SLAs that guarantee the QoS between the consumer and provider domains, but this approach presents some drawbacks. There is an added Smart contract deployment latency [210]. This added delay is due to the writing operation on the ledger. Reading operations on a Blockchain ledger are immediate, but the writing alters the ledger state. The speed of writing mainly depends on the consensus mechanism (e.g., Proof-of-Work, Proof-of-State, Byzantine Fault Tolerant, etc.), the network size, number of newly issued transactions, etc. Furthermore, the use of a third-party entity (oracle) for monitoring QoS metrics denatures the distributed concept, transforming it into a hybrid version of a centralized solution.

Our proposed design (Fig. 5.2) can be realized, (*i*) with a single Smart contract as an auctioneer; or (*ii*) without a Smart contract, on an application-based Blockchain.

The use of a single Smart contract, in our vision, defines a neutral set of rules that reflect the federation steps from Section 3.2.1 in the role of an auctioneer. The use of a reverse-auction model enables consumer domains to have customized federation announcements and provider domains diverse bids. In [59], we showcased the use of a single Smart contract for federation in a dynamic environment. Compared to the multiple Smart contracts case, the main di erence is that the deployment delay is omitted. The Smart contract can record all the domains' interactions on a single Blockchain address. These records are used as proof that all procedures have been performed correctly and to enable billing.

Application-based Blockchain for federation vs. Smart contract Extensive use of Smart contracts may exponentially increase the ledger storage, due to the number of writing operations, even if it is a single Smart contract. As a consequence, new joining domains may encounter significant delays in syncing the ledger of the Blockchain, known as a *scaling issue*. New application-based Blockchains emerged to diminish the *scaling issue*. In comparison to general-purpose Blockchains, containing multiple Smart contracts (e.g., Ethereum), application-based Blockchains propose a single application per Blockchain, or vice versa. Examples of these are Tendermint Cosmos SDK, Polkadot, etc.

Both approaches have advantages and drawbacks. The adoption of each of the approaches may highly depend on their performances. Hence in the followuing, we try to characterize the performance of di erent platforms for a simple federation scenario.

5.6. Performance of di erent consensus mechanisms to a federation scenario

In the following section we describe the experimental scenario and setup used to evaluate the performance of di erent consensus mechanisms. The obtained results are elaborated for each Blockchain platform in terms of execution time and utilization of resources.

5.6.1. Experimental scenario

The experimental setup contains three independent administrative domains. Each administrative domain containing an orchestrator, underlying infrastructure and a Blockchain node. The characteristics of each component are described in the following Section 5.6.2. The experimental scenario is divided into two parts: federation and healing.

The federation part simulates an extension of a network service through federation. The federation procedure is considered successful when two hosts from the consumer domain are able to maintain a continuous communication (with no packet loss for finite amount of time) with another two hosts from a provider domain.

At the start, the consumer domain announces the desired extension of the service to be federated by sending a transaction. Both provider domains receive the announcement transaction and generate a bid-o er as a transaction, containing all the service details and prices. The consumer domain receives the o ers (transactions) and elects a winning provider domain which selection may be based on various things. In our case, we elect the domain using first-come-first-serve (FCFS) strategy. Both domains receive the consumer selection outcome, and the winning provider (#1) starts the deployment of the federated service. The provider #2 returns to idle state. While the deployment is running, the consumer domain sends the connection details through a transaction. Upon deployment of the federated service, the provider #1 and the consumer domain establish the interconnection between both domains using VxLAN The federated service is up and running promptly after the deployment finished and inter-connection is established. Beside utilizing the service, the consumer domain starts to continuously monitor the connection for if satisfies a zero packet loss requirement. In our experimental scenario, once the winning provider deploys the federated service is keeping it up and running for 10 seconds.

While in the second phase - healing, the federated service fails, and it is healed by performing a new federation procedure with another provider domain (provider #2). The healing is successful when the two hosts establish again an uninterrupted communication with hosts from the provider #2, similarly as in the federation procedure. In our experiments, the federated service is set to fail after 10 seconds which marks the start of the healing part. The consumer domain upon detectiion of two consecutive packet losses, issues a new federation announcement. For the new federation procedure, the provider #1 is blacklisted as an unreliable domain which leaves provider #2 as only winner. The provider #2 deploys the newly healed federated service using the same deployment steps.

Here we summarize the exact events measured during the experiments:

1. Service announced - *consumer*
2. Announce received - *providers*
3. A bid o er sent to consumer - *providers*
4. Winner choosen and broadcasted - *consumer*

5. Winner announcement received - *providers*
6. Deployed federated service - *winning provider*
7. Connection details sent to winning provider - *consumer*
8. E2E Service running - *consumer and winning provider*
9. Service stopped - *winning provider*

Note that the events of the healing service are measured in the same order. Additionally the 10 seconds countdown of the federated service starts at event (6), and it is stopped in the last step (9).

5.6.2. Experimental setup

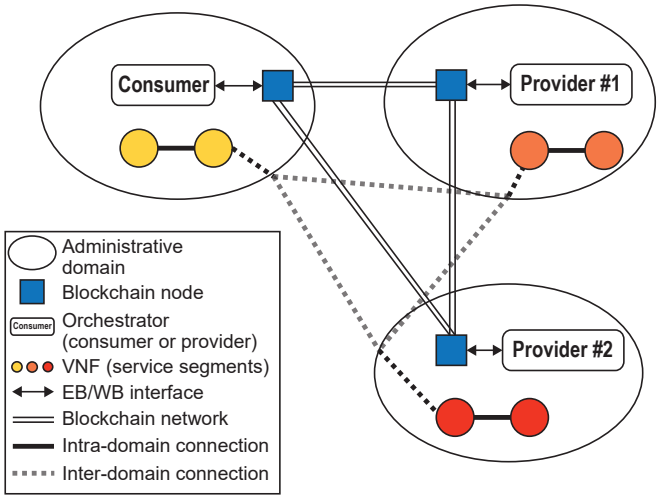


Figure 5.3: Experimental setup

The experimental setup is shown on Fig. 5.3. Each administrative domain consists of two host machines. The orchestrator and the underlying infrastructure are coupled in a mininet VM, an Ubuntu 14.04 virtual machine with 2 CPU cores, 2 GB of RAM, and 5 GB of disk memory. Each of the blockchain nodes are in di erent machine, an Ubuntu 18.04 virtual machine with 2 CPU cores, 6 GB of RAM, and 25 GB of disk. Besides the di erent dependencies both on the mininet and Blockchain platforms, the decoupling would represent a real integration of Blockchain nodes into an existing infrastructure of a service provider or a mobile operator.

5.6.3. Proof-of-work consensus profiling

First, we executed the experimental scenario using the Ethereum platform with Proof-of-work consensus mechanism. Note that the three PoW Ethereum nodes were mining simultaneously, competing each of them for the block reward. Each orchestrator (consumer

or provider) is posting the transactions directly to the local Blockchain node. Figure 5.4 presents the occurrence of all events listed in the previous section (Sec. 5.6.1). The narrow ticks represent the mean occurrence time of the events. Note that some of them may overlap due to very high variance range. The variance for each event is represented by a transparent bar which represents the variance range. Hence in the Figure 5.4, the variance ranges overlap with each other thus lowering the color transparency. Some areas are color dense due to overlap of multiple variance ranges for several events.

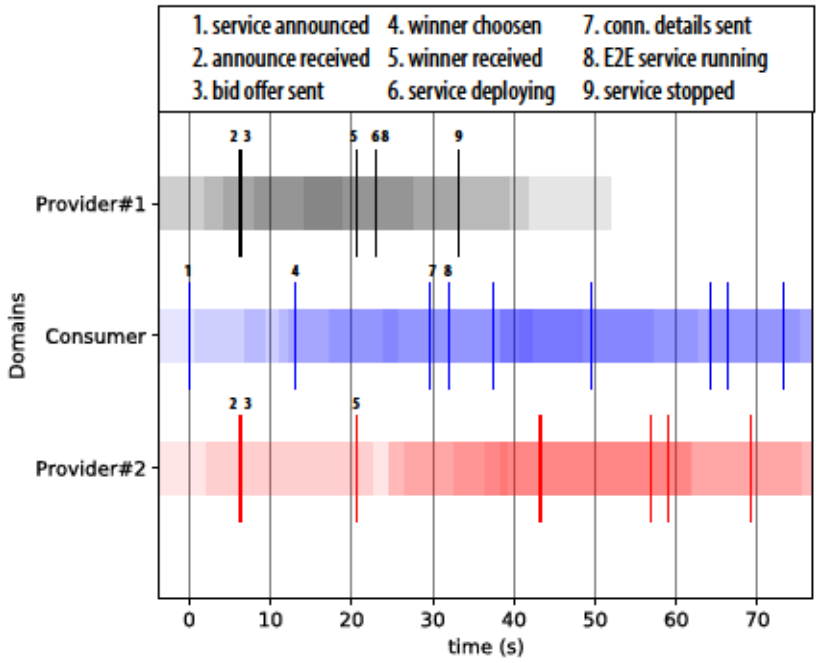


Figure 5.4: PoW event variance

As mentioned before the provider domain is deploying and keeping the service up and running for limited time of 10 seconds. From Fig. 5.4 is visible that in the case of PoW, the service is barely consumed by the consumer domain, due to high variance in transaction propagation.

For better view of what is happening in each Blockchain node, we monitored the CPU usage, the memory usage, the storage and network receiving for the duration of the experiments. Figure 5.5 is presenting the profiling obtained for the duration of 20 consecutive experiments. From the obtained results, it is clear that the PoW consensus mechanism is saturating the CPU in each node up to 100%. The memory usage is constant while runing the experiments. Due to exchange of pending transactions and mined blocks, both the disk and the network activities are at moderate level.

5.6.4. Proof-of-authority consensus profiling

We repeated the experimental scenario for Ethereum platform using PoA (Clique) consensus mechanism. The experiments consist 20 consecutive repetitions. On Fig. 5.6 are

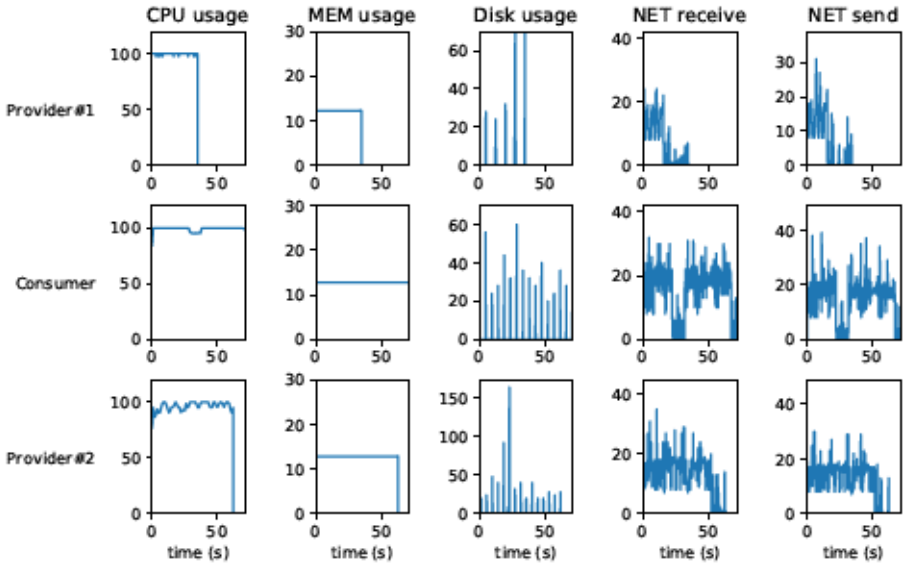


Figure 5.5: PoW profiling

shown the event occurrences. In the PoA case, compared to PoW, the submitted transactions are mined more regularly, which reflects in lower federation time as well as lower variance ranges per events. The service federation is established within 15 seconds with no overlapping average times of events occurrences. The completed time of the PoA experiment is less than 50 seconds while in the PoW case is over 70 seconds.

Similar as in the PoW case, events overlap and are triggered in the same mainly due to inter-block times.

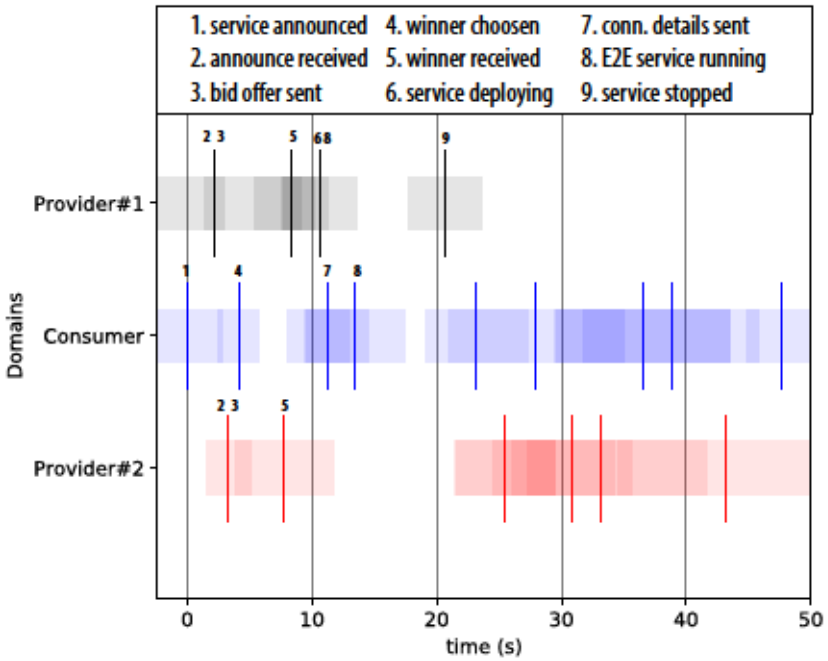


Figure 5.6: PoA event variance

The profiling of the PoA Blockchain nodes is presented in Fig. 5.7. The most evident

is the low CPU usage. In contrast to PoW, the CPU load is less than 10% with small peaks in the Provider #1 domain. These peaks are due to the Provider #1 domain being the last validator and sealer of each newly created block. As mentioned in Section 5.2.4, the last validator is in charge of running the smart contract bytecode and sealing the block. The memory consumption is similar to the memory consumption of the PoW Ethereuem. There is a significant increase of disk and network activity. Even though the disk activity peaks are not significantly higher, the network activity of PoA is around 100% higher than the PoW driven Ethereum.

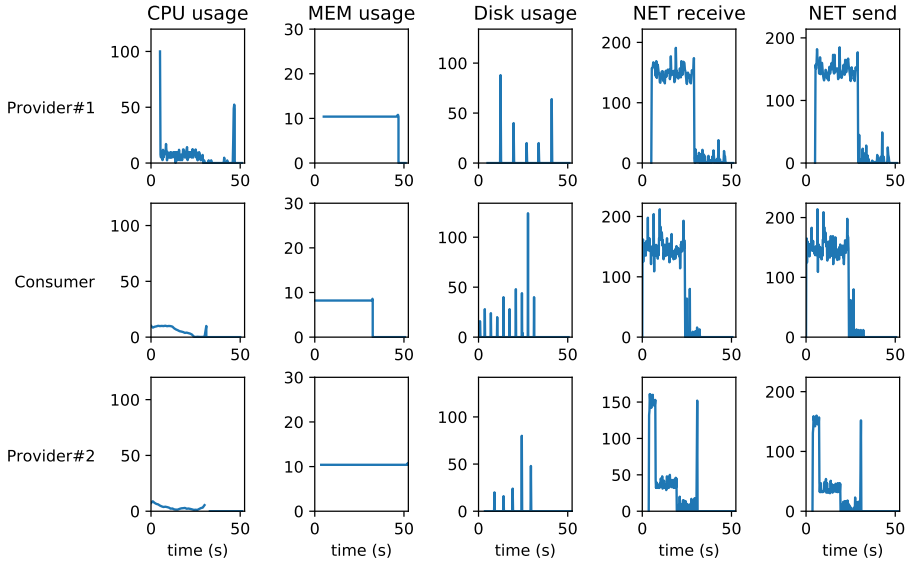


Figure 5.7: PoA profiling

5.6.5. Practical Byzantine tolerance consensus profiling

On Fig. 5.8 are shown the experimental results obtained from the Tendermint platform using the PBFT consensus mechanism. From the obtained results, the average occurrence times of the events have lower variance compared to the the PoW or the PoA results. The system and execution stability is generally preserved for the all repetitive trials. The average completion time is lower than both the Ethereum PoA and the Ethereum PoW. The transaction propagation is almost instant.

The performance of the Tendermint platform is displayed on Fig. 5.9. The CPU load shows that the Tendermint platform is very e cient is appending and exchanging transaction. The validation is not computationally demanding. Since the Tendermint is application-based Blockchain, only a single application can run on top of the Blockchain. In this case it is the federation application. Thus the CPU is significantly lower in contrast to the Ethereum PoW platform, where as a general purpose Blockchain many smart contracts can run on top. However there are not many di erences compared to the Ethereum PoA. Memory-wise, the Tendermint platform takes over around 10% of the available memory, similar to the Ethereum platform. On the other hand, the disk activity is signif-

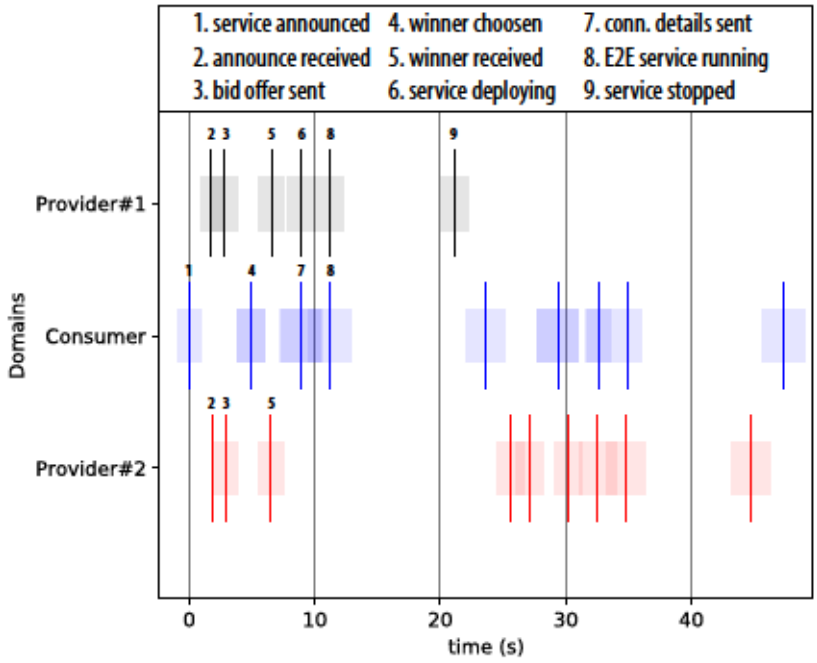


Figure 5.8: PBFT event variance

icantly increased compared to the Ethereum platform. This can potentially be problematic on the long run, mainly depending on the storage hardware used for the Tendermint nodes. The network activity is in the range of the Ethereum PoW platform, with increased picks when new federation announcements are submitted, mainly due to increased data exchange.

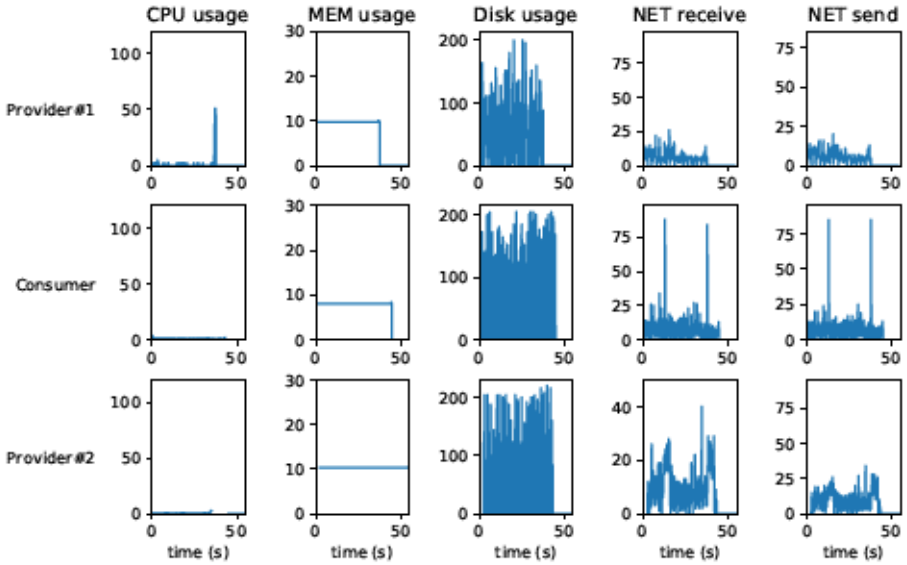


Figure 5.9: PBFT profiling

5.6.6. Proof-of-stake consensus profiling

The last evaluated consensus mechanism, the Cosmos PoS platform the averaged event occurrences are shown on Fig. 5.10. The average time to complete a federation is *sim*26 seconds which is shorter in duration than Ethereum PoW, but longer than Ethereum PoA and PBFT Tendermint. The variance is significantly lower than Ethereum PoW.

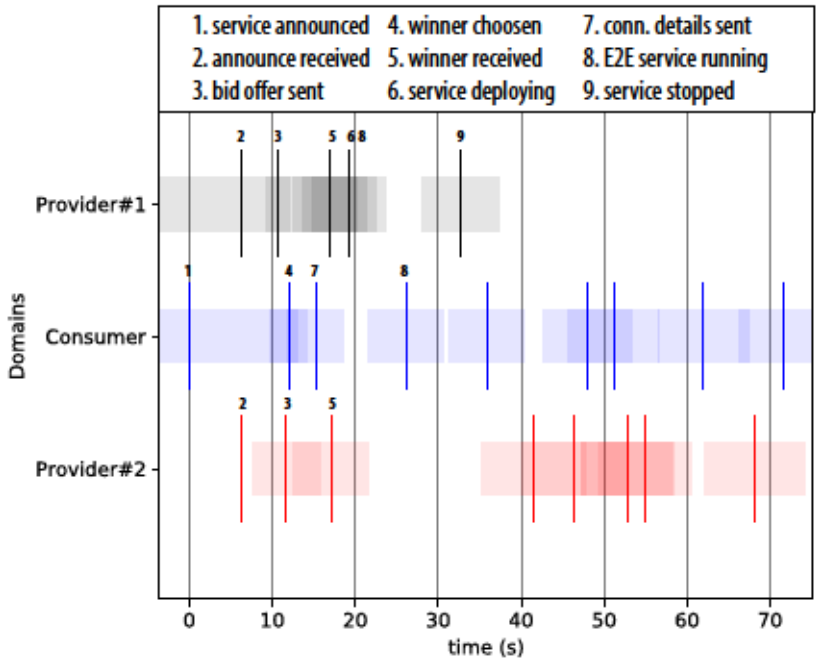


Figure 5.10: PoS event variance

The performance analysis of the Cosmos PoS is shown on Fig. 5.11. Even though the Cosmos platform is built on top of Tendermint, the computational overhead in verifying all the blocks is evident in the CPU load. The CPU increase of up to 50% is related to the generation of transactions from the given nodes. When a node is only validating and relaying blocks, the CPU load drops significantly, as in the case with Provider #1. The memory usage is standard up to 10% for all Blockchain platforms.

The storage activity is relatively high as in the Tendermint case. However the network activity is significantly higher than rest of the Blockchain platforms. In our view, this is due to the increased size of data exchanged. The Cosmos PoS is an application of Tendermint itself, which by default adds an data overhead.

5.7. Discussion

In this section we elaborate over the evaluated results presented in Section 5.6. Table 5.3 summarizes the following elaboration. Besides the measured results, we generated additional empirical metrics through the setup and running of the experiments which we find them useful for evaluation of the platforms. These empirical metrics are: setup com-

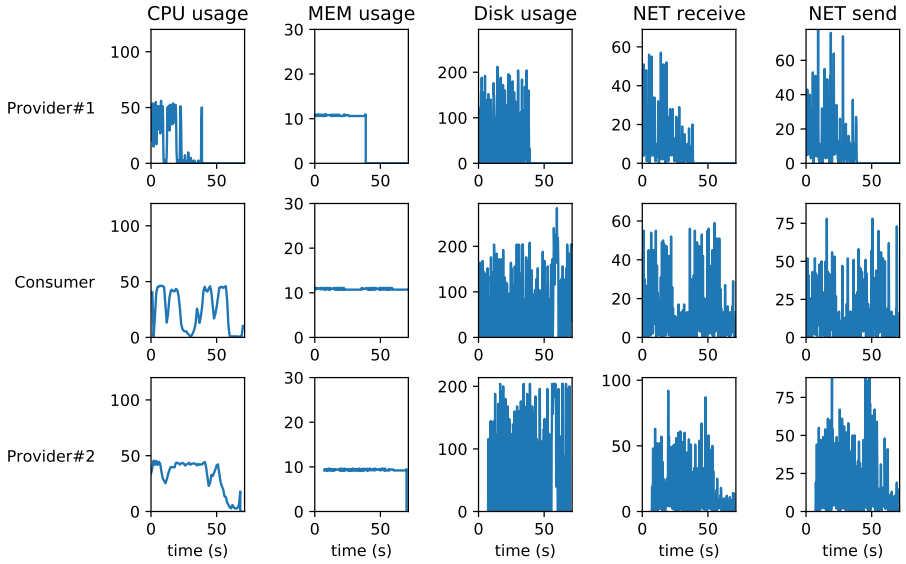


Figure 5.11: PoS profiling

plexity, application development complexity, public chain portability, support for multiple applications and community support through documentation. Note that these are more subjective metrics given we have 5 developing experience on Ethereum platform. However, we argue that these metrics provide significant insight into the transition and adoption of Blockchain as a technology for NSF or other network applications, given the maturity level of the Blockchain technology (at the time of publishing).

The average federation time shows the overhead of the NSF application if it is realized through the application of Blockchain technology [54], [216], [217]. The di erent consensus mechanisms, as we previously evaluated, have di erent security characteristics. To that end, the choice of more time-e cient consensus is a trade-o for choosing less secure and more centralized or permissioned systems. For example, the Ethereum PoW would be more suitable for open federation, where the participants does not demand stringent authentication procedures and anonymity is allowed. On the other hand, PBFT Tendermint or Ethereum PoA would be more suitable for rapidly changing dynamic environment, where a service demands a volatile edge infrastructure [59].

There is a big distinction in the CPU utilization for each of the consensus mechanisms. In the case of Tendermint and Ethreuem PoA, the Blockchain process activity has low e ect on the CPU usage. The saturated CPU utilization in Ethereum PoW demands higher performance computational infrastructure.

In terms of memory usage, every platform use around 10% memory usage. The disk activity might be severe for the long-run, especially in the case of Tendermint and Cosmos platforms. Except in the Ethereum PoA case, there is a low network overhead which is suitable for federation of network services as well as other applications.

As mentioned before, the observed metrics are useful for future application of any of the evaluated Blockchain platforms. The setup complexity is straight-forward and

Table 5.3: Consensus mechanisms and platforms comparison

		PoW - Ethereum	PoA - Ethereum	PBFT - Tendermint	PoS - Cosmos
Measured	Avg. federation time	32 seconds	14 seconds	11 seconds	26 seconds
	CPU utilization	100%	10%	3%	50%
	Memory utilization	10%	10%	10%	10%
	Disk utilization	Moderate	Moderate-Low	High	High
	Network activity	Low	Moderate-High	Low	Low
Empirical	Setup complexity	Low	Medium	High	High
	App development complexity	Solidity (medium)	Solidity (medium)	Golang (medium-high)	Golang (medium-high)
	Support for multiple applications	Yes	Yes	No	Partially
	Portability to public Blockchain	Simple	Simple	Complex	Complex
	Community support	High	Medium-high	Low	Medium-low

well documented for both Ethereum PoW and PoA. The access points are well-defined, with various tools for deployment of smart contracts (e.g., Tru e, Hardhat). The block creation process is familiar to the original Bitcoin block creation process. The complexity of setting up Tendermint and Cosmos private Blockchain instances is significantly higher. Although running a single node environment is straight-forward, the setup of multiple networks is not well documented and not very intuitive. We also want to note that this was the case at the time of setting up the experimental environment which might be improved afterwards.

The application development is not significantly di erent in terms of application logic. The Ethereum Virtual Machine (EVM) provide universal functions, definitions (e.g., addressing, balances) and variables (block numbers, states) that are not di er significantly between di erent EVM compilers. Smart contracts can be interconnected with other smart contracts and interact. An application might be distributed over several smart contracts. In Tendermint, it is up to the service providers to develop all the utility libraries on top (addressing, balances, etc.). Cosmos contains some of the default utilities, however it still demands very detailed application code which defines behaviors at each stage of the block creation.

The support for multiple applications or smart contracts might be crucial for future implementations. Both Ethereum platforms support running multiple smart contracts over the same Blockchain (EVM) instance. In this case the computational utilization is not (significantly) increasing for every new Blockchain smart contract in Ethereum PoW.

This is not the case the PBFT Tendermint and PoS Cosmos platforms which demand newly deployed Blockchain instance for each new application. Although there are Cosmos extensions that allow for enabling an Ethereum Virtual Machine (EVM) to run over Cosmos [218], the Cosmos performance might be degraded due to high overhead. Hence, in our view Cosmos has partial support for running multiple applications at the same Blockchain.

The portability to a public Blockchain (main network) is tightly related with the support for multiple applications. In the case with Ethereum, it is a straight-forward process that requires use of the provided tools (e.g., Tru e, Hardhat), or well defined APIs. There is no public Tendermint network, and in case of Cosmos, the portability is not straightforward. Although the Inter-communication Blockchain Protocol (IBC) is designed to allow di erent Blockchain application instances to be able to communicate, the process is not simple.

Finally, the development communities of Ethereum, Tendermint and Cosmos is significantly di erent. Ethereum has already established and very active community. The Tendermint and Cosmos community is tightly working together and although they are quite centralized, the development is very active and constantly improving with the goal to catch-up the Ethereum.

5.8. Edge Robotics using Blockchain

5.8.1. Motivation

In recent years, Edge robotics emerged as a consequence of the rapid development of Edge computing to address the network performance (e.g., high latency, unpredictable jitter) related challenges that Cloud-based robotic applications experience[219]. By placing computing and storage resource near the edge, robotic systems can execute applications closer to the robots resulting in more predictable communication and overall better system performance. For the market, the Edge robotics services are an opportunity for mobile robots to be employed in accomplishing a range of manual tasks (e.g., security surveillance, cleaning, delivery of goods, collecting fruits, ehealth emergency response, sports video coverage, etc.). The linchpin of the Edge robotics is the constant robots connectivity over the access network and the available real-time information about the connectivity. This information enables e ective adaptation of the robot operations to the actual status of the communication. The high mobility of robots demands change of the point of access in the access network which is currently feasible within a single administrative domain. What happens if a robot needs to leave an administrative domain (network coverage) in order to finalize a task? In such cases, an Edge robotics service require fast and short-lasting expansion of the service footprint over multiple administrative domains (e.g., delivery of goods for a big day-lasting events, emergency response for large area, video streaming of cycling events, etc.).

The federation as a 5G networks concept for NFV and MEC, enables orchestration of resources and services across multiple administrative domains. Virtualized access networks enable the robotic service providers to request on-demand deployment of virtualized access point, in an external administrative domain at a specific location through the federation process. With the introduction of the Fog concept, where volatile and low-power consumption devices are used as access network, federation extends the eco-system heterogeneity and variance in the access network coverage. Multiple administrative domains can simultaneously deploy virtualized wireless networks over range of hardware devices thanks to Fog, Edge, MEC and NFV concepts.

As a consequence, a higher number of involved administrative domains, eligible to provide on-demand federation of services and resources, increase the risk of security threats, maintaining SLAs, privacy violations, and etc. The DLT is a potential solution to counter the negative byproducts of the federation process. The Blockchain as a DLT, by default provides trust, security and cryptography to participants. Leveraging the DLTs, the administrative domain can discover, negotiate and federate services on-the-fly.

In this paper, our goal is to (*i*) propose a DLT federation concept in the Edge robotics environment, (*ii*) apply the DLT federation concept on a real Edge robotics test-bed and (*iii*) evaluate the performance of the solution.

5.8.2. Related works in Edge robotics

Following the principles of Edge robotics, [220] elaborates on the edge-computing friendly functionalities in healthcare robots and discusses the corespondent edge computing techniques in order to materialize wireless driven healthcare robotic services. Moreover, an example of system architecture that exploits the edge to achieve o oading for computationally expensive localization and mapping is presented in [221]. In [222], the authors present the possibilities of deploying AI based dynamic robotic control in the edge of the network to self-balance service robot and pick up a box automatically. The experimental test-bed and scenario (described in the following sections) is partially implemented in our previous work [38].

5.8.3. Federation in Edge robotics

In this section, first we dive into the Edge robotics service and realization through MEC in NFV. Then, as a consequence of the dynamic and volatile environment, we propose the edge federation concept. Finally, we explain how DLT can be applied for private, secure, and trusty edge federation.

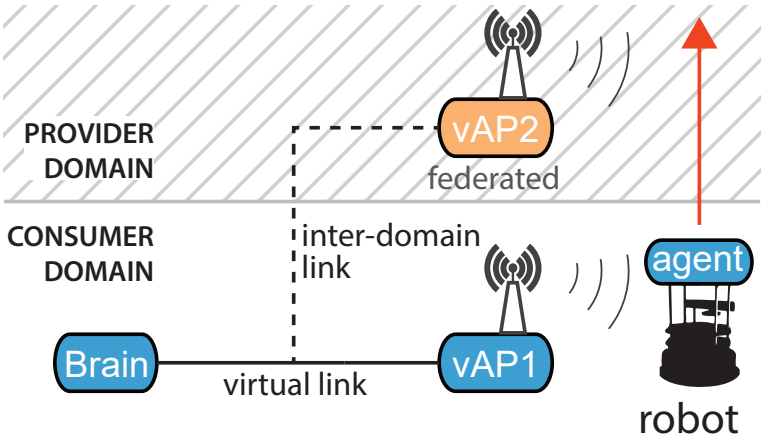


Figure 5.12: Edge service

5.8.4. Edge robotics: MECinNFV-based service

Driven by the opportunities that are o ered by operating at the edge of the network (e.g., proximity, low-latency, network context information), ETSI created the MEC framework as one of the early implementations of Edge computing. In this work, the Edge robotics service relies on the MEC in NFV reference architecture [13] [223]. In this realization of the MEC, the key components of the architecture (e.g., MEC platform, MEC applications, and MEC services) are realized as virtualized network functions (VNFs) over a virtualized infrastructure. To that end, the Edge robotics service is represented by MEC apps distributed between robots and MEC hosts. The points of access (e.g., virtual access point) are represented by a MEC apps as well, while MEC services provide real-time radio network information or robot localization information through a MEC platform. The Edge robotic service can use this information to dynamically adapt the robot operations.

The combination of radio context information and location coordinates allow the robot to move within the boundaries of a single administrative domain. Our proposal is that through application of service federation, an Edge robotics service would not be limited (to single AD) and it would be able to extend the desired service footprint at anytime, anywhere. A simplified service federation of an Edge robotics service is illustrated on Fig. 5.12. All colored blocks represent MEC apps as VNFs. The blue blocks present the MEC apps of an exemplary Edge robotics service deployed in a consumer domain. The "Brain" contains the control logic that provides movement instructions to a robot "agent", through the virtual access point ("vAP1"). The robot, via the "agent", executes the movement commands using its actuators and provides real-time sensor data back to the "Brain". In short, this is a closed-loop which allows the "Brain" to control the robot to accomplish di erent tasks (for more details refer to [38]). When the robot leaves the coverage area of the consumer domain, a service federation is initiated by the consumer domain. The service federation includes deployment of new virtual access point ("vAP2") in a provider domain that can ensure extended network coverage. The federated "vAP2" establishes an overlay connection to the "Brain" through an inter-domain link. Once the end-to-end connectivity is established, the closed-loop between the "Brain" and the robot

continues through the federated "vAP2" without any service interruption.

5.8.5. Service federation procedures

In our solution we focus on the service federation rather than resource federation. In service federation a consumer domain (orchestrator) requests an extension of a service (or part of a service) to be deployed over a provider domain. The provider domain (orchestrator) oversees the complete deployment process of the service extension. While in resource federation the provider domain only provides available resources (e.g., computing or networking) to the consumer domain, and the deployment of the service extension is executed by the consumer domain. In order to successfully complete a service federation [90] [216], there are several federation procedures that are executed in sequence:

Registration - initial procedure through which the administrative domains establish their peer-to-peer inter-connectivity or register to a central entity. The registration procedure characterizes the type of federation, which can be relatively open or strictly closed. As an open federation can be considered when external new domains can more easily register to the peer-to-peer or centralized interaction. The closed federation includes pre-defined participants with strict policies and rules, manually set and defined by the ADs.

Discovery - in this procedure the participating ADs periodically broadcast or exchange among themselves information on their capabilities to provide services or (computing networking) resources. Each AD creates and continuously updates a global view of the available service or resources at the external ADs.

Announcement - this procedure is triggered by the consumer domain, once it has been decided the need to federate part of a service in an external peering domain. An announcement is broadcast to all potential provider ADs. The announcement conveys the requirements for a given service or set of resources. In the centralized case, the central entity is used as a proxy.

Negotiation - the potential provider ADs receive the announced o er, analyze if they can satisfy the requirements and send back a positive or negative answer. The positive answer includes the pricing of the service.

Acceptance deployment - the consumer AD analyzes all collected answers and chooses an o er of a single provider domain. The selection process is entirely left to the consumer AD's internal policies and preferences. The consumer domain sends an acceptance reply to the chosen provider AD. The provider AD starts the deployment of the requested *federated* service.

Usage Charging - once the provider AD deploys the federated service, it notifies the consumer AD and sends all necessary information for the consumer AD to

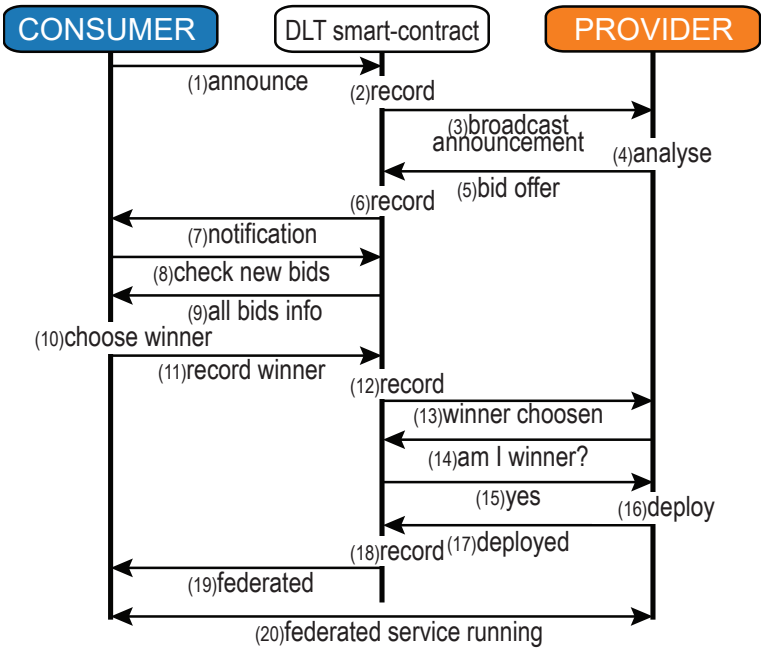


Figure 5.13: Sequence message diagram for Federation Smart-Contract and administrative domains during federation

include the federated service as part of the end-to-end service deployment. From that on, the provider AD starts charging for the federated service during its lifecycle, until it is terminated.

Please note that the security privacy and trust among the participating ADs is vital in all the aforementioned procedures. Actually, due to competitive reasons, any AD (e.g., mobile operators, cloud providers, etc.,) would not reveal much information regarding the underlying internal infrastructure or the full capabilities for service deployments.

5.8.6. Applying DLT for federation

Depending on how the service federation procedures (described in Sec. 5.8.5) are realized, the sequential completion of the whole federation process can take more than a minute or even an hour. In a dynamic and heterogeneous environment, where the underlying infrastructure of each domain is continuously modified, the state can change in order of seconds.

To boost the federation process in secure manner, our idea is to squeeze the whole service federation process (from Sec. 5.8.5) to run on a DLT. More specifically, the federation procedures to be stored and deployed on a Federation smart-contract (SC) which is running on top of a permissioned blockchain. The design of the Federation SC is completely open. Our focus in the smart-contract design is to maintain neutrality and privacy while overseeing the federation procedures that involve all ADs.

Each domain sets up a single node as part of the peer-to-peer blockchain network. The distributed nature of the blockchain network allows scalability while maintaining

the security. The ADs communicate with the Federation SC through transactions. The transactions are recorded in the blocks. The sealing or generation of blocks depends on the consensus protocol. The choice of the consensus protocol would determine the speed and the security level of the federation process. For example, the *Proof-of-Authority* (PoA) consensus increases the speed, while the *Proof-of-Work* (PoW) mechanism increases the security of the blockchain.

Each new joining AD establishes connectivity with at least a single node in the blockchain network using a new and locally deployed node. Then, it registers to the Federation SC with a single-transaction registration using its unique blockchain address. In the single-transaction registration the Federation SC records the information of the registering AD and its service footprint. This way the registration procedure explained in Sec. 5.8.5 is relatively simple to be realized. Once the registration procedure is successfully completed, the AD is ready to consume or provide federated services.

Fig. 5.13 presents the interactions of registered ADs with the Federation SC for a single service federation process. The registered ADs can participate as consumers or providers in the federation process. When a consumer AD needs a federated services, it creates a federation **announcement** (step 1). The announcement is sent as a transaction to the Federation SC which records the announcement as a new auction process on the blockchain (step 2). Then, the Federation SC broadcasts the auction to all registered ADs (step 3). Note that the address of the consumer AD is hidden in the broadcast announcement in order to protect the AD's privacy and prevent the rest of the ADs to passively collect information. Thus, the **discovery** phase is omitted in the design of the Federation SC. Instead, our approach is using a single-blinded reverse auction [224], where a consumer AD anonymously creates an announcement o er and the rest of the potential provider ADs are bidding for it. Therefore, once the broadcast announcement is received, the potential providers analyze the requirements and place a bid o er to the Federation SC (step 4 & 5). Each received o er is mapped and recorded by the Federation SC (step 6).

In our vision the Federation SC is used more as a tool for maintaining neutrality and privacy than a governing or an authority member in the federation process. As a result, the bidding process is controlled by the consumer AD. That way the consumer AD has the full control and freedom to apply any selection policies (e.g., prioritize given o ers, select the lowest price o er, etc.). In other words, the consumer domain is oversees the **negotiation** and **acceptance** procedures. Therefore, the consumer AD is notified for any new bidding o er and it polls the Federation SC to obtain the information of each bidding o er (step 7, 8 & 9). Once the consumer AD selects a provider AD (e.g., winning provider), it closes the auction in the Federation SC (step 10 & 11). The winning provider is recorded by the Federation SC, which immediately broadcasts message to all participating ADs that the federation announcement has finished and a winner is chosen (step 12 & 13). Each of the participating ADs attempts to obtain the details in order to deploy the federated service. As shown on Fig. 5.13, only the winning provider AD has the granted access to

the information (step 14 & 15).

The information that the provider domain obtains can vary depending of the trust-level of the participating ADs. If the participating ADs want to maintain privacy, having low trust level towards other ADs or **untrusty communication**, the service deployment information is limited (e.g., descriptor to be used, consumer's endpoint to establish data-connectivity). If the participating ADs have higher level of trusts or **trusty communication**, the information that the provider domain can access is broader (e.g., database of resources, storage, di erent endpoints, etc.). At this point the negotiation and acceptance phases (of Sec. 5.8.5) are completed and the **deployment** of the federation service has started (step 16).

Once the deployment is concluded, the provider AD confirms the operation by sending transaction to the Federation SC (step 17). The Federation SC records the successful deployment and initiates charging for the federated service (step 18). The **charging** can be applied through micropayment channels [151]. The micropayment channel applied on the blockchain can enable single non-bias charging record that is immutable for both the consumer AD and provider AD.

At the end, the Federation SC notifies the consumer AD of successful federated service deployment (step 19 & 20). The consumer AD leverages the running federated service until is needed, then terminates the service through the Federation SC. The termination procedure is omitted in this work.

5.8.7. Experimental setup

To prove the feasibility of the DLT federation for Edge robotics (of Sec. 5.8.3) and evaluate the solution, we have deployed an experimental test-bed which on top of it we run trusty & untrusty experimental scenario.

The experimental test-bed (shown on Fig. 5.14) consists of a robot, Ethereum blockchain node and two administrative domains (ADs) - consumer and provider domain - with their underlying infrastructure. The test-bed is deployed along a hallway in the University Carlos III of Madrid.

The consumer domain infrastructure consists of two MEC hosts depicted as host 1 and host 2 on Fig. 5.14. KVM and LXD virtualization is running on top of host 1, while only LXD on top of host 2. Both hosts are orchestrated by the Consumer orchestrator which in this case is a simple custom developed orchetrator determined for the whole scenario process. An on-boarded Edge robotics service is similar to the service described in Fig. 5.12. The Consumer orchestrator deploys the Edge robotics service over the underlying infrastructure (host 1 & host 2) through the distributed Virtualized Infrastructure Manager (VIM) - Fog05[10]. As described in Sec. 5.8.5, the Edge robotics service is deployed as VNF-MEC apps (shown as blue rounded boxes on Fig. 5.14):

[10]https: fog05.io

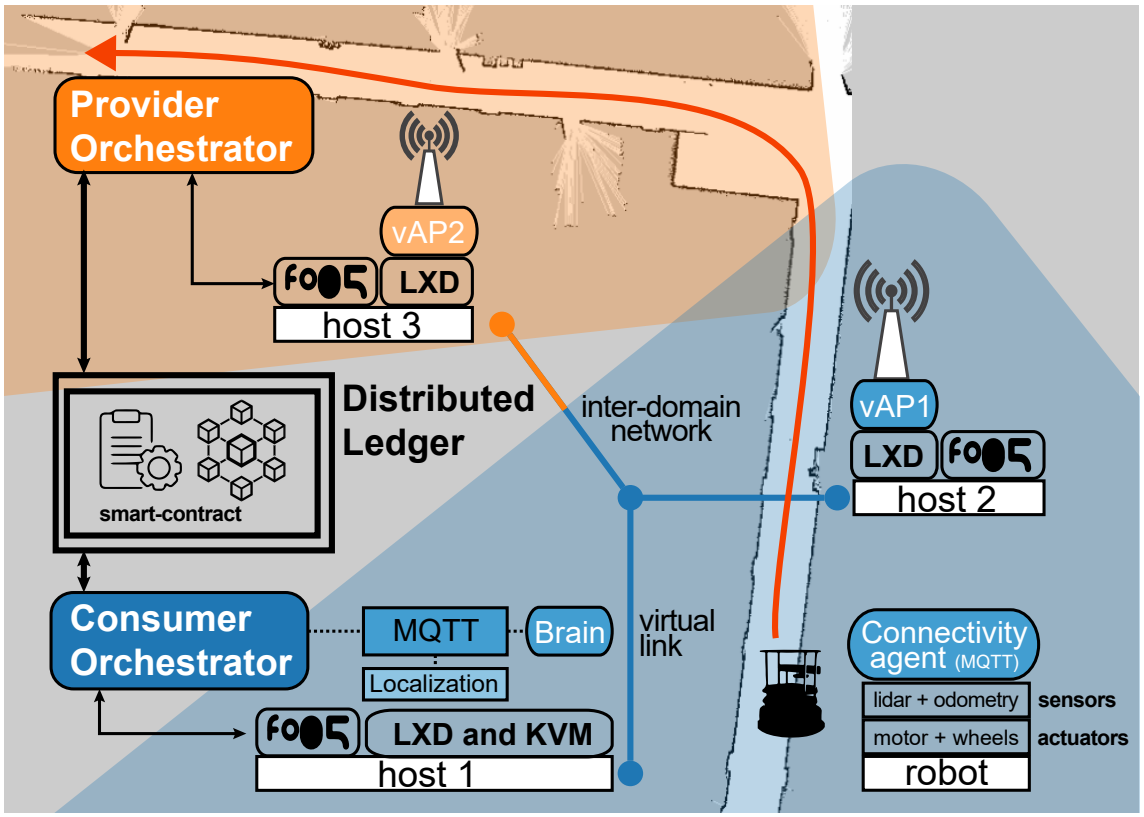


Figure 5.14: Edge robotics experimental test-bed & scenario

Brain is a MEC app deployed over host 1.

vAP1 is deployed over host 2 as hostapd MEC app, inter-connected through virtual link to Brain.

Connectivity agent is deployed over the robot hardware. The robot hardware consists of motor wheels as actuators, 802.11 connectivity, and sensors (lidar & odometry).

A MQTT broker is substituting the role of a MEC platform. The *Brain*, as a main MEC application, is consuming a Localization MEC service via the MQTT broker.

The provider domain is isolated from the consumer domain. Contains a single host (illustrated as host 3 on Fig. 5.14). The Provider orchestrator is a replica of the consumer orchestrator that orchestrates the virtualized infrastructure (LXD) through new instance of Fog05. The provider orchestrator has only the on-boarded image of the *vAP2* MEC application on Fog05 (illustrated as orange rounded box on Fig. 5.14).

The Distributed Ledger contains two instances of Ethereum blockchain. The instances are deployed over a virtual machine on a server at the University network. Both instances contain the Federation SC described in Sec. 5.8.6. The first instance is running *Proof-of-Authority* (PoA) consensus for trusty communication, and the second instance *Proof-of-Work* (PoW) for untrusty communication. In-depth consensus mechanism comparison is out of scope for this work.

The experimental scenario is mimicking a real use-case where the robot is instructed to deliver goods or clean an area at the University, following a path as illustrated with the red line on Fig. 5.14. In order to finalize the task, the robot needs to drive from the blue (consumer) domain to the area of coverage of the orange (provider) domain. The *Brain* is aware of the real-time robot's location by consuming the Localization MEC Service. The *Brain* triggers the federation procedure to the consumer orchestrator when the robot approaches the boundaries of the vAP1 coverage. On triggering event, the consumer orchestrator proceeds with the federation procedure as described in Sec. 5.8.6 and Fig. 5.13. The provider domain, as a winner, establishes an overlay inter-domain link to the consumer domain, and deploys the *vAP2* (as depicted on Fig. 5.14). After the deployment of the federated *vAP2* has finished, the provider orchestrator confirms the deployment to the Federation SC by storing the BSSID of the deployed *vAP2*. The consumer domain delivers this information to the *Brain*. Finally, the *Brain* instructs the robot, or the *Connectivity agent*, to switch connectivity to the BSSID of *vAP2*. The *Connectivity agent* connects to the *vAP2* while the closed-loop (*Brain* to robot) is not broken. The closed-loop data in both directions starts passing through the overlay inter-domain link.

5.8.8. Results

In this section we are evaluating the time performance of the Edge robotics federation using DLT by running the experimental scenario as described previously. We run the experimental scenario using (*i*) the PoA-based blockchain instance that uses trusty communication between the domains, and (*ii*) the experimental scenario with the default PoW-based consensus and untrusty communication. As already mentioned, in the untrusty communication the consumer provides only the inter-domain link endpoint, and the provider domain provides back the BSSID of the *vAP2*, upon deployment. We made a number of experimental runs for each of the PoA-based and PoW-based scenarios. In the rest of the section we present the average times for each step in the process.

Three graphs of the time it takes to finalize all the federation procedures are shown on Fig. 5.15. In all graphs the time bars are colored:

> orange - for all federation related procedures as described in Sec. 5.8.5 5.8.6 and Fig. 5.13.
>
> blue - for all procedures that involve deployment of the Edge robotics service or part of it.

To that end, the top graph of Fig. 5.15 presents the accumulated times of the federation procedures in both consumer and provider domain. The average federation time is 19.038 seconds - or the time it takes from the trigger at the consumer orchestrator to the robot connected to the *vAP2*. The break-down in all phases that occur in the consumer domain is presented in the middle graph of Fig. 5.15. It takes 12.97 seconds for the deployment

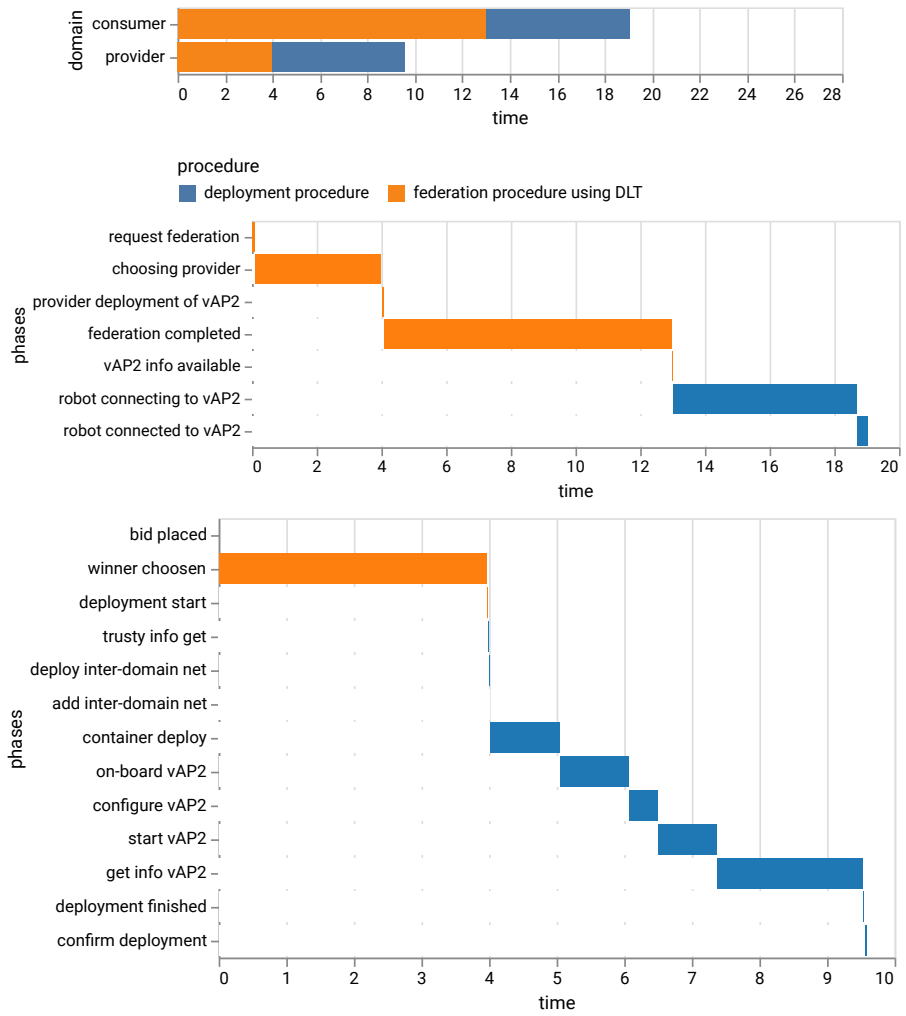


Figure 5.15: Federation using trusty communication - PoA consensus:
(top) summarized phase times; (middle) consumer AD; (bottom) provider AD;

of the *vAP2* to be confirmed at the consumer domain (or phase "federation completed"). In other words, the consumer domain retrieves the BSSID of *vAP2* in provider domain in 12.97 seconds. Then it takes around 6 seconds. for the *Brain* to instruct the robot to discover *vAP2*, disconnect from *vAP1*, and connect to *vAP2*.

The bottom graph of Fig. 5.15 breaks down all the phases in the provider domain, that occur within the previously mentioned 12.97 seconds. The negotiation or bidding process until the provider domain is elected as a winning provider takes 3.98 seconds. More specifically, it takes 3.98 seconds from the time that the provider domain receives the broadcast announcement (shown on Fig. 5.13) until the deployment starts. The establishment of the inter-domain link, on-boarding & instantiation of the *vAP2* takes additional 5.58 seconds.

The results of the PoW-based scenario and untrusty communication is shown in the Fig. 5.16. The graph shows only the accumulated times for both domains. Compared to the PoA-based solution, it is clear that the PoW-based solution takes significantly more time to negotiate and complete the federation process using the Blockchain DLT. Due

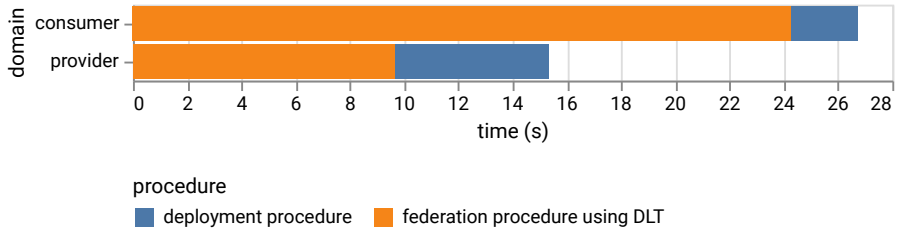


Figure 5.16: Federation using untrusty communication - PoW consensus: summarized times

to the PoW consensus mechanism the "federation completed" phase is completed within 24.3 seconds, nearly double the time of the PoA-based solution.

5.8.9. Remarks on federation in dynamic scenarions using Blockchain for life-cycle management and Edge robotics

In this chapter, we elaborated what is Blockchain as a technology, how it is used in di erent vertical industries, and how di erent consensus mechanisms can be applied to the process of Network Service Federation. We provided an initial design on how the Blockchain based solution can be suited for already established NFV MANO administrative domains with an underlying virtualized infrastructure. We are confident that any of the Blockchain applications can successfully improve the NSF process, especially in dynamic scenarios with unknown users, despite the unpredictability and unreliability of the involved domains or networking resources.

We set up an experimental scenario where we tested the performance of four di erent consensus mechanisms mainly for life-cycle procedures in a NFV environment: instantiation and healing. Through the measurements and the process of adapting the scenario for each of the platforms, we managed to provide additional empirical observation of how each of the platforms may a ect the NSF process. From our experience, every platform and consensus mechanism has its own benefits and drawbacks. In our view, the choice mainly depends of the application nature and the longevity of the solution.

Later, we showcased the application of DLT for federation of an Edge robotics service in a real scenario over an experimental test-bed. To the best of our knowledge, it is the first work that applies federation in an Edge robotics scenario.

Results show that a complete federation process is concluded in around 19 seconds while using more e cient (e.g., PoA) consensus mechanism for more trusted environment. In 42% of the federation time, the consumer domain generates announcement, collects bids and chooses a winning provider domain. In a more distributed environment where large number of unknown domains are expected to join and interact, the PoW consensus mechanism is the preferred option. In this case, the federation concludes in 28 seconds.

In the future, we plan to analyze more consensus mechanisms and execute the scenario using real NFVO MANO infrastructure while federating an end-to-end NFV network service.

6. FEDERATION USING MACHINE LEARNING

6.1. Motivation

Through the technologies such as NFV and Network Slicing, a range of specific requirements is satisfied for each vertical industry. Vertical services are translated into network services (NFV-NS) containing all requirements and instructions to be deployed over underlying network and computational infrastructure. The dynamic slicing enables the optimal resource allocation that reduces the cost of the mobile operators. The 5GT platform leverages the network slicing to e ciently orchestrate di erent NFV network services (NFV-NSs) over single or multiple domains.

In this chapter, we take the role of a service provider at a mobile edge (with limited resource capacity) that aims at maximizing its long-term revenue. On the one hand, in order to satisfy stringent service requirements, service providers need to over-dimension their infrastructure to face system dynamics with high reliability, which results in additional cost and waste of resources. On the other hand, service providers have limited coverage and footprint, and may not have enough resources in certain areas where the service is requested. In this way, when facing a deficit of resources to accommodate new service requests, they need to lease services or resources from another provider according to already-established terms and service level agreements, i.e., *service federation*.

The concept of service orchestration across multiple administrative domains [225] as federation is one of the key features of the 5GT platform. The federation enables each 5GT administrative domain (e.g., mobile operator) to provide broader spectrum of services to the vertical customers with low-cost access to infrastructure capacity and global service coverage in external domains. Through the use of optimized orchestration strategies (e.g., Machine Learning, Artificial Intelligence, ...) [226]–[228], operators may significantly increase their profit.

The main goal of this chapter is to (*i*) formulate a service deployment decision problem with an aim to maximize the administration domain's revenue; (*ii*) and to propose a solution to the decision problem, evaluated through simulation of federation scenarios. Such decisions have important implications on the final price o ered to potential customers.

The first part of this chapter considers simpler scenario, using fixed resource pricing. However, the price associated with federated resources has complex time dynamics [229]. Specifically, the end-user price depends on several variables, such as the availability of resources over time, the demands of service requests, and other business factors, *which are unknown to the requester*. Price fluctuations may lead to service rejections, even when there are resources available in the federated domain, due to negative financial rev-

enue when infrastructure costs are overly high. The second part extends the work using dynamic pricing.

6.2. Related work

In [230] and in Chapter 3, federation mechanisms are classified as (*i*) open federation, where the connectivity between administrative domains changes dynamically; or as (*ii*) pre-established federation, where connections are fixed using business contracts and service level agreements.

There are already several platforms that enable a federation. In the 5GEx project [52], the administrative domains use UNIFY [231], which allows them to expose and exchange information about available resources for federation. In the 5G-TRANSFORMER project [54], [91], [232], [233], a service orchestrator module provides a pre-established service federation to peering administrative domains. A similar approach is adopted by the 5Growth project [234], [235], as the platform enables various federation approaches such as multi-level multi-domain orchestration or open federation using distributed ledger technologies [236]. These works [52], [54], [91], [231]–[236] provide detailed technical and architectural workflows of how federation can be realized in various scenarios for di erent use cases but do not present algorithmic solutions to actually make decisions. This chapter complements these existing works, providing a tool that can be adapted to generate profitable federation decisions in most of the described solutions and platforms.

In 2012, the work in [237], a federation is identified as a challenging mechanism to tackle in virtual network embeddings. The work described in [238] proposes an adaptation of an Alternating Direction Method of Multipliers (ADMM) based algorithm, named Alternating Directions Dual Decomposition (AD3) [239]. The adapted algorithm solves the Virtual Network Embedding (VNE) problem in a decentralized fashion, and in a multi-domain scenario with each domain o ering fixed pricing. [240] formulates the VNE problem in a scenario of non-cooperative domains that bid prices o ered to deploy incoming Virtual Network Functions Forwarding Graphs (VNF-FG). The authors of [240] propose a framework based on Actor-Critic [241] agents for domains to decide the bidding prices, and for clients to maximize the number of deployed VNF-FGs. These works [237], [238], [240] study the VNE problem in-depth but focus on generating decisions that are technically e cient rather than economically profitable. Only [240] uses a Cost-based First Fit (CFF) heuristic algorithm to decide for low-price resources while others do not consider real price dynamics.

The same applies to [242], which proposes a heuristic to assess the VNE in multi-domain networks. The proposed solution, called consolidation-based, is a greedy approach that gives preference to the deployment of VNFs in master paths before service function chains (SFC) su er from branching. The heuristic is enhanced with a feedback mechanism that prevents itself from deploying the SFC over links and servers that have

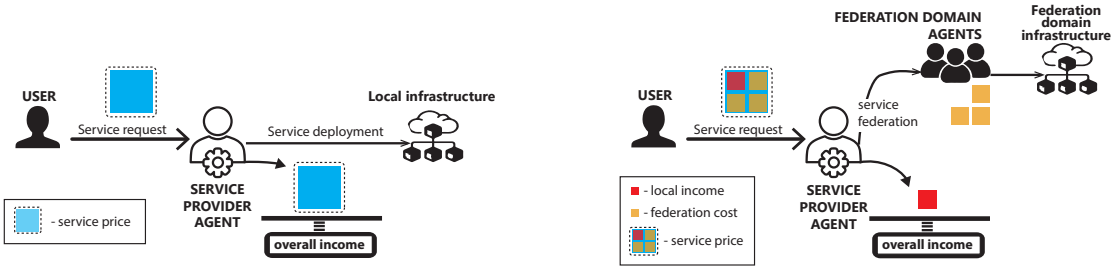
recently failed. They assume static costs and revenues.

The work in [243] presents a distributed solution to compute a VNE in multi-domain networks. The algorithm is inspired by a large-scale graph processing [244] system that uses message-passing to decentralize the computation of the embedding of incoming VNF-FGs. The proposed algorithm iterates over what authors call "super steps", until each domain has locally deployed a part of the VNF-FG. Finally, a master node collects all feasible solutions proposed by each domain and selects the best. The solution ignores costs in the multi-domain infrastructure and the authors lay focus on scalability.

The authors in [245] focus on the problem of migrating service VNFs among domains that belong to a cooperative federation. Inspired by the flow state migration problem[246], the paper proposes an algorithm that coordinates each domain orchestration, so as to assess the migration in a finite time, and satisfying non-functional requirements. The work ignores the price associated with hosting VNFs across di erent domains.

The work in [247] o ers a complete view of a multiple provider federation in 5G networks, and experimental validation of a heuristic approach on top of the described federation model. The work presents an abstraction of the resources that each provider o ers to its neighbors within the federation. The abstraction, called the Bis-Bis node, represents a graph with an abstraction of the resources and connections o ered to the peering providers. The authors use a heuristic algorithm that is based on a greedy backtracking approach [248]. The algorithm is evaluated by means of scalability and running time in a multiple provider experimental setup. In the experiments, the authors in [247] assume fixed prices.

Regarding dynamic pricing scenarios, works such as [249] and [250] tackle resource allocation in mobile edge computing (MEC) and heterogeneous cloud scenarios in an auction-based manner. [249] proposes TCDA (Truthful Combinatorial Double Auction), a solution to determine both the pricing and resource allocation in a MEC scenario where mobile devices bid to obtain resources in the Edge. TCDA solves the associated optimization problem, and ensures that the pricing and resource allocation satisfies properties as social welfare maximization, locality constraints, and budget balance; among others. [250] solves the o oading of computing tasks in a heterogeneous cloud scenario where also users' mobile phones can execute o oaded tasks. Mobile phone users ask and o er resources to o oad and accommodate computing tasks. As in [249], users bid to other mobile users to o oad their tasks, and [250] proposes a greedy algorithm to solve both the allocation of such tasks, and derive payments of the auctioning phase. The proposed greedy reverse auctioning algorithm shows near optimal results by means of utility, execution time, and energy consumption. Moreover, it also satisfies economic properties as truthfulness, and individual rationality (as [249]); and it has been implemented as an Android app. Though both works [249], [250] provide good insights about allocating heterogeneous resources when the auction prices are controlled by the platform, their solutions focus on auction-based platforms, not on scenarios when prices are not under the



(a) Service local deployment

(b) Service federation

Figure 6.1: Business model

platform control, which is the most common scenario. Moreover, [249], [250] do not use real-world pricing data in their analysis and have to rely on assumptions.

6.3. Business model

Inspired by the market of cloud services, we consider a system where a *service provider* o ers services at a *service price* with rate that may vary over time depending on the operator's pricing model. In case a vertical user is willing to pay such a price to deploy service, it makes the request, which arrives at the system at time t, and leaves the system at time $t \quad D$, where D is the deployment time. Once a vertical user request arrives at the service provider, it is the service provider task to decide the best location to deploy the service: either on its own infrastructure, or on another domain within the federation it belongs to — see Fig. 6.1b. Hence, upon each service deployment requested by a vertical user, the service provider can take an action whether the service is deployed locally, deployed in the federated domain, or rejected. Note that the vertical user is not aware of the available resources in the service provider virtualized infrastructure (nor in the federation). Vertical users are only aware of the service price o ered for their request.

The goal is to maximize the long-term revenue of the service provider. The pricing model does have an impact on the arrival process of the service requests: intuitively, lower prices incentivize a higher vertical user arrival rate. Importantly, however, once there is an agreement between customer (e.g., vertical user) and provider (e.g., service provider), the customer pays the agreed fee for every time slot T during which the service is active. In contrast, however, should the service be deployed in the federated domain, the service provider has to pay the federated domain agents a time-varying fee that depends on the dynamics in the federation. If the service is deployed neither locally nor in the federated domain, then the service will be rejected and, in this case, the customer does not pay the service fee, and thus there is no income for the service provider. This business model allows us to exploit opportunistically (uncertain) price fluctuations, which can provide substantial cost savings, yet provide certainty to the end-users, which is essential for vertical sectors.

As a result, every time instance t we have two concurrent cash flows:

(Fig. 6.1a) The service provider uses local resources to grant the request, and therefore the service provider's income is maximum;

(Fig. 6.1b) The service provider uses federated resources, and therefore the provider gets the service price deducted by the *federation cost*, which fluctuates over time.

In this way, we can denote the agent's income, which represents the instantaneous revenue of the service provider, at time t as sum of service prices for the services running on top of the local infrastructure and the sum of the profit from the federated services. In case the service provider runs out of local resources, its agent can federate the service at a cost for the service provider. Hence, the availability of resources has an impact on the instantaneous revenue of the service provider.

6.4. Problem statement

Service federation and resource federation provide the 5GT service providers with various deployment options. In order to increase the revenue and to avoid resource shortage, it is important that for each deployment and or scaling of NFV-NS, the 5GT-SO generates a profitable decision without significant increase of processing and re-calculation for available resources.

In section 6.4.1, we formulate a Reinforcement learning (RL) based decision problem of the 5GT framework, that requires a solution to generate straight-forward deployment decision for each nested NFV-NS, and scale-up request.

Section 6.4.2 presents the intuitive "greedy" approach to generate a decision that maximizes the profit, and section 6.4.3 reformulates the RL-based decision algorithm as an optimization problem to maximize the profit.

Throughout this section we do not consider networking resources, nor a shortage of federated (external) resources. Although they are easily absorbed by the formulation, we decided not to include them to ease the problem readability. For simplification, we use network service to refer to NFV-NS.

6.4.1. Problem description

Let us consider a time-slotted system t : $1, 2, \ldots,$. At the beginning of each slot, (*i*) we may receive one request to deploy a network service (or segment of a network service); or (*ii*) services already deployed may leave the system. A service request may arrive at instant t asking for $c^{(t)}$ CPUs, $m^{(t)}$ memory, and $d^{(t)}$ disk. And by that time the local domain will have $C^{(t)}, M^{(t)}, D^{(t)}$ CPU, memory, and disk, respectively.

The system state is represented as a vector $s^{(t)}$ $\left(c^{(t)}, m^{(t)}, d^{(t)}, C^{(t)}, M^{(t)}, D^{(t)}\right)$. An agent, in the local domain (i.e., the 5GT-SO itself), takes an action $a^{(t)}$ upon a service

arrival at t. The agent can choose whether to consume its own resources for the incoming service ($a^{(t)}$ 0), to ask another domain to deploy it ($a^{(t)}$ 1), or to reject the service ($a^{(t)}$ 2).

The chosen action a ects the instant reward $r^{(t)}$ that the local domain receives. An instant reward is the economic profit which the administrative domain receives per service deployment (locally or in federated domain). Indeed, in our system, the instant reward is determined by the state-action pair, i.e., $r^{(t)}$ $r(s^{(t)}, a^{(t)})$. And as long as the service arriving at t can be deployed locally, our system satisfies

$$r(s^{(t)}, 0) \quad r(s^{(t)}, 1) \quad r(s^{(t)}, 2) \tag{6.1}$$

We assume that the system state $s^{(t)}$ follows a distribution E, unknown *apriori*. Hence, the goal is to find an adequate policy $\pi(s)$: that maps a system state to an action. The above is modeled with an Markov Decision Process (MDP) with transition probability $p(s^{(t\ 1)}\ s^{(t)}, a^{(t)})$, and the agent uses a policy π [251] to give a trajectory of states, actions and rewards $h_{1:T}$: $(s_1, a_1, r_1, \ldots, s_T, a_T, r_T)$ over $\mathbb{R}$. The return from a state at time t is defined as the sum of discounted future rewards

$$R(t) \quad \sum_{i\ t}^{T} \gamma^{(i\ T)} r(s^{(i)}, a^{(i)})$$

with discounted factor γ $[0, 1]$. This is a classic reinforcement learning (RL) problem and the agent's task is to *learn* a policy that maximizes the expected return from the start distribution J $\mathbb{E}_{r^{(i)},s^{(i)}\ E,a^{(i)}\ \pi}[R^{(1)}]$.

It is convenient for RL problems to describe an action-value function describing the expected return after taking an action $a^{(t)}$ given a state $^{(t)}$ (i.e., following policy π) as

$$Q^{\pi}(s^{(t)}, a^{(t)}) \quad \mathbb{E}_{r^{(i\ t)},s^{(i\ t)}\ E,a^{(i\ t)}\ \pi}[R^{(t)}\ \ s^{(t)}, a^{(t)}]$$

and particularly, its recursive representation (Bellman equation):

$$\begin{aligned} &Q^{\pi}(s^{(t)}, a^{(t)}) \\ &\mathbb{E}_{r^{(t)},s^{(t\ 1)}\ E}\left[r(s^{(t)}, a^{(t)}) \quad \gamma \mathbb{E}_{a^{(t\ 1)}\ \pi}[Q^{\pi}(s^{(t\ 1)}, a^{(t\ 1)})]\right] \end{aligned} \tag{6.2}$$

Since the expectation depends only on E, the agent can learn Q^{μ} o -policy, using transitions which are generated from a di erent stochastic process like in Q-learning [252].

6.4.2. A greedy approach

For comparison, we introduce a straightforward "greedy" approach to solve the decision problem by locally deploying an incoming service, as long as there are enough local resources. We latter (see section 6.5) refer this algorithm as the *checker* solution, since it checks the availability of local resources before deciding to federate, or locally deploy a service.

According to the RL problem defined in section 6.4.1, this translates into:

$$a^{(t)} \quad \begin{cases} 0, & c^{(t)} \quad C^{(t)} \quad m^{(t)} \quad M^{(t)} \quad d^{(t)} \quad D^{(t)} \\ 1, & \text{otherwise} \end{cases} \tag{6.3}$$

This approach never rejects a service request, as it assumes that other administrative domains can always host the service. Thus, the instant reward $r^{(t)}$ for federating a service is below the one obtained with a local deployment (as stated in (6.1)), since it is considered that the consumer domain is paying a hosting fee to the provider domain.

6.4.3. Optimization formulation

To check the goodness of a RL solution, we reformulate the problem in section 6.4.1 as an optimization problem.

We use binary variable $a_i^{(t)}$ 1 to abbreviate $a^{(t)}$ i, i 0, 1, 2. Note that $a_i^{(t)}$ 0 means $a^{(t)} \neq i$. Then, $C_f^{(t)}$, $M_f^{(t)}, D_f^{(t)}$ the CPU, memory and disk resources (respectively) freed at time t due to a service leaving. The instant when such leaving service arrived is denoted by $p^{(t)}$ t.

These variables help us to impose the resource conservation constraints:

$$C^{(t)} \quad C^{(t\ 1)} \quad a_0^{(t)} c^{(t\ 1)} \quad a_0^{(p^{(t\ 1)})} C_f^{(t\ 1)} \tag{6.4}$$

$$M^{(t)} \quad M^{(t\ 1)} \quad a_0^{(t)} m^{(t\ 1)} \quad a_0^{(p^{(t\ 1)})} M_f^{(t\ 1)} \tag{6.5}$$

$$D^{(t)} \quad D^{(t\ 1)} \quad a_0^{(t)} d^{(t\ 1)} \quad a_0^{(p^{(t\ 1)})} D_f^{(t\ 1)} \tag{6.6}$$

which state that at time t available resources must consider deployed and freed resources at t 1. As well resources should always stay above zero

$$C^{(t)} \quad 0, M^{(t)} \quad 0, D^{(t)} \quad 0, \quad t \tag{6.7}$$

and only one action is performed at each instant t

$$a_0^{(t)} \quad a_1^{(t)} \quad a_2^{(t)} \quad 1 \tag{6.8}$$

With constraints (6.4) - (6.8) the total reward (r $\sum_t r^{(t)}$) is maximized taking as objective function:

$$\max \sum_t \sum_{i\ 0,1,2} r(s^{(t)}, a_i^{(t)}) a_i^{(t)} \tag{6.9}$$

6.5. Algorithm and simulation results

This section evaluates the performance of a RL solution that solves the deployment decision problem in a federated environment. Section 6.5.1 presents a simple Q-learning solution that solves the RL problem of section 6.4.1.

Section 6.5.2 presents the setup used to derive simulation results. Then, section 6.5.3 begins showing how Q-learning approximates to the optimal solution with an adequate selection of parameters. Finally, end of section 6.5.3 removes the unlimited federated resources assumption, and analyzes how it a ects the Q-learning performance.

6.5.1. Q-learning algorithm

Based on the RL problem defined in section 6.4.1, we derived a Q-learning algorithm that generates decision for each incoming network service deployment at instant t.

The algorithm (Algorithm 2) presents how the agent performs the deployment decision. It decides which action to make for every service request happenning in episode $[0, t_{\text{end}}]$. Then it repeats the whole episode EP times and returns the actions vector $a_t \,{}_0^{t_{\text{end}}}$ to which it has converged. In the beginning, the agent obtains information regarding the state of the system. The state reflects the available computational resources in the environment. Thus at the start, it is assumed that all resources in both (local and federated) domains are available with no network service running (i.e. $state \quad 0$ at time $t \quad 0$).

At some point, a new network service request may arrive. Once the service arrives in the system, the agent decides an action (e.g., local deployment, federation, rejection).

The well-known RL technique, the Q-learning algorithm, uses a state-action matrix Q_T to maximize the reward while iterates through states. Typically the states are the rows and the actions are the columns of the Q-learning matrix. In our work, there are only three actions (i.e., local deployment, federation and rejection), while the number of states depend of the system configuration (i.e., amount of computational resources). Algorithm 2 represents state $s^{(t)}$ as a vector of the available computational resources. Thus, Q_T has $(N^{dim(S)})$ entries with $N \quad \max_i s_i^{(t)}$ being the maximum capacity among all resources represented in the state vector, and $dim(S)$ the number of di erent resources considered (e.g., $dim(S) \quad 3$ if CPU, memory and disk are represented in the state vector).

At the beginning ($t \quad 0$), the Q-learning table is initialized to all zeros. Therefore the agent picks random action at the start, initiating the *learning* process.

Once a decision has been made, the deployment is executed locally, federated, or the service is rejected. The completed action *transits* the environment to a new state and generates instant reward. This means if the network service has been successfully deployed in the local domain, the agent can calculate the remaining available resources (i.e. the new state) and the immediate revenue of the deployment. However, if the agent

Data: environment, EP
Result: $a_t \ ^{t_{\text{end}}}_{0}$
$s^{(t)}$ 0, e 0, r 0;
t_{end} environment.lastService();
while e EP **do**
  Q_T 0;
  t 0;
  r 0;
  while t t_{end} **do**
    if $s^{(t)}$ **then**
      $s^{(t)}$ environment.getState(t);
    end
    $a^{(t)}$ $max_a \left\{ Q_T[s^{(t)}, a] \right\}$ $\frac{1}{e}$ unif[0, 2];
    environment DeployService($s^{(t)}, a^{(t)}$);
    (s , $r^{(t)}$) environment.current(t);
    $Q_T[s^{(t)}, a^{(t)}]$ (1 α)$Q_T[s^{(t)}, a^{(t)}]$ $\alpha \left(r^{(t)} \quad \gamma max_a Q_T[s\ , a] \right)$;
    s s ;
    r r $r^{(t)}$;
    t t 1;
  end
  e e 1;
end

Algorithm 2: Q-learning decision

decided for local deployment despite the lack of local resource, no transition to other state occurs, but a negative reward is generated for the performed action.

The instant reward and state transition enables the agent to *learn* thanks to the Bellman equation (6.2). Additional fixed parameters represent the learning rate (α) and the discount factor (γ). The chosen values for the learning rate and the discount factor directly impact on the performance of the algorithm, this matter is evaluated in section 6.5.3. Afterwards the state of the system is updated with the new state and the instant revenue is added to the total revenue.

For each next incoming network service, all the steps of the algorithm are executed. The agent is continuously *learning* and every next decision for an action is more knowledge-driven.

6.5.2. Simulation environment

We have set-up a simulation environment consisting of a local domain and external - federated domain. Both domains are capable of network service deployment that contains only computational resources (i.e. CPU, memory, disk). The local domain is configured to have limited amount of computational resources, whereas the federated domain is optionally set to have both limited and infinite computational resources.

Incoming requests for network service deployment are based on an Poisson arrival process of big B and small S services. The parameters for both big B and small S services are shown in Table 6.1. The arrival process in the environment simulates 30 days of Poission arrivals of big and small services.

Each service request can be either *i)* deployed locally, *ii)* federated or *iii)* rejected. Whatever algorithm is used, the agent gets full reward(r_{big} or r_{small}) for local deployment; for federation, the consumer domain agent gets reward 1; and for rejection the reward is zero. In case the agent decides to locally deploy or federate a service over a full infrastructure, the agent receives penalty(r_{big} or r_{small}).

Table 6.1: Service arrivals

	Big B	Small S
Arrival rate	λ_B	λ_S $6\lambda_B$
CPU	10	1
Memory	20 GB	2 GB
Storage	1024 GB	20 GB
Life-time	5-10 days	1-4 days
Revenue	r_{big} $10r_{small}$	r_{small}

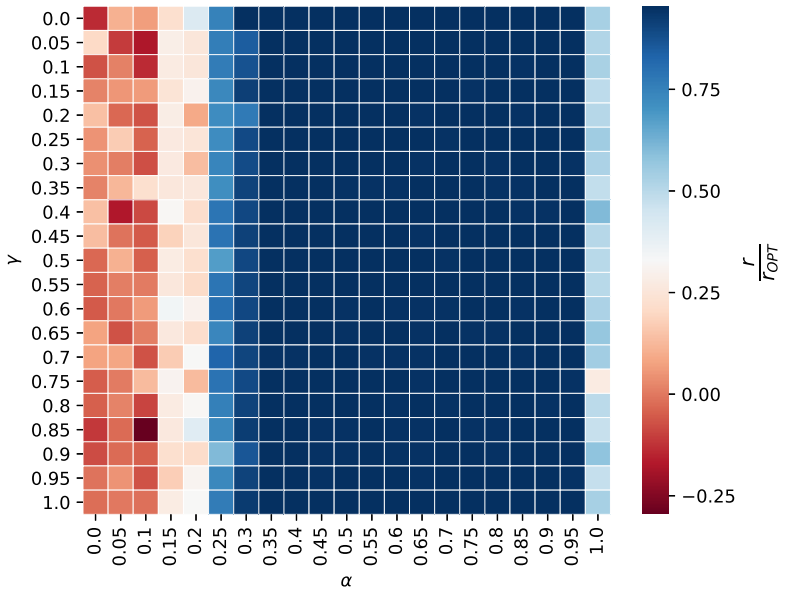


Figure 6.2: Possible (α, γ) combinations under the assumption of unlimited federated resources.

6.5.3. Performance evaluation

To evaluate the performance of the Q-learning algorithm described in 6.5.1, we performed set of simulation experiments.

First, the performance of the Q-learning algorithm depends on how the learning rate (α) and the discount factor (γ) are set up. To tune it and derive the best tuple (α, γ), we performed a set of simulations (400 simulations) exploring the combinations in the value range [0, 1] with step 0.5 for each variable. Using the environment described in previous section 6.5.2, the local resources are finite (e.g. CPU 10; Memory 100GB, Storage 400GB), where the federated domain has unlimited resources. Fig. 6.2 shows the simulation results, presenting the total revenue that the algorithm produces for each tuple (α, γ). The x-axis presents the range of values chosen for the learning rate (α), the y-axis represent the discount factor (γ). The gradient color bar (on the right) present the indicator for the normalized reward revenue (i.e., obtained revenue r divided by the optimal policy π revenue r_{OPT}), ranging from low (*dark-red*) to high (*dark blue*). The results intelligibly show that the learning rate is the major contributor to profitable performance of the Q-learning algorithm. The learning rate's best performance is in the [0.35, 0.95] range, with negligible value range for the discount factor.

The performance evaluation for the Q-learning algorithm proceeds with 80-episodes simulation using the tuple values (α 0.95, γ 0.9). Each episode runs the same arrival process of small and big services and the episode begins with all available local resources (CPU 10; Memory 100GB, Storage 400GB) and unrealistically infinite resources in the federated domain. I.e., the environment is reset to initial conditions at the start of each

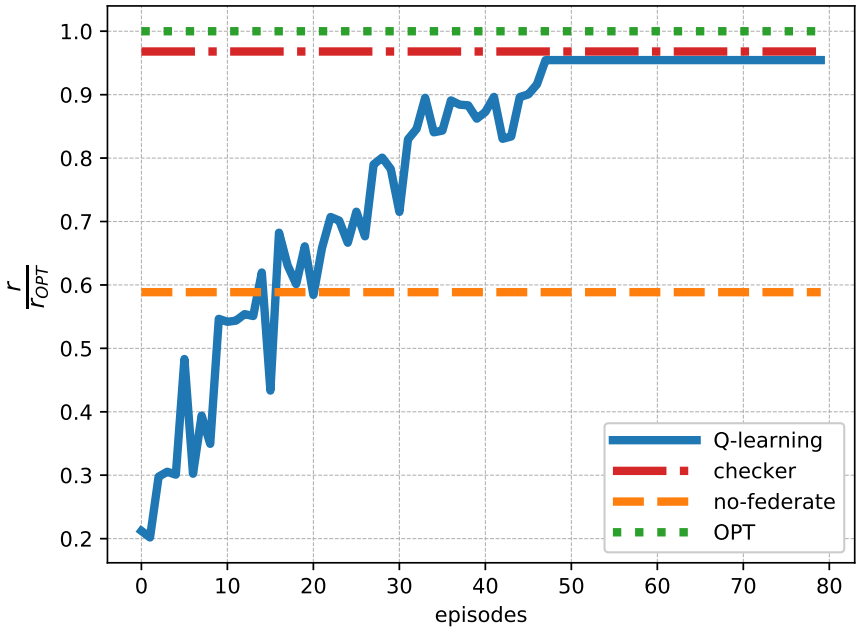


Figure 6.3: Convergence of best (α, γ) combination under the assumption of unlimited federated resources.

episode, while the Q-learning algorithm *learns* for the whole duration of the experiment.

The simulation results shown on Fig. 6.3, present all the episodes on the x-axis and the normalized revenue (per episode) on the y-axis. The solid blue curve presents the Q-learning reward evolution as the episodes increase. In the Fig. 6.3 we also introduced the results from three benchmark approaches:

The orange (slash-slash) line represent the total reward of a single domain system, or if the federation option does not exist and there is no federated domain.

The red (slash-dotted) line represent the total revenue of the "greedy" approach described in section 6.4.2. Note that, here the agent first performs additional processing (e.g. scanning local resource capacity, checking databases, etc.,) before generating decision. If scan results are negative (insu cient resources), the agent federates the service in the federated domain. There are zero rejections in this case and the decision is made after the solution is checked and confirmed.

The green (dotted) line represent the optimal solution generated from the optimization formulation of section 6.4.3 through the use of AMPL [253] and Gurobi [254].

Given a service arrival realization, results in Fig. 6.3 show that the Q-learning algorithm performs little over 95% of the optimal profit after 48 episodes. The "greedy" *checker* approach is slightly better (< 1.5%) than the Q-learning algorithm. The "greedy" *checker* for each check adds more than *1.68 sec* [62] processing time to generate *a posteriori* decision. The added time corresponds to the size of the local domain resources that

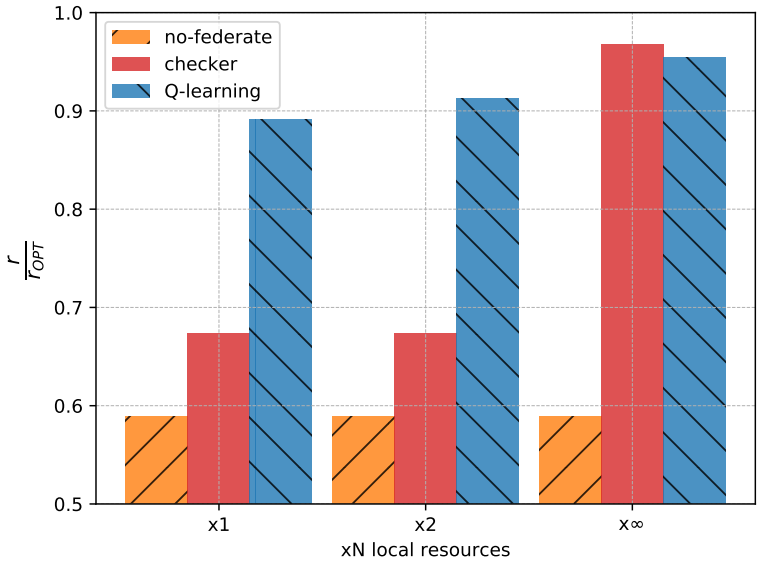


Figure 6.4: Performance of Q-learning as federation resources increase. Training lapse of EP 200 episodes.

needs to be checked. On the contrary, the Q-learning generates *a priori* decision. Having a stable scenario with infinite resources in the federated domain enables a near optimal solution for the "greedy" *checker*.

For more realistic evaluation of the Q-learning algorithm, we performed additional simulation experiments where the federation domain has finite resources. Therein the federation domain has been set to [*x1*, *x2*] times the local domain respectively. State vector $s^{(t)}$ now has three more components to represent the available CPU, memory and disk in the federated domain. Thus, $dim(S)$ duplicates, and the state space grows from (N^3) to (N^6), where N $\max_i s_i^{(t)}$. Due to dynamicity of the scenario, the agent can not obtain precise status of the external resources. Therefore the "greedy" *checker* performs only local check and tries to federate a service request with no rejection. A penalty is applied when the agent tries to deploy over a full federated domain.

Figure 6.4 compares the [*x1*, *x2*] scenarios to the infinite scenario of Fig. 6.3. Despite the growth in the state space, the Q-learning outperforms the "greedy" *checker* within 200 training episodes. Results imply that without the perfect knowledge in the "greedy" *checker*, the Q-learning can accommodate and *learn* the system dynamicity. Existence of penalty probability in the "greedy" *checker* is due to imprecise and highly complex operation to obtain information for all external resources.

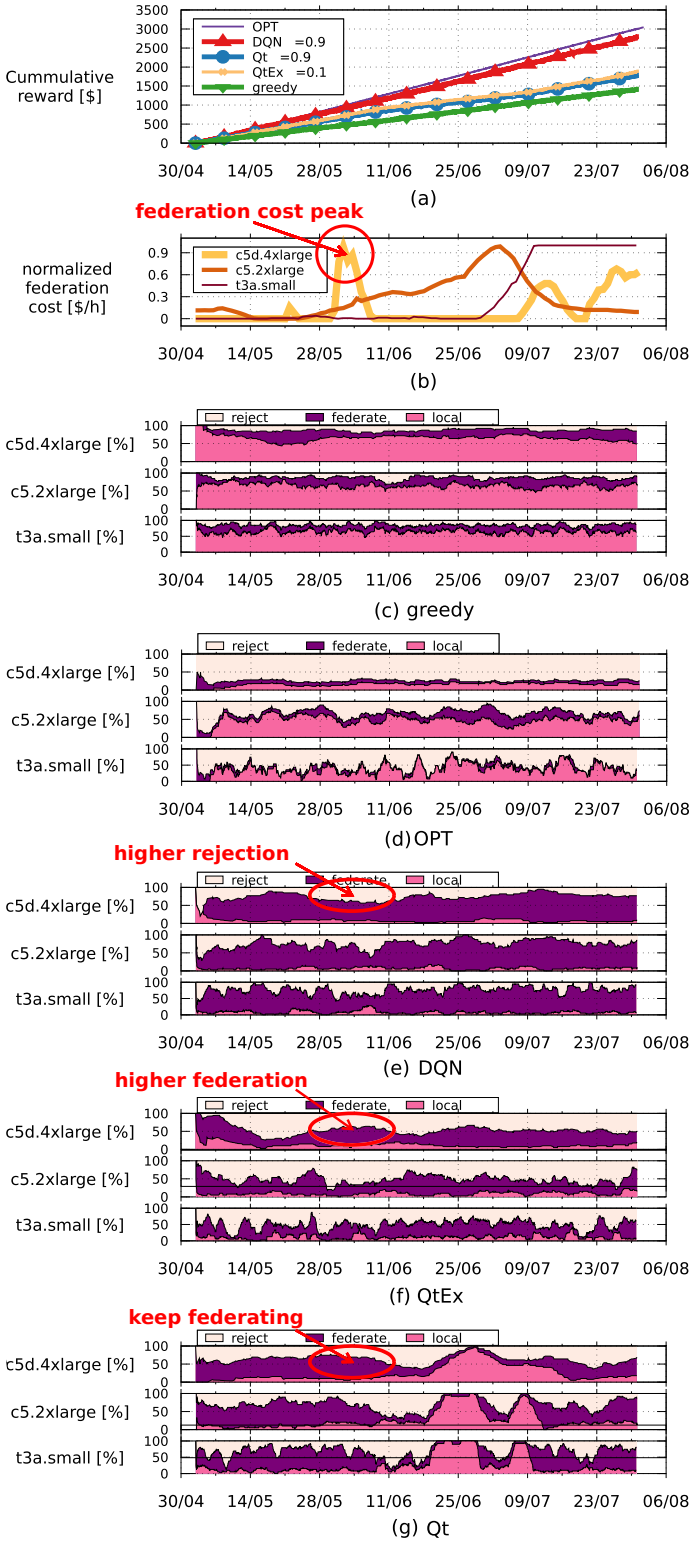


Figure 6.5: (a) Cumulative reward of each solution during for a dynamic pricing dataset; (b) the normalized *federation cost* over time per service; and the percentage of instances rejected, federated, or locally deployed by (c) Greedy, (d) Optimal solution, (e) Deep Q Network, (f) Q-learning exploration, and (g) Q-learning simple solution.

6.6. Application of more complex reinforcement learning algorithms and dynamic pricing

The previous work has been extended for more complex scenario which involves dynamic pricing for each network service o ered to a vertical user.

As previously explained in the business model in Section 6.3, the service provider charges the vertical user at the service arrival time for the deployment of the service. Then the service provider can decide which deployment option is the most profitable. In the case of the dynamic pricing, the optimal decision for each service deployment does not entirely depend on the resource requirements, but often depends of the arrival time of the service requested by the vertical user. The federation cost for a given service is linear to the arrival rate of new services. For example, if the number of services being federated is increasing, the federated domain may increase their federation cost. Thus a service provider in order to make a near-optimal profitable decision, needs to federate at the lowest federation cost. More in-depth is provided in [255].

The performed experiments used a real data from a cloud provider which o ers three di erent kind of services. These services depict the resource requirements for each of them represented as *t3a.small* - low, *c5.2xlarge* - medium, and *c5d.4xlarge* - high. On Figure 6.5-b are shown the federation costs for each of the three o ered services for a time duration of three months. Note that these are real data obtained for the period between 30th of April 2020 and 27th of July 2020. We applied four di erent algorithms: (*i*) greedy; (*ii*) Deep Q Network (DQN); (*iii*) Q-learning - from Section 6.5.1; and (*iv*) a basic Q-learning without any state exploration. For bench-marking purposes, the optimal solution is presented as well. In the Figure 6.5 is presented the evolution of all decisions with the respect of the as a percentage of all service requests received at each time instant. Figure 6.5-a presents the cumulative reward that clearly shows the near optimal revenue (around 90%) is achieved through the use of more complex techniques such as DQN. Even though the Q learning algorithm is not able to depict sudden federation cost peak as the DQN, both Q-learning techniques manage to achieve 30% higher cumulative reward.

6.7. Remarks on generating federation decision using reinforcement learning techniques

This chapter studies the performance of Q-learning to generate a profitable deployment decision in a federated ecosystem. Results show that Q-learning algorithms can provide the 5GT administrative domains near optimal earnings without additional time-consumption in an ideal scenario setup. In more realistic scenarios, the Q-learning solution outperforms the "greedy" *checker* approach, with better computational e ciency and *apriori* decision.

In a real environment where service price o ering is dynamically changing as well

as the federation cost, there are more complex reinforcement learning algorithms that outperform the Q-learning algorithm. Such example is DQN which is able to reach 90% of the optimal revenue even for sudden peaks in price o erings.

7. CONCLUSION

This thesis aims to improve evolution of the mobile communications through the integration of two pillar technologies - MEC and NFV - into the existing and future 5G systems. After providing an introduction of both technologies, the thesis dives into horizontal integration of MEC into NFV. The proposed integration is generated through deep analysis and categorization of the expected NFV integration issues reported in ETSI MEC [13]. An experimental scenario that emulates Edge robotics as a MEC deployment has been deployed at the University premises presenting the benefits of MEC deployments. The deployed scenario is taken as a starting point to showcase a proposed solution of the MEC in NFV integration through step-by-step deployment of the introduced Edge robotics MEC application in a virtualized NFV MANO environment.

The second part of the thesis mainly focuses on how both MEC and NFV can play pivotal role into enabling vertical industries to deploy their services in a multi-domain heterogeneous environments. The main feature that enables multi-domain deployment - federation - is defined at the beginning. Beside the definition itself, the thesis generates a characterization of the federation. Based on the di erent aspects, the main focus is the deployment of network services or resources as well as the deployment environments. After describing the design and implementation of the federation feature in di erent H2020 projects (5G-TRANSFORMER and 5G-Coral), the thesis focuses on presenting how the federation is achieved in static and dynamic environments. For the static environments, an e-health application is used to showcase the federation of Edge MEC resources that are federated closer to an emergency patient to satisfy requirements of AR VR application used by paramedics.

Following the static environment, the thesis tackles more complex problem of enabling multi-domain deployment of vertical End-to-end (E2E) services in dynamic environments. Since the dynamic environments are rapidly changing, with urgency of establishing agreements between domains, this thesis proposes the use of Blockchain technology to tackle the main issues. The main characteristics of Blockchain are described at the beginning with the main focus of how the dynamic challenges are overcome through Blochain application. The applicability is demonstrated by an experimental scenario involving three domains with the use of di erent Blockchain platforms and consensus mechanisms. Results show the federation execution time and profiling the computing resources. Additionally, an Edge Robotics scenario showcases the main benefits of real-life integration of the Blockchain federation for dynamic environments.

Finally, the thesis focuses in applying ML Reinforcement algorithms - Q-learning - to present how service providers and operators would benefit from the implementation of the federation feature for deployment of multi-domain E2E vertical services. In this part, the service provider perspective is taken with the final goal of generating profitable

decisions for network service deployments. The work is extended towards more realistic setups (e.g., using dynamic pricing).

BIBLIOGRAPHY

[1] H. Q. Tran, C. Van Phan, and Q.-T. Vien, "An overview of 5g technologies," *Emerging Wireless Communication and Network Technologies*, pp. 59–80, 2018.

[2] W. Xiang, K. Zheng, and X. S. Shen, *5G mobile communications*. Springer, 2016.

[3] M. Series, "Imt vision–framework and overall objectives of the future development of imt for 2020 and beyond," *Recommendation ITU*, vol. 2083, p. 0, 2015.

[4] N. Alliance, "5g white paper," *Next generation mobile networks, white paper*, vol. 1, 2015.

[5] S. Redana *et al.*, "5g ppp architecture working group: View on 5g architecture, version 3.0," 2019.

[6] Rysavy Research and 5G Americas, *Global 5G: Rise of a transformational technology*,
, (Accessed on 01 15 2022).

[7] 3GPP, *Technical Specification Group Services and System Aspects; Release 15 Description; Summary of Rel-15 Work Items, TR 21.915, v. 15.0.0*, Sep. 2019.

[8] ——, *Technical Specification Group Services and System Aspects; Release 16 Description; Summary of Rel-16 Work Items (Release 16), TR 21.915, v. 16.1.0*, Jan. 2022.

[9] ——, *Technical Specification Group Services and System Aspects; Release 17 Description; Summary of Rel-17 Work Items (Release 17), TR 21.917, v. 0.1.0*, Nov. 2021.

[10] ——, *Technical Specification Group Services and System Aspects; Release 18 Description; Summary of Rel-18 Work Items (Release 18), TR 21.918, v. 0.1.0*, Nov. 2021.

[11] ETSI, "Network Functions Virtualisation (NFV); Architectural Framework," European Telecommunications Standards Institute (ETSI), Group Specification (GS) NFV 002 v1.1.1, Oct. 2013.

[12] M. Ersue, "Etsi nfv management and orchestration-an overview," *Presentation at the IETF*, vol. 88, 2013.

[13] ETSI, "Mobile Edge Computing (MEC); Deployment of Mobile Edge Computing in an NFV environment," European Telecommunications Standards Institute (ETSI), Group Report (GR) 017 V1.1.1, Feb. 2018.

[14] ——, "Mobile Edge Computing (MEC); Framework and Reference Architecture," European Telecommunications Standards Institute (ETSI), Group Specification (GS) 003 v2.1.1, Jan. 2018.

[15] ——, “Mobile Edge Computing (MEC); Radio Network Information API,” European Telecommunications Standards Institute (ETSI), Group Specification (GS) 012 v1.1.1, Jul. 2017.

[16] ——, “Multi-access Edge Computing (MEC); WLAN Information API,” European Telecommunications Standards Institute (ETSI), Group Specification (GS) 028 v2.0.1, Jul. 2018.

[17] ——, “Mobile Edge Computing (MEC); Location API,” European Telecommunications Standards Institute (ETSI), Group Specification (GS) 013 v1.1.1, Jul. 2017.

[18] ——, “Mobile Edge Computing (MEC); Bandwidth Management API,” European Telecommunications Standards Institute (ETSI), Group Specification (GS) 015 v1.1.1, Oct. 2017.

[19] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, “Collaborative mobile edge computing in 5g networks: New paradigms, scenarios, and challenges,” *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, Apr. 2017. DOI: .

[20] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, “A Survey of Research on Cloud Robotics and Automation,” *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 398–409, Apr. 2015. DOI: .

[21] 5G-CORAL, “5G-CORAL initial system design, use cases, and requirements,” Deliverable 1.1, Apr. 2018.

[22] 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)*, Jun. 2018, pp. 1–9. DOI: .

[23] M. Erol-Kantarci and S. Sukhmani, “Caching and computing at the edge for mobile augmented reality and virtual reality (ar vr) in 5g,” in *Ad Hoc Networks*, Y. Zhou and T. Kunz, Eds., Cham: Springer International Publishing, 2018, pp. 169–177.

[24] H. Hu *et al.*, “Intelligent video surveillance based on mobile edge networks,” in *2018 IEEE International Conference on Communication Systems (ICCS)*, Dec. 2018, pp. 286–291. DOI: .

[25] T. Iwai and A. Nakao, “Demystifying myths of mec: Rethinking and exploring benefits of multi-access mobile edge computing,” in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, Oct. 2018, pp. 1–4. DOI: .

[26] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-edge computing architecture: The role of mec in the internet of things," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 84–91, Oct. 2016. DOI: .

[27] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. ZHANG, "Mobile-edge computing for vehicular networks: A promising network paradigm with predictive o -loading," *IEEE Vehicular Technology Magazine*, vol. 12, no. 2, pp. 36–44, Jun. 2017. DOI: .

[28] G. Valecce, S. Strazzella, and L. A. Grieco, "On the interplay between 5g, mobile edge computing and robotics in smart agriculture scenarios," in *Ad-Hoc, Mobile, and Wireless Networks*, M. R. Palattella, S. Scanzio, and S. Coleri Ergen, Eds., Cham: Springer International Publishing, 2019, pp. 549–559.

[29] 3GPP, "Service requirements for next generation new services and markets," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 22.261 v16.4.0, Jun. 2018.

[30] 5TONIC, "5TONIC: Open-research and innovation laboratory for 5G technologies," [Online], 2018, [Accessed 13 Sep. 2018]. [Online]. Available: .

[31] X. Wang, L. Wang, A. Mohammed, and M. Givehchi, "Ubiquitous manufacturing system based on Cloud: A robotics application," *Robotics and Computer-Integrated Manufacturing*, vol. 45, pp. 116–125, 2017, Special Issue on Ubiquitous Manufacturing (UbiM). DOI: .

[32] R. Rahimi *et al.*, "An Industrial Robotics Application with Cloud Computing and High-Speed Networking," in *2017 First IEEE International Conference on Robotic Computing (IRC)*, Apr. 2017, pp. 44–51. DOI: .

[33] A. Hammad, S. S. Ali, and A. S. T. Eldien, "A novel implementation for FastSLAM 2.0 algorithm based on cloud robotics," in *2017 13th International Computer Engineering Conference (ICENCO)*, Dec. 2017, pp. 184–189. DOI: .

[34] J. Wan, S. Tang, Q. Hua, D. Li, C. Liu, and J. Lloret, "Context-Aware Cloud Robotics for Material Handling in Cognitive Industrial Internet of Things," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2272–2281, Aug. 2018. DOI: .

[35] J. Karjee, S. Behera, H. K. Rath, and A. Simha, "Distributed Cooperative Communication and Link Prediction in Cloud Robotics," in *2017 IEEE International Conference on Sensing, Communication and Networking (SECON Workshops)*, Jun. 2017, pp. 1–7. DOI: .

[36] ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration; Os-Ma-Nfvo reference point - Interface and Information Model Specification," European Telecommunications Standards Institute (ETSI), Group Specification (GS) ETSI GS NFV-IFA 013 V2.4.1, Feb. 2018.

[37] ——, "Network Functions Virtualisation (NFV); Management and Orchestration; Ve-Vnfm reference point - Interface and Information Model Specification," European Telecommunications Standards Institute (ETSI), Group Specification (GS) ETSI GS NFV-IFA 008 V3.1.1, Aug. 2018.

[38] K. Antevski, M. Groshev, L. Cominardi, C. Bernardos, A. Mourad, and R. Gazda, "Enhancing edge robotics through the use of context information," in *Proceedings of the Workshop on Experimentation and Measurements in 5G*, ACM, 2018, pp. 7–12.

[39] A. Reznik *et al.*, "Developing software for multi-access edge computing," *ETSI White Paper*, vol. 20, 2017.

[40] ETSI, "Network Functions Virtualisation (NFV); Management and Orchestration," European Telecommunications Standards Institute (ETSI), Group Specification (GS) ETSI GS NFV-MAN 001 V1.1.1, Dec. 2014.

[41] G. Cattaneo, F. Giust, C. Meani, D. Munaretto, and P. Paglierani, "Deploying cpu-intensive applications on mec in nfv systems: The immersive video use case," *Computers*, vol. 7, no. 4, p. 55, 2018.

[42] F. Giust *et al.*, "MEC deployments in 4G and evolution towards 5G," *ETSI, White Paper, under publication in February*, 2018.

[43] F. Giust *et al.*, "Multi-access edge computing: The driver behind the wheel of 5g-connected cars," *CoRR*, vol. abs 1803.07009, 2018. arXiv: . [Online]. Available: .

[44] V. Sciancalepore, F. Giust, K. Samdanis, and Z. Yousaf, "A double-tier mec-nfv architecture: Design and optimisation," in *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*, Oct. 2016, pp. 1–6. DOI: .

[45] T. V. Doan, G. T. Nguyen, A. Kropp, and F. H. P. Fitzek, "APMEC: an automated provisioning framework for multi-access edge computing," *CoRR*, vol. abs 1805.09251, 2018. arXiv: . [Online]. Available: .

[46] T. Doan-Van, A. Kropp, G. T. Nguyen, H. Salah, and F. H. P. Fitzek, "Programmable first: Automated orchestration between mec and nfv platforms," in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan. 2019, pp. 1–2. DOI: .

[47] G. A. Carella, M. Pauls, T. Magedanz, M. Cilloni, P. Bellavista, and L. Foschini, "Prototyping nfv-based multi-access edge computing in 5g ready networks with open baton," in *2017 IEEE Conference on Network Softwarization (NetSoft)*, Jul. 2017, pp. 1–4. DOI: .

[48] T. Taleb, P. A. Frangoudis, I. Benkacem, and A. Ksentini, "Cdn slicing over a multi-domain edge cloud," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2019. DOI: .

[49] S. Arora, P. A. Frangoudis, and A. Ksentini, "Exposing radio network information in a mec-in-nfv environment: The rnisaas concept," 2019.

[50] L. Yala, P. A. Frangoudis, and A. Ksentini, "Latency and availability driven vnf placement in a mec-nfv environment," in *2018 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2018, pp. 1–7. DOI: .

[51] L. Cominardi, "Multi-domain federation: scope, challenges, and opportunities," in *Workshop in 3rd IEEE Conference on Network Softwarization, Bologna, Italy*, IEEE Conference on Network Softwarization, Jul. 2017.

[52] C. J. Bernardos et al., "5gex: Realising a europe-wide multi-domain framework for software-defined infrastructures," *Transactions on Emerging Telecommunications Technologies*, vol. 27, no. 9, pp. 1271–1280, 2016.

[53] A. Oliva et al., "5G-TRANSFORMER: Slicing and Orchestrating Transport Networks for Industry Verticals," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 78–84, 2018. DOI: .

[54] J. Baranda et al., "Realizing the Network Service Federation Vision: Enabling Automated Multidomain Orchestration of Network Services," *IEEE Vehicular Technology Magazine*, vol. 15, no. 2, pp. 48–57, 2020.

[55] 5G-CORAL et al., *Deliverable 3.2 - refined design of 5g-coral*, , (Accessed on 30 01 2022).

[56] A. Francescon et al., "X–mano: Cross–domain management and orchestration of network services," in *2017 IEEE Conference on Network Softwarization (NetSoft)*, IEEE, 2017, pp. 1–5.

[57] GSMA, "Operator platform concept whitepaper," Feb. 2020, (Accessed on 02 22 2021). [Online]. Available: .

[58] A. Boubendir et al., "Federation of cross-domain edge resources: A brokering architecture for network slicing," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*, IEEE, 2018, pp. 415–423.

[59] K. Antevski et al., “Dlt federation for edge robotics,” in *2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, IEEE, 2020, pp. 71–76.

[60] R.B. Uriarte et al., “Distributed service-level agreement management with smart contracts and blockchain,” *Concurrency and Computation: Practice and Experience*, e5800, 2020.

[61] V. Scoca et al., “Smart contract negotiation in cloud computing,” in *2017 IEEE 10Th international conference on cloud computing (CLOUD)*, IEEE, 2017, pp. 592–599.

[62] J. Mangues-Bafalluy et al., “5g-transformer service orchestrator: Design, implementation, and evaluation,” in *2019 European Conference on Networks and Communications (EuCNC)*, Jun. 2019, pp. 31–36. DOI:
.

[63] ETSI, “Network Functions Virtualisation (NFV); Management and Orchestration,” European Telecommunications Standards Institute (ETSI), Group Specification (GS) NFV 001 v1.1.1, Dec. 2014.

[64] L. Valcarenghi et al., “A framework for orchestration and federation of 5g services in a multi-domain scenario,” in *Proceedings of the Workshop on Experimentation and Measurements in 5G*, ser. EM-5G’18, Heraklion, Greece: ACM, 2018, pp. 19–24. DOI: . [Online]. Available:
.

[65] X. Li et al., “Service orchestration and federation for verticals,” in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, Apr. 2018, pp. 260–265. DOI: .

[66] K. Antevski et al., “Resource orchestration of 5g transport networks for vertical industries,” in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2018, pp. 158–163. DOI:
.

[67] 5G-CORAL et al., *Deliverable 3.1 - initial design of 5g-coral orchestration and control system*, : 5g-coral.eu wp-content uploads 2018 06 D3.19802.pdf.

[68] H. Ullah, N. Gopalakrishnan Nair, A. Moore, C. Nugent, P. Muschamp, and M. Cuevas, “5G Communication: An Overview of Vehicle-to-Everything, Drones, and Healthcare Use-Cases,” *IEEE Access*, vol. 7, pp. 37 251–37 268, 2019.

[69] J. Lloret, L. Parra, M. Taha, and J. Tomás, “An architecture and protocol for smart continuous ehealth monitoring using 5g,” *Computer Networks*, vol. 129, pp. 340–351, 2017, Special Issue on 5G Wireless Networks for IoT and Body Sensors. DOI: . [Online]. Available:
.

[70] T. Mohammed, A. Albeshri, I. Katib, and R. Mehmood, "Ubipriseq—deep reinforcement learning to manage privacy, security, energy, and qos in 5g iot hetnets," *Applied Sciences*, vol. 10, no. 20, p. 7120, 2020.

[71] N. Janbi, I. Katib, A. Albeshri, and R. Mehmood, "Distributed artificial intelligence-as-a-service (DAIaaS) for smarter IoE and 6G environments," *Sensors*, vol. 20, no. 20, p. 5796, 2020.

[72] T. Muhammed, R. Mehmood, A. Albeshri, and I. Katib, "UbeHealth: a personalized ubiquitous cloud and edge-enabled networked healthcare system for smart cities," *IEEE Access*, vol. 6, pp. 32 258–32 285, 2018.

[73] E. M. Abou-Nassar, A. M. Iliyasu, P. M. El-Kafrawy, O.-Y. Song, A. K. Bashir, and A. A. Abd El-Latif, "DITrust chain: towards blockchain-based trust models for sustainable healthcare IoT systems," *IEEE Access*, vol. 8, pp. 111 223–111 238, 2020.

[74] A. A. Abd El-Latif, B. Abd-El-Atty, W. Mazurczyk, C. Fung, and S. E. Venegas-Andraca, "Secure data encryption based on quantum walks for 5G Internet of Things scenario," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 118–131, 2020.

[75] Y. Liu, J. Peng, J. Kang, A. M. Iliyasu, D. Niyato, and A. A. Abd El-Latif, "A secure federated learning framework for 5G networks," *IEEE Wireless Communications*, vol. 27, no. 4, pp. 24–31, 2020.

[76] H. Moustafa, E. M. Schooler, G. Shen, and S. Kamath, "Remote monitoring and medical devices control in ehealth," in *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, 2016, pp. 1–8.

[77] G. Muhammad, M. F. Alhamid, M. Alsulaiman, and B. Gupta, "Edge computing with cloud for voice disorder assessment and treatment," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 60–65, 2018.

[78] B. W. Munzer, M. M. Khan, B. Shipman, and P. Mahajan, "Augmented reality in emergency medicine: A scoping review," *J Med Internet Res*, vol. 21, no. 4, e12368, Apr. 2019. DOI: . [Online]. Available: .

[79] K. Klinker, M. Wiesche, and H. Krcmar, "Digital transformation in health care: Augmented reality for hands-free service innovation," *Information Systems Frontiers*, Jun. 2019. DOI: .

[80] L. Carenzo, F. L. Barra, P. L. Ingrassia, D. Colombo, A. Costa, and F. Della Corte, "Disaster medicine through google glass," *European Journal of Emergency Medicine*, vol. 22, no. 3, pp. 222–225, 2015.

[81] A. Follmann, M. Ohligs, N. Hochhausen, S. K. Beckers, R. Rossaint, and M. Czaplik, "Technical support by smart glasses during a mass casualty incident: A randomized controlled simulation trial on technically assisted triage and telemedical app use in disaster medicine," *Journal of medical Internet research*, vol. 21, no. 1, e11939, 2019.

[82] MarketWatch, *Mhealth market: Industry size, growth, analysis and forecast of 2023*,

, Accessed on: 05 Mar 2020.

[83] C.vMeyer, *Social life of health information*,

, Accessed on: 05 Mar 2020.

[84] E. H. Bradley *et al.*, "Strategies for reducing the door-to-balloon time in acute myocardial infarction," *New England Journal of Medicine*, vol. 355, no. 22, pp. 2308–2320, 2006.

[85] E. B. Wu, N. Arora, A. C. Eisenhauer, and F. S. Resnic, "An analysis of door-to-balloon time in a single center to determine causes of delay and possibilities for improvement," *Catheterization and Cardiovascular Interventions*, vol. 71, no. 2, pp. 152–157, 2008.

[86] B. Shi, J. Yang, Z. Huang, and P. Hui, "O oading guidelines for augmented reality applications on wearable devices," in *Proceedings of the 23rd ACM international conference on Multimedia*, 2015, pp. 1271–1274.

[87] J. Dolezal, Z. Becvar, and T. Zeman, "Performance evaluation of computation o oading from mobile device to the edge of mobile network," in *2016 IEEE Conference on Standards for Communications and Networking (CSCN)*, IEEE, 2016, pp. 1–7.

[88] S. Sukhmani, M. Sadeghi, M. Erol-Kantarci, and A. El Saddik, "Edge caching and computing in 5g for mobile ar vr and tactile internet," *IEEE MultiMedia*, vol. 26, no. 1, pp. 21–30, 2018.

[89] M. Erol-Kantarci and S. Sukhmani, "Caching and computing at the edge for mobile augmented reality and virtual reality (ar vr) in 5g," *Ad Hoc Networks*, pp. 169–177, 2018.

[90] J. Baranda et al., "Realizing the network service federation vision: Enabling automated multidomain orchestration of network services," *IEEE Vehicular Technology Magazine*, vol. 15, no. 48–57, pp. 48–57, 2020. DOI:

.

[91] ——, "NFV Service Federation: enabling Multi-Provider eHealth Emergency Services," in *2020 International Conference on Computer Communications (INFOCOM'20)*, Jul. 2020.

[92] K. Antevski *et al.*, "Resource orchestration of 5g transport networks for vertical industries," in *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, IEEE, 2018, pp. 158–163.

[93] J. Martın-Pérez, L. Cominardi, C. J. Bernardos, A. de la Oliva, and A. Azcorra, "Modeling mobile edge computing deployments for low latency multimedia services," *IEEE Transactions on Broadcasting*, vol. 65, no. 2, pp. 464–474, 2019.

[94] J. Martın-Peréz, F. Malandrino, C.-F. Chiasserini, and C. J. Bernardos, "Okpi: All-kpi network slicing through e cient resource allocation," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, IEEE, 2020, pp. 804–813.

[95] *Mixed reality toolkit*,
, Accessed on: 03 Feb 2020.

[96] *Mapbox*, , Accessed on: 03 Feb 2020.

[97] W. Pasman and F. W. Jansen, "Distributed low-latency rendering for mobile ar," in *Proceedings IEEE and ACM International Symposium on Augmented Reality*, 2001, pp. 107–113.

[98] M.-C. Chambrin, "Alarms in the intensive care unit: How can the number of false alarms be reduced?" *Critical Care*, vol. 5, no. 4, pp. 1–5, 2001.

[99] E. A. Blackstone, A. J. Buck, and S. Hakim, "The economics of emergency response," *Policy Sciences*, vol. 40, no. 4, pp. 313–334, 2007.

[100] L. R. Barnes, E. C. Gruntfest, M. H. Hayden, D. M. Schultz, and C. Benight, "False alarms and close calls: A conceptual model of warning accuracy," *Weather and Forecasting*, vol. 22, no. 5, pp. 1140–1147, 2007.

[101] A. Sunyaev, "Distributed ledger technology," in *Internet Computing*, Springer, 2020, pp. 265–299.

[102] S. Yrjölä, "How could blockchain transform 6g towards open ecosystemic business models?" In *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, IEEE, 2020, pp. 1–6.

[103] Y. Du, Z. Wang, and V. Leung, "Blockchain-enabled edge intelligence for iot: Background, emerging trends and open issues," *Future Internet*, vol. 13, no. 2, p. 48, 2021.

[104] T. Hewa, G. Gür, A. Kalla, M. Ylianttila, A. Bracken, and M. Liyanage, "The role of blockchain in 6g: Challenges, opportunities and research directions," in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, IEEE, 2020, pp. 1–5.

[105] H. Xu, P. V. Klainea, O. Oniretia, B. Caob, M. Imrana, and L. Zhang, "Blockchain-enabled resource management and sharing for 6g communications," *arXiv preprint arXiv:2003.13083*, 2020.

[106] T. Nguyen, N. Tran, L. Loven, J. Partala, M.-T. Kechadi, and S. Pirttikangas, "Privacy-aware blockchain innovation for 6g: Challenges and opportunities," in *2020 2nd 6G Wireless Summit (6G SUMMIT)*, IEEE, 2020, pp. 1–5.

[107] A. Murphy, "An analysis of the financial crisis of 2008: Causes and solutions," *An Analysis of the Financial Crisis of*, 2008.

[108] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21 260, 2008.

[109] *Bitcoin open source implementation of p2p currency satoshi nakamoto institute*, , (Accessed on 04 26 2021).

[110] N. Popper, "Decoding the enigma of satoshi nakamoto and the birth of bitcoin," *New York Times*, vol. 15, 2015.

[111] P. Lemieux, "Who is satoshi nakamoto?" *Regulation*, vol. 36, no. 3, pp. 14–16, 2013.

[112] *Bitcoin's origin story remains shrouded in mystery. here's why it matters*, Availble https: www.cnbc.com, (Accessed on 04 26 2021).

[113] N. Szabo, "The idea of smart contracts," *Nick Szabo's Papers and Concise Tutorials*, vol. 6, 1997.

[114] D. Efanov and P. Roschin, "The all-pervasiveness of the blockchain technology," *Procedia Computer Science*, vol. 123, pp. 116–121, 2018.

[115] N. Abdullah, A. Hakansson, and E. Moradian, "Blockchain based approach to enhance big data authentication in distributed environment," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, IEEE, 2017, pp. 887–892.

[116] K. Okupski, "Bitcoin developer reference," in *Eindhoven*, 2014.

[117] P. T. Duy, D. T. T. Hien, D. H. Hien, and V.-H. Pham, "A survey on opportunities and challenges of blockchain technology adoption for revolutionary innovation," in *Proceedings of the Ninth International Symposium on Information and Communication Technology*, 2018, pp. 200–207.

[118] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, 2017, pp. 2567–2572.

[119] J. Conley *et al.*, "Encryption, hashing, ppk, and blockchain: A simple introduction," Vanderbilt University Department of Economics, Tech. Rep., 2019.

[120] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An overview of blockchain technology: Architecture, consensus, and future trends," in *2017 IEEE international congress on big data (BigData congress)*, IEEE, 2017, pp. 557–564.

[121] L. W. Cong and Z. He, "Blockchain disruption and smart contracts," *The Review of Financial Studies*, vol. 32, no. 5, pp. 1754–1797, 2019.

[122] S. S. Gupta, *Blockchain*. John Wiley & Sons, Inc, 2017.

[123] M. B. Taylor, "The evolution of bitcoin hardware," *Computer*, vol. 50, no. 9, pp. 58–66, 2017.

[124] *Ethereum (eth) blockchain explorer*, , (Accessed on 05 06 2021).

[125] *Bitcoin block reward halving countdown*, , (Accessed on 05 06 2021).

[126] D. Guegan, *Public Blockchain versus Private blockhain*, Documents de travail du Centre d'Economie de la Sorbonne 2017.20 - ISSN : 1955-611X, Apr. 2017. [Online]. Available: .

[127] S. M. H. Bamakan, A. Motavali, and A. B. Bondarti, "A survey of blockchain consensus algorithms performance evaluation criteria," *Expert Systems with Applications*, p. 113 385, 2020.

[128] P. Tasca and C. J. Tessone, "Taxonomy of blockchain technologies. principles of identification and classification," *arXiv preprint arXiv:1708.04872*, 2017.

[129] S. Perera, S. Nanayakkara, M. Rodrigo, S. Senaratne, and R. Weinand, "Blockchain technology: Is it hype or real in the construction industry?" *Journal of Industrial Information Integration*, vol. 17, p. 100 125, 2020.

[130] F. Casino, T. K. Dasaklis, and C. Patsakis, "A systematic literature review of blockchain-based applications: Current status, classification and open issues," *Telematics and Informatics*, vol. 36, pp. 55–81, 2019.

[131] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, "Consortium blockchain for secure energy trading in industrial internet of things," *IEEE transactions on industrial informatics*, vol. 14, no. 8, pp. 3690–3700, 2017.

[132] *Ethereum (eth) blockchain explorer - etherchain.org - 2021*, , (Accessed on 11 25 2021).

[133] *Ethereum Virtual Machine (EVM)*, , (Accessed on 11 25 2021).

[134] *Eip-225: Clique proof-of-authority consensus protocol*, , (Accessed on 11 25 2021).

[135] *Clique PoA protocol*, , (Accessed on 11 25 2021).

[136] W. Wang *et al.*, "A survey on consensus mechanisms and mining strategy management in blockchain networks," *IEEE Access*, vol. 7, pp. 22 328–22 370, 2019.

[137] *PoA Network Whitepaper*, , (Accessed on 11 25 2021).

[138] M. Schä er, M. Di Angelo, and G. Salzer, "Performance and scalability of private ethereum blockchains," in *International Conference on Business Process Management*, Springer, 2019, pp. 103–118.

[139] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 203–226.

[140] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10 127–10 149, 2019.

[141] J. Kwon, "Tendermint: Consensus without mining," *Draft v. 0.6, fall*, vol. 1, no. 11, 2014.

[142] A. Amoordon and H. Rocha, "Presenting tendermint: Idiosyncrasies, weaknesses, and good practices," in *2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)*, IEEE, 2019, pp. 44–49.

[143] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," Ph.D. dissertation, 2016.

[144] F. Saleh, "Blockchain without waste: Proof-of-stake," *The Review of financial studies*, vol. 34, no. 3, pp. 1156–1190, 2021.

[145] L. Bach, B. Mihaljevic, and M. Zagar, "Comparative analysis of blockchain consensus algorithms," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 2018, pp. 1545–1550.

[146] *Cosmos whitepaper.md at master · cosmos cosmos*, , (Accessed on 04 20 2021).

[147] S. Schulte, M. Sigwart, P. Frauenthaler, and M. Borkowski, "Towards blockchain interoperability," in *International Conference on Business Process Management*, Springer, 2019, pp. 3–10.

[148] L. X. Lin, "Deconstructing decentralized exchanges," *Stanford Journal of Blockchain Law Policy*, vol. 2, 2019.

[149] M. Deloitte, "Blockchain@ media: A new game changer for the media industry," *Blockchain Institute*, 2017.

[150] A. Dutra, A. Tumasjan, and I. M. Welpe, "Blockchain is changing how media and entertainment companies compete," English, *MIT Sloan Management Review*, vol. 60, no. 1, pp. 39–45, Fall 2018, Copyright - Copyright Massachusetts Institute of Technology, Cambridge, MA Fall 2018; Última actualización - 2020-04-02; SubjectsTermNotLitGenreText - United States–US. [Online]. Available: .

[151] H. G. et al., "An e cient micropayment channel on ethereum," in *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, Springer International Publishing, 2019, pp. 211–218.

[152] R. Matulionyte, "Can copyright be tokenized?" *Available at SSRN 3475214*, 2019.

[153] C. Li and B. Palanisamy, "Incentivized blockchain-based social media platforms: A case study of steemit," in *Proceedings of the 10th ACM Conference on Web Science*, ser. WebSci '19, Boston, Massachusetts, USA: Association for Computing Machinery, 2019, pp. 145–154. DOI: . [Online]. Available: .

[154] A. Tresise, J. Goldenfein, and D. Hunter, "What blockchain can and can't do for copyright," en, in *28 Australian Intellectual Property Journal 144*, Available at SSRN: Aug. 2018. [Online]. Available: .

[155] *Binded - copyright made simple*, , (Accessed on 07 28 2020).

[156] *Cefriel videosign prototype- camera action*, , (Accessed on 07 28 2020).

[157] *Everdreamsoft*, , (Accessed on 07 29 2020).

[158] *Vibehub.io*, , (Accessed on 07 30 2020).

[159] O. Esteban, M. Ariel, J. Yemel, and A. Manuel, "Decentraland white paper," Technical Report. Decentraland, Tech. Rep., 2017.

[160] I. J. Jensen, D. F. Selvaraj, and P. Ranganathan, "Blockchain technology for networked swarms of unmanned aerial vehicles (uavs)," in *2019 IEEE 20th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, 2019, pp. 1–7.

[161] P. Mehta, R. Gupta, and S. Tanwar, "Blockchain envisioned uav networks: Challenges, solutions, and comparisons," *Computer Communications*, vol. 151, pp. 518–538, 2020.

[162] *AIAA SciTech Forum*, San Diego, CA, United States: NTRS, 2019.

[163] V. Chamola, V. Hassija, V. Gupta, and M. Guizani, "A comprehensive review of the covid-19 pandemic and the role of iot, drones, ai, blockchain, and 5g in managing its impact," *IEEE Access*, vol. 8, pp. 90 225–90 265, 2020.

[164] H. Kantur and C. Bamuleseyo, *How smart contracts can change the insurance industry: Benefits and challenges of using blockchain technology*, 2018.

[165] Z. Chen, A. Xu, H. Wen, Y. Zhang, and X. Xu, "Aviation terminal data security architecture based on blockchain," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1575, 2020, p. 012 062.

[166] S. Kar, V. Kasimsetty, S. Barlow, and S. Rao, "Risk analysis of blockchain application for aerospace records management," in *SAE Technical Paper*, SAE International, Mar. 2019. DOI: . [Online]. Available: .

[167] T. Zwartkruis, "Blockchain takes o : Explorative study into the desirability, feasibility and viability of a blockchain technology-enabled platform to smooth dry-operational aircraft lease transitions.,"
, Ph.D. dissertation, Delft University of Technology, 2019.

[168] T.-T. Kuo, H.-E. Kim, and L. Ohno-Machado, "Blockchain distributed ledger technologies for biomedical and health care applications," *Journal of the American Medical Informatics Association*, vol. 24, no. 6, pp. 1211–1220, Sep. 2017. DOI: . eprint:
. [Online]. Available:
.

[169] M. Hölbl, M. Kompara, A. Kamišalić, and L. Nemec Zlatolas, "A systematic review of the use of blockchain in healthcare," *Symmetry*, vol. 10, no. 10, p. 470, 2018.

[170] J. M. Roman-Belmonte, H. D. la Corte-Rodriguez, and E. C. Rodriguez-Merchan, "How blockchain technology can change medicine," *Postgraduate Medicine*, vol. 130, no. 4, pp. 420–427, 2018, PMID: 29727247. DOI:
. eprint: .
[Online]. Available: .

[171] T. McGhin, K.-K. R. Choo, C. Z. Liu, and D. He, "Blockchain in healthcare applications: Research challenges and opportunities," *Journal of Network and Computer Applications*, vol. 135, pp. 62–75, 2019.

[172] A. Hasselgren, K. Kralevska, D. Gligoroski, S. A. Pedersen, and A. Faxvaag, "Blockchain in healthcare and health sciences—a scoping review," *International Journal of Medical Informatics*, vol. 134, p. 104 040, 2020.

[173] C. Pirtle and J. Ehrenfeld, *Blockchain for healthcare: The next generation of medical records?* 2018.

[174] A. F. Hussein, N. ArunKumar, G. Ramirez-Gonzalez, E. Abdulhay, J. M. R. Tavares, and V. H. C. de Albuquerque, "A medical records managing and securing blockchain based system supported by a genetic algorithm and discrete wavelet transform," *Cognitive Systems Research*, vol. 52, pp. 1–11, 2018.

[175] Y. Chen, S. Ding, Z. Xu, H. Zheng, and S. Yang, "Blockchain-based medical records secure storage and medical service framework," *Journal of medical systems*, vol. 43, no. 1, p. 5, 2019.

[176] A. Dubovitskaya, Z. Xu, S. Ryu, M. Schumacher, and F. Wang, "Secure and trustable electronic medical records sharing using blockchain," in *AMIA annual symposium proceedings*, American Medical Informatics Association, vol. 2017, 2017, p. 650.

[177] A. Ekblaw, A. Azaria, J. D. Halamka, and A. Lippman, "A case study for blockchain in healthcare:"medrec" prototype for electronic health records and medical research data," in *Proceedings of IEEE open big data conference*, vol. 13, 2016, p. 13.

[178] Y. R. Park, E. Lee, W. Na, S. Park, Y. Lee, and J.-H. Lee, "Is blockchain technology suitable for managing personal health records? mixed-methods study to test feasibility," *Journal of medical Internet research*, vol. 21, no. 2, e12533, 2019.

[179] L. Ismail, H. Materwala, and S. Zeadally, "Lightweight blockchain for healthcare," *IEEE Access*, vol. 7, pp. 149 935–149 951, 2019.

[180] J.-A. Hanssen Rensaa, D. Gligoroski, K. Kralevska, A. Hasselgren, and A. Faxvaag, "Verifymed–a blockchain platform for transparent trust in virtualized healthcare: Proof-of-concept," *arXiv*, arXiv–2005, 2020.

[181] F. Jamil, L. Hang, K. Kim, and D. Kim, "A novel medical blockchain model for drug supply chain integrity management in a smart hospital," *Electronics*, vol. 8, no. 5, p. 505, 2019.

[182] R. Kumar and R. Tripathi, "Traceability of counterfeit medicine supply chain through blockchain," in *2019 11th International Conference on Communication Systems Networks (COMSNETS)*, IEEE, 2019, pp. 568–570.

[183] M. Sahoo, S. S. Singhar, and S. S. Sahoo, "A blockchain based model to eliminate drug counterfeiting," in *Machine Learning and Information Processing*, Springer, 2020, pp. 213–222.

[184] R. Anand, K. Niyas, S. Gupta, and S. Revathy, "Anti-counterfeit on medicine detection using blockchain technology," in *Inventive Communication and Computational Technologies*, Springer, 2020, pp. 1223–1232.

[185] M. Malinverno, J. Mangues-Bafalluy, C. E. Casetti, C. F. Chiasserini, M. Requena-Esteso, and J. Baranda, "An edge-based framework for enhanced road safety of connected cars," *IEEE Access*, vol. 8, pp. 58 018–58 031, 2020.

[186] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, p. 91, 2015.

[187] A. Dorri, M. Steger, S. S. Kanhere, and R. Jurdak, "Blockchain: A distributed solution to automotive security and privacy," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 119–125, 2017.

[188] D. Lu *et al.*, "Reducing automotive counterfeiting using blockchain: Benefits and challenges," in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, IEEE, 2019, pp. 39–48.

[189] S.-K. Kim, C. Y. Yeun, E. Damiani, Y. Al-Hammadi, and N.-W. Lo, "New blockchain adoptation for automotive security by using systematic innovation," in *2019 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific)*, IEEE, 2019, pp. 1–4.

[190] L. Sang-Oun, J. Hyunseok, and B. Han, "Security assured vehicle data collection platform by blockchain: Service provider's perspective," in *2019 21st International Conference on Advanced Communication Technology (ICACT)*, IEEE, 2019, pp. 265–268.

[191] *Artificial intelligence: Bosch teaches cars how to learn and take appropriate action - bosch media service*, Availble https: www.bosch-presse.de, (Accessed on 09 09 2020).

[192] *How blockchain can help to prevent odometer fraud*, Availble https: blog.bosch-si.com, (Accessed on 09 09 2020).

[193] P. Fraga-Lamas and T. M. Fernández-Caramés, "A review on blockchain technologies for an advanced and cyber-resilient automotive industry," *IEEE Access*, vol. 7, pp. 17 578–17 598, 2019.

[194] A. R. Pedrosa and G. Pau, "Chargeltup: On blockchain-based technologies for autonomous vehicles," in *Proceedings of the 1st Workshop on Cryptocurrencies and Blockchains for Distributed Systems*, 2018, pp. 87–92.

[195] C. Decker and R. Wattenhofer, "A fast and scalable payment network with bitcoin duplex micropayment channels," in *Symposium on Self-Stabilizing Systems*, Springer, 2015, pp. 3–18.

[196] C. F. Chiasserini, P. Giaccone, G. Malnati, M. Macagno, and G. Sviridov, "Blockchain-based mobility verification of connected cars," in *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*, IEEE, 2020, pp. 1–6.

[197] M. Singh and S. Kim, "Branch based blockchain technology in intelligent vehicle," *Computer Networks*, vol. 145, pp. 219–231, 2018.

[198] M. Cebe, E. Erdin, K. Akkaya, H. Aksu, and S. Uluagac, "Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles," *IEEE Communications Magazine*, vol. 56, no. 10, pp. 50–57, 2018.

[199] M. Baza, M. Nabil, N. Lasla, K. Fidan, M. Mahmoud, and M. Abdallah, "Blockchain-based firmware update scheme tailored for autonomous vehicles," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, IEEE, 2019, pp. 1–7.

[200] N. Hackius and M. Petersen, "Blockchain in logistics and supply chain: Trick or treat?" In *Digitalization in Supply Chain Management and Logistics: Smart and Digital Solutions for an Industry 4.0 Environment. Proceedings of the Hamburg International Conference of Logistics (HICL), Vol. 23*, Berlin: epubli GmbH, 2017, pp. 3–18.

[201] G. Perboli, S. Musso, and M. Rosano, "Blockchain in logistics and supply chain: A lean approach for designing real-world use cases," *IEEE Access*, vol. 6, pp. 62 018–62 028, 2018.

[202] I. M. Ar, I. Erol, I. Peker, A. I. Ozdemir, T. D. Medeni, and I. T. Medeni, "Evaluating the feasibility of blockchain in logistics operations: A decision framework," *Expert Systems with Applications*, vol. 158, p. 113 543, 2020. DOI: . [Online]. Available: .

[203] Y. Madhwal and P. B. Panfilov, "Blockchain and supply chain management: Aircrafts'parts'business case.," *Annals of DAAAM Proceedings*, vol. 28, pp. 1051–1056, Jan. 2017.

[204] C. Wickboldt and N. Kliewer, "Blockchain for workshop event certificates - a proof of concept in the aviation industry," in *ECIS*, 2019.

[205] V. Ortega, F. Bouchmal, and J. F. Monserrat, "Trusted 5g vehicular networks: Blockchains and content-centric networking," *IEEE Vehicular Technology Magazine*, vol. 13, no. 2, pp. 121–127, 2018.

[206] J. P. Queralta, L. Qingqing, Z. Zou, and T. Westerlund, "Enhancing autonomy with blockchain and multi-acess edge computing in distributed robotic systems," in *The Fifth International Conference on Fog and Mobile Edge Computing (FMEC). IEEE*, 2020.

[207] K. Katsalis et al., "Multi-domain orchestration for nfv: Challenges and research directions," in *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*, IEEE, 2016, pp. 189–195.

[208] R.B. Uriarte et al., "Blockchain-based decentralized cloud fog solutions: Challenges, opportunities, and standards," *IEEE Communications Standards Magazine*, vol. 2, no. 3, pp. 22–28, 2018.

[209] ETSI, "ETSI ISG PDL 003 V1.1.1, Permissioned Distributed Ledger (PDL); Application Scenarios," Dec. 2020.

[210] ——, "ETSI ISG PDL 004 V1.1.1, Permissioned Distributed Ledgers (PDL) Smart Contracts System Architecture and Functional Specification," Feb. 2021.

[211] *2020 global blockchain survey*, https: www2.deloitte.com, (Accessed on 10 02 2020).

[212] G. Karame, "On the security and scalability of bitcoin's blockchain," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 1861–1862.

[213] P. Fairley, "Blockchain world - feeding the blockchain beast if bitcoin ever does go mainstream, the electricity needed to sustain it will be enormous," *IEEE Spectrum*, vol. 54, no. 10, pp. 36–59, 2017.

[214] T. Hepp, M. Sharinghousen, P. Ehret, A. Schoenhals, and B. Gipp, "On-chain vs. o -chain storage for supply-and blockchain integration," *it-Information Technology*, vol. 60, no. 5-6, pp. 283–291, 2018.

[215] B. Nour et al., "A blockchain-based network slice broker for 5g services," *IEEE Networking Letters*, vol. 1, no. 3, pp. 99–102, 2019. DOI: .

[216] J. Baranda *et al.*, "Nfv service federation: Enabling multi-provider ehealth emergency services," in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2020, pp. 1322–1323. DOI: .

[217] K. Antevski, L. Girletti, C. J. Bernardos, A. de la Oliva, J. Baranda, and J. Mangues-Bafalluy, "A 5g-based ehealth monitoring and emergency response system: Experience and lessons learned," *IEEE Access*, vol. 9, pp. 131 420–131 429, 2021.

[218] *Ethermint - Ethereum Virtual Machine (EVM) as a Cosmos SDK module*, , (Accessed on 12 10 2021).

[219] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 398–409, 2015.

[220] S. Wan, Z. Gu, and Q. Ni, "Cognitive computing and wireless communications on the edge for healthcare service robots," *Computer Communications*, vol. 149, pp. 99–106, 2020.

[221] S. Dey and A. Mukherjee, "Robotic slam: A review from fog computing and mobile edge computing perspective," in *Adjunct Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing Networking and Services*, ser. MOBIQUITOUS 2016, Hiroshima, Japan: ACM, 2016, pp. 153–158. DOI: .

[222] N. Tian *et al.*, "A fog robotic system for dynamic visual servoing," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 1982–1988.

[223] K. A. et al., "On the integration of nfv and mec technologies: Architecture analysis and benefits for edge robotics," *Computer Networks*, vol. 175, p. 107 274, 2020. DOI: .

[224] S. D. Jap, "The impact of online reverse auction design on buyer–supplier relationships," *Journal of Marketing*, vol. 71, no. 1, pp. 146–159, 2007. DOI: .

[225] ETSI, "Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Report on architecture options to support multiple administrative domains," European Telecommunications Standards Institute (ETSI), Group Specification (GS) NFV 028 v3.1.1, Jan. 2018.

[226] C. N. et al., "Machine learning aided orchestration in multi-tenant networks," in *2018 IEEE Photonics Society Summer Topical Meeting Series (SUM)*, Jul. 2018, pp. 125–126. DOI: .

[227] D. M. et al., "Artificial intelligence for elastic management and orchestration of 5g networks," *IEEE Wireless Communications*, pp. 1–8, 2019. DOI: .

[228] Y. W. et al., "Network management and orchestration using artificial intelligence: Overview of etsi eni," *IEEE Communications Standards Magazine*, vol. 2, no. 4, pp. 58–65, Dec. 2018. DOI: .

[229] G. George, R. Wolski, C. Krintz, and J. Brevik, "Analyzing aws spot instance pricing," in *2019 IEEE International Conference on Cloud Engineering (IC2E)*, IEEE, 2019, pp. 222–228.

[230] X. Li *et al.*, "Service orchestration and federation for verticals," in *2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, IEEE, 2018, pp. 260–265.

[231] *Unify-d2.2-final_architecture.pdf*, , (Accessed on 12 10 2020).

[232] J. B. et al., "Composing services in 5g-transformer," in *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, ser. Mobihoc '19, Catania, Italy: ACM, 2019, pp. 407–408. DOI: .

[233] J. Baranda *et al.*, "5g-transformer meets network service federation: Design, implementation and evaluation," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, IEEE, 2020, pp. 175–179.

[234] X. Li *et al.*, "5growth: An end-to-end service platform for automated deployment and management of vertical services over 5g networks," *IEEE Communications Magazine*, vol. 59, no. 3, pp. 84–90, 2021.

[235] C. Papagianni *et al.*, "5growth: Ai-driven 5g for automation in vertical industries," in *2020 European Conference on Networks and Communications (EuCNC)*, 2020, pp. 17–22. DOI: .

[236] K. Antevski and C. J. Bernardos, "Federation of 5g services using distributed ledger technologies," *Internet Technology Letters*, e193, 2020.

[237] D. Stezenbach, M. Hartmann, and K. Tutschku, "Parameters and challenges for virtual network embedding in the future internet," in *2012 IEEE Network Operations and Management Symposium*, IEEE, 2012, pp. 1272–1278.

[238] P. T. A. Quang, A. Bradai, K. D. Singh, G. Picard, and R. Riggio, "Single and multi-domain adaptive allocation algorithms for vnf forwarding graph embedding," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 98–112, 2019. DOI: .

[239] A. F. T. Martins, M. A. T. Figueiredo, P. M. Q. Aguiar, N. A. Smith, and E. P. Xing, "Ad3: Alternating directions dual decomposition for map inference in graphical models," *Journal of Machine Learning Research*, vol. 16, no. 16, pp. 495–545, 2015. [Online]. Available:
.

[240] P. T. A. Quang, A. Bradai, K. D. Singh, and Y. Hadjadj-Aoul, "Multi-domain non-cooperative vnf-fg embedding: A deep reinforcement learning approach," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 886–891. DOI:
.

[241] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, 2000, pp. 1008–1014.

[242] G. Li, H. Zhou, B. Feng, and G. Li, "Context-aware service function chaining and its cost-e ective orchestration in multi-domain networks," *IEEE Access*, vol. 6, pp. 34 976–34 991, 2018. DOI: .

[243] Q. Zhang, X. Wang, I. Kim, P. Palacharla, and T. Ikeuchi, "Service function chaining in multi-domain networks," in *2016 Optical Fiber Communications Conference and Exhibition (OFC)*, 2016, pp. 1–3.

[244] G. Malewicz *et al.*, "Pregel: A system for large-scale graph processing," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '10, Indianapolis, Indiana, USA: Association for Computing Machinery, 2010, pp. 135–146. DOI: . [Online]. Available: .

[245] J. C. Cisneros, S. Yangui, S. E. Pomares Hernández, J. C. Pérez Sansalvador, and K. Drira, "Coordination algorithm for migration of shared vnfs in federated environments," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 252–256. DOI: .

[246] Yang Wang, Gaogang Xie, Zhenyu Li, Peng He, and K. Salamatian, "Transparent flow migration for nfv," in *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, 2016, pp. 1–10. DOI: .

[247] B. Sonkoly *et al.*, "5g applications from vision to reality: Multi-operator orchestration," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 7, pp. 1401–1416, 2020. DOI: .

[248] B. Németh, B. Sonkoly, M. Rost, and S. Schmid, “E cient service graph embedding: A practical approach,” in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, 2016, pp. 19–25. DOI: .

[249] L. Ma and X. Wang and X. Wang and L. Wang and Y. Shi and M. Huang, “TCDA: Truthful Combinatorial Double Auctions for Mobile Edge Computing in Industrial Internet of Things,” *IEEE Transactions on Mobile Computing*, no. 01, pp. 1–1, 2021. DOI: .

[250] B. Zhou, S. N. Srirama, and R. Buyya, “an auction-based incentive mechanism for heterogeneous mobile clouds,” *Journal of Systems and Software*, vol. 152, pp. 151–164, 2019. DOI: .
[Online]. Available: .

[251] R. Bellman, “Dynamic programming and stochastic control processes,” *Information and Control*, vol. 1, no. 3, pp. 228–239, 1958. DOI: . [Online]. Available: .

[252] C. J. C. H. Watkins, “Learning from delayed rewards,” 1989.

[253] R. e. a. Fourer, “Ampl. a modeling language for mathematical programming,” 1993.

[254] I. Gurobi Optimization, “Gurobi optimizer reference manual,” *URL http: www. gurobi. com*, 2015.

[255] J. Martın-Pérez, K. Antevski, A. Garcia-Saavedra, X. Li, and C. J. Bernardos, “Dqn dynamic pricing and revenue driven service federation strategy,” *IEEE Transactions on Network and Service Management*, vol. 18, no. 4, pp. 3987–4001, 2021.