Private 5G Networks
for Connected Industries

Deliverable D4.3

Specification and Implementation of Advanced Functionalities

Co-funded by the Horizon 2020 programme
of the European Union in collaboration with Taiwan

Date of Delivery: 04.10.2022
Project Start Date: 01.10.2019 **Duration:** 36 Months

Document Information

Project Number:	861459
Project Name:	Private 5G Networks for Connected Industries

Document Number:	D4.3
Document Title:	Specification and Implementation of Advanced Functionalities
Editor:	Mickael Maman (CEA)
Authors:	Mickael Maman (CEA) Ngoc-Lam Dinh (CEA) Emilio Calvanese Strinati (CEA) Cheng-Yi Chien (Chunghwa Telecom, CHT) Jiun-Cheng Huang (Chunghwa Telecom, CHT) Yueh-Feng Li (Chunghwa Telecom, CHT) Sergio Barbarossa (SAP) Stefania Sardellitti (SAP) Paolo Di Lorenzo (SAP) Francesco Pezone (SAP) Daniele Munaretto (ATH) Daniele Ronzani (ATH) Marco Centenaro (ATH) Nicola di Pietro (ATH) Jack Shi-Jie Luo (ITRI) Shuo-Peng Liang (ITRI) Winson Huang (ANI) Louis Lu (III) PoYu Liao (III) Frank Chih-Wei Su (III)
Dissemination Level:	Public
Contractual Date of Delivery:	30.09.2022
Work Package	WP 4
File Name:	861459-5G CONNI-D4.3-Specification and implementation of advanced functionalities-V1.0.docx

Revision History

Ver-sion	Date	Comment
0.1	20.06.2022	First ToC
0.2	26.07.2022	CEA contribution for RAN
0.3	16.08.2022	ATH and III contributions
0.4	30.08.2022	ATH, CHT, ITRI, ANI contributions
0.41	30.08.2022	Update of Executive summary, advanced functionality conclusion (section 6.4) and global conclusion (section 7)
0.44	13.09.2022	SAP contributions and conclusions
1.0	20.09.2022	Final version

Executive Summary

WP4 (Technical Enablers for Industrial Applications) covers Mobile Edge Computing (MEC) cloud development, industrial application technical development, radio network technical development, and core network technical development for industrial field. The main goal of this work package is to ensure that industrial use cases can be successfully implemented on private 5G networks for industrial requirements, including high data rates (eMBB) and low latency (URLLC).

D4.3 describes the final implementation of private 5G networks building blocks and the specification of advanced functionalities. This deliverable is both an extension of D4.1 and D4.2 and covers future functionalities (e.g., joint optimization of enabling technologies (radio, core, MEC), goal-directed communications, and some enhancements of existing deterministic URLLC protocols providing minimum QoS requirements). These innovative components have fueled the lab integration and experimental results of the building blocks will be reported in D5.3.

In task 4.1, the 5G CONNI project built a RAN system composed of CDU, RU, and CPE as RAN implementation. gNodeB and CPE have been deployed in ITRI IMTC for industrial application and have been optimized to meet the requirements (e.g., bandwidth and low latency) of the selected use case.

In task 4.2, 5G CONNI project proposed two complementary 5G Core networks: 5G core prototype and lightweight orchestration framework. The components of the 5G core prototype are defined as self-contained, independent and reusable network functions (NFs) together with a well-defined Service Based Interface (SBI) using HTTPv2 that can be used to invoke services. The 5GC provides interfaces and integrates with Application Network (AN), Mobile Edge Computing (MEC), and gNB. The 5GC contains the resources for setting up and maintaining the traffic between the gNBs and Application Network can support multiple gNBs. The 5GC is capable to achieve throughput to 40Gbps and data plane latency less than 1ms. The 5G Core also provides interfaces to support Network OAM (Operations Administration and Maintenance) functions. 5G CONNI project also worked on the ETSI NFV-like instantiation and orchestration of 5G mobile core network components via OSM Release Eleven. With the designed VNFDs, it is possible to deploy a mobile core network with off-the-shelf, standard-compliant MANO implementations such as OSM and ONAP. 5G CONNI project focused on the implementation of the semantic of the Ve-Vnfm reference point, to make OSM able to directly configure the core network in an ETSI standard approach. A ProxyAPI module has been implemented and added into the VNF to allow 5GC management and configuration in a standard way. A simple Web interface displays the 5GC VNF configuration status and progress. In order to have a unique centralized point of monitoring, 5G CONNI project has updated the orchestration framework with OSM metric collection. Thus we performed the OSM approach to retrieve VIM metrics (CPU, memory, disk, and network usage) and to visualize them in its Prometheus/Grafana instances. As a complementary work, we developed a provisioning procedure of a Network Slice Subnet (NSS) modeled as a Network Service (NS), composed of several VNFs from potentially different vendors. It includes a Network Management System (NMS) conforming to the 3GPP Service-Based Management Architecture (SBMA), an ETSI MANO orchestrator, and a Network Function Virtualization Infrastructure (NFVI).

In task 4.3, two implementations of MEC are proposed by 5G-CONNI for the European and Taiwanese testbeds: the hybrid 5GC solution and the bump-in-the-wire solution. Hybrid 5GC solution consists in the splitting of CP and UP functions through OSM, giving more flexibility for the mobile network deployment. The 5GC CP network functions (e.g., AMF, UDM, SMF, etc.) is deployed in the central NFVI server, while the 5GC UPF, the only component acting as UP function, is deployed in the edge NFVI server. This approach allows the support of edge

computing, with the consequent possibility of installing a MEC platform. MEC 5G SA based on a bump-in-the-wire architecture develops handover, multi-PDU sessions and multi-QoS flows functionalities. The bump-in-the-wire solution proposes a MEC platform with virtualization capability and VNF management. The virtualization capability is responsible for virtualizing edge computing functions, industrial applications and any needed applications to the MEC platform to realize flexible deployment. The VNF management supports the essential management demands of ETSI NFV specifications and adjusts the performance of VNFs to achieve the required transmission delay, execution performance and resource management.

In Task 4.4, 5G CONNI project worked on three vertical use cases, namely (1) process diagnostics using Augmented/Virtual Reality (2) cloud based controller for CNC and (3) the multi-site use case. For the process diagnostics, remote rendering technique has been adopted and the 3D model of the demo site and target machine have been built and loaded in the GPU workstation and rendered by Dassault System software. The viewport of 3D scene can be updated with the IMU information from user device so that a smooth user experience of navigating through the 3D scene can be achieved.

5G CONNI project also investigates advanced functionalities for future private 5G networks. First, 5G CONNI proposes a dynamic resource allocations for URLLC. This novel dynamic decision maker framework enables a reliable, resource and delay-optimized scheduling suitable for dynamic URLLC scenarios (e.g., intermittent traffic source rate, time-varying channel). Based on Lyapunov optimization, it takes into account the traffic arrival at the network layer, the queue behaviors at the data link layer and the risk that the applied decision might trigger packet loss. The trade-off between the resource efficiency, latency and reliability is achieved by the timing and intensity of decisions.

Second, 5G CONNI proposes a dynamic strategy for resource allocation in the edge cloud, aimed at finding the optimal trade-off between energy consumption and service delay, taking into account both computational and radio resources. More specifically, we considers the service placement, incorporating also the storage resources necessary to run virtual machines in the edge cloud. The solution is based on Lyapunov stochastic optimization and extended to a distributed learning problem, exploiting federated learning as an effective mechanism to achieve globally optimal solutions without requiring all the exchange of data between peripheral devices and edge server, but only exchange of parameter updates.

Finally, 5G CONNI proposes a goal-oriented communication scheme, a very novel communication strategy that has the potentials to outperform the conventional Shannon-based approach in all cases where communication occurs to fulfill a common goal between source and destination. We generalizes the approach exploiting the information bottleneck principle as a driving principle, by incorporating convolutional neural networks, used as encode/decode pairs, trained offline and used adaptively online during the communication process.

Table of Contents

List of Figures

List of Tables

List of Acronyms

3GPP	3rd Generation Partnership Project
4G	4th Generation
5G	5th Generation
5GC	5G Core
5G CONNI	5G for Connected Industries
5GS	5G System
AF	Application Function
AMF	Access and Mobility Management Function
AP	Access Point
API	Application Programming Interface
AR	Augmented Reality
AS	Angular Spread
ATH	Athonet
AUSF	Authentication Server Function
CEA	Commissariat à l'énergie atomique et aux énergies alternatives
CHT	ChungHwa Telecom
CN	Core Network
CNC	Computerized Numerical Control
COVID-19	Corona Virus Disease 2019
CP	Control Plane
CPE	Customer Premises Equipment
CU	Central Unit
DS	Delay Spread
DU	Data Unit
E2E	End-to-End
ECC	ECoreCloud
eMBB	Enhanced Mobile Broadband
EML	Edge Machine Learning
EPC	Evolved Packet Core
ENI	Experimental Networked Intelligence
ETSI	European Telecommunications Standards Institute
FCAPS	Faults, Configuration, Accounting, Performance, and Security
FoF	Factories of the Future
gNB	Gigabit Node B, 5G Base Station
GTP	GPRS Tunneling Protocol
HARQ	Hybrid ARQ, Hybrid Automatic Repeat Request
HHI	Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute
IEEE	Institute of Electrical and Electronics Engineers
III	Institute for Information Industry
IMTC	Intelligent Machine Tool Center
InF	Indoor Factory
IOC	Information Object Classes

ITRI	Industrial Technology Research Institute
I-UPF	Intermediate User Plane Function
KPI	Key Performance Indicator
LOS	Line of Sight
MAC	Medium Access Control
MANO	Management and Orchestration
MEC	Mobile Edge Cloud / Multi-access Edge Computing
MEE	Mobile Edge Enabler
MIB	Management Information Base
MIMO	Multiple Input, Multiple Output
MNO	Mobile Network Operator
MnS	Management Services
MOI	Managed Object Instance
NFMF	Network Function Management Functions
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
NLOS	Non Line of Sight
NMS	Network Management System
NR	5G New Radio
NRF	Network Repository Function
NRM	Network Resource Model
NS	Network Service
NS3	Network Simulator Version 3
NSA	5G Non Stand Alone
NSD	Network Service Descriptor
NSS	Network Slice Subnet
NSSMF	Network Slice Subnet Management Function
OAM	Operation, Administration and Maintenance
OAM&P	Operation, Administration, Maintenance and Provisioning
OSM	ETSI’s Open Source MANO
PCF	Policy Charging Function
PDCP	Packet Data Convergence Protocol
PHY	Physical Layer
PPS	Pulse Per Second
QoS	Quality of Service
RAN	Radio Access Network
RIB	Routing Information Base
RLC	Radio Link Control
RRC	Radio Resource Control
RU	Radio Unit
SA	(5G) Stand Alone / Services and System Aspects
SAP	University La Sapienza
SBA	Service Based Architecture

SBMA	Service-Based Management Architecture
SF	Shadow Fading
SMF	Session Management Function
TR	Technical Report
TS	Technical Specification
UC	Use Case
UDM	Unified Data Management
URLLC	Ultra-Reliable Low Latency Communication
UP	User Plane
UPF	User Plane Function
VDU	Virtual Deployment Units
VIM	Virtual Infrastructure Manager
VM	Virtual Machine
VNF	Virtualized Network Function
VNFC	VNF Component
VNFD	VNF Descriptor
VNFM	Virtualized Network Function Manager
VR	Virtual Reality
VSDC	Vendor-Specific Data Containers
WP	Work Package

1 Introduction

WP4 (Technical Enablers for Industrial Applications) covers Mobile Edge Computing (MEC) cloud development, industrial application technical development, radio network technical development, and core network technical development for industrial field. The main goal of this work package is to ensure industrial use cases can be implemented on private 5G networks successfully for industrial requirements, including high data rates (eMBB) and low latency (URLLC).

1.1 Scope

D4.3 provides the specification and implementation of advanced functionalities for private 5G networks. This deliverable is both an extension of D4.1 and D4.2 and covers future functionalities (e.g., joint optimization of enabling technologies (radio, core, MEC), goal-directed communications, and some enhancements of existing deterministic URLLC protocols providing minimum QoS requirements)

1.2 Structure

The document is divided in six sections: the first four sections describe the implementation of 5G CONNI private network from radio network to industrial applications and the last section is related to advanced functionalities (beyond the demonstrator). Section 2 investigates radio network technical enablers and provides a 5G RAN system composed of a CPE and a gNB. Section 3 focused on core network technical enablers and proposes two complementary solutions: a lightweight orchestration framework and a 5G core prototype. Section 4 investigates mobile edge cloud enablers supporting the requirements of smart factories in 5G eMBB and URLLC scenarios. Two implementations of MEC are proposed by 5G-CONNI for the European and the Taiwanese testbeds: the hybrid 5GC solution and the bump-in-the-wire solution. Section 5 details the implementation of the selected use cases: (i) process diagnostics using AR/VR, (ii) cloud based controller for CNC and (iii) the multi-site use case. Section 6 rethinks the network in a holistic manner by jointly optimizing all enabling technologies and proposes several algorithms on service placement, goal oriented communication and dynamic resource scheduling allocations for URLLC. Finally, Section 7 concludes this deliverable.

2 Radio Network Technical Enablers

The objectives of Task 4.1 are the development of the 5G radio access network. This work is based on the industrial use cases generated by WP1.

2.1 5G RAN system composed of CPE and gNB

In D4.2 of WP4 task, the related specification of 5G RAN system, including CPE and gNB were presented. In this section, the progress of features which is going be used in WP5 will be introduced. Figure 1 shows the 5G CPE and gNB built for this project.

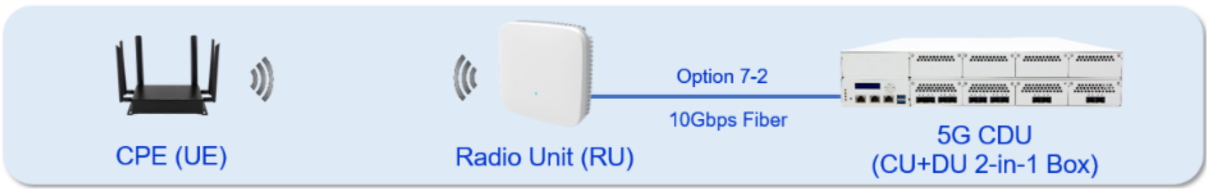


Figure 1: 5G RAN

To support the use cases which requires bandwidth and low latency separately in WP5, 5G RAN system should have some QoS features, shown in Table 1. To comply with 3GPP industrial standard, the 5QI (5G QoS Identifier) is introduced. Table 2 shows part of the 5QI characteristic and 5QI value of 3, 6 and 9 were implemented in the RAN system.

Table 1: Use Cases to be supported

Use Case	Application	5QI	Requirement (Throughput or Latency)
1	Data Collection	9	UL TCP > 16Mbps
2	AR Diagnostic	9	UL TCP > 3Mbps; DL TCP > 50Mbps E2E latency < 30ms
3	Flexible Work holding	5	E2E latency < 20ms
4	Robot in European filed	n/a	Stable jitter

Table 2: 5QI to QoS characteristics mapping

5QI Value	Resource Type	Default Priority Level	Packet Delay Budget	Packet Error Rate	Example Services
3	GBR (Guaranteed Bit Rate)	30	50ms	10^{-3}	Real Time Gaming, V2X messages
5	Non-GBR	10	100ms	10^{-6}	IMS Signaling
6		60	300ms	10^{-6}	Video (Buffered Streaming TCP based)
9		90	300ms	10^{-6}	

Sever times of testing for 5G RAN were conducted to verify if the system is able to comply with the requirement of use cases. The current result is shown in below Table 3.

Table 3 : 5G RAN Test Result

Use Case	Use Case	Requirement	Test Result
1	Data Collection	UL TCP > 16Mbps	UDP DL: 580 Mbps UDP UL: 37.7 Mbps TCP DL: 563 Mbps TCP UL: 38 Mbps E2E latency: 28.7 ms
2	AR Diagnostic	UL TCP > 3Mbps; DL TCP > 50Mbps E2E latency < 30ms	
3	Flexible Work holding	E2E latency < 20ms	
4	Robot in European filed	Stable Jitter	1.82 ms

In the task of WP5, whole 5G system would be deployed in the field. In order to better comply with the requirement, the performance and stability optimization for 5G RAN is still on-going.

2.2 Conclusions

In this section, some development progress of the 5G RAN which composed the RU, DU and CPE is presented and the data were collected during the end to end test for different use cases.

The achievement in this task can be contributed to the integrated verification task of WP5 in the field.

3 Core Network Technical Enablers

Task 4.2 focused on the development of the core network components to realize private local 5G networks to meet the requirements of the envisioned industrial application.

Two complementary activities have been carried out.

- Lightweight orchestration framework
- 5G Core prototype

3.1 NFV-like lightweight orchestration framework for the core network

The final activities concerning the NFV orchestration framework outline some advancements and results in terms of mobile core deployment, standardization, and exploitation of orchestration functionalities, mostly related to metrics and deployment automation.

3.1.1 Orchestration framework update

The final virtualization and orchestration framework architecture is shown in Figure 2.

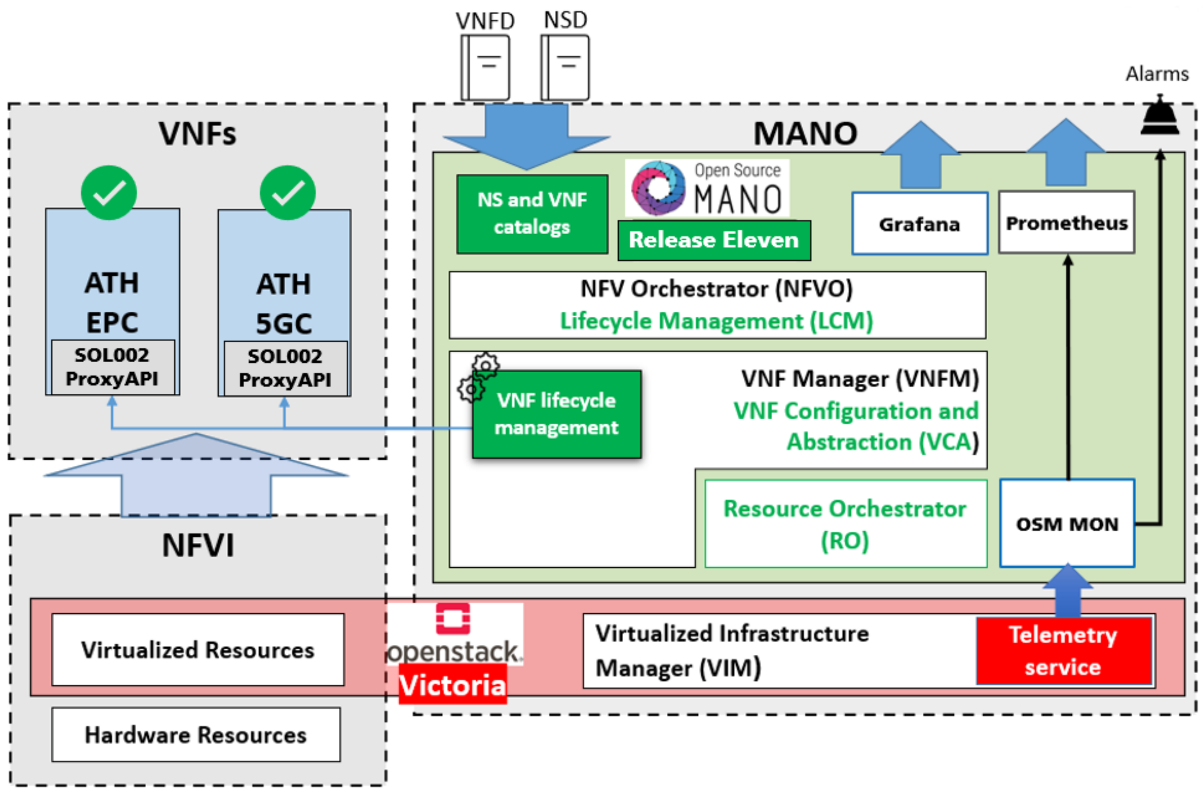


Figure 2: Final NFV and orchestration testbed architecture.

Some minor updates have been done in the testbed. A relevant change is the upgrade of Open Source Mano (OSM) to Release Eleven, one of the last software versions.

3.1.2 5GC Deployment Implementation and Automation

As central part of the research work, the mobile core deployment research activity has been moved from EPC to 5GC adoption. Accordingly, the NS/VNF packages, scripts and VM images have been completely transformed or changed in order to adapt to the new core generation.

Referring to what was mentioned in the previous deliverable (D4.2), the final objective was to deploy and configure a 5GC, running as a VNF. The activity is focused on the analysis and implementation of the semantic of the Ve-Vnfm reference point (ETSI SOL002), to make OSM able to directly configure the core network in a ETSI standard approach.

As done for EPC, we worked on making 5GC SOL002 standard compliant in the configuration process. Indeed, the Athonet 5GC can be configured adopting its proprietary API interface. However, if a network element is placed as a VNF within the context of the ETSI VNF standard architecture, the configuration should be performed through the Ve-Vnfm reference point. Here, the standard ETSI SOL002 outlines the set of operations to be carried out between the VNFM and the VNF.

Currently, neither the 5GC nor OSM provide the ETSI SOL002 standard compliance for the VNF configuration; in this work, our interest focuses on the implementation of this feature in our testbed deployment. For this reason, a new module, called *SOL002 ProxyAPI (more briefly, ProxyAPI)*, has been implemented, similarly for the EPC configuration; its purpose is to expose ETSI SOL002 configuration API to then translate the standard request into proprietary configuration request towards the 5GCs The final deployment architecture is shown in Figure 3, where two VNF components, the Athonet 5GC and the ProxyAPI are included into the main VNF, and three main steps are illustrated: onboarding, instantiation and configuration. In the following, some descriptions will refer to this figure.

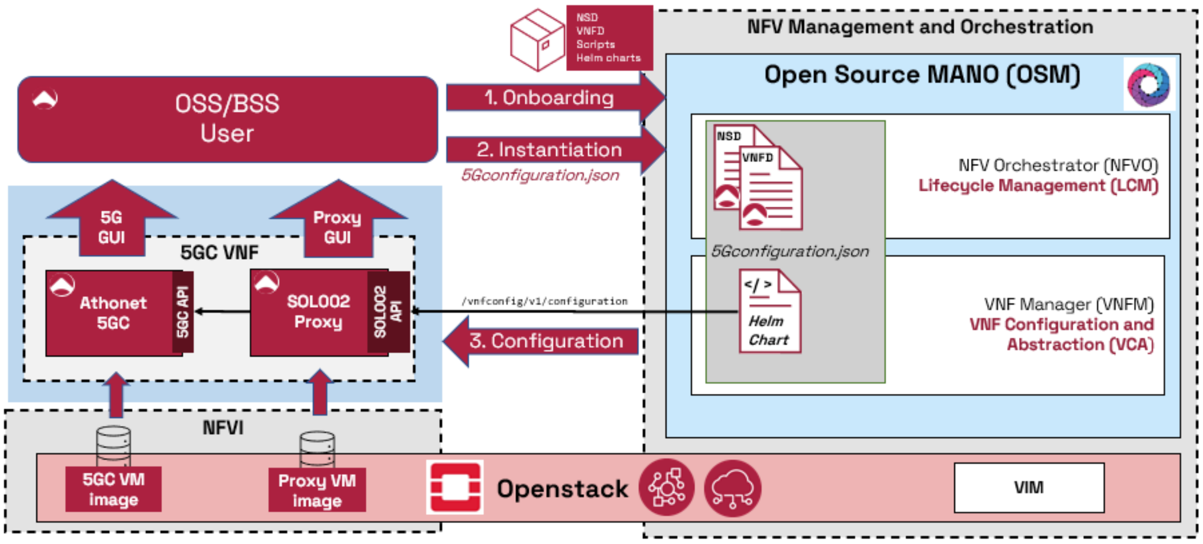
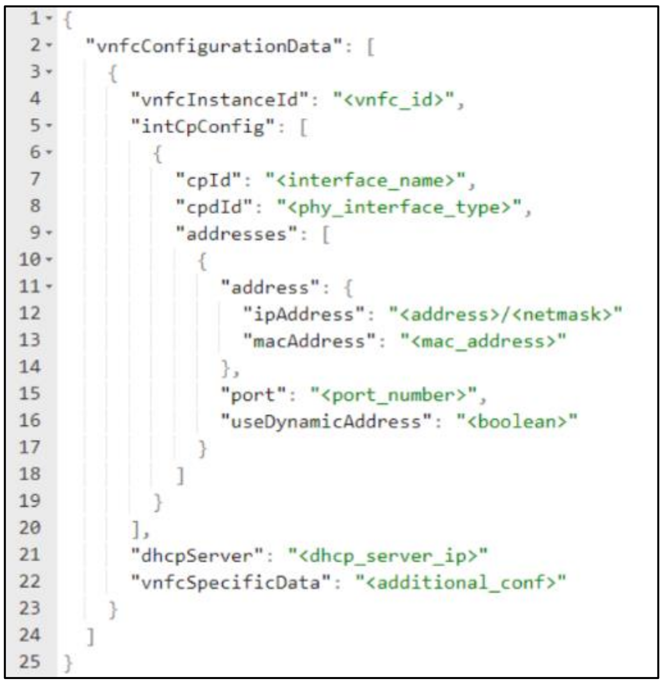


Figure 3: Final testbed architecture.

To make possible the standardization of the configuration, we intervened on both sides of the Ve-Vnfm connection point.

3.1.2.1 VNFM (OSM) Side

As we mentioned before, the VNFM module of OSM doesn't support SOL002 specifications. On the other hand, OSM adopts Proxy charms or Helm charts for VNF configuration, two frameworks provided for vendors who want to implement their own VNFM and so, their configuration approach compatible with their product. In our case we exploit the Helm chart to design the SOL002 VNF configuration request towards the 5GC. This Helm chart is included into the VNF package (see D4.2, sec. 3.1.2 for more information about VNF on-boarding, the first process in Figure 3, which loads NSD, VNF and the Helm chart into OSM). The latest SOL002 specifications (V4.3.1 – 2022-07) define the data model for VNF configuration request, as shown in Figure 4.



```
{
  "vnfcConfigurationData": [
    {
      "vnfcInstanceId": "<vnfc_id>",
      "intCpConfig": [
        {
          "cpId": "<interface_name>",
          "cpdId": "<phy_interface_type>",
          "addresses": [
            {
              "address": {
                "ipAddress": "<address>/<netmask>"
                "macAddress": "<mac_address>"
              },
              "port": "<port_number>",
              "useDynamicAddress": "<boolean>"
            }
          ]
        }
      ],
      "dhcpServer": "<dhcp_server_ip>"
      "vnfcSpecificData": "<additional_conf>"
    }
  ]
}
```

Figure 4: Data model of VNF configuration request payload.

The content of the data model considers all the configuration of a generic VNF component (in our case, the 5GC component), to associate each VNFC connection point to a physical interface of the VM, along with inclusion of network routes, IP addresses, and other parameters; however, part of specific 5GC settings (functional configurations) is not supported by this model. To solve the limitation, we exploited the optional field `vnfcSpecificData` made available by the API, whose scope is to allow vendors to perform specific configuration that cannot be done through the other SOL002 parameters: it is a generic field which supports data in *key/value* pairs, such as provided by YAML or JSON files, for additional settings. We adopt this field to include all the 5GC NFs configuration related to the private mobile network.

Helm Charts

A further advancement, respect to the previous work, is the adoption of Helm charts instead of Proxy charms (Juju). The new framework is directly deployed on K8s cluster (the same where OSM runs) to make pods for VNF configuration. The approach is similar to Juju, with the difference that containers are used, instead of virtual machines. This results in a lighter and faster deployment, as shown in the results section.

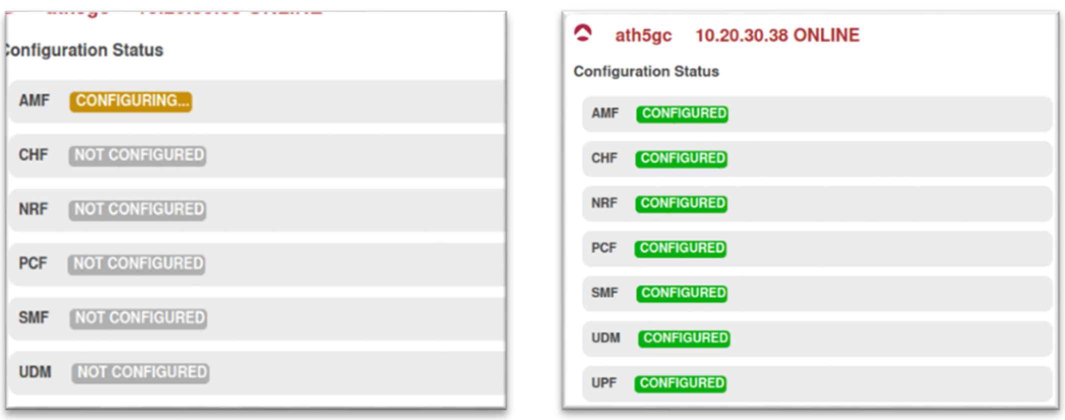
In order to configure the 5GC, the system needs three main information pieces, which are properly retrieved by the chart:

- **IP address of the ProxyAPI**: since the Helm chart functions need to talk with the ProxyAPI, the chart itself is binded to that VDU in the VNFD. Doing that, OSM automatically assigns its IP address to a variable that can be used within the chart itself.
  5GC configuration data: this information is passed as file parameter during the NS instantiation. The Helm chart, then, parses and elaborates this content and makes a SOL002-compliant data information to be sent to the ProxyAPI, which will further process the data to properly configure the 5GC.
- **IP address of the 5GC VM**: this information is needed by the ProxyAPI to know where to forward the proprietary configuration request. Hence, this information must be included into the configuration request by the Helm chart. However, assuming that the IP address allocation is dynamic, the chart does not have the IP information of other

VDUs, except the one it was associated with. For this purpose, a special module in the chart has been designed in order to query OSM NBI and ask for the specific IP address assigned afterwards to the 5GC VM. This IP address is then included into the SOL002 vnfcConfigurationData, to be used by the ProxyAPI.

3.1.2.2 VNF (5GC) Side

The 5GC exposes proprietary APIs for its management and configuration. Consequently, we must also adapt its endpoints to allow configuration in a standard way. For this purpose, the ProxyAPI module has been implemented and added into the VNF, together with the 5GC. In practice, it translates a standard VNF configuration request (called by the Helm chart within the VNFM) to the set of core specific configuration requests. This implementation is similar to the ProxyAPI realized for the EPC configuration, but several improvements have been included, e.g., a simple Web interface to display the 5GC VNF configuration status and progress; in the specific, this web console lists which NFs are being configured and shows the status of their configuration process (Figure 5).



onfiguration Status

AMF	CONFIGURING...
CHF	NOT CONFIGURED
NRF	NOT CONFIGURED
PCF	NOT CONFIGURED
SMF	NOT CONFIGURED
UDM	NOT CONFIGURED

ath5gc 10.20.30.38 ONLINE

Configuration Status

AMF	CONFIGURED
CHF	CONFIGURED
NRF	CONFIGURED
PCF	CONFIGURED
SMF	CONFIGURED
UDM	CONFIGURED
UPF	CONFIGURED

Figure 5: ProxyAPI web interface showing the configuration process.

In Figure 6, the set of HTTP requests for the configuration process is illustrated. When the Helm chart receives the trigger for configuration, it makes a set of requests to OSM for retrieving information about the instantiated VNF, in particular the 5GC IP address. Then, a PATCH request is sent to the ProxyAPI to forward all the 5GC configuration, and finally the request payload is processed, and proprietary 5GC configuration request is then sent to the actual 5GC, which applies the configuration itself.

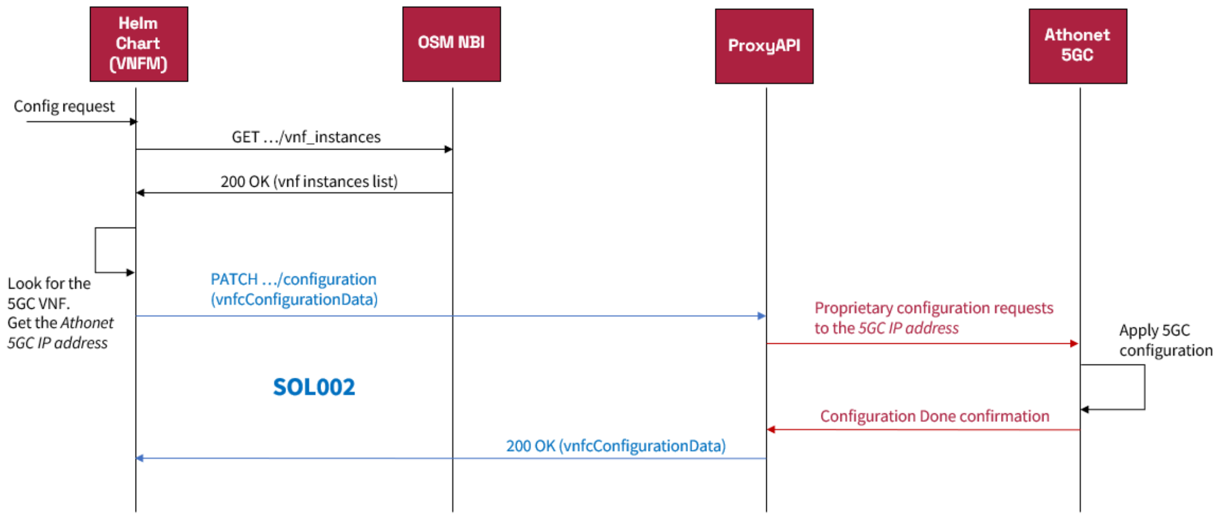

Helm Chart (VNFM)
OSM NBI
ProxyAPI
Athonet 5GC
Config request
GET …/vnf_instances
200 OK (vnf instances list)
Look for the 5GC VNF. Get the *Athonet 5GC IP address*
PATCH …/configuration (vnfcConfigurationData)
SOL002
Proprietary configuration requests to the *5GC IP address*
Apply 5GC configuration
Configuration Done confirmation
200 OK (vnfcConfigurationData)

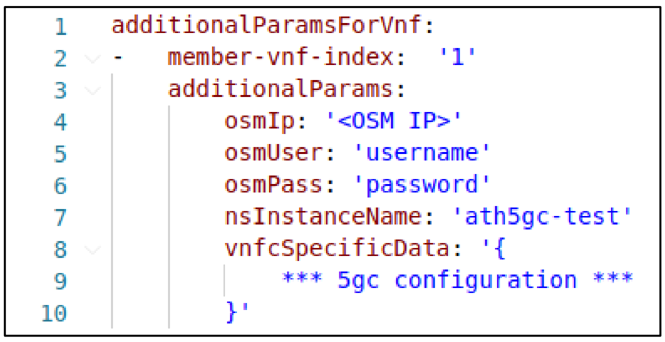
Figure 6: Flow chart representing the set of requests for the 5GC configuration among deployment components.

3.1.3 Deployment and Performance Results

The final 5GC deployment prototype has been successfully tested in the framework. The instantiation is performed by OSM command line interface, with the following command:

```
osm ns-create \
--ns_name ath5gc-test \
--nsd_name ath-5gc-proxy-ns \
--vim_account openstack01 \
--config_file 5g_configuration.yaml
```

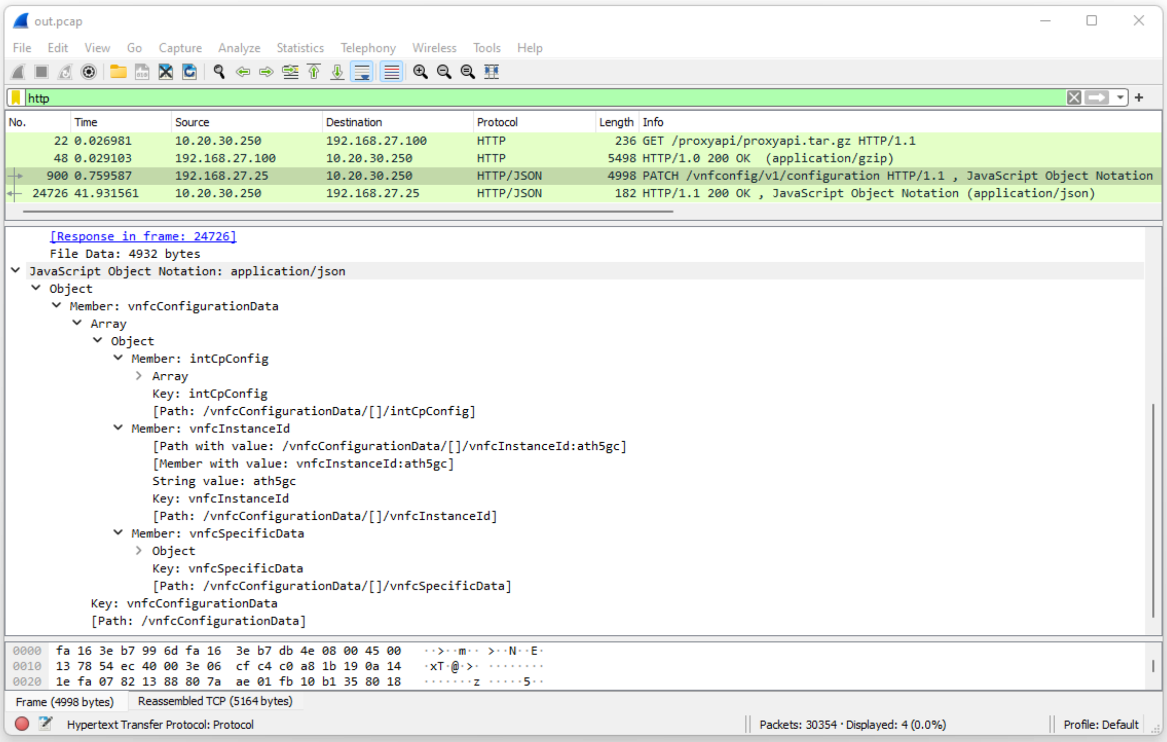
where `config_file` indicates the inclusion of an additional configuration file (5g_configuration.yaml) to be included into the NS descriptor. This file includes the necessary information to properly configure the 5GC with the ProxyAPI, together with the configuration itself. The structure of the config file is shown in Figure 7 . In particular, the *vnfcSpecificData* field hosts all the needed 5GC configuration written in *json* format.



```
additionalParamsForVnf:
-   member-vnf-index:  '1'
    additionalParams:
        osmIp: '<OSM IP>'
        osmUser: 'username'
        osmPass: 'password'
        nsInstanceName: 'ath5gc-test'
        vnfcSpecificData: '{
            *** 5gc configuration ***
        }'
```

Figure 7: Additional data information for NS instantiation and 5GC VNF configuration.

Figure 8 shows the network trace of a configuration request example made by the Helm chart towards the ProxyAPI. This pcap trace has been captured in the ProxyAPI node. The packet 900 represents the PATCH configuration request, consisting of the properly URI and the payload, shown below, structured according to the SOL002 data model presented above. Finally, once the ProxyAPI properly configured the 5GC with its proprietary APIs, it replies to the Helm chart with a 200 (OK) response to confirm the execution.


out.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http

```
No.    Time       Source           Destination      Protocol   Length Info
    22 0.026981   10.20.30.250     192.168.27.100   HTTP          236 GET /proxyapi/proxyapi.tar.gz HTTP/1.1
    48 0.029103   192.168.27.100   10.20.30.250     HTTP         5498 HTTP/1.0 200 OK  (application/gzip)
   900 0.759587   192.168.27.25    10.20.30.250     HTTP/JSON    4998 PATCH /vnfconfig/v1/configuration HTTP/1.1 , JavaScript Object Notation
 24726 41.931561  10.20.30.250     192.168.27.25    HTTP/JSON     182 HTTP/1.1 200 OK , JavaScript Object Notation (application/json)

    [Response in frame: 24726]
    File Data: 4932 bytes
JavaScript Object Notation: application/json
  Object
    Member: vnfcConfigurationData
      Array
        Object
          Member: intCpConfig
            Array
            Key: intCpConfig
            [Path: /vnfcConfigurationData/[]/intCpConfig]
          Member: vnfcInstanceId
            [Path with value: /vnfcConfigurationData/[]/vnfcInstanceId:ath5gc]
            [Member with value: vnfcInstanceId:ath5gc]
            String value: ath5gc
            Key: vnfcInstanceId
            [Path: /vnfcConfigurationData/[]/vnfcInstanceId]
          Member: vnfcSpecificData
            Object
            Key: vnfcSpecificData
            [Path: /vnfcConfigurationData/[]/vnfcSpecificData]
      Key: vnfcConfigurationData
      [Path: /vnfcConfigurationData]

0000  fa 16 3e b7 99 6d fa 16  3e b7 db 4e 08 00 45 00   ··>··m·· >··N··E·
0010  13 78 54 ec 40 00 3e 06  cf c4 c0 a8 1b 19 0a 14   ·xT·@·>· ········
0020  1e fa 07 82 13 88 80 7a  ae 01 fb 10 b1 35 80 18   ·······z ·····5··

Frame (4998 bytes)   Reassembled TCP (5164 bytes)
Hypertext Transfer Protocol: Protocol     Packets: 30354 · Displayed: 4 (0.0%)     Profile: Default
```

Figure 8: pcap trace captured in the ProxyAPI node during the configuration operation.

5GC instantiation and configuration delay have been measured to compare with the previous obtained delay (with Juju). Figure 9 illustrates the set of delays respect to all the instantiation and configuration process. Some procedures are changed because of Helm usage instead of Juju.

We can notice a first improvement in the VM instantiation: in D4.2, the VM instantiation was found to be about 130 seconds; thanks to the framework improvement and lighter VMs, we reached an instantiation time of 32 seconds.

The most important improvement is obtained in the actual 5GC configuration, where the Helm environment is immediately ready in the Kubernetes system, and the Helm chart installation takes about 21 seconds. Finally, also the actual 5GC configuration process results faster respect to the EPC counterpart. Considering all the tasks, the total configuration process takes about 107 seconds, compared to the 363 seconds obtained from the EPC optimized configuration (see D4.2).

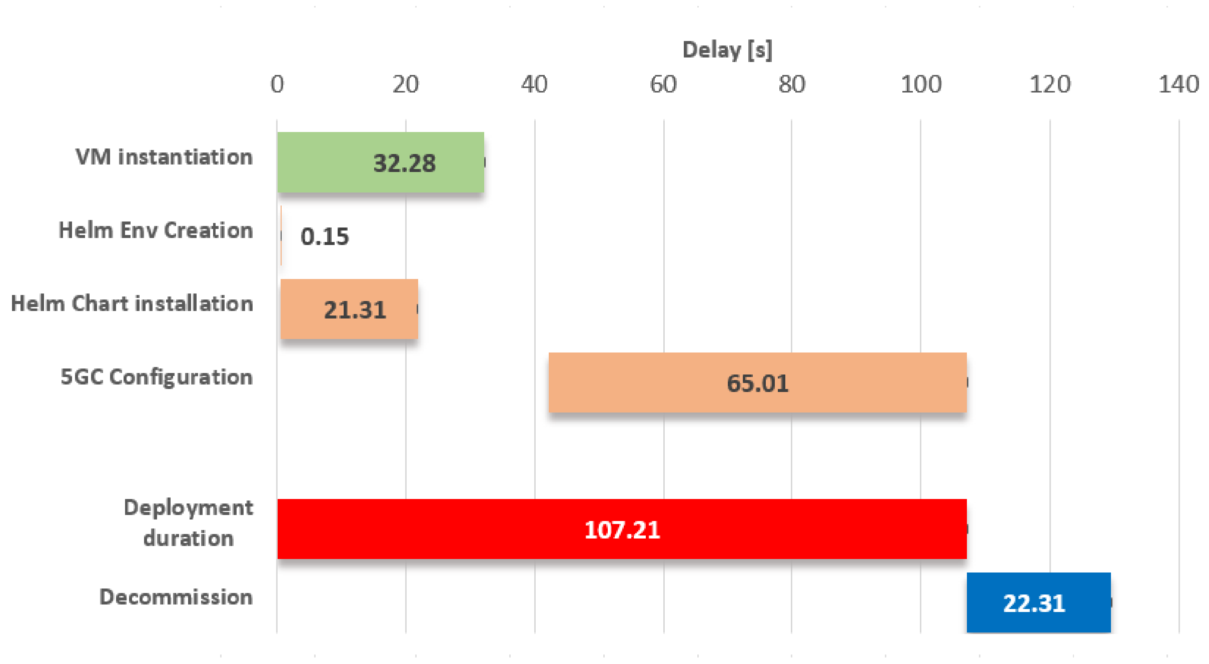


Figure 9: 5GC deployment delay with configuration through Helm chart.

3.1.4 Network Slice Subnet orchestration with OSM

As a complementary work, we developed a provisioning procedure of a Network Slice Subnet (NSS) modeled as a Network Service (NS), composed of several VNFs. The work [ISH22] was presented during a live demo at the *IEEE International Conference on Network Softwarization, 27 June–1 July 2022, Milan, Italy.* The proof of concept developed within the framework of 5G CONNI's WP4 demonstrates the implementation of an on-demand provisioning procedure of a Network Slice Subnet composed of VNFs from potentially different vendors. It includes a Network Management System (NMS) conforming to the 3GPP Service-Based Management Architecture (SBMA), an ETSI MANO orchestrator, and a Network Function Virtualization Infrastructure (NFVI).

3GPP has a long tradition of specifying the Operation, Administration, Maintenance and Provisioning (OAM&P) of Telecommunications Networks [TS32101]. Management functions are separated into layers, going from the *Element Management* layer to the *Network*, *Service*, and *Business* Management layers. Management at the *Service* and *Business Management* layers is focused on processes and is compatible with TMForum specifications [TS32101]. Management in the *Network* and *Element Management* layers, which takes care of the management of *Faults, Configuration, Accounting, Performance, and Security*, so-called FCAPS management, is adopted from the ITU-T Telecommunications Management Network [ITU00].

With the advent of VNFs communicating over virtualized networks, the framework has been extended to include *Lifecycle Management*. ETSI has developed the Management and Orchestration (MANO) framework for the lifecycle management and orchestration of virtualized resources made available by an NFVI–- which can abstract the functionalities provided by cloud-like infrastructure management systems [GSNFV]- and has been integrated by 3GPP in its own SBMA [TS28500].

Lastly, 5G is based on the concept of Network Slices [TS23501], logical 5G Systems (5GSs), abstractions over the concrete network, realized by means of physical or virtualized resources.

Network slicing is essentially a management concept whereby a 5GS’s resources are allocated wholly or in part to logical or, in analogy to other technologies, virtual 5GSs.

SBMA, like the 5G control plane, is service-based [TS28533]. It makes use of so-called Management Services (MnS) utilizing a REST paradigm: ProvMnS for for lifecycle management and configuration; FaultSupervisionMnS for fault supervision; StreamingDataReportingMnS and FileDataReportingMnS for performance monitoring and reporting.

MnSs act upon Managed Object Instances (MOIs), abstract views of the 5GS’s resources. MOIs are instances of Information Object Classes (IOCs) which, like object-oriented models, have attributes but no methods. The IOCs are modelled in the Network Resource Model (NRM) [TS28541], a tree structure similar to an LDAP tree. The set of MOIs representing a network and its components is called its Management Information Base (MIB). Management Systems are populated with the MIB which provides the management view of the network and through the modification of the MOI attributes, provides the means to control it.

Although similar to SNMP in concept, there are some relevant differences. Unlike SNMP, the MOs are identified by Distinguished Names (DN) a la LDAP, the MIB tree is not rooted in some universally known MO, the protocol is HTTP and is thus extensible, the NRM is JSON-based and not ASN.1, and vendor-specific extensions are allowed through the use of Vendor-Specific Data Containers (VSDC). Moreover, the MIB itself is not necessarily known a-priori, but is populated dynamically as network slices get created and VNFs instantiated.

The scope of our work was to develop a proof-of-concept NMS for the creation and configuration of a Network Slice Subnet (NSS), constituent part of a Network Slice, composed of VNFs from potentially different vendors [TS28530].

3.1.4.1 Related works and our contribution

A survey of the end-to-end slicing models advanced by various organizations is presented in [CDD+21] and although 3GPP’s approach to network slicing is described therein, the focus is on the Transport Network.

Isolated experimentation platforms realized as Network slices in a Pan-European network are envisioned in [OTR20], and although the 3GPP network slicing model is described, attention is on TMForum-based service ordering.

The actual realization of the slice after it has been ordered is the subject of [OL+18], and is based entirely on the ETSI MANO, totally ignoring the 3GPP OAM, even ascribing the role of an Element Manager to an SDN Controller.

Going beyond the cited state of the art, our setup is grounded in the 3GPP SBMA, which makes use of the ETSI MANO Framework to realize Network Slice Subnets and VNFs. 3GPP SBMA then manages the slices, also allowing for NFs that may not be virtualized, such as RAN equipment.

We employ vendor-specific NF Management Functions (NFMF), developed for the purpose of this work, to cater for the differences in VNF implementation. Those that employ specific techniques, such as e.g. micro services, fetch VNF instance information from ETSI MANO and pass it to the VNFs in dedicated VSDCs. The NFMFs can also take care of any MANO non-compliance to ETSI specifications as is the case with OSM, which, despite being an ETSI initiative, is not totally compliant.

3.1.4.2 Proof of Concept’s Setup and Demonstration Steps

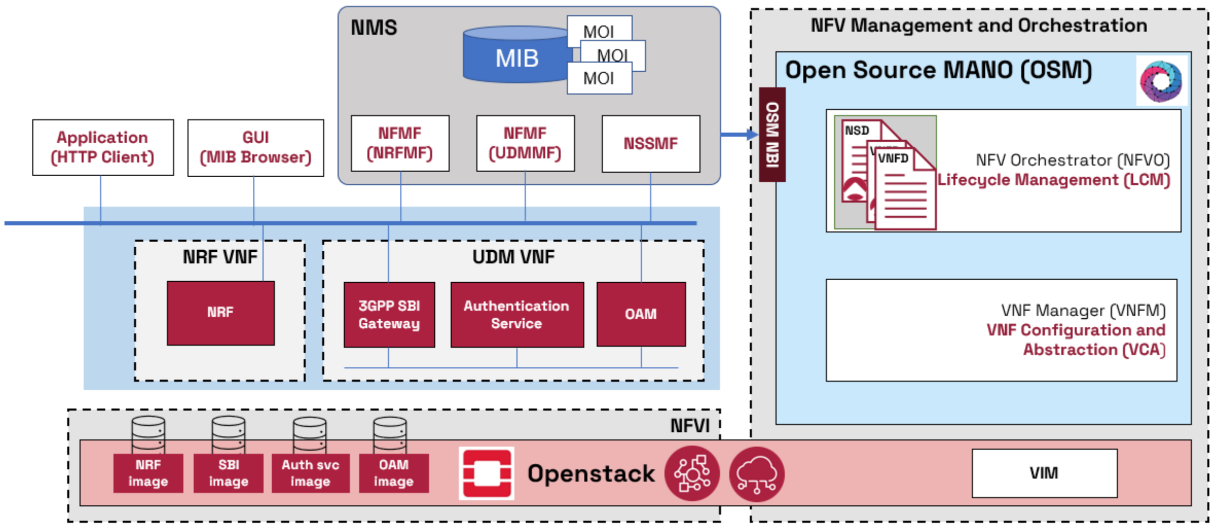


Figure 10: Components of the proof of concept on for on-demand creation, configuration, and control of a network slice subnet.

Our work demonstrates the provisioning, configuration and control of the exemplary NSS shown in Figure 10. It is composed of two NFs, a Unified Data Management (UDM), and a Network Repository Function (NRF) [TS23501], implemented by different developers, mimicking vendor-independence. The UDM is implemented as an agglomeration of independent micro services and is SBMA-aware, whereas the NRF is monolithic and does not expose SBMA-compliant management services.

We consider these two NFs to be sufficient to demonstrate an NSS for the purposes of our prototype, also because they do not depend on the functionality of other NFs in the 5G core network [TS23501]. In addition, each UDM micro service is modelled as a VNF Component (VNFC), thus permitting the demonstration of the instantiation of a VNF with multiple VNFCs.

Starting with an NMS that is not managing any VNF or Network Slice, we provision the NSS by invoking the ProvMns operation AllocateNssi [TS28531]. The VNFs are instantiated in OpenStack (release Victoria) by OSM (release Ten) and MOIs are created representing the resources of the NSS. This can be verified by invoking the ProvMns operation GetMoiAttributes on the SubNetwork resource, or through the GUI of our demonstrational setup.

Using command-line-interface tools to invoke HTTP requests towards the exposed NF interfaces, it is also possible to verify that the instantiated VNFs do indeed work and that their state can be controlled by changing the relative MOI attributes either through ProvMns operations or through the GUI.

The setup, hosted at Athonet's Research and Innovation labs, is made up of a 3GPP SBMA-compliant NMS, including the NSSMF and the NFMFs, and OSM, a fairly stable open-source implementation of the ETSI MANO Framework, configured to use OpenStack as the NFVI, employing a Dell R640 server. OSM provides an OpenStack Virtual Infrastructure Manager (VIM) for instantiating and orchestrating Virtual Machines (VMs) and Networks. The NMS is deployed in a different PC. The NMS hosts the root SubNetwork MOI, which namecontains the MOIs of Management Nodes and other Managed Entities.

Creation and operation of a Network Slice can be follows the four phases described in [TS28530]. Over the described setup, it is possible to demonstrate the following three: Preparation, Commissioning, and Operation.

1. *Network Slice Design and Preparation Phase*: in this phase [TS28530], the NSS, including its networking, is designed, and that information is provided to the NMS and NFV MANO.
   The NSS corresponds to a Network Service (NS) in NFV MANO [TS28541], and is described in a Network Service Descriptor (NSD) [GSNFVIFA]. The NSD specifies the required VNFs – described in VNF Descriptors (VNFDs) - and their networking requirements. Each VNF is made up of one or more VNFCs - described as Virtual Deployment Units (VDUs) in the VNFD [GSNFVIFA+] - and specify the component's Compute, Storage, and Network resource requirements. The NSD and the constituent VNFDs are defined in yaml files and together with the software images of the constituent VNFs are on-boarded in OSM by means of its GUI or the command-line interface.
   From the SBMA point of view, the NSS and the constituent NFs are all part of a Sub-Network [TS28530]. The NSS is associated with a Slice Profile, which specifies the characteristics of the NSS, including the PLMN and the Slice Service Type. The characteristics and their exact meaning are specified by GSMA [GSMA20].
   The supported Slice Profile together with the NSD Id which must be used to instantiate the corresponding NS in ETSI MANO is registered in the NSSMF.
   The vendor-specific NFMFs together with the VNF products they will manage are registered in the NMS so that it can create entries in it MnsRegistry to their ProvMns interfaces. This information is then be used by the NSSMF to request the creation of VNF-related MOIs.
   Slice Profile registration in the NSSMF and NFMF registration in the NMs are not subject to standardization.
2. *Network Slice Commissioning Phase*: In this phase, the NSS is created. The NSSMF requests ETSI MANO to create and instantiate an NS instance based on the specified NSD [GSNFVSOL].
   The NSSMF fetches the information on the instantiated NS from ETSI MANO, determines the instantiated VNFs, together with the allocated virtualized resources, including networking, and requests the NFMFs of the VNFs to create the VNF-related MOIs. NMS receives notifications of the creation of the relative MOIs and updates its view of the MIB.
   Finally, the NSSMF creates a MOI for the instantiated NSS and notifies the NMS. A fragment of the resulting MIB is represented in Figure 11.
3. *Network Slice Operation Phase*: Through the command-line interface or the GUI, it is possible to invoke ProvMnS operations on the MOIs and to inspect their attributes. In particular, it is possible to see which NFs make up the NSS, which 5G services they offer and the relative Service Access Points, what the state of those services is, and whether the UDM is registered in the NRF or not. In particular, the NSS is initially not active with the 5G services provided by its constituent VNFs unavailable, with *administrativeState* set to "LOCKED". It is possible to activate the NSS by setting its MOI's *administrativeState* to "UNLOCKED". This leads to the "unlocking" of the constituent VNFs, enabling the registration of the UDM in the NRF and thus its discovery by other NFs. One can then verify the effective activation of the NSS by invoking the 5G services of the NFs.

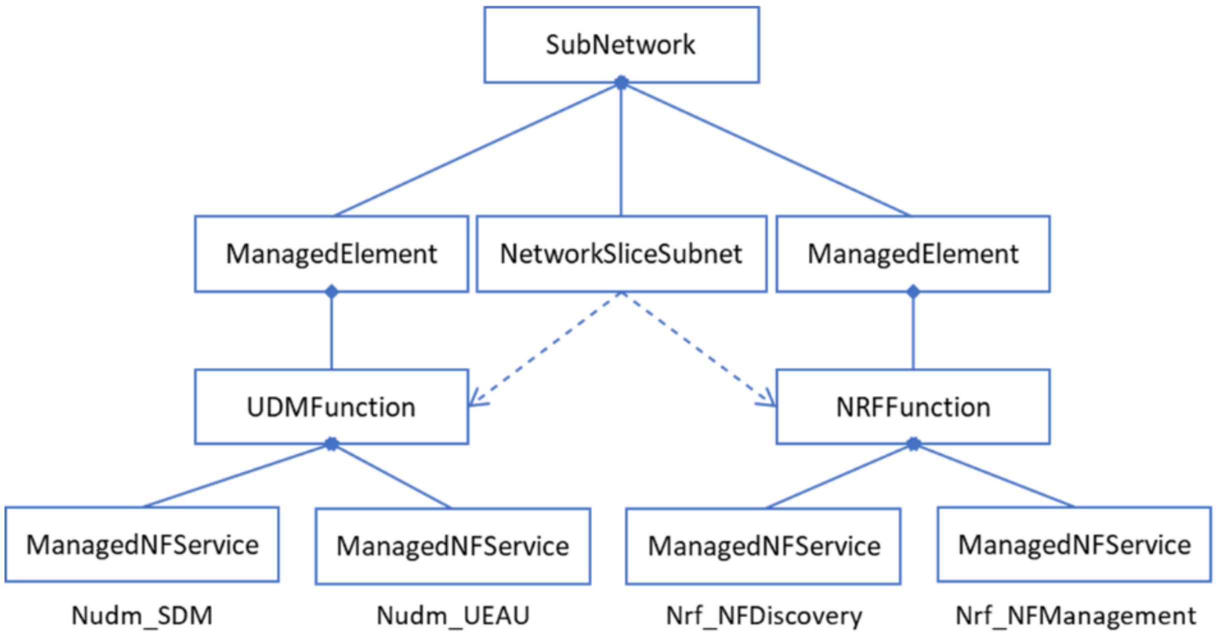


Figure 11: MIB fragment created for the demonstrated NSS.

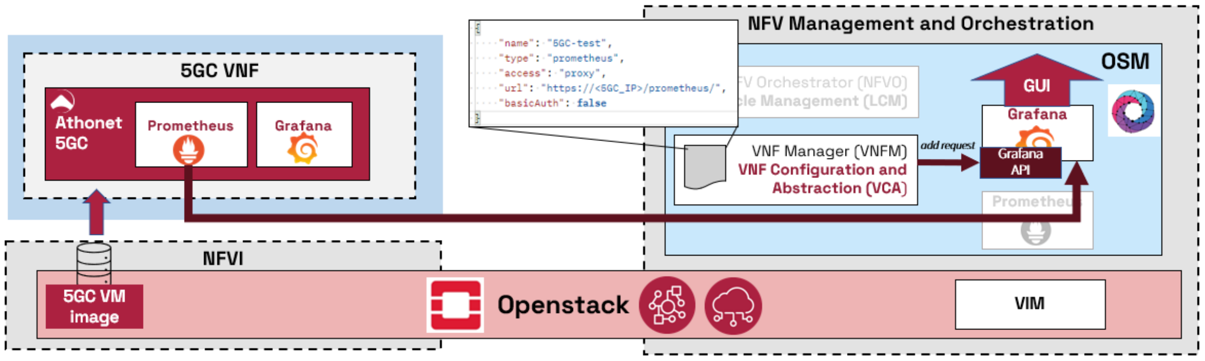
3.1.5 5GC VNF-specific metric collection

Another research activity related to the orchestrator is about OSM metric collection. In the previous deliverable version (D4.2), we described and performed the OSM approach to retrieve VIM metrics (CPU, memory, disk, and network usage) and to visualize them in its Prometheus/Grafana instances. In updating the orchestration framework, the next objective was to find a way to forward VNF-specific metrics (in our case the Athonet 5GC metrics/KPIs) to OSM in order to have a unique centralized point of monitoring.

However, OSM does not support a standard and clear way to retrieve VNF-specific metrics. As mentioned by the OSM team, a Juju functionality, called Juju Metrics, shall be leveraged to collect those metrics, and eventually trigger scaling/healing processes. Unfortunately, this mechanism results to have the following drawbacks:

- It is broken in the latest versions of OSM (release > Eight);
- It is not well documented, and the current documentation refers to the old framework;
- It implies a ssh access to the target VM to read data.

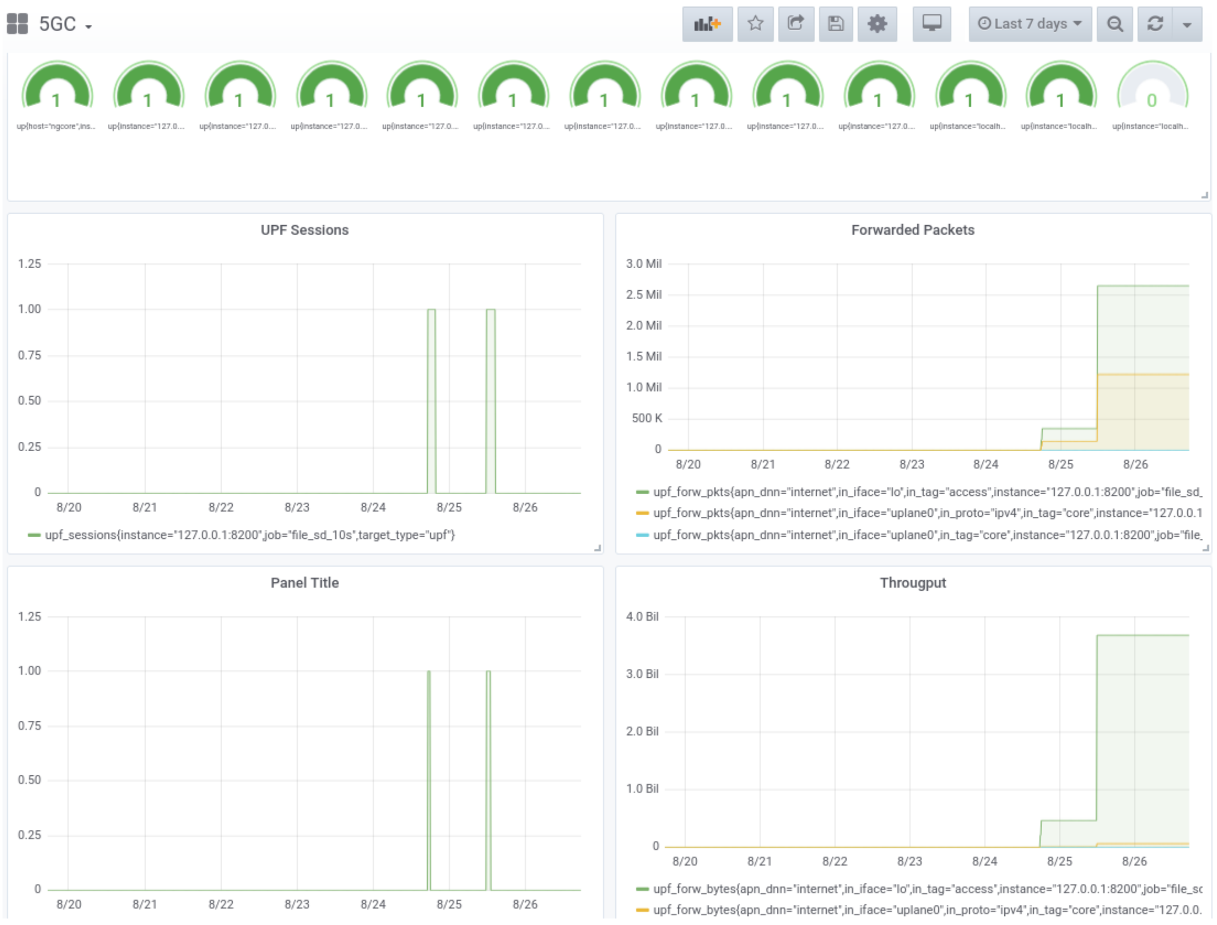
Several alternatives to Juju have been considered to collect VNF metrics. The one of them that has been implemented in our framework deals the addition of 5GC Prometheus as data source in OSM Grafana, in order to visualize and customize all the metrics' graphs. This approach has been chosen because it can programmatically make an automatic HTTP request for a data source addition by adoption of dedicated Proxy Charm/Helm Charts during the 5GC VNF instantiation. In Figure [], this approach is illustrated; note the VNFM (where Proxy/Helm runs) that makes the request for adding the 5GC Prometheus data source towards the Grafana API. The HTTP request specifies all the needed information to retrieve metrics from the target source (e.g., IP address and source type).



```
{
    "name": "5GC-test",
    "type": "prometheus",
    "access": "proxy",
    "url": "https://<5GC_IP>/prometheus/",
    "basicAuth": false
}
```

Figure 12: 5GC metrics collection configuration in OSM's Grafana.

Once Prometheus has been added as a data source, Grafana makes the metrics available to create plots. *Figure 13* shows a group of graphs created and customized showing some of the metrics made available by 5GC (e.g., status of NFs, number of UPF sessions, forwarded packets, throughput).



upf_sessions{instance="127.0.0.1:8200",job="file_sd_10s",target_type="upf"}

upf_forw_pkts{apn_dnn="internet",in_iface="lo",in_tag="access",instance="127.0.0.1:8200",job="file_sd_
upf_forw_pkts{apn_dnn="internet",in_iface="uplane0",in_proto="ipv4",in_tag="core",instance="127.0.0.1
upf_forw_pkts{apn_dnn="internet",in_iface="uplane0",in_tag="core",instance="127.0.0.1:8200",job="file_

upf_forw_bytes{apn_dnn="internet",in_iface="lo",in_tag="access",instance="127.0.0.1:8200",job="file_sc
upf_forw_bytes{apn_dnn="internet",in_iface="uplane0",in_proto="ipv4",in_tag="core",instance="127.0.0.

Figure 13: Example of plot creation and customization from 5GC metrics.

3.2 5G Core prototype

The 5GC provides interfaces and integrates with Application Network (AN), Mobile Edge Computing (MEC), and gNB. The 5GC contains the resources for setting up and maintaining the traffic between the gNBs and AN can support multiple gNBs. The 5GC is capable to achieve throughput to 40Gbps and data plane latency less than 1ms.

Local traffic within the UPF could be routed locally via the co-located Mobile Edge Computing (MEC). This feature provides some additional flexibility in the location of the traffic to the local application network. The 5GC solution from III consists of six key network elements, as shown in Figure 14:

- Access and Mobility Management Function (AMF)
- Session Management Function (SMF)
- User Plane Function (UPF)
- Authentication Server Function (AUSF)
- Unified Data Management (UDM)
- Policy Control Function (PCF)

The 5G Core also provides interfaces to support Network OAM (Operations Administration and Maintenance) functions. In order to communicate with OAM system efficiently, III 5GC is modified to send data to OAM through MQTT (Message Queuing Telemetry Transport) protocol, and the architecture is shown in Figure 15.

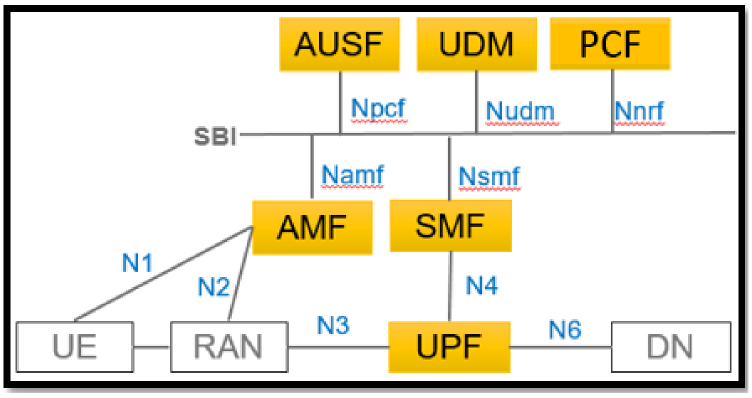


Figure 14: III 5G Core architecture

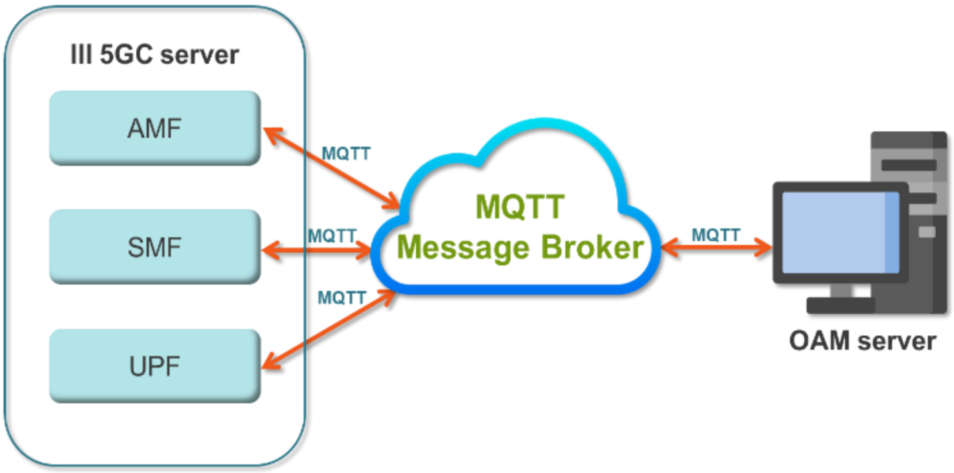


Figure 15: III 5G Core with MQTT support

3.3 Conclusions

Obtaining a lean and efficient orchestration system, compliant with the ETSI standard, was one of the main objectives of T4.2. This framework is suitable for scenarios in which immediate deployment of a complete and configured network is required. In this respect, the prototype of the orchestration framework was proposed. It includes OSM and some specific implementations by ATH, which speed up the instantiation and configuration of a 5GC system. In this case, the configuration process and semantics have been implemented following the standard ETSI specifications (SOL002) in order to demonstrate the possibility of being agnostic to the specific vendor product (on both VNF and MANO sides). In addition, in order to monitor 5GC metrics and alarms in a single point of control, the orchestration system has been modified in that sense. Additional complementary activities have been conducted leveraging the implemented orchestration prototype.

Moreover, it is important to highlight that adherence to standards is essential in a multi-vendor world. The work presented in Section 3.1.4.2 demonstrates how it is possible to implement a standard-compliant system to manage a 5GS, and in particular to manage its network slicing aspects, with relatively little system integration effort, and compatible with the combination of components from different vendors into an open environment (i.e., not subject to vendor lock-in).

The 5G core components are defined as self-contained, independent and reusable network functions (NFs) together with a well-defined Service Based Interface (SBI) using HTTPv2 that can be used to invoke services. By this way, throughput can be achieved to 40Gbps and data plane latency can be reduced to less than 1ms. Also, MQTT support is provided to support 5G OAM functions efficiently.

4 Mobile Edge Cloud Enablers

The objective on task 4.3 is to develop MEC technologies to be deployed in the project's testbeds, which support the requirements of smart factories in 5G eMBB and URLLC scenarios. Sections 4.1 and 4.2 present the MEC implementation methods in 5G-CONNI for the European and the Taiwanese testbeds, respectively. The hybrid 5GC solution is implemented in European testbed and the bump-in-the-wire solution is implemented in Taiwanese testbed

4.1 MEC based hybrid architecture

Hybrid 5GC solution consists in the splitting of CP and UP functions, giving more flexibility for the mobile network deployment. This approach allows the support of edge computing, with the consequent possibility of installing a MEC platform. This solution was considered for the European testbed of 5G-CONNI and in D4.2 the creation and integration of a second NFVI node at Athonet premises has been described to allow the deployment of such a hybrid scenario in the testbed.

Based on what was envisaged previously, the actual deployment of a 5GC with separated CP and UP through OSM has been finalized. According to what has been designated, the 5GC CP network functions (e.g., AMF, UDM, SMF, etc.) is deployed in the central NFVI server, while the 5GC UPF, the only component acting as UP function, is deployed in the edge NFVI server.

4.1.1 Non-Public Network deployment emulation

As simulation scenario for testing a private hybrid network architecture on the provided framework, we emulated the deployment of a typical 5G Non-Public Network (NPN). NPNs are mobile networks for private use, and recently are attracting attention in academic and industrial contexts. (e.g., Industry 4.0, smart grids, public safety, mission-critical communication, Internet of Things (IoT), indoor communication and positioning). In particular, what we emulated are two deployment options:

- Standalone NPN (SNPN), a 5GS deployment option for private use which leverages a novel approach for identifying such network, that is, via the combination of a PLMN identifier and a network identifier (NID).
- Public Network Integrated NPN (PNI-NPN), a 5GS deployment option adopted when integration between the NPN and the PLMN can be desirable, especially when the private entity does not shoulder the burden of the entire NPN management.

In Figure 16, 5G NPN deployment options. The left-most solution is the SNPN solution powered by the hybrid cloud. The one in the center is the full-on-site SNPN, where the entire SNPN is deployed at the edge cloud, in proximity of the RAN. Finally, the right-most solution is a PNI-NPN, where the grey blocks represent the PLMN domain and the blue ones a private network slice comprising the central PLMN CP as well as dedicated user plane function at the edge.

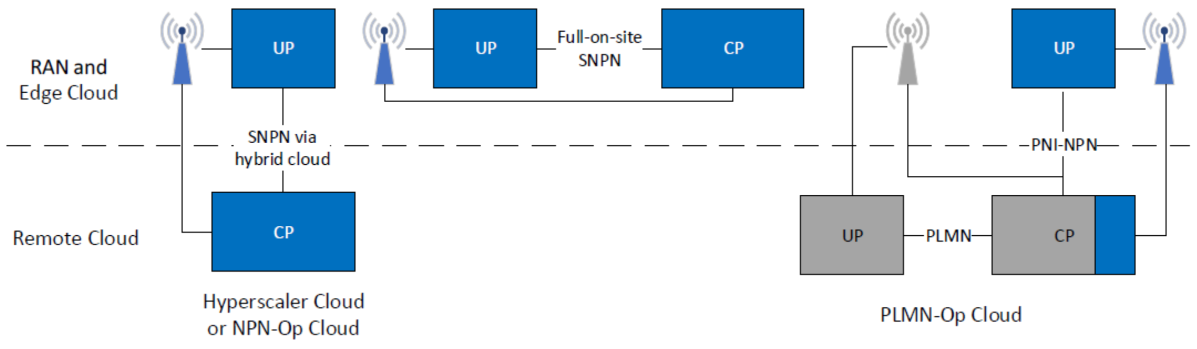


Figure 16: Three 5GC NPN deployment options.

In our framework, we have been emulating the presented NPN architectures. Let's go back to the characteristics of the two physical machines, presented in the previous deliverable:

- **Central server**: Dell PowerEdge R640 server based on two Intel(R) Xeon(R) Silver 4210R CP @ 2.40 GHz and 64 GB of memory.
- **Edge server**: Commodity Desktop PC equipped with an Intel(R) Core(TM) i7-2600 @ 3.40 GHz and 16 GB of memory running Ubuntu 20.04.

In particular, the central server node simulates a remote cloud datacenter, while the edge node simulates a constrained edge server deployed on the premises of the private entity or network tenant. Dedicated NS and VNF descriptors and configurations have been provided to deploy the scenario, specifying which private 5GC instance must be placed in the central and edge server and which NFs have to be enabled/disabled in those instances.

Results

Both SNPN and PNI-NPN models have been deployed on the infrastructure and their deployment automated using OSM descriptors. As for the SNPN model, the entirety of the CN functionalities is deployed on the edge node. As for the PNI-NPN model, the NPN is served as a network slice of a PLMN deployed on the remote cloud, with which it shares all the control plane functionalities. Instead, the user plane functionalities of the NPN are not shared and are deployed independently on the edge cloud in order to preserve the advantages of a MEC architecture. Both proprietary (Athonet Griffone) and open-source (Open5GS) 5GC implementations have been tested.

4.1.2 MEC IoT Service Experimental Evaluation

As part of T4.3 activities, ATH worked on the technical definition and feasibility of the MEC IoT Service being standardized in MEC033 - IoT API specification [ETSIMEC]. The MEC IoT Service (IoTS) enables the integration of IoT platforms in the MEC system, and it exposes IoT APIs to enable the configuration, discovery and provisioning of IoT systems.

Figure 17 shows a MEC IoTS scenario, which considers an edge facility comprising a MEC host served by a data plane. The figure also shows a usage example of IoT framework setting: when an IoT administrator needs to configure an IoT system, the following components must be configured:

- The IoT platform itself (not MEC compliant), comprising a dedicated message bus (MB);
- The IoT device, equipped with a mobile network transceiver and a USIM;

- The IoT application which interacts with the IoT platform.

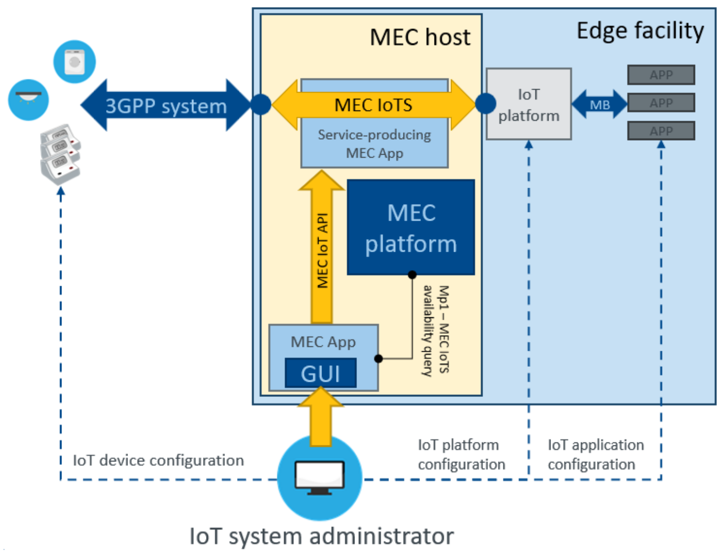


Figure 17: MEC IoT API framework architecture.

In the testbed, ATH instantiated the following components:

- An E-UTRAN simulator (srsRAN);
- An Athonet EPC;
- A dedicated VNF for a MEC IoTS mock-up.

All these components have been properly configured to communicate among themselves. The complete framework architecture is illustrated in Figure 18, showing the three mentioned components instantiated as VNFs on the NFVI. As shown, the orchestration functionalities are not leveraged for this demonstration.

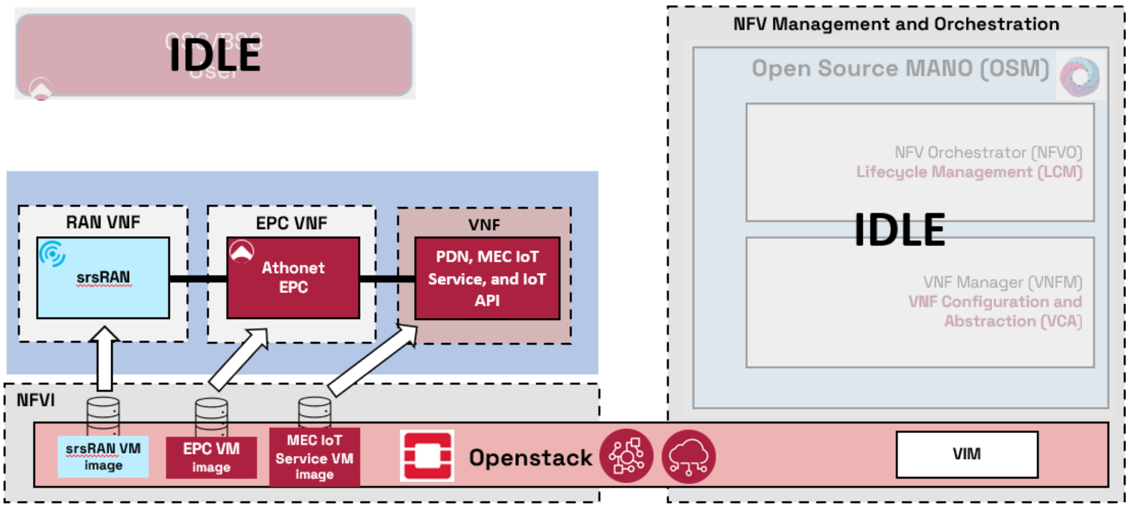


Figure 18: IoT service and API system architecture deployed in ATH premises.

MEC IoT Service Demonstration Results

In order to demonstrate the whole system working, several IoTS API calls have been implemented and tested, mostly related to device and IoT platform registration on the MEC system. We report here two examples of operation we performed.

The first one is the “*Device Registration*” operation, which allows a service consumer to register a new device, and then to create an association between a device and a traffic rule. During the execution of this call, the service consumer requests a device registration with a POST method, sending all the required information, and then the IoTS acknowledges the operation sending the list of MEC platforms where the device is registered.

The second operation is “*IoT Platform Registration*”, which allows a service consumer to register an IoT platform on the MEC platform. The registration request payload information is based on *IotServicePlatformInfo* resource type, which represents the information associated to an IoT platform.

4.2 MEC based bump-in-the-wire architecture

Facing the fast-changing market demands, smart factories need characteristics of steady operation, flexible deployment and efficient performance to adapt to various requirements of machining processes. Therefore, we meet the above features of smart factories by implementing a MEC platform with virtualization capability and VNF management. The virtualization capability is responsible for virtualizing edge computing functions, industrial applications and any needed applications to the MEC platform to realize flexible deployment. After virtualizing, the VNF management has to support the essential management demands of ETSI NFV specifications and adjust the performance of VNFs to achieve the required transmission delay, execution performance and resource management. These two capabilities can obtain more stable and efficient machining processes by implementing the MEC platform.

VNF Management can bring benefits such as VNF resource management and lifecycle management that can help smart factories manage their industrial applications flexibly. The MEC platform also supports the network adjusting to achieve the performance for network requirements. The MEC platform in the Taiwanese deployment of the 5G CONNI project deployed MEE VNF and ITRI IMTC’s Cloud-based Controller for Fixture System application VNF. The MEC platform uses the PCI-passthrough network technology to process the required transmission delay and throughputs. The MEC platform also monitors the MEE VNF and ITRI IMTC’s Cloud-based Controller for Fixture System application VNF for continuous operation in smart factories. The industrial applications onboard the MEC platform can extend their industrial application scale conveniently and deploy new industrial applications faster.

In the industrial application virtualization stage, MEC platforms need to consider hardware resources, software drivers and network capabilities of VNFs to meet industrial application requirements. Therefore, MEC platforms must efficiently allocate hardware resources, such as CPU, memory and single root I/O virtualization, to satisfy VNFs hardware requirements and support more VNFs. Then, MEC platforms maintain execution performance and compatibility of industrial applications after virtualization by installing various software drivers. After that, MEC platforms also have to manage the network capabilities of VNFs containing routing paths, transmission performance, and network resource management. At last, enterprises will effectively virtualize industrial applications to MEC platforms by implementing the above methods.

In the Taiwanese deployment of the 5G CONNI project, we implement the use case of Cloud-based Controller for Fixture System on our MEC platform by virtualizing the cloud controller of

ITRI IMTC, as shown in Figure 19. Cloud-based Controller for Fixture System has three prime demands: command transmission requirement, application execution time and stable network capabilities. The command transmission requirement needs continuously transmit a series of data to the factory equipment periodically, so our MEC platform provides PCI-passthrough technology to achieve high transmission and low latency to satisfy the requirement. And the application execution time needs the cloud controller to complete a lot of computing actions in a short time, which has to improve the performance of the cloud controller after virtualization by installing drivers and micro services on the MEC platform. After that, the stable network capabilities require the MEC platform to provide a steady and efficient data channel from VNFs of the cloud controller to factory equipment and a less packet jitter method. Finally, we virtualized the cloud controller successfully, which realized the MEC platform to support the use case of Cloud-based Controller for Fixture System on IMTC's smart factory. Figure 21 also shows that remote users can manage applications deployment and maintain continuous operation of applications through the MEC Platform.

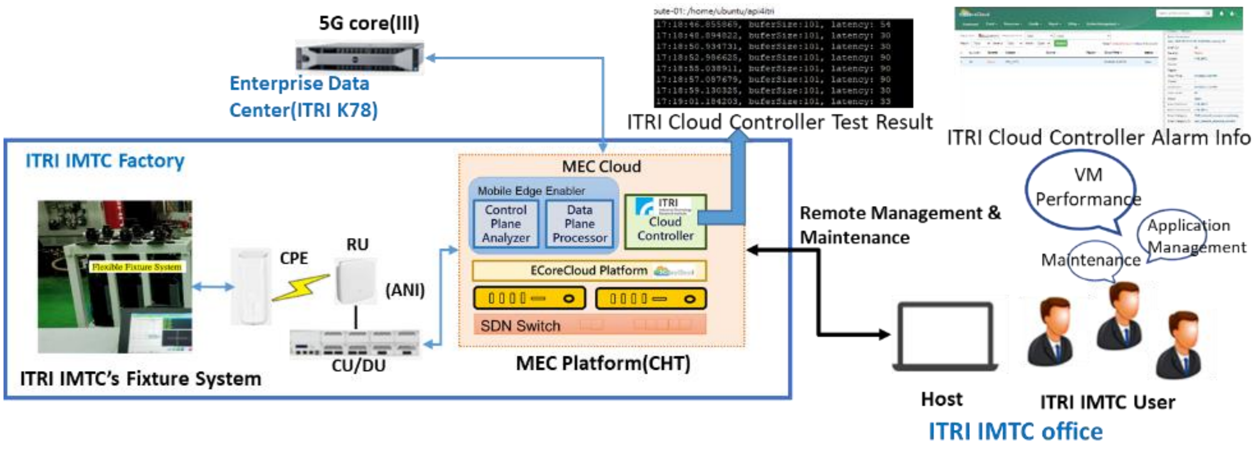

```
pute-01:/home/ubuntu/api4itri
17:18:46.855869, buferSize:101, latency: 54
17:18:48.894822, buferSize:101, latency: 30
17:18:50.934731, buferSize:101, latency: 30
17:18:52.986625, buferSize:101, latency: 90
17:18:55.038911, buferSize:101, latency: 90
17:18:57.087679, buferSize:101, latency: 90
17:18:59.130325, buferSize:101, latency: 30
17:19:01.184203, buferSize:101, latency: 33
```

Figure 19 : Cloud-based Controller for Fixture System integration with MEC Platform

4.3 Conclusions

The deployment of hybrid architectures is one of the key points of 5G CONNI project. In its lab premises, ATH has designed and deployed a multi-node framework consisting of a central and an edge node, which represent a typical hybrid architecture. Both the nodes are equipped with Openstack NFVI, in order to host the VNFs (in form of VMs) that are deployed by OSM.

As deployment tests, several NPN architectures have been simulated in the infrastructure, i.e., SNPN and PNI-NPN deployment models. Their deployment has been automated using OSM NS and VNF descriptors, and its total execution time has been measured. Finally, ATH has experimented the execution of the MEC IoT Service onto the set infrastructure.

MEC based on bump-in-the-wire architecture is the Taiwanese deployment of the 5G CONNI project. CHT provides the MEC platform including MEE VNF for traffic steering and ECore-Cloud for Cloudified NF/APP operations management described in D4.1. The MEC platform has been verified the performance for UC-2 Augmented/Virtual Reality for Process Diagnosis industrial application described in D4.2, and additional use case Cloud-based Controller for Fixture System described above. The MEC platform also provides the virtualization capability and VNF management that helps smart factories deploy the private 5G network and industrial applications integration more quickly and conveniently.

5 Industrial Application Enablers

The objective of this task is to implement the use cases selected in Task 1.1 for a proof-of-concept demonstration. Initial implementation of use cases selected in Task 1.1 at ITRI shop floor has been described in D4.2. The final application implementations are as follows.

5.1 Advanced Functionalities in Use cases

While the implementation of use case 1 has been finished and described in D4.2, there is no further functionalities at this stage. However, functionalities in use case 1 act as key foundation in use case 2 and multi-site use case and may continue to evolve during the final test and integration stage of this project.

5.1.1 Use case 2 prototype: Process Diagnosis Using Augmented/Virtual Reality

Implementation architecture for the Using Augmented/Virtual Reality for Process Diagnosis use case is shown in Figure 20. Remote rendering technique has been adopted in the implementation of this use case. The 3D model of the demo site and target machine have been built and loaded in the GPU workstation and rendered by Dassault System software. With the help of the nVidia CloudXR package, the rendered scene can be streamed to user device such as head-mount display, tablet or cell phone. The viewport of 3D scene can be updated with the IMU information from user device so that a smooth user experience of navigating through the 3D scene can be achieved.

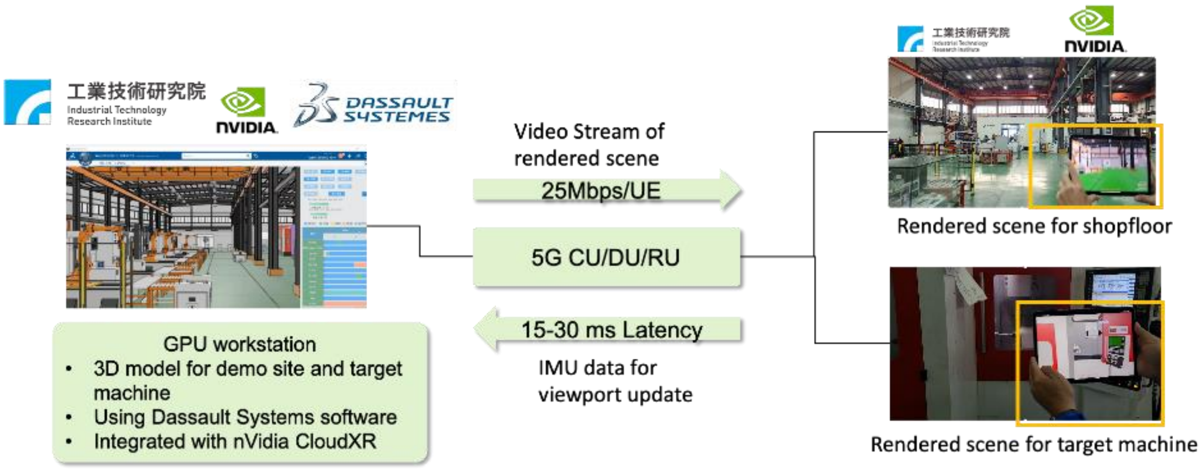

工業技術研究院
Industrial Technology Research Institute
NVIDIA
DASSAULT SYSTEMES
GPU workstation
• 3D model for demo site and target machine
• Using Dassault Systems software
• Integrated with nVidia CloudXR
Video Stream of rendered scene
25Mbps/UE
5G CU/DU/RU
15-30 ms Latency
IMU data for viewport update
Rendered scene for shopfloor
Rendered scene for target machine

Figure 20: System Architecture of UC2

Figure 21 shows the specification of the information shown in 3D scene of the target machine, where

1. Production management information: current work order for the workpiece and progress of machining process
2. Machine Operational Data: machine coordinate, spindle speed, federate, spindle loading
3. Sensing Data: 3-axis vibration data from spindle and workpiece, totally 6 channels of sensing data

The snapshot of the 3D scene is shown in Figure 22 and Figure 23. Users can toggle switch between full machine and simplified viewport according to diagnostic scenario.

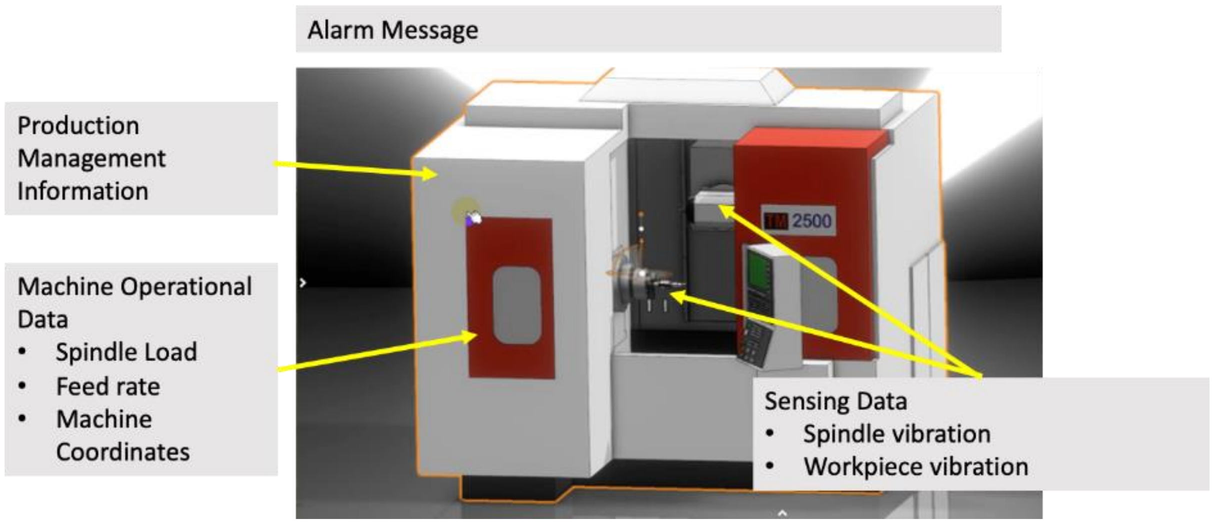


Figure 21: Specification of the 3D scene of the target machine

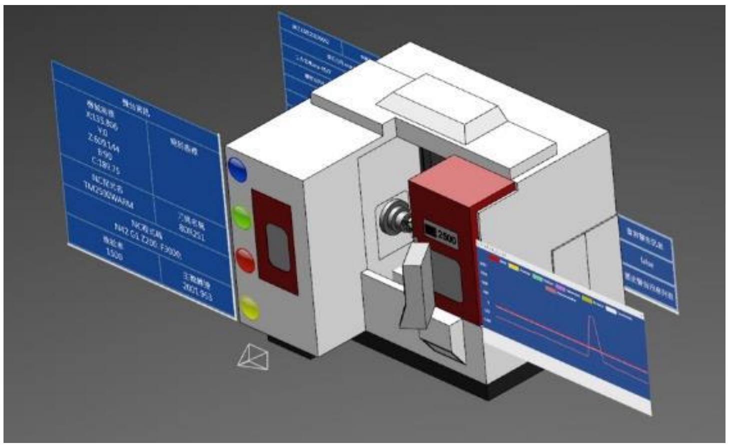


Figure 22: Snapshot of the 3D scene of the target machine (full machine viewport)

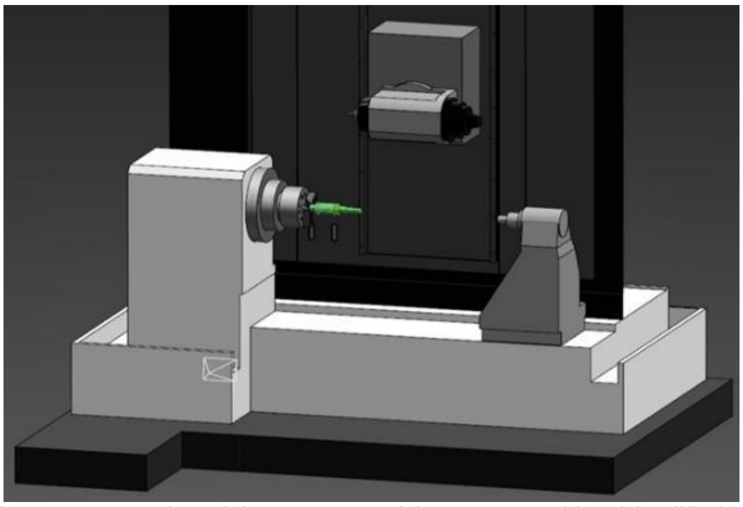


Figure 23: Snapshot of the 3D scene of the target machine (simplified viewport)

Machining error can be calculated from machine operational and sensing data during machining process and output in terms of point cloud. The point cloud of error distribution can then be shown in the 3D scene via remote rending as shown in Figure 24.

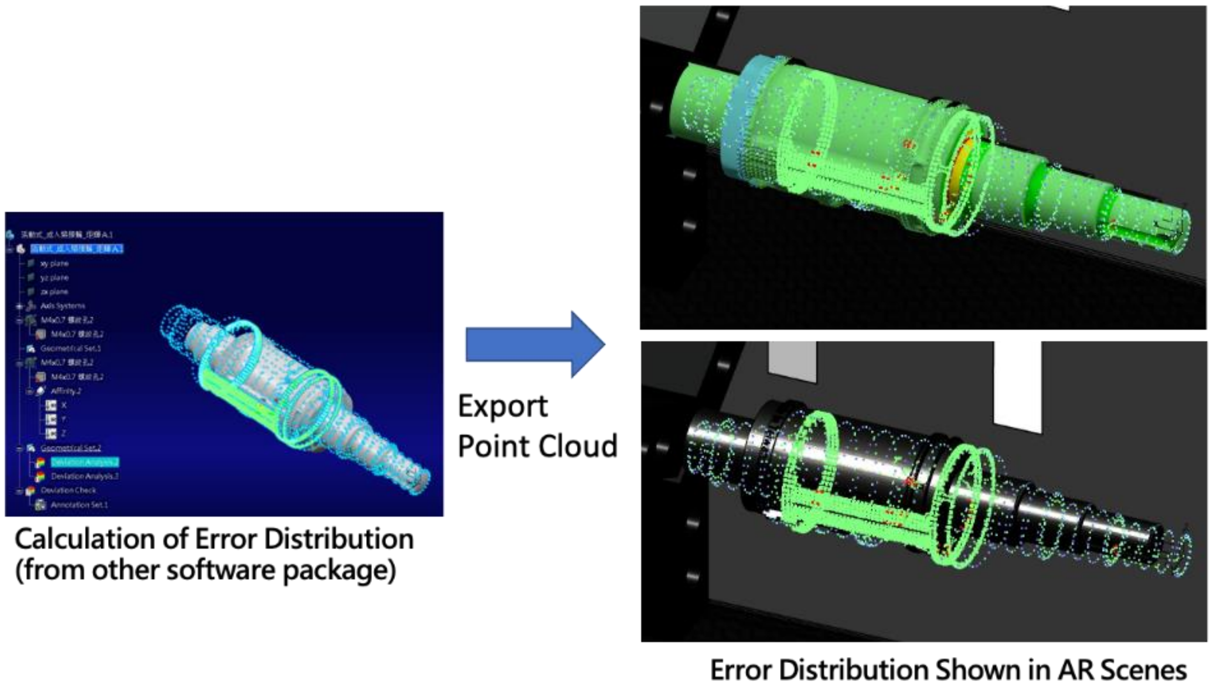

Export
Point Cloud
Calculation of Error Distribution
(from other software package)
Error Distribution Shown in AR Scenes

Figure 24: Calculation and showing of error distribution

5.1.2 Use case 3 prototype: Cloud-Based Controller for CNC

The cloud-based CNC software and the test machine has been constructed and tested under distributed network architecture (shown in Figure 25) where the motion command generation, motion command execution modules are separated. As shown in Figure 25, which is a flexible fixture system used in aerospace part machining. In the test scenario, a steel plat work piece is installed on the fixture system and exited by a shaker to simulate the vibration during machining process. To observe the effect of the vibration suppression, a cup of water was placed on the steel plat. The detected vibration signal is also shown on the sensor data acquisition system, as shown in Figure 26. When the vibrator sends a fixed vibration frequency at 35Hz, the sensor detected and send back to cloud controller. The controller sends a series of motion command packages(40 motion commands package) at 10ms time span in response to the vibration an reduced the overall vibration amplitude at 35Hz.

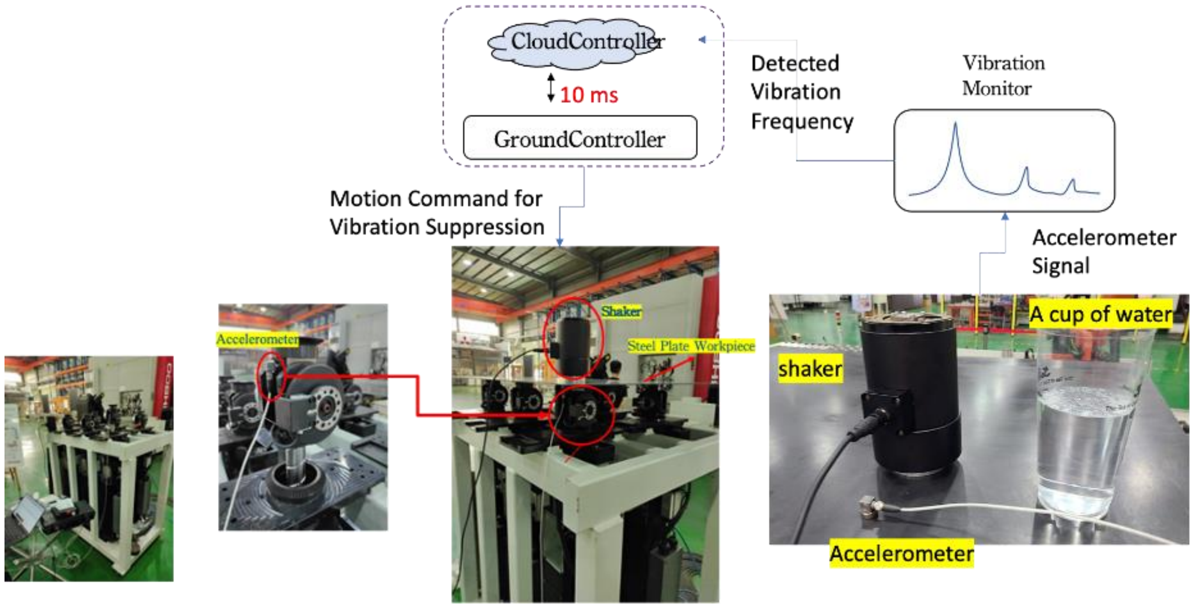


Figure 25: The test architecture of the cloud-based controller for CNC

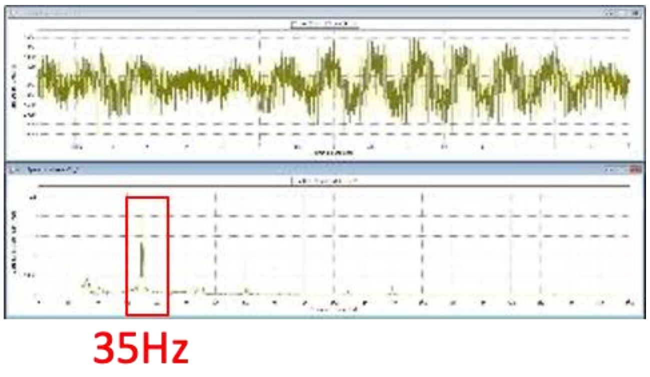


Figure 26: Detected vibration frequency

5.2 Multi-site use cases

The implementation architecture is shown in Table 4

Table 4 : Proposed architectures for multi-site use case

Architecture	Implementation Requirements
Remote Rendering with nVidia CloudXR SDK (with Dassault Soft license) (*CloudXR 3.0 not support Hololens 2)	• Workstation PC(nVidia Pascal GPU) with Dassault 3DExperience, linked with MEC. Use the Dassault product as-it-is. • Development effort base on Dassault SDK is required to link 3Dexcite with WebApi from ITRI • CloudXR developer license is re-quired

Implementation architecture is shown in Figure 27. All the required software packages, third-party components have been integrated and tested at ITRI site and will duplicate to BOSCH site

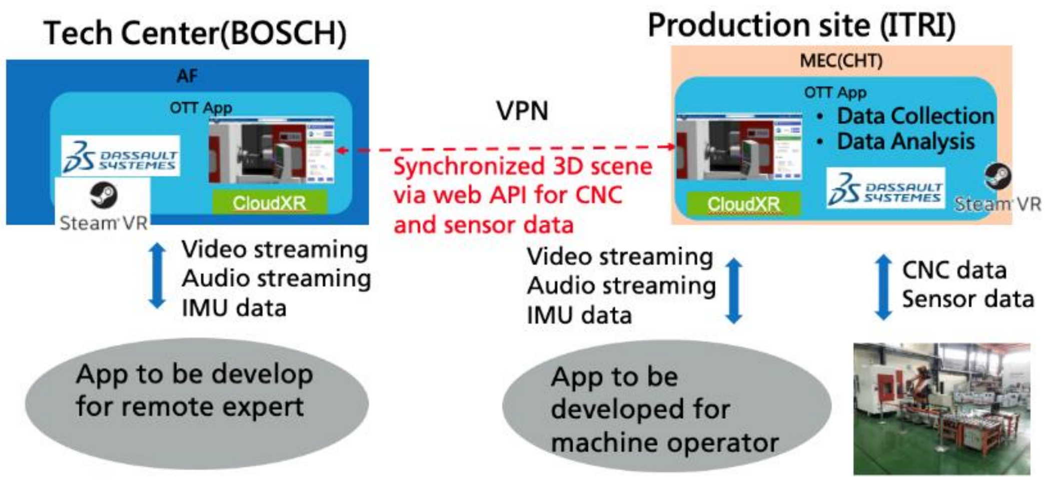


Figure 27: The implementation architecture of the multi-site use case

5.3 Conclusions

Three vertical use cases have been implemented in ITRI site, namely (1) Process Diagnostics by CNC and Sensing Data Collection (2) Using Augmented/Virtual Reality for Process Diagnosis (3) Cloud-based CNC. Among these implemented use cases, (1) & (2) were implemented on a five-axis machine tool and (3) was implemented on a flexible fixture system, which is a specialized machine to test the cloud-based controller.

For use case (1) the implementation was completed, production data, machine operation data and sensor data can be access by application of use case 2 or other data analysis modules. For use case (2), 3D model of the target machine have been created and deployed in a remote rendering server, which will then linked with MEC. Machine operator and remote expert now use the same application to conduct process diagnosis, these two instances of application share the same 3D machine model and synchronized by the machine data from the WebAPI server developed by IMTC. The prototype of cloud CNC (i.e. use case 3) has been developed and tested on the flexible fixturing system by sending vibration suppression motion commands from cloud controller to the ground controller to evaluate overall performance.

All these use cases have been tested under existing Wi-Fi infrastructure in the ITRI site and estimated to be able to test under 5G private network by the end of November. Performance data and comparison between Wi-Fi and 5G will be shown in the D5.3 deliverable.

For the multi-site use case, it is basically an extension of use case 2. The required software and hardware have been setup in both European and Taiwan side. 3D model for the target machine have been constructed by ITRI and shared to BOSCH and will conduct full connection test by the end of November.

6 Advanced functionalities

In this section, we investigates advanced functionalities for future private 5G networks: dynamic resource allocations for URLLC, dynamic strategy for resource allocation in the edge cloud and goal-oriented communication.

6.1 Advanced resource scheduling optimizations

The 5G and beyond network enables the exploitation of new emerging use cases, such as Ultra-Reliable Low Latency Communication (URLLC). Advanced resource scheduling optimizations are required to jointly reduce latency and improve reliability while maintaining appropriate efficiency. While many mechanisms exist in the literature for the first two, it is still unclear how to efficiently utilize resources while maintaining reliable, low latency communications [SYQ17]. Given the requirements for delay and reliability, two approaches to activate available resources are proposed. On the one hand, reactive strategies activate additional resources on demand, which enable efficient resource utilization, but significantly increase latency, as the demand for additional communication resources is not instantaneous. On the other hand, proactive approaches systematically apply additional resources to stretch the latency below the deadline and usually consider the worst case with a margin. Therefore, this approach implies a high cost in terms of resource utilization, especially when worst-case impairments are very rare. Our goal is to design an early decision maker, as patented in [MDC21], defining one or more decision moments to dynamically optimize the resource scheduling by adapting reactive-proactive modes to cope with various dynamic scenarios. The efficiency-latency-reliability trade-off is achieved by the timing and intensity of the decisions. The earlier (resp. stronger) the decision is made, the greater the latency gain (resp. reliability gain) at the cost of resource efficiency, and vice versa.

To highlight the benefits of early decision making in resource scheduling, Figure 28 illustrates the probability density function when the system reacts by setting a series of actions to achieve a latency gain. The clusters represent the latency when a transmission or several retransmissions (RTXs), are required for the receiver to decode the packet. At the end of each cluster, the system knows whether the packet was successfully delivered or not. Figure 28 shows how decisions made at different times (e.g., parallel RTXs at actions $a_1$and $a_2$can reduce latency at the cost of resource efficiency (i.e. after action a_1, packets that only needed one RTX were allocated two RTXs).

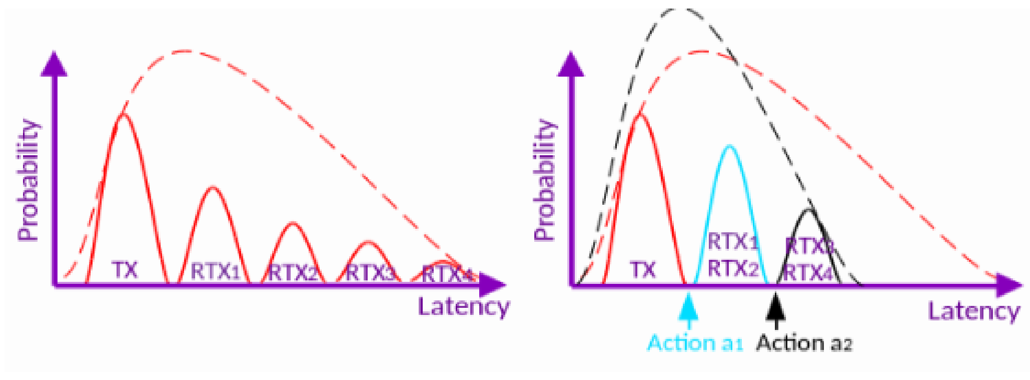


Figure 28: Early decision making scheme

In order to highlight the importance of the above-mentioned trade-off for improving URLLC communications, we investigate the early decision maker in the well-known Hybrid Automatic Repeat reQuest (HARQ) retransmission protocol. In the literature, adaptation of the HARQ strategy is usually achieved by adapting the modulation and coding scheme [PDN14], the transmission power [JBSL16] and the maximum number of retransmissions [MAA16] but rarely at the scheduler level. A K-repetition scheme \[LSK21] and a proactive scheme with early termination [LDE+21] have been proposed, allowing for a number of redundant retransmissions upon receipt of the acknowledgment by the sender. By doing so, one can opportunistically decode the packet at the receiver in a shorter time at the expense of inefficient resource usage [JAB17]. However, the adaptation of HARQ strategies at the scheduling level in a rapidly changing environment is limited in current research.

The randomness of bursty traffic and time-varying channel pose critical problems for URLLC, thus, dynamic scheduling (i.e. queue aware and channel aware) is required. In [HZS+22], they proposed a Closed-Loop ARQ protocol that dynamically re-allocates the remaining resources between uplink and downlink slots upon the result of last uplink transmission. In [WLCE19], the transmission decision of the scheduling under delay and power constraints is based on data packet arrival, occupancy of the transmission queue and time-varying channel. In [HCF20], a joint transmission - computation optimization achieves an optimal tradeoff between power and latency by taking into account the system dynamics. Hereafter, we propose a resource efficient, delay optimal, reliable scheduling adapted to dynamic scenarios (e.g., time-varying channel and traffic).

In D4.2 deliverable, we proposed a novel scheduling methodology (combining reactive and proactive resource allocation strategies) specifically devised for URLLC services. Our ultimate objective was to characterize the level of proactivity required to cope with various scenarios. Specifically, we proposed to operate at the scheduling level, addressing the trade-off between reliability, latency and resource efficiency. We offer an evaluation of the proposed methodology in the case of the well-known Hybrid Automatic Repeat reQuest (HARQ) protocol in which the proactive strategy allows a number of parallel retransmissions instead of the "send-wait-react" mode. To this end, we proposed some deviations from the HARQ procedure and benchmark the performance in terms of latency, reliability outage and resource efficiency as a function of the level of proactivity. Afterwards, we highlight the critical importance of proactive adaptation in dynamic scenarios (i.e. with changing traffic rates and channel conditions). In this section, we formulate the two dynamic resource scheduling problems by considering the traffic arrival in the network layer, the queue behaviors in the data link layer and the risk of applying vulnerable decision which causes packet loss. The first proposed solution (**proactive**) applies decisions dynamically based on channel, traffic dynamics and long-term packet loss. The second proposed solution (**dynamic**) exploits a better trade-off between latency, reliability and resource efficiency, is aware of long-term or short-term reliability requirements and dynamically adapts to the traffic behavior and channel impairments.

We consider end-to-end performance by developing a system-level simulator based on NS-3 [PLBG19] applying to the 5G New Radio (NR). This simulator handles several HARQ processes and measures the latency between the transmitter Radio Link Control (RLC) layer and the receiver RLC layer assuming that transmission buffer size is infinity. We therefore consider both the queuing latency at the scheduler (due to reactive/proactive approaches) and (re)transmission latency (i.e. PHY/MAC).

6.1.1 System model

In this section, we describe the system model making the trade-off between the latency, the reliability and the resource efficiency. A series of actions $a_j \in \{a_0, \dots, a_{max}\}$ is made at the corresponding action slot t. We can define two queues: The arrival rate queue $Q_1(t)$ is the RLC transmission buffer and contains the application packets. After completing the scheduler operations at MAC layer, the gNB prepares a Transport Block (TB) whose data is extracted from $Q_1(t)$ and sends it over the air. The scheduling rate queue $Q_2(t)$ keeps a copy of this TB and takes into account the ongoing scheduling processes that are not yet decoded at the UE side. Due to the dynamic nature of not only the traffic but also the channel behaviour, the lengths of $Q_1(t)$ and $Q_2(t)$ can be considered as random variables. The state of $Q_1(t)$ and $Q_2(t)$ demonstrated a two-stage queuing system whose length should be minimized.

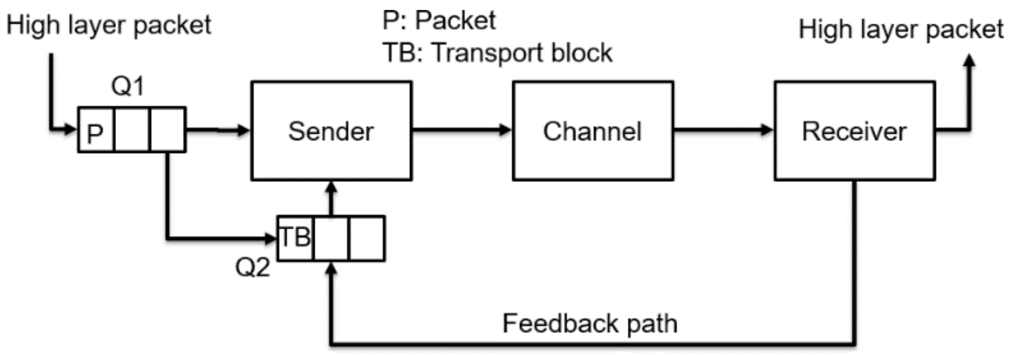


Figure 29: System model

The queuing dynamic is defined as follows:

$$Q_1(t+1) = \max\{Q_1(t) - \alpha_a \times TB_{a_0}, 0\} + A_1(t)$$

$$Q_2(t+1) = \max\left\{Q_2(t) - (1 - \alpha_a) \times 1_{TB} \times TB_{a_j}, 0\right\} + A_2(t)$$

where $Q_1(t+1)$ are the backlogs of the queue i at the action slot $t+1$. $A_1(t)$ represents the total amount of high layer packets that arrive Q_1 at time t. During this action slot, an amount of TB_{a_j} will be served. The indicator function 1_{TB}, in the second equation, is equal to 1 if the scheduling process of TB is successful and is 0, otherwise. If the first transmission of TB_{a_0} is a failure, $A_2(t) = TB_a$ will be added to Q_2, otherwise $A_2(t) = 0$ as the scheduling process of TB_{a_0} is ending. In order to control which queue will be served, we introduced the control variable α_a (1 and 0 mean serving $Q_1(t)$ and $Q_2(t)$ respectively). Knowing that ongoing processes have a higher priority, $\alpha_a = 0$ $Q_2(t) > 0$.

To calculate the TB size, the TDMA-based scheduler takes into account the number of OFDM symbols allocated to user in a single time slot ($1 \leq N_{OFDM} \leq 12$), as well as the choice of the Modulation and Coding Scheme (MCS) *m*, the numerology *Num*, the bandwidth *BW*, as follows:

$$TB = \frac{N_{OFDM} \times BW}{8 \times SCS(Num)} \times M(m) \times CR(m)$$

where $SCS(Num)$ is the sub-carrier spacing at the numerology Num, $M(m)$, $CR(m)$ are the modulation order and code rate of the selected MCS *m*, respectively. These values are given in Table 5.1.3.1 of 3GPP TS 38.214 [3GPP21].

Concerning the improvement of the diversity order of the communication, $U_{tx} = M_{BS} \times N_{BS}$ and $S_{rx} = M_{UE} \times N_{UE}$ Uniform Linear Array (ULA) antennas are installed at the gNB and UE, respectively. The transmission power at the gNb is set to P_{tx} dBm.

In order to evaluate the performance of the system in a highly changing environment, we modeled the Indoor Factory- sparse clutter scenario (InF-SL) according to the 3GPP technical report [3GPP19].

This model clarifies the channel behaviors based on antenna modeling, slow-fading factors such as shadowing, propagation loss and fast-fading factors due to multi path. Initially, the propagation loss is calculated based on whether the path is in visibility or not (LOS or NLOS) and can be derived:

$$PL_{LoS} = 31.84 + 21.50 \times \log_{10}(d) + 19.00 \times \log_{10}(f_c)$$

$$PL_{NLoS} = 33.00 + 25.50 \times \log_{10}(d) + 20.00 \times \log_{10}(f_c)$$

where d is the instantaneous 3D distance between the UE and gNb in meter and f_c is the center frequency in GHz.

According to the channel statistics, the probability of LOS communication Pr_{LoS} between the transmitter and receiver which is based on the distance d and the scenario parameter $k = -\frac{d_{clutter}}{\ln(1-r)}$ can be derived as follows:

$$Pr_{LoS} = e^{-\frac{d}{k}}$$

where $d_{clutter} = 10$ is the typical clutter size in meter and $\mathrm{r} = 0.3$ is the clutter density. In this study, we consider the UE mobility with respect to the gNb, so the shadowing effect plays an important role in the channel status and it has been modelled as in [ITU19].

To better understand the benefits of good decision making for proactive HARQ in improving Radio Access Network (RAN) latency and resource efficiency, Figure 30 shows the schemes of (A) classical reactive HARQ procedure, (B) fixed repetitions of 3 RTXs and (C) dynamic redundancy applied to the number of RTXs.

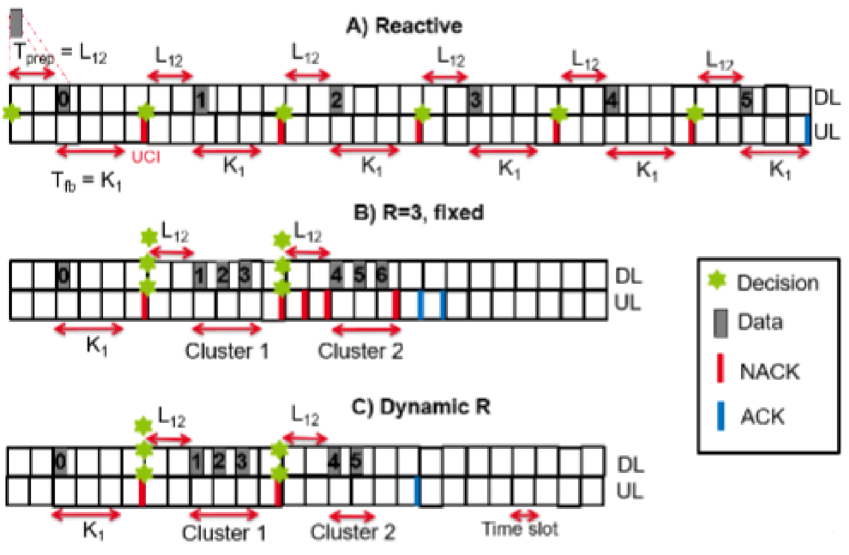


Figure 30: Different DL HARQ procedures: (A) classical, (B) fixed R=3 parallel RTXs and (C) Dynamic and proactive RTXs

In Figure 30A, a delay L_{12} is introduced to demonstrate the TB preparation time from the gNb scheduler to the antenna. Then, a feedback will be encoded together with an Uplink Control Information (UCI) message and sent back to the gNB after $T_{fb} = K_1$ slot(s), thus illustrating the processing time at the UE. In 5G NR standard, this processing time reflects a delay between the reception of the UL grant in the DL (PDCCH) and the transmission of the corresponding UL data (Physical Uplink Shared Channel (PUSCH)) [PLGB19]. Afterwards, the gNB has the information about the corrupted HARQ process on the UE side and decides to retransmit the erroneous TB after L_{12} slots. This process continues 5 times until the corrupted TB is successfully decoded by the UE. By doing this, the resource are perfectly utilized, but the latency could be unacceptable for URLLC communications.

Figure 30B shows the fixed repetition of corrupted TB with a fixed redundancy level R=3. This means that the first cluster which will occupy 3 consecutive slots, will be reserved for retransmissions. If it fails, the second cluster with the same redundancy level will be allocated. As shown in Figure 30B, latency is significantly improved at the cost of resource efficiency (RTX6 is useless when the TB has been decoded after 5 retransmissions).

By dynamically selecting the proactive redundancy level for each cluster, Figure 30C shows better performance in terms of reduced latency and improved resource efficiency. In this design, to reduce the control overhead due to multiple feedbacks to the transmitter, we grouped their feedbacks into a single feedback that represents the current proactive retransmission status.

In this work, a TB is successfully decoded by the receiver if the Signal-to-Noise-plus-Interference-Ratio (SINR) reaches the target SINR. We consider an incremental redundancy HARQ, thus each RTX contains different coded bits than the previous one. According to [LWE+20], a singular value of $SINR^{(r)}$after *r* RTXs is computed quantitatively based on a set of received Resource Blocks (RBs) ω, their $SINR_x^{(r)}$ $(\forall x \in w)$ and the previous RTX, i.e. $SINR_{tb_n}^{(r-1)}$ as follows:

$$SINR_{tb_n}^{(r)} = -\beta \times \ln(\frac{1}{|\omega|} \times \sum_{x\in\omega} e^{-\frac{SINR_x^{(r)}+SINR_{tb_n}^{(r-1)}}{\beta}})$$

β is a constant depending on the MCS as specified in [LWE+20].

6.1.2 Problem formulation

6.1.2.1 *Proactive HARQ*

In this work, we limit the number of decisions into c_{max} clusters of proactive RTXs. Each cluster $c_j \in \{c_0 \dots c_{max}\}$ can include r_{c_j} RTXs. c_0 corresponds to the initial TB transmission. The decision maker we designed will dynamically choose the size of each cluster c_j (i.e, r_{c_j}) to reduce both RAN latency and resource waste.

With respect to the resources allocated for proactive RTXs of a TB_n, the decision maker selects an element-wise positive resource allocation vector $(r_{n,c_0}, r_{n,c_1}, \dots, r_{n,c_{max}})$ that satisfies the following condition:

$$1 \leq \sum_j r_{n,c_j} \leq R_{max}$$

where R_{max} is the maximum number of RTXs for a TB_n. In the case where a TB is not decoded at receiver after R_{max} RTXs, the TB that contains many application packets is considered a loss. $r_{min} \leq r_{n,c_j} \leq r_{max}$ constrains the number of proactive RTXs at cluster c_j not to exceed a value r_{max}. We can define an objective function f_{obj} that is akin to the average resource allocation to be provided for each TB as follows:

$$f_{obj} = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{j} r_{n,c_j}$$

The decision of how to optimally select r_{n,c_j} is based on various factors, such as the current status of $Q_1(t)$, $Q_2(t)$,, the current cluster index c_j and the current aggregated SINR. According to Little's law, the average delay is related to the queue length. Therefore, the first constraint of our problem is both to minimize the average long-term queue length $\overline{Q_1(t) + Q_2(t)}$, and to make their average rate stable. By definition, a stochastic process Q(t) is stable at the average rate if:

$$\lim_{t \to \infty} \frac{\mathbb{E}\{Q(t)\}}{t} = 0$$

In the following, we present an additional constraint that is associated with the average long term risk. The visible risk in our problem is when the decision maker at cluster c_{max} chooses to serve the HARQ process in Q_2 by insufficient allocation of $r_{n,c_{max},}$ to recover the corrupted TB:

$$\zeta n = \mathbb{P}[\left(SINR_{tb_n}^{\sum_{j=0}^{max} r_{n,c_j}} \leq SINR^t \middle| SINR_{tb_n}^{\sum_{j=0}^{max-1} r_{n,c_j}} \right)]$$

where $SINR_{tb_n}^{\sum_{j=0}^{max} r_{n,c_j}}$ is the SINR of TB_n after $c_{max,}$ clusters of RTXs and $SINR^t$ is the target SINR.

So, the long-term average risk across all TBs is:

$$\bar{\zeta} = \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} \zeta n$$

The constraint is then to guarantee the expectation of the long-term risk $\bar{\zeta}$ below a predefined value ζ_0. For that purpose, we introduce a Z(t) which is incremented each time slot t by $\bar{\zeta} - \zeta_0$ each time $\bar{\zeta}$ exceeds ζ_0.

$$Z(t+1) = \max(Z(t) + \bar{\zeta} - \zeta_0 , 0)$$

Hence, the constraint of satisfying the long-term risk becomes a stability constraint on the average rate of the queue Z(t).

$$\lim_{t \to \infty} \frac{\mathbb{E}\{Z(t)\}}{t} = 0$$

To summarize, our optimization problem($\mathcal{P}_1$) is to minimize the objective function f_{obj} subject to several constraints:

$$\min f_{obj} \quad (\mathcal{P}_1)$$

$$\text{s.t. } \lim_{t\to\infty} \frac{\mathbb{E}\{Q_i(t)\}}{t} = 0, \forall i \in \{1,2\}; \qquad (\mathcal{C}_{1,2})$$

$$\lim_{t\to\infty} \frac{\mathbb{E}\{Z(t)\}}{t} = 0; \qquad (\mathcal{C}_3)$$

$$1 \le \sum_j^{c_{max}} r_{n,c_j} \le R_{max}; \, (\mathcal{C}_4)$$

$$0 \le r_{n,c_j} \le r_{max} \; (\mathcal{C}_5)$$

6.1.2.2 Dynamic HARQ

The objective is to optimally select r_{a_j} based on various factors, such as the current status of the $Q_1(t)$, $Q_2(t)$, the current action index a_j and the risk that the applied decision causes loss. The main reliability constraint is to reduce the risk of the last action $\zeta(a_{max})$ below a predefined value ζo. However, the constraint associated with poor decision making must be defined for each upcoming action, not just for the last action. We define the risk for the current action $\zeta(a_j)$. In this case, the procedure has to retrigger other actions later, which consumes not only time and resources but also the reliability of the communication, when we are close to the maximum number of actions allowed. The index of the current action (i.e a_j) is thus very important. Clearly, the higher j is, the greater the sensitivity of TB loss will be if a wrong decision is applied, and the earlier the action (i.e low j) is, the higher the total number of RTXs can be. We define an objective function f_{obj} as the weighed sum of average number of resources allocated to each TB and the current risk, as follows:

$$f_{obj} = \lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \sum_{a_0}^{a_{max}} r_{n,a_j} \times 1_{a_j} + \alpha \times f(a_j) \times \zeta(a_j)$$

where the indicator function 1_{aj} is equal to 0 if the action a_j is successful (i.e. $\zeta(a_j) < \zeta_0$) and is 1, otherwise. $\alpha \ge 0$ is a constant value trading off risk and resource allocation. A higher value of α implies greater importance of risk minimization over the number of resources allocated (i.e. reliability over resource efficiency). The function $f(a_j)$ increases with the action index a_j. In our study, we consider $f(a_j) = j$.

Thus, our optimization problem $\mathcal{P}_2$ is to minimize the objective function f_{obj} subject to several constraints:

$$\min_{\{r_{n,a_j}\}_{n,a_j}} \; f_{obj} \;\; (\mathcal{P}_2)$$

$$\text{s.t. } \lim_{t\to\infty} \frac{\mathbb{E}\{Q_i(t)\}}{t} = 0, \forall i \in \{1,2\}; \qquad (\mathcal{C}_{1,2})$$

$$r_{min} \times 1_{a_j} \le r_{n,a_j} \le r_{max} \times 1_{a_j}, \forall a_j \le a_{max} \; (\mathcal{C}_3)$$

$(\mathcal{C}_{1,2})$ concern the stability constraint of the queues $Q_{1,2}(t)$. $(\mathcal{C}_3)$ limits the number of decisions into a_{max} actions and constrains the maximal number of proactive RTXs at action a_j to r_{max}.

6.1.3 Proposed solutions

6.1.3.1 Proactive HARQ

Our proactive decision maker algorithm is based on Lyapunov's optimization tools to solve the optimization problem $(\mathcal{P}_1)$. To simplify the design of the sequential decision maker, we assume that (i) the first decision is the same as the reactive decision (i.e. only one slot is reserved for the TB transmission) and (ii) proactive RTXs in the same cluster share the same HARQ feed-back.

First, we define the current state in the slot t as $\Theta(t) = (Q_1(t), Q_2(t), Z(t))$ and the Lyapunov function as follows:

$$L(\Theta(t)) = \frac{1}{2}[Q_1^2(t) + Q_2^2(t) + Z^2(t)]$$

Next, we define the one-slot conditional Lyapunov drift $\Delta(\Theta(t))$ representing the expected change of the Lyapunov function over a slot as follows:

$$\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t)) \mid \Theta(t)\}$$

By minimizing both $\Delta(\Theta(t))$ and f_{obj}, we can solve the problem $(\mathcal{P}_1)$ because the queues are stable in terms of average rate and the objective function is minimized . However, according to [NEE14], there is a performance-delay trade-off between these dual objective optimizations that can be parameterized by a constant V. By setting a large positive value to V, the control algorithm will favor minimizing the objective function f_{obj}over the stability of the average rate queues. Our ultimate objective now is to minimize the following Lyapunov-drift-plus-penalty function:

$$g(t) = \Delta(\Theta(t)) + V.\mathbb{E}\{f_{obj} \mid \Theta(t)\}$$

Its upper bound, $\gamma(t)$,, can be derived as follows for any action, any possible value of $\Theta(t)$ and any parameter V>0 [NEE14]:

$$\gamma(t) = B + V.\mathbb{E}\{f_{obj} \mid \Theta(t)\} + \sum_{i=1}^{2} Q_i(t).\mathbb{E}\{A_i(t) - b_i(t) \mid \Theta(t)\} + \mathbb{E}\{Z(t).(\bar{\zeta} - \zeta_0) \mid \Theta(t)\}$$

where B is a constant that satisfies:

$$B \geq \frac{1}{2}\sum_{i=1}^{2} \mathbb{E}\{A_i^2(t) - b_i^2(t) \mid \Theta(t)\} + \frac{1}{2}\mathbb{E}\{(\bar{\zeta} - \zeta_0) \mid \Theta(t)\} - \sum_{i=1}^{2} \mathbb{E}\{A_i(t).\min\{Q_i(t), b_i(t)\} \mid \Theta(t)\}$$

and $b_i(t)$ is the quantity of TB_n at time slot t that the queue i can process.

$$b_i(t) = \begin{cases} TB_n^{r_{n,1}} \; if \; i = 1 \\ TB_n^{r_{n,c_j}}.1_{TB_n^{r_{n,c_j}}} \; if \; i = 2 \\ 0 \; otherwise \end{cases}$$

Under Slater's condition [NEE14], the drift-plus-penalty algorithm provides a time-averaged $\mathcal{O}\left(\frac{1}{\varepsilon}\right)$ of the expected size on all actual queues.

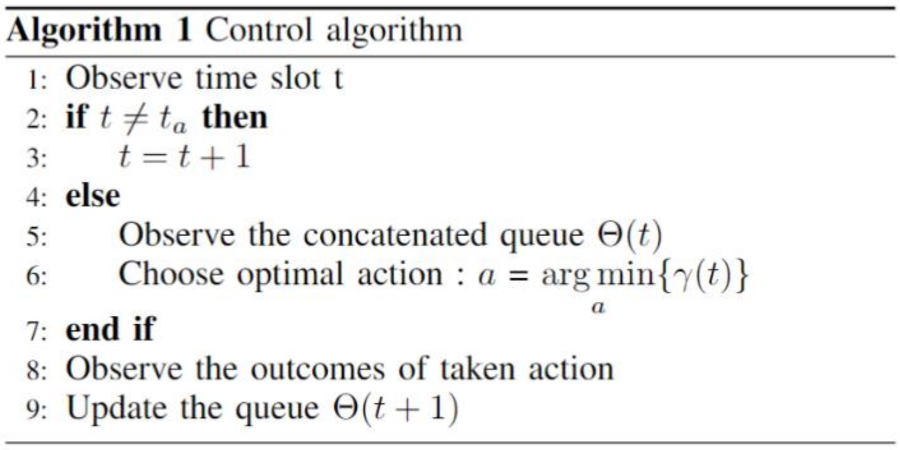
$$\lim_{t\to\infty}\sup\frac{1}{t}\sum_{\tau=0}^{t-1}\overline{Q_1(\tau)+Q_2(\tau)} \leq \frac{B+C+V[f_{obj,max}-f_{obj,min}]}{\varepsilon}$$

where C, ε>0 are constant values.

By increasing the value of V minimizes f_{obj} at a cost of higher average queue length $\overline{Q_1(\tau)+Q_2(\tau)}$ and vice versa.

Through the opportunistic minimization framework of a conditional expectation [NEE14], by minimizing $\gamma(t)$, the upper bound of the dual objective optimization, we can guarantee that the optimization problem $(\mathcal{P}_1)$. will be satisfied.

Therefore, the design of our algorithm will be based on the control action a at the decisive time t_a and will choose the control action that exhaustively minimizes function $\gamma(t)$ as follows:



```
Algorithm 1 Control algorithm
1: Observe time slot t
2: if $t \neq t_a$ then
3:     $t = t + 1$
4: else
5:     Observe the concatenated queue $\Theta(t)$
6:     Choose optimal action : $a = \arg\min_{a}\{\gamma(t)\}$
7: end if
8: Observe the outcomes of taken action
9: Update the queue $\Theta(t + 1)$
```

Figure 31: Control algorithm

Since the optimal action is chosen based on the exhaustive search minimizing the designed function $\gamma(t)$ and the action space is bounded from r_{min} to r_{max}, the computation complexity is relatively low and does not lead to an increase in the transmission processing time.

6.1.3.2 Dynamic HARQ

Our dynamic decision maker algorithm is based on Lyapunov's optimization tools to solve the optimization problem $(\mathcal{P}_2)$. We define the current state in the slot t as $\Theta(t) = (Q_1(t), Q_2(t)))$ and the Lyapunov function as follows:

$$L(\Theta(t)) = \frac{1}{2}[Q_1^2(t) + Q_2^2(t)]$$

Next, we define the one-slot conditional Lyapunov drift $\Delta(\Theta(t))$ representing the expected change of the Lyapunov function over a slot as follows:

$$\Delta(\Theta(t)) = \mathbb{E}\{L(\Theta(t+1)) - L(\Theta(t)) \mid \Theta(t)\}$$

By minimizing both $\Delta(\Theta(t))$ and f_{obj}, we can solve the problem $(\mathcal{P}_2)$ because the queues are stable in terms of average rate and the objective function is minimized . However, according to [NEE14], a performance-delay trade-off between these dual objective optimizations can be parameterized by a constant V. By setting a large positive value to V, the control algorithm will favor minimizing the objective function f_{obj} over the stability of the average rate queues. Our objective is then to minimize the following Lyapunov-drift-plus-penalty function:

$$g(t) = \Delta(\Theta(t)) + V.\mathbb{E}\{f_{obj} \mid \Theta(t)\}$$

As defined in [NEE14], the upper bound, $\gamma(t)$, can be derived for any action, any possible value of $\Theta(t)$ and any parameter $V > 0$ as follows:

$$\gamma(t) = B + V.\mathbb{E}\{f_{obj} \mid \Theta(t)\} + \sum_{i=1}^{2} Q_i(t).\mathbb{E}\{A_i(t) - b_i(t) \mid \Theta(t)\}$$

where B is a constant that satisfies:

$$B \geq \frac{1}{2}\sum_{i=1}^{2} \mathbb{E}\{A_i^2(t) - b_i^2(t) \mid \Theta(t)\} - \sum_{i=1}^{2} \mathbb{E}\{A_i(t).\min\{Q_i(t), b_i(t)\} \mid \Theta(t)\}$$

Through the opportunistic minimization framework of a conditional expectation [NEE14], by minimizing $\gamma(t)$, the upper bound of the dual objective optimization, we can guarantee that the optimization problem $(\mathcal{P}_2)$ will be satisfied.

6.1.4 Results

6.1.4.1 Proactive HARQ

In this section, we compare the performance of several HARQ strategies (i.e. our proactive adaptation algorithm, a fixed proactive and a reactive) between a gNB and single mobile user. Initially, the UE is placed at a distance d_0 from the gNb and moves away from it with constant speed and direction (v_x , v_y). In this work, packets are generated in distinct ON and OFF periods that follow the Internet Protocol (IP) traffic model. The average duration of the ON and OFF periods are t_{ON} and t_{OFF} , respectively. In the ON state, packets of variable size are generated with an arrival rate of λ_{ON} and fill $Q_1(t)$. The performance is evaluated in terms of RAN latency, reliability outage, packet loss and resource efficiency.

We define the resource efficiency as the ratio of the number of radio resources required for a TB to be received by the receiver to the number of radio resources allocated by the scheduler. We also define the RAN latency as the times between the arrival of IP packets in the RLC layer of the gNB and their arrival in the IP layer at the UE side. In the scheduling process, K_1 and L_{12} are modelled to illustrate the feedback processing time and data preparation time at the UE and gNB, respectively. For simplicity, we assumed that the core network latency and propagation delay are negligible.

Table 5 summarizes all parameters including communication band, transmit power P_{tx}, target Block Error Rate (BLER) ε_t, maximum number of proactive cluster c_{max} and proactive retransmission per cluster r_{max} and risk threshold ζ_0.

Table 5 : Simulation parameters

Parameters	Values
Packet Size	Exponential(50) Bytes
ON-OFF Traffic	$t_{on} = t_{off} = 2.5$ ms
Velocity (v_x,v_y)	(4,4) m/s
d_0	110m
(K_1, L_{12})	(2,2) time slots
R_{max}	10
(f_c, BW)	(3.5 GHz, 50 MHz)
P_{tx}	8 dBm
$Numerology$	1
(U_{tx}, S_{rx})	$(4 \times 4, 2 \times 2)$
$BLER\ \epsilon_t$	10^{-4}
$(m, \eta_{S.eff}, CR)$	(5 , 0.7402 , 0.3701)
c_{max}	5
(r_{min}, r_{max})	(2, 5)
ζ_o	0.05

Figure 32 illustrates how the objective function and resource efficiency behave as a function of the parameter V. For small value of V, our algorithm tends to minimize the expected changes in the Lyapunov function, i.e $\Theta(t)$ rather than the number of allocated resources i.e. f_{obj}. By doing so, a large number of radio resources are generously provided to each TB for proactive RTXs, so the average resource allocation is high and the resource efficiency is low. When the value of V increases, the focus is on minimizing the objective function and fewer resources are allocated. At a certain value of V (i.e. around 50), we found that a minimum value of the objective function is reached (i.e. around 3.2). This is because at a high value of V, the algorithm favors allocating smaller number of parallel RTXs on each cluster and thus more proactive RTX clusters are required for each TB. Therefore, in our future evaluations, we will set V to 60.

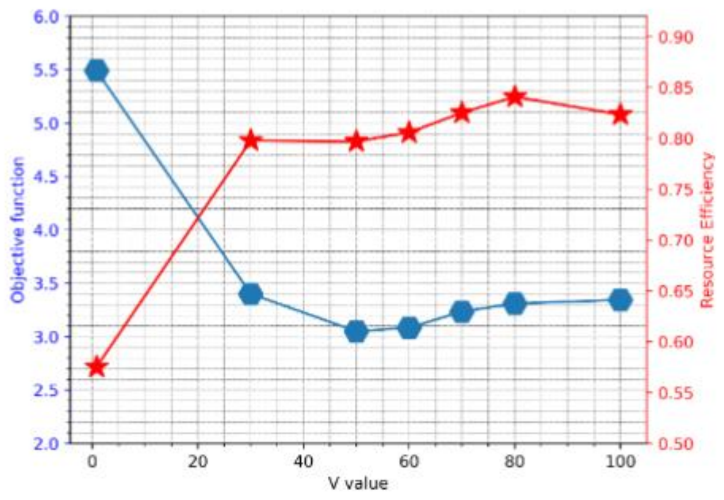


Figure 32: Objective function and resource efficiency as a function of V

Figure 33 compares the Complementary Distribution Function (CDF) of latency for different HARQ schemes: (i) reactive HARQ scheme (i.e. R=1) with a maximum number of 10 RTXs, (ii) 5 RTX clusters where each cluster contains 2 proactive RTXs (2-2-2-2-2 clusters), (iii) 4 RTX clusters where the first three clusters contain 3 proactive RTXs and the last cluster contains a single RTX (3-3-3-1 clusters) and (iv) our proposition with a maximum of 5 clusters. Figure 33 shows that the reactive HARQ scheme performs the worst compared to the other solutions due to the high RTT cost associated with triggering many RTXs. By enabling two parallel RTXs per received NACK, the latency can be improved at the cost of decreasing resource efficiency to around 0.82 compared to 1 in the reactive case. The latency can be further improved with 3 proactive RTXs per cluster, but the resource efficiency drops significantly to 0.69 when we redundantly provide radio resources for a TB to be successfully decoded.

Concerning the performance of our algorithm with the adaptation of proactive HARQ, the performance of latency is slightly better than the others cases. As the channel condition and traffic behavior varies dynamically, our dynamic decision maker will wisely select different redundancy level according to the current queues status and instantaneous channel condition. As the result, we can enhance the latency while keeping a good level of resource efficiency at around 0.8.

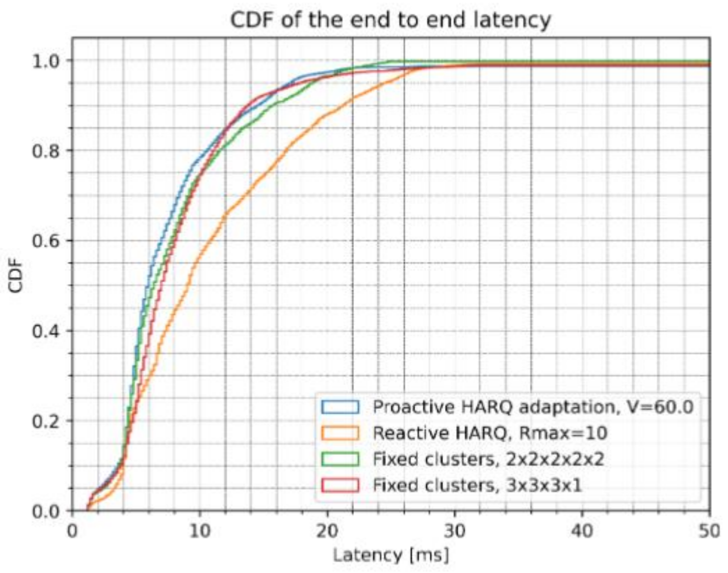


Figure 33: CDF of latency for reactive RTX, fixed 2-parallel RTX, fixed 3-parallel RTXs and our dynamic adaptation algorithm with V=60

To show the temporal variation in MAC layer delay of the different HARQ schemes for intermittent URLLC traffic scenarios under time-varying channel conditions, Figure 34 compares the latency between the departure of a TB at the sender's MAC layer and its successful arrival at the receiver's MAC layer. The reactive HARQ scheme experiences more peaks as it needs more time to successfully decode a packet when critical errors occur.

When fixed proactive HARQ schemes are applied, 2-parallel strategy reduces the peak delay and 3-parallel strategy further improves the delay when the radio resources allocated in parallel help decode the packet faster. Our proposed algorithm also reduces the number and amplitudes of peaks when it dynamically selects an optimal level of proactive redundancy for each cluster, thus achieving a better trade-off between reserving more radio resources to recover the corrupted packet faster and less to maintain a minimum objective function.

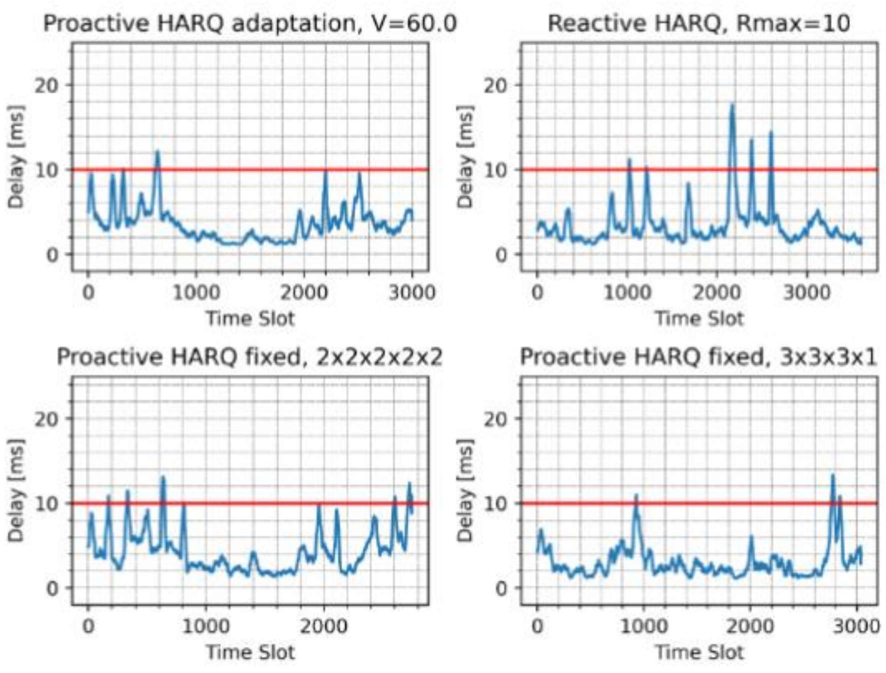


Figure 34: Evolution of MAC delay between HARQ schemes

Figure 35 shows the latency as a function of V value for an outage level of 0.9 and 0.95, respectively. Certain values of V make the outage latency minimal because they allow a better balance between the number of resources allocated for each packet and the stability of the queues. As the result, the effect of overloaded buffer is better managed and leads to a good latency outage. On the other hand, by setting V too high or too low, this balance is no longer efficiently controlled and it causes a queuing effect due to the fact that too many clusters are used (large V) or too many retransmissions in each cluster are set (small V).

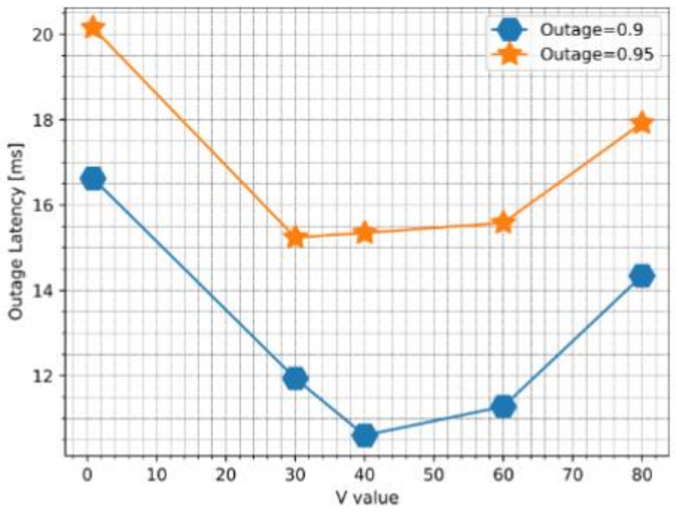


Figure 35: Outage latency as the function of V value

In addition, various performance metrics such as application packet loss (APP loss), resource efficiency as well as average delay and its standard deviation for different HARQ schemes are discussed. First, the APP loss is guaranteed to be between 0.8 % and 1 % because the maximum number of retransmissions (i.e. $R_{max} = 10$) is applied in all cases. Second, we find the largest average delay and its standard deviation in reactive HARQ whose values are around

10.77 ms and 6.57 ms, respectively. These coupled values are enhanced to roughly 8.14 ms, 4.91 ms and 8.1 ms, 4.5 ms in fixed 2-parallel and 3 parallel HARQ schemes at the expense of resource efficiency which decreases to 0.82 and 0.69, respectively. By dynamically deciding the number of proactive retransmissions at V=60, the average delay and jitter are slightly better than other HARQs which are about 7.2 ms and 4 ms while maintaining a good resource utilization level at about 0.8.

6.1.4.2 Dynamic HARQ

Performance is evaluated in terms of RAN latency, packet loss and resource efficiency. We define resource efficiency as the ratio of the number of radio resources required for a TB to be successfully received to the number of radio resources allocated by the scheduler. We also define RAN latency as the time between the arrival of IP packets in the RLC layer of the gNB and their arrival in the IP layer at the UE side.

Figure 36 shows the evolution of resource efficiency (solid line) as well as the average total number of radio resources allocated (dash line) as the function of V and α. We selected three values of α (i.e. 0, 2 and 15) that depend on the awareness of reliable transmission's objective. When reliability is not considered (α=0), our algorithm tends to spend less radio resources for each action and thus, resources are used efficiently. When V increases, we put more emphasis on minimizing resource allocation, so resource efficiency is further improved. When α appears and grows, the goal of reducing packet loss is also taken into account. The decision maker adapts to the channel conditions and allows more generous allocations for each action and this leads to high resource allocation with high standard deviation and low resource efficiency.

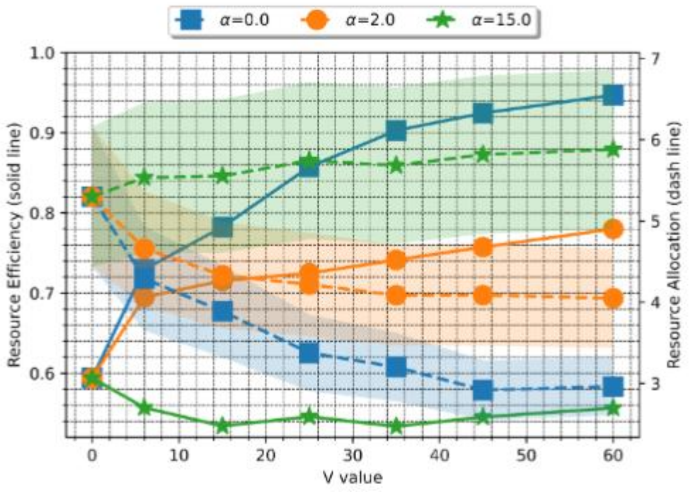


Figure 36: Resource allocation and efficiency as a function of optimization parameters

Reliability of communication is guaranteed at the cost of low resource utilization as shown in Figure 37. When α is high, the transmission error is significantly low and the communication reliability no longer depends on the V-value (i.e., 99.5% and 98.5% of the total packets successfully reach the IP layer at the UE side for $\alpha = 15$ and $= 2$, respectively). However, the dependent relationship between the transmission reliability and the V-value is observed for $\alpha = 0$. In this case, we barely follow the minimization of the number of resources allocated for each action rather than the reliability of its transmission, thus, we noticed more error-prone transmissions when V increases.

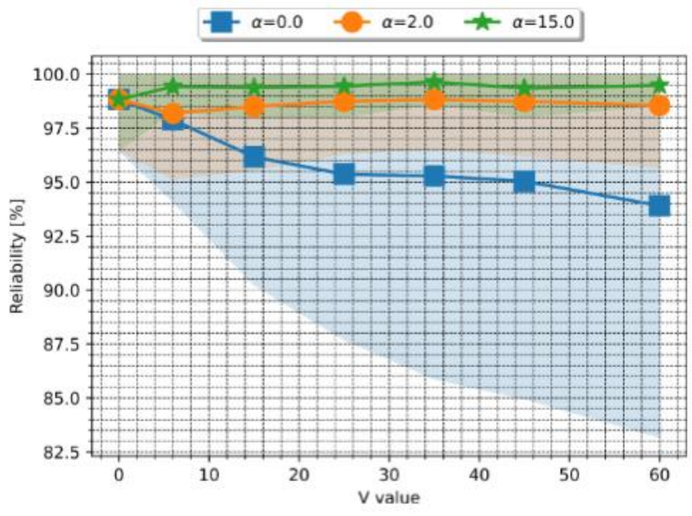


Figure 37: Reliability as a function of optimization parameters

Figure 38 shows that the average latency at the RAN mainly depends on α. Redundant radio resources are scheduled when α is high to improve reliability, but this can result in increased queuing delay as incoming packets must wait longer in the queue before being served.

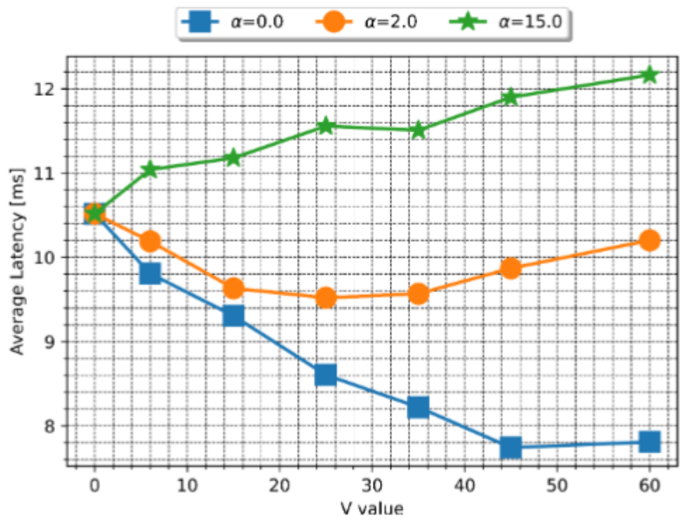


Figure 38: Average Latency as a function of optimization parameters

Figure 39 compares the Complementary Distribution Function (CDF) of latency for different HARQ schemes: (i) Classic HARQ procedure, (ii) Fixed number of parallel RTXs, (iii) Proactive HARQ adaptation with a fixed maximum number of RTXs ($R_{max} = 10$) and (iv) our proposed optimization (Dynamic HARQ) in which $R_{max} = \sum_{a_0}^{a_{max}} r_{n,a_j}$. According to Figure 36, Figure 37 and Figure 38, we select two pairs of (V, α) parameters: (25, 2) for good reliability and considerably low latency and (60, 0) for very good resource efficiency and latency.

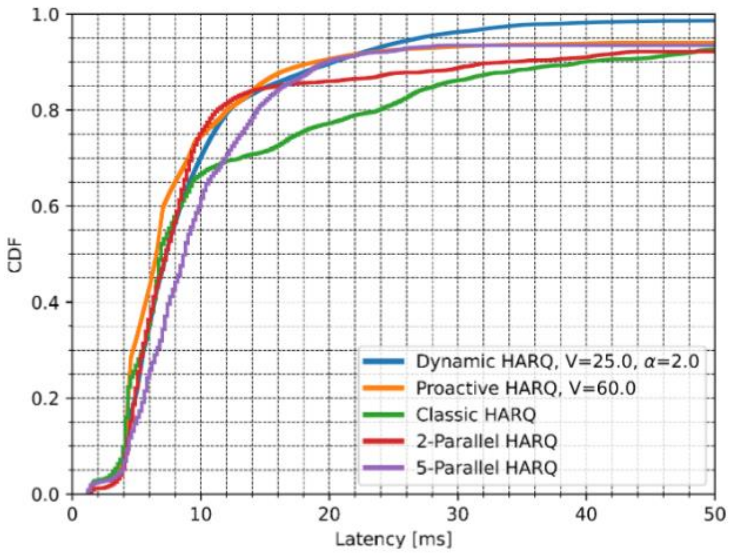


Figure 39: CDF of latency for reactive RTX, fixed 2-parallel RTX, fixed 5-parallel RTXs, our proactive adaptation algorithm with V=60 and our Dynamic HARQ with V=25 and $\alpha = 2$

As expected, the latency of Classic HARQ is the highest and spreads out over time. 2-parallel and 5-parallel HARQ improve latency at the cost of decreasing resource efficiency to 0.8 and 0.6, respectively due to the lack of adaptation when needed. Dynamic HARQ offers two tradeoffs. When V=60 and α =0, we improve resource efficiency and latency but not reliability. When V=25 and α =2, we improve reliability at the cost of a slight degradation in latency in the best case.

6.1.5 Conclusions

In this study, we demonstrated the application of dynamic decision maker framework that enables a novel proactive HARQ design to cope with a time-varying channel and intermittent traffic source rate. Based on Lyapunov stochastic optimization tool, a mathematical framework is proposed to understand the performance-delay trade-off by minimizing the objective function of the total resource allocation and the total queue length that is parameterized by a V value. Our results revealed that an appropriate selection of V enables the dynamic selection of proactive retransmission to overcome the defect of a long RTT in the reactive scheme while maintaining a good level of resource efficiency that is considered a drawback of the fixed proactive schemes. Then we have extended this work to dynamic resource scheduling. We proposed a reliable, resource and delay-optimized scheduling suitable for dynamic scenarios (e.g., random bursty traffic, time-varying channel) based on Lyapunov optimization. It takes into account the traffic arrival at the network layer, the queue behaviors at the data link layer and the risk that the applied decision might trigger packet loss. The trade-off between the resource efficiency, latency and reliability is achieved by the timing and intensity of decisions and can be parameterized with V and α .

6.2 Dynamic service placement

The goal of this activity is to extend the dynamic optimization of computation and communication resources in the mobile edge cloud, already reported in D4.1 and D4.2, in two main directions: i) implementation of novel algorithms for dynamic service placement in the edge cloud, incorporating the assignment of mobile devices to radio access points and edge servers; ii) implementation of novel *adaptive* and *distributed* federated learning (FL) mechanisms suitable for the edge cloud.

6.2.1 Dynamic service placement in the edge cloud

This activity builds on the activity carried out in WP3 on service placement in the static case and it extends it to a dynamic scenario, where the channels between the mobile monitoring nodes and the edge server may change randomly over time and the packet generation process is also dynamic and random, with unknown statistics. The resources to be optimized include the association of each peripheral device to a pair of radio access point and edge server, plus the allocation of storage and CPU clock rates at the edge server, transmit power at the mobile nodes, and routing strategies. Our objective is to propose a dynamic algorithm enabling energy-efficient mobile edge computing with end-to-end delay guarantees, optimizing the placement and routing of network services jointly with radio and computational resources. We handle the randomness of the channel and packet arrival rates in a very efficient way by merging Lyapunov stochastic optimization and approximating algorithm for binary problems. This allows us to jointly optimize the communication parameters (i.e., transmit power of the mobile nodes), storage and computational resources (i.e., server CPU clock rates assigned by the edge server to each task), while solving, concurrently, the virtual service placement and the request routing problem.

Dynamic resource problems have already been considered in the technical literature. The novelty of our approach is that we handle jointly the assignment problem, which is inherently an integer programming problem, and the resource allocation problem, which deals with real variables. The proposed algorithm works in a per-slot fashion that optimizes the selection of the current variables, slot-by-slot, by taking into account all average constraints and values of the objective function, achieved up to the current slot. The complexity of the overall problem is efficiently handled by proposing a two-time-scale optimization algorithm that works as follows: (fine scale tuning) within each time slot, we allocate the radio and computational resources, for a given assignment of mobile devices to radio access points and edge servers; (coarse scale tuning) every *m* time slots, we update the assignment. The overall strategy builds on a Lyapunov stochastic optimization algorithm to allocate radio and computational resources to properly satisfy all requests. Since the statistics of channels and packet arrival rates are unknown, the optimization performed in each slot is not necessarily optimal in the average sense. Nevertheless, thanks to Lyapunov stochastic optimization, the optimization problem is converted into the stability problem of a set of queues representing the current values of the performance metrics and of the constraints (e.g., power consumption or average service delay). Even though the statistics of the channel and packet arrival rates are not known, enforcing the stability of the queues allows us to guarantee the (asymptotic) achievement of the optimal values, in the average sense.

The scenario is composed of a set of N access points (APs) with co-located edge servers (EDs) having storage and computation capabilities. We assume that each edge server is able to run virtual machines (VMs) or containers, implementing either network functionalities or services. The N wireless access points cover overlapping local areas and serve a set of K mobile devices, each one requesting a single delay-sensitive service. The timeline is discretized into time slots $t \epsilon \mathcal{T} = \{1,2, \dots, T\}$ of duration τ, during which the wireless channel is supposed to be static (i.e., τ is supposed to be smaller than the channel coherence time).

To enable a network service $s \in S$, at each time slot t, a device k uploads an amount of data $a_s^k(t)$ to the associated server n, which provides an amount J_s^k of workload (intended as the number of CPU cycles per information unit). The mobile devices can get wireless access to a subset $\mathcal{N}_k$ of AP/ESs that fall within a distance $d_{kn} < d_{th}$. Otherwise, the users' requests can be routed to a central core cloud $\mathcal{C}_L$, located at a greater distance from the end users, which stores the entire set of services and is equipped with very large computation capabilities. Clearly, to be routed to the core cloud, the devices must first access the wireless network through one of the near APs.
To formulate the proposed dynamic placement optimization strategy, we start introducing the variables and constraints involved in our problem.

Service Placement and Request Routing

To handle placement and routing, we introduce the two sets of binary optimization variables:

$$\begin{aligned} \mathbf{x}(t) &= \left(x_{sn}(t) \in \{0,1\} : s \in \mathcal{S}, n \in \mathcal{N}\right) \\ \mathbf{y}(t) &= \left(y_{kn}(t) \in \{0,1\} : k \in \mathcal{K}, n \in \mathcal{N}_C\right) \end{aligned} \qquad (a)$$

where $x_{sn}(t)$ indicates if node n stores the service s (and the associated data m_s) at time t, $y_{kn}(t)$ denotes the assignment of user k to server n at time t, and $\mathcal{N}_C = \mathcal{N} \cup \mathcal{C}_L$ represents the set of all possible servers (i.e., ESs plus core cloud).
At a given slot t, a device performs a single service request to either the central cloud or to one of the nearby ESs (belonging $\mathcal{N}_k$), which must already store the specific application. Mathematically, this translates into imposing the following constraints on variables $\mathbf{x}(t)$ and $\mathbf{y}(t)$:

$$\begin{aligned} &\sum_{n \in \mathcal{N}_k \cup \{\mathcal{C}_L\}} y_{kn}(t) = 1, && \forall t,k, \\ &y_{kn}(t) = 0, && \forall t,k,n \notin \mathcal{N}_k \cup \{\mathcal{C}_L\}, \qquad (b) \\ &y_{kn}(t) \le x_{sn}(t), && \forall t,k,s,n \in \mathcal{N}. \end{aligned}$$

We also take into account the limited storage and computation capabilities of the ESs, denoted by M_n and F_n, respectively. Thus, the following capacity constraints hold:

$$\sum_{s=1}^{S} x_{sn}(t)\, m_s \le M_n \quad \forall t, n \in \mathcal{N}, \quad (c)$$

$$\sum_{k=1}^{K} y_{kn}(t)\, f_{kn}(t) \le F_n \quad \forall t, n \in \mathcal{N}, \quad (d)$$

where $f_{kn}(t)$ is the amount of CPU clock rate (cycles per second) assigned by ES n at time t to process data and enable the application requested by device k.

Energy Consumption

Let us now define the main sources of energy consumption that we aim to minimize. The first one is associated with the transmission of data between user k and server n. We assume that each server is aware of the channel state information and allocates nonoverlapping portions of bandwidth to each connected device. Thus, letting $R_{kn}(t)$ be the available transmission rate, the energy spent by a device k to offload data to a server n is

$$\mathcal{E}^{T}_{kn}(t) = \frac{\tau c_1}{h_{kn}(t)}\left(e^{R_{kn}(t)c_2} - 1\right)$$

where $h_{kn}(t)$ is the channel gain of the transmission link, $c_1 = N_0 B_{kn}$, $c_2 = \ln 2 / B_{kn}$, with B_{kn} denoting the assigned bandwidth, and N_0 being the noise power spectral density.
On the computing side, the energy spent by the ES CPU to process the offloaded data is evaluated considering a cubic relation with the CPU clock speed [YUAN06] $\mathcal{E}^{\mathrm{C}}_{kn}(t) = \kappa_c \left(f_{kn}(t)\right)^3$, where κ_c is the effective switched capacitance of the processor. The overall system energy consumption, in each time slot, is then the sum of the energy spent for communication and computation tasks and it can be written as:

$$\mathcal{E}^{sys}(t) = \sum_{n\in\mathcal{N}_C}\sum_{k=1}^{K} y_{kn}(t)[\mathcal{E}^{T}_{kn}(t) + \mathcal{E}^{\mathrm{C}}_{kn}(t)].$$

End-to-End Average Delay

The dynamicity of the system is modeled using a set of processing queues, which help to quantify the end-to-end (E2E) latency experienced by the data from their generation at the device side up to the end of processing at the server side. We assume that the computation is always offloaded to the ESs, i.e. there is no local processing at the device side. The uplink transmission queue is a device's local buffer that takes as input a set of data of length $a_s^k(t)$, necessary to enable a desired service, and evolves according to:

$$Q_{kn}^{\mathrm{T}}(t+1) = \max\left(0, Q_{kn}^{\mathrm{T}}(t) - b_{kn}^{\mathrm{T}}(t)\right) + a_s^k(t)$$

where $b_{kn}^{\mathrm{T}} = \tau R_{kn}(t)$ represents the output data at time t, i.e., how the local queue is drained. Then, the servers collect an amount of data $\tilde{b}^{\mathrm{T}}_{kn}(t) = \min\left(Q_{k,n}^{\mathrm{T}}(t), b_{kn}^{\mathrm{T}}(t)\right)$ uploaded by the mobile devices in a computation queue whose length evolves as:

$$Q_{kn}^{\mathrm{C}}(t+1) = \max\left(0, Q_{kn}^{\mathrm{C}}(t) - b_{kn}^{\mathrm{C}}(t)\right) + \tilde{b}_{kn}^{\mathrm{T}}(t),$$

where $b_{kn}^{\mathrm{C}}(t) = f_{kn}(t)\tau / J_s^k$ is the amount of data draining the computation queues. Finally, letting $Q_{kn}^{\mathrm{tot}}(t) = Q_{kn}^{\mathrm{T}}(t) + Q_{kn}^{\mathrm{C}}(t)$, we impose an average delay constraint by upper-bounding the total average queue backlog as:

$$\mathrm{E}\left\{Q_{kn}^{\mathrm{tot}}(t)\right\} \le Q_{kn}^{\mathrm{avg}} = D_k^{avg}\,\mathrm{E}\left\{a_s^k(t)/\tau\right\},$$

where D_k^{avg} depends on the specific application, and $\mathrm{E}\left\{a_s^k(t)/\tau\right\} = \lambda_k$ are the average arrival rates. The expectation should be taken with respect to the random wireless channel and data arrival statistics. However, in our formulation these statistics are not not supposed to be known a priori and we handled this issue resorting to stochastic optimization, as explained next.

6.2.1.1 Problem Formulation

The problem is formulated as the minimization of the long-term system energy consumption, under long-term average delay constraints, with respect to placement, assignment, communication, and computation variables. Mathematically, the problem we wish to solve can be formulated as follows:

$$
\begin{aligned}
(\mathrm{P}_0) \qquad & \min_{\boldsymbol{\Phi}(t)} \ \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \mathrm{E}\left\{\mathrm{E}^{\mathrm{sys}}(t)\right\} \\
& s.t. \qquad (a),(b),(c),(d) \\
& (e)\ \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\mathrm{E}\left\{Q_{kn}^{\mathrm{tot}}(t)\right\} \le Q_{kn}^{\mathrm{avg}},\ \forall k,n\in \mathrm{N}_k; \\
& (f)\ 0 \le R_{kn}(t) \le y_{kn}(t) R_{kn}^{\max}(t), \quad \forall t,k,n\in \mathrm{N}_k; \\
& (g)\ f_{kn}(t) \ge 0,\ \forall t,k,n\in \mathrm{N}_k
\end{aligned}
$$

with $\Phi(t)=[\mathbf{x}(t),\mathbf{y}(t),\{R_{kn}(t)\}_{k,n},\{f_{kn}(t)\}_{k,n}]$ and $R_{kn}^{\max}(t)$ $R_{kn}^{\max}(t)$ denoting the maximum achievable transmission rate during slot t, for any established wireless link. The constraints in (P_0) have the following meaning: (a),(b),(c) and (d) impose the structural constraints on the placement and routing variables; (e) constrains the long-term average of the total queues to be bounded by Q_{kn}^{avg} (i.e., bounded average delay), for all $k,n\in \mathrm{N}_k$; (f) sets the constraints on the transmission rate between user k and server n; finally, (g) imposes that all CPU clock rates must be non-negative. Problem (P_0) is very complicated to solve, since it involves expectations over unknown distributions of random variables such as wireless channels and data arrivals. Nevertheless, in the sequel we provide an elegant dynamic solution based on Lyapunov stochastic optimization [NEE10].

Dynamic Solution via Stochastic Optimization

The first challenge of problem (P_0) is to handle the long-term constraints (e) without assuming prior knowledge on the statistics of radio channels and data arrivals. To tackle this issue, we resort to stochastic optimization [NEE10], converting the long-term constraints (e) into the problem of stability of properly defined *virtual* queues. In particular, to capture the state of the system in terms of constraint violations, we introduce the virtual queue $Z_{kn}(t+1)$ for each device k associated to server n, which evolve as [NEE10] :

$$Z_{kn}(t+1)=\max\left(0, Z_{kn}(t)+Q_{kn}^{\mathrm{tot}}(t+1)-Q_{kn}^{\mathrm{avg}}\right),$$

for all $k,n\in \mathrm{N}_k$. Following stochastic optimization arguments [NEE10] and defining the cumulative queue parameters $\widetilde{Q}_{kn}^{T}(t)=2Q_{kn}^{T}(t)+Z_{kn}(t)-2Q_{kn}^{C}(t)$ and $\widetilde{Q}_{kn}^{C}(t)=2Q_{kn}^{C}(t)+Z_{kn}(t)$, we obtain the following deterministic per-slot problem at time t:

$$(\mathcal{P}_1) \quad \min_{\Phi(t)} \; V\mathcal{E}^{\text{sys}} - \sum_{n\in\mathcal{N}_C}\sum_{k=1}^{K} y_{kn}\left(b_{kn}^{\text{T}}\widetilde{Q}_{kn}^{\text{T}} + b_{kn}^{\text{C}}\widetilde{Q}_{kn}^{\text{C}}\right)$$

$$s.t. \quad (a),(b),(c),(d)$$

$$f)\; 0 \le b_{kn}^{\text{T}} \le \min\left(b_{kn}^{\text{T,max}}, Q_{kn}^{\text{T}}\right) \quad \forall t,k,n\in\mathcal{N}_k$$
$$g)\; 0 \le b_{kn}^{\text{C}} \le \min\left(b_{kn}^{\text{C,max}}, Q_{kn}^{\text{C}}\right) \quad \forall t,k,n\in\mathcal{N}_k$$

where we recall that $b_{kn}^{\text{T}} = \tau R_{kn}$ and $b_{kn}^{\text{C}} = f_{kn}\tau / J_s^k$; also $b_{kn}^{\text{T,max}}$ and $b_{kn}^{\text{C,max}}$ are the information units computed with maximum achievable radio and computation rates; V is a positive parameter weighting the objective function of (P_0), i.e. the energy consumption, with respect to the long-term constraint violation. Problem (P_1) depends, jointly, on placement, routing, communication and computation variables. However, as we will show in the sequel, for a fixed vale of the assignment variables, we are able to derive the communication and computation variables in closed form. Thus, we propose a two time-scale solution procedure that updates placement and routing parameters at a coarser time-scale, whereas communication and computation resources are then optimized at a faster time-scale, under the given placement and routing variables. The details of the proposed procedure are illustrated in the sequel.

Placement and Routing Optimization

For given values of transmission and computation resources and all queues, (P_1) boils down to the following JSPRR problem:

$$(\mathcal{P}_2) \quad \min_{\Phi_1} V\,\mathcal{P}^{\text{sys}} - \sum_{n\in\mathcal{N}_C}\sum_{k=1}^{K} y_{kn}\left(b_{kn}^{\text{T}}\,\widetilde{Q}_{kn}^{\text{T}} + b_{kn}^{\text{C}}\,\widetilde{Q}_{kn}^{\text{C}}\right)$$
$$s.t. \quad (a),(b),(c),(d)$$

where $\mathbf{\Phi}_1 = [\mathbf{x},\mathbf{y}]$. From the constraints in (a), this problem is a linear integer programming problem with combinatorial complexity. To solve it, albeit approximately, but in a numerically efficient manner, we switch to a relaxed version of (P_2), where the integer constraints in (a) are converted into real variables $x_{sn}\in[0,1]$ and $y_{kn}\in[0,1]$, i.e., we introduce box constraints over real variables. Then, we apply a linear programming (LP) tool, and a final probabilistic rounding based on the approximation algorithm proposed in [POU20]. Finally, since this solution might be unfeasible, we apply a last refinement on the found optimal integer variables, to guarantee feasibility while having minimum impact on the objective function.

Communication and Computation Resources Optimization

Once the placement and routing variables are optimized, (P_1) boils down into a convex optimization problem, separable between radio and computation resource allocation within the edge network, thus leading to low-complexity solutions. Specifically, from (P_1), solving the communication sub-problem, we get the following optimal closed form solution

$$R_{kn}^* = \max\left(0, \min\left(\tilde{R}_{kn}, \frac{1}{c_2}\ln\left(\frac{\tilde{Q}_{kn}^T h_{kn}}{V c_1 c_2}\right)\right)\right)$$

where $\tilde{R}_{kn} = \min\left(R_{kn}^{\max}, Q_{kn}^{\mathrm{T}} / \tau\right)$, while, solving the computation sub-problem, the optimal CPU frequencies are given by

$$f_{kn}^* = \max\left(0, \min\left(\tilde{f}_{kn}, \sqrt{\frac{J_s^k \tilde{Q}_{kn}^{\mathrm{C}}}{3V\kappa_c}}\right)\right)$$

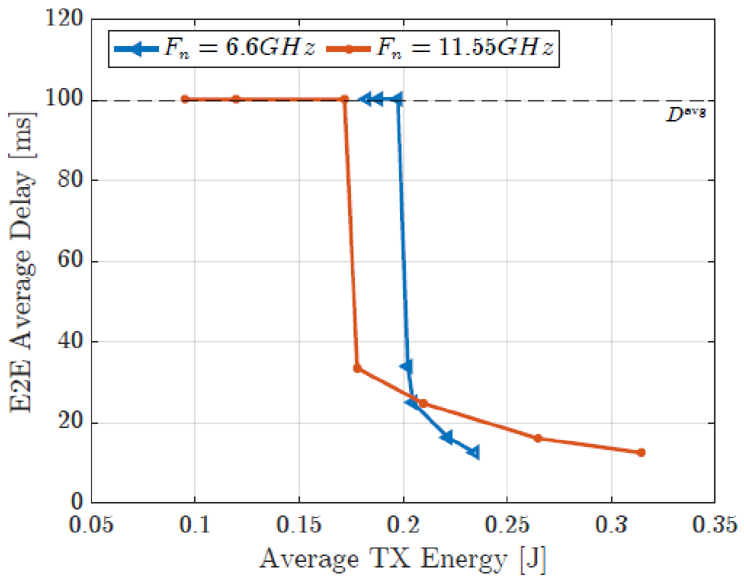
where $\tilde{f}_{kn} = \min\left(F_n, Q_{kn}^{\mathrm{C}} / \tau J_s^k\right)$. If we have $\sum_{k=1}^{K} f_{kn}^* \geq F_N$, the optimal solution can be found through a water-filling algorithm, searching for Langrangian multiplier values μ satisfying constraints $f_{kn}^* = [\tau \tilde{Q}_{kn}^{\mathrm{C}} / J_s^k - \mu]^+$ and the computation budget allocation.

6.2.1.2 *Numerical Results*

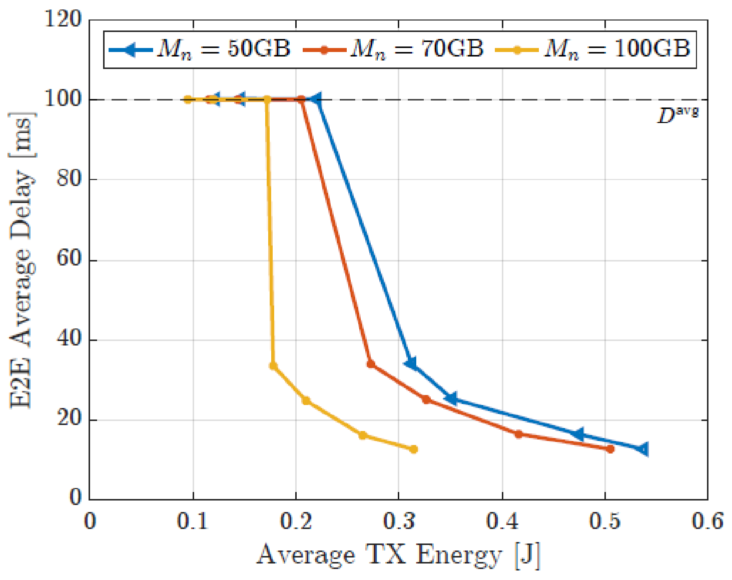
In this section, we evaluate the performance of our proposed algorithm. In our simulations, we consider $N = 8$ edge servers regularly deployed on a grid network of dimensions 500m $\times$ 400m, and $K = 320$ user devices uniformly distributed over the edge nodes coverage regions (each of 150m). Each device transmits over an equal, constant, bandwidth $B_{kn} = 0.5$ MHz, with 100 mW maximum transmission power, at a carrier frequency of 28 GHz. The devices can request a network service drawn from a library of $S = 320$ different applications, with storage requirements $m_s \in [0.5, 2]$ GB; the data are generated from Poisson distribution with arrival rates $a_s \in [0.1, 1]$ KB, in each time slot of duration $\tau = 5$ ms. The coarse time scale used to update the assignment is equal to 500 t. The number of CPU cycles needed to process a data unit is expressed as $1/J_s$ with $J_s \in [0.01, 0.05]$.

In Figure 40 (a), we can see the optimal tradeoff between average energy consumption and average delay, using two different clock rates of the edge severs' CPU. In this example, the constraint on the average service delay is *100* ms. Each point on the tradeoff curve is obtained by tuning the single optimization variable V. As expected, increasing the average energy consumption, the service delay decreases. When the overall energy consumption is very low, the average service delay equals the constraint. However, above a certain energy level, the system can achieve lower delays.

In parallel, in Figure 40 (b) we analyze the effect of providing more storage capability to each edge server, to make it able to store more virtual machines. More specifically, the figure shows again the trade-off between average energy consumption and average service delay, but for three different values of storage capacity of each server. From the figure, we observe that, increasing the storage capacity of each server, the proposed algorithm is able to achieve a better energy-delay tradeoff, thanks to its capability to optimize the association between mobile devices and edge servers and to provide a better resource allocation within the edge cloud.



(a)



(b)

Figure 40: Performance evaluation of JSPRR and computation offloading optimization algorithm

6.2.2 Energy-efficient adaptive federated learning

Federated learning (FL) has recently received significant attention in the technical literature as a distributed learning strategy that does not require the peripheral sensors to send the raw data to the fusion center (edge server), but only learning parameter updates. Differently from previous works appeared in the literature that mainly focused on a static learning task, where the learning process runs up to convergence and then it stops, we propose an adaptive FL strategy performing continuous learning, adaptation, and tracking of the parameters of interest. The scenario of interest is depicted in Figure 41, where a set of peripheral devices send only gradient updates to the edge server. FL is inherently amenable for privacy-preserving applications, because it does not entail the direct transmission of raw data, but only gradients of the local objective function. Furthermore, FL is useful because it does not force every peripheral device to send all the local data to the edge server.

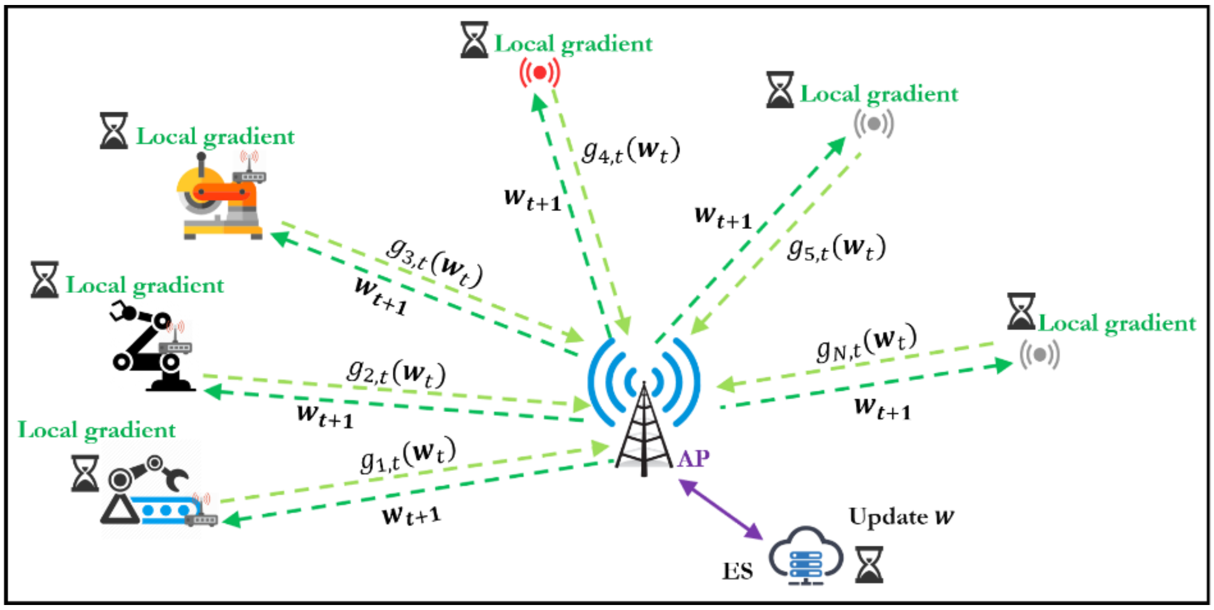


Figure 41: Scenario for federated learning

Hinging on Lyapunov stochastic optimization, SAP developed a dynamic resource allocation strategy for FL that works at the same time-scale of the gradient-based algorithm, while optimizing on the fly radio parameters (e.g., the selection of the set of transmitting devices, their transmit powers and data rates) and computational resources (e.g., CPU clock rates of the peripheral devices and of the edge server) in order to strike the best trade-off between overall energy consumption, service delay (incorporating communication and computation times), and performance (e.g., convergence rate, accuracy or mean-squared error) of the adaptive FL task. The joint optimization of communication, computation, and learning, make the proposed approach markedly different with respect to the previous approaches hinging on Lyapunov optimization [Sun20], [Zhou20], [Huang21]. In our work, particular emphasis is devoted to the definition and the online control of proper performance metrics in both a model-based scenario, where the metrics are known in closed-form, and in a data-driven case, where the performance is inferred online from data. In both cases, the method is able to adaptively minimize the average power needed for the FL task, while guaranteeing average service delays and learning performance. The proposed strategy is customized to adaptive federated Least Mean Squares (LMS) estimation and deep convolutional neural network training. To handle the convergence of communication and learning in a single framework, it is fundamental to merge these aspects with a formal mathematical analysis of the communication and computation power consumptions, delays, and learning performance metrics, expressed in terms of accuracy of the learning task, rate of convergence and adaptation. This mathematical analysis, also exploited in the proposed online solution, represents one of the main distinctive features of our work with respect to the state of the art. The technical details are available in [Bat22]. In Figure 42, we report an example of performance showing how the the proposed method is able to guarantee, asymptotically, the desired average accuracy in a learning task. In particular, in Figure 42, we show the mean square distortion achieved in a prediction problem as a function of time (expressed in terms of time slot index), obtained with three different specifications of the learning accuracy. From the figure, we can see that to achieve a better accuracy it is necessary, as expected, to wait longer. Each curve refers to a specified value of the average accuracy and we can check that in every setting, the method is able to converge to the desired performance value.

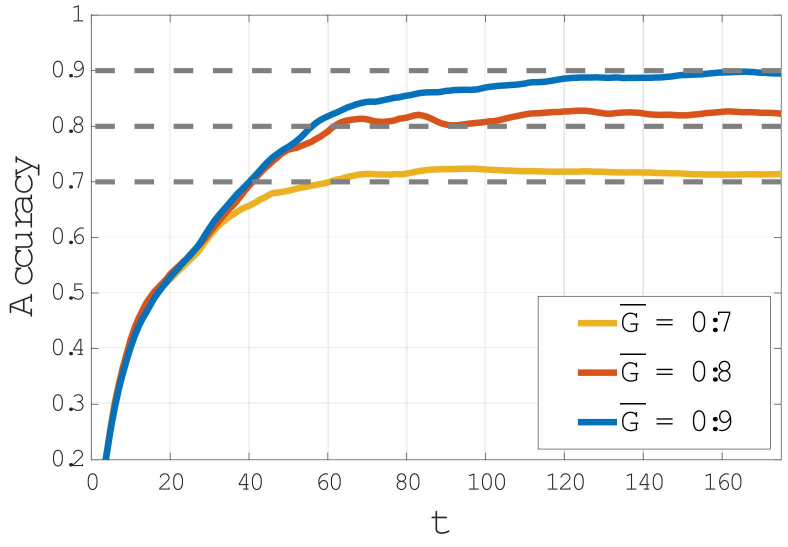


Figure 42: Accuracy vs. time

The trade-off between average power consumption and service delay is reported in Figure 43, where we compare the proposed approach with some alternative techniques.

In particular, we considered the following strategies for comparison:

i. Equal Rate Policy (referred to as Equal Rate): All the agents always transmit with a fixed number of quantization bits (to match a certain learning accuracy), the remote and local frequencies are fixed, and the uplink rate is equally adapted for all the agents to match the latency constraint;

ii. Fixed Scheduling & Quantization Bits Policy (referred to as Fixed S&B): All the agents always transmit with a fixed number of quantization bits (to match a certain learning accuracy), whereas remote frequency, local frequencies and uplink rates are optimized via Lyapunov Optimization;

iii. Joint Optimization with Random Scheduling and Quantization Bits (referred to as Random Joint): It is our joint procedure, but the scheduling and quantization bits are not assigned via the proposed greedy method, but rather with a random search of comparable complexity, meaning that multiple random realizations of the variables are checked to select the best one.

We can see from Figure 43 that the proposed method is able to strike the best trade-off among all competitive strategies.

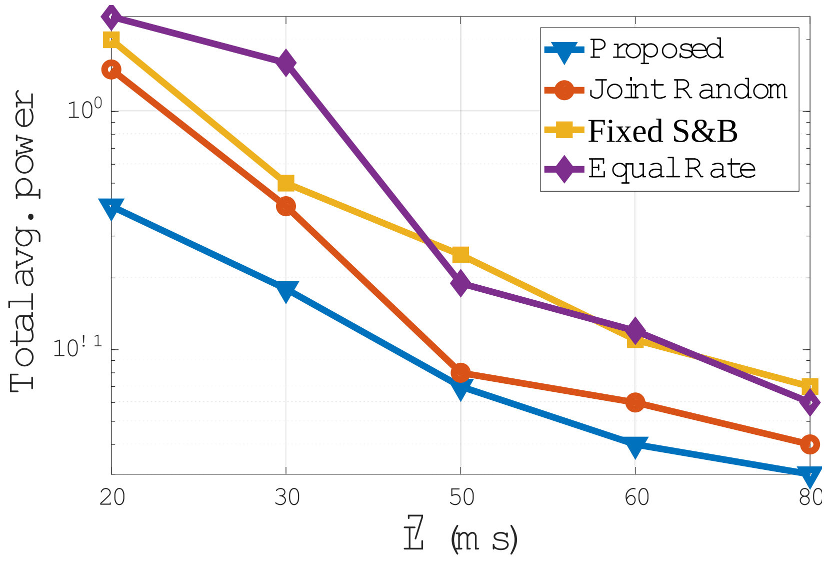


Figure 43: Average power consumption vs. average delay-Comparison between alternative strategies.

6.3 Goal oriented communications

SAP has proposed a new approach to goal-oriented communications, a research area that is starting receiving significant attention in the literature and it might represent a breakthrough in future networks. The idea underlying goal-oriented communications is that, whenever the interaction between source and destination occurs to fulfill a common goal, e.g. the control of a manufacturing process, the design of the communication blocks, e.g. source, and channel encoders, should be performed not to recover the transmitted bits, but rather to fulfill the goal motivating the exchange of information in an effective way, i.e. by minimizing energy consumption or service delay. This change of perspective has deep consequences on the design of the communication blocks.

The use of Deep neural networks (DNN) has been proposed to design a joint source/channel coding (JSCC), as an alternative to the conventional cascade of source and channel encoders. The JSCC based on DNN has been shown to achieve superior performance in the finite blocklength regime for image retrieval applications [Bou19], with respect to conventional communication architectures. The design of the JSCC encoder focusing directly on the recognition accuracy rather than performing image reconstruction and then classification separately, was investigated in [Chi19]. In [Jan20], the authors proposed an image retrieval scheme where, instead of sending the image, the feature vectors are first extracted and then mapped into channel input symbols, while the noisy channel output is used by the server to retrieve the most relevant images, without involving any explicit channel code. More recently, a Transformer-based approach has also been investigated in [Xie21] to support both image and text transmission.

Differently from previous works, SAP proposed a principled way to design the goal-oriented communication scheme, based on the information bottleneck (IB) principle. The basic idea is the following. Denoting by X the observed random variable and by Y the result of the decision (learning) task, the idea is to design the source encoder, at the sensor side, able to compute a compressed version T(X) of X, in order to minimize the complexity of the representation of X (measured in number of bits), while at the same time being able to keep (and send) only what is relevant to reconstruct Y. This principle can be expressed mathematically as the solution of the following optimization problem

$$\min I(X;\, T) - \beta\, I(T;\, Y)$$

where the minimization is carried out with respect to the conditional probability of *T*, given *X*. The first term, computed as the mutual information between *X* and *T*, represents the complexity of the representation of the source data, while the second term, computed as the mutual information between *T* and *Y*, represents the relevance of the encoded data *T* with respect to the decision variable Y; β is a (positive) trade-off parameter: small values of β yield more compact representations, but worse decisions, while large values lead to less compact representations but better quality decisions. This problem is known as the IB problem. Even though the problem is nonconvex, it has been shown that, in case of discrete random variables, the problem can be solved exactly using an iterative algorithm [Tis00]. Furthermore, in case of Gaussian random variables, the problem admits an exact solution in closed form.

The IB provides the theoretical foundation to review the communication paradigm, because it provides a principled way to compress the data and send only what is strictly relevant to reconstruct the required decision variables *Y directly*, rather than recovering X and then computing Y. This approach is particularly useful in the scenario considered in 5G-CONNI because it allows the monitoring sensors to reduce their data rates, thus reducing energy consumption and delay.

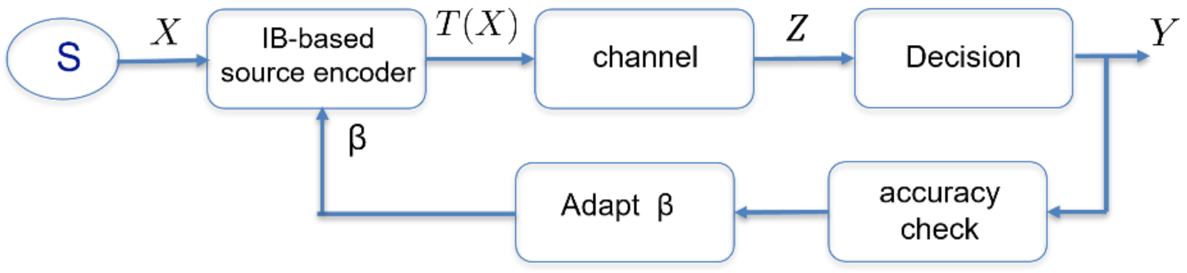


Figure 44: Goal-oriented communication based on the IB principle

In our work, we have promoted the idea of exploiting the IB principle as a possible driving principle of goal-oriented communication, proposing the scheme illustrated in Figure 44.

More specifically, we have extended the IB principle in two main directions: i) rather than fixing the trade-off parameter β once and for all, we made it adaptable online, as a function of the overall performance of the system, evaluated in terms of average energy consumption, average delay and average accuracy; ii) we modified the IB principle to make it suitable to the applications of interest for 5G-CONNI, by defining the objective function as

$$\min I(X;\, T) + \beta\, MSE(T;\, Y)$$

where $MSE(T;\, Y)$ indicates the mean square estimate of *Y*, given *T*. This formulation is more amenable for implementation because in many applications it could be very difficult to compute the mutual information between the encoded variable T and Y, while the mean square estimate is in general easier to compute and more directly associated to the learning tasks.

Some numerical results are useful to grasp the potentials of the proposed approach. The first example refers to the case in which the signal observed by the sensors can be modeled as a Gaussian random process. In such a case, we have derived closed form expressions for the encoder, which help us to better understand the behavior of the proposed communication strategy. As a numerical example, in Figure 45 we report the average power consumption vs. the (normalized) MSE, for different values of the average service delay. These curves represent the optimal trade-off between the three above mentioned performance indicators: average power consumption, delay and accuracy.

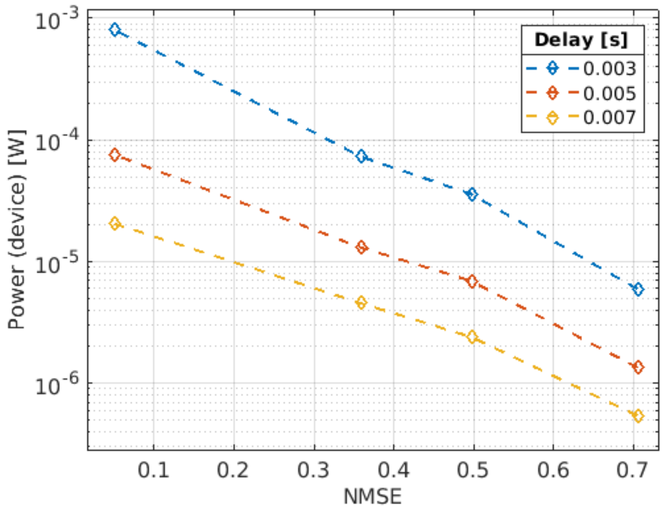


Figure 45: Average power-delay-accuracy trade-off achieved with IB-based communications.

The numerical results shown in Figure 45 have been obtained under the simplifying assumption of Gaussian random processes. To generalize the approach, for example to image transmission and recognition from remote, we generalized the IB-based approach by incorporating two convolutional neural networks (CNN), one at the transmit side, acting as a trainable encoder, and the other at the receive side, acting as a trainable decoder. The overall scheme is sketched in Figure 46.

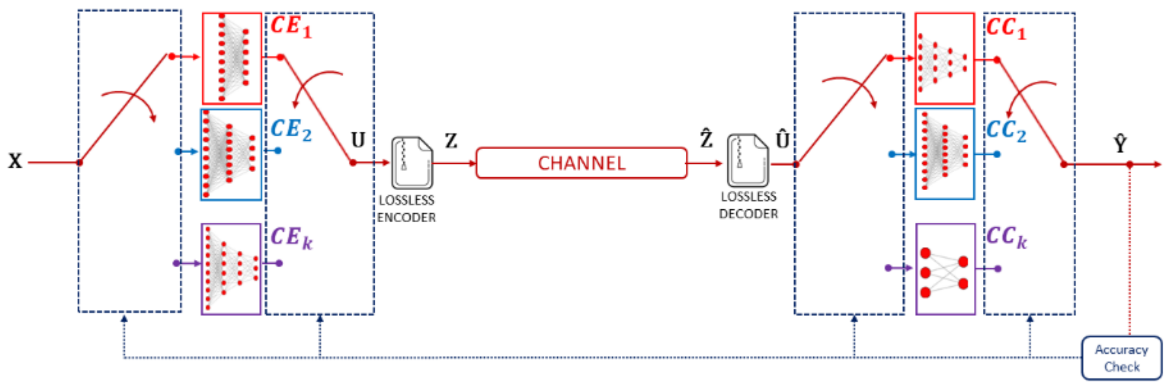


Figure 46:Goal-oriented communication scheme based on the IB principle implemented with variable size CNNs..

The architecture of the transmit CNN is chosen in order to introduce a bottleneck, obtained by adjusting the dimension of the CNN output vector (i.e., the number of output neurons in the last layer of the transmit CNN). The CNN at the transmit side is trained offline from a training set of images. Each CNN at the transmit side has its counterpart at the receive phase, which acts as a decoder. Each pair of transmit and receive CNN is trained jointly, using realistic channel models in the middle, according to the scheme shown in Figure 46.

To be able to adjust the bottleneck as a function of the channel state, we train a set of CNNs having different sizes of the transmit CNN output vector. At the end of the training phase, we have a set of CNNs pairs, having different inner size (the dimension of the transmit CNN output and receive CNN input), which can be selected during the communication phase as a pair of encoders/decoders. The adaptation is obtained by incorporating a feedback loop that adjusts the inner dimension as a function of the online performance. The system works in a slotted fashion and, in each slot, it selects the CNN pair most suitable for the situation at hand, depending on the channel state and on the current values of energy consumption, delay and accuracy. An example of performance, expressed in terms of image classification rate, is reported in Figure 47.

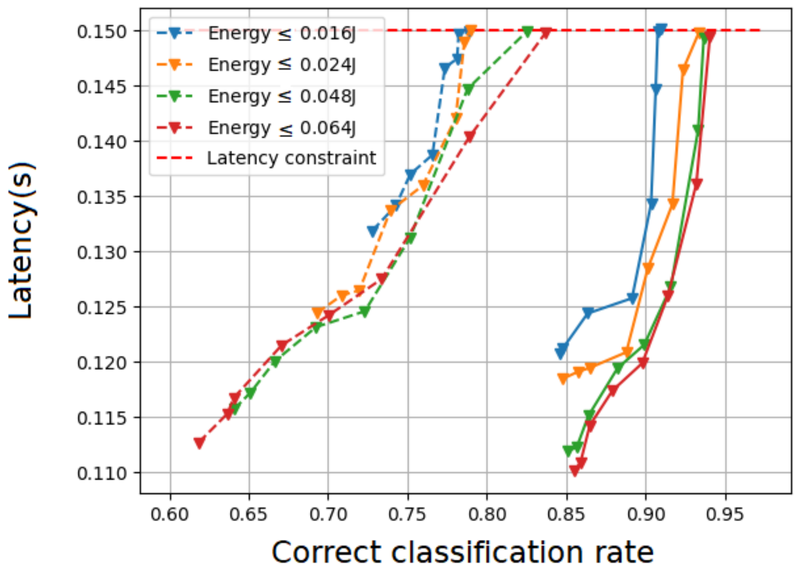


Figure 47 Average delay vs. correct image classification rate, for different energy consumption constraints.

The figure shows how the proposed scheme can adjust the selection of the encoder/decoder pair in order to achieve the desired trade-off between the performance of the learning task (image correct classification rate) and the system key performance indicators, i.e. energy consumption and delay.

6.4 Conclusions

In this section, we proposed three advanced functionalities for future private 5G networks. The first functionalities is dynamic resource allocations for URLLC. This novel dynamic decision maker framework enables a reliable, resource and delay-optimized scheduling suitable for dynamic URLLC scenarios (e.g., intermittent traffic source rate, time-varying channel). Based on Lyapunov optimization, it takes into account the traffic arrival at the network layer, the queue behaviors at the data link layer and the risk that the applied decision might trigger packet loss. The trade-off between the resource efficiency, latency and reliability is achieved by the timing and intensity of decisions.

Then, we have proposed a dynamic strategy for resource allocation in the edge cloud, aimed at finding the optimal trade-off between energy consumption and service delay, taking into account both computational and radio resources. More specifically, we considered the service placement, incorporating also the storage resources necessary to run virtual machines in the edge cloud. Even though the technical problem exhibits a very high complexity, we derived a numerically efficient solution hinging on Lyapunov stochastic optimization. Then, we extended the approach to a distributed learning problem, exploiting federated learning as an effective mechanism to achieve globally optimal solutions without requiring all the exchange of data between peripheral devices and edge server, but only exchange of parameter updates. The proposed method has been shown to compare favorably with a set of alternative techniques.

Furthermore, we have proposed a goal-oriented communication scheme, a very novel communication strategy that has the potentials to outperform the conventional Shannon-based approach in all cases where communication occurs to fulfill a common goal between source and destination. This is the case in the scenarios of interest for 5G-CONNI because for example, the sensors monitor the manufacturing process and send data to an edge server that typically runs a machine learning task on the received data to detect possible anomalies or to predict the behaviors of some critical parameter. In such a case, what really matters is not the recovery of all transmitted bits at the receiver side, but rather the achievement of a decision with the desired accuracy and reliability. This change of perspective leads to a different formulation of the communication task. We showed how to approach this problem exploiting the information bottleneck principle as a driving principle. We generalized the approach by incorporating convolutional neural networks, used as encode/decode pairs, trained offline and used adaptively online during the communication process.

7 Conclusions

D4.3 describes the final implementation of private 5G networks building blocks and the specification of advanced functionalities. This deliverable is an extension of D4.1 and D4.2. These innovative components have fueled the lab integration and experimental results of the building blocks will be reported in D5.3.

In task 4.1, 5G CONNI project built a RAN system composed of CDU, RU, and CPE as RAN implementation. gNodeB and CPE have been deployed in ITRI IMTC for industrial application and have been optimized to meet the requirements (e.g., bandwidth and low latency) of the selected use case.

In task 4.2, 5G CONNI project proposed two complementary 5G Core networks: 5G core prototype and lightweight orchestration framework. The 5G core components are defined as self-contained, independent and reusable network functions (NFs) together with a well-defined Service Based Interface (SBI) using HTTPv2 that can be used to invoke services. The 5GC provides interfaces and integrates with Application Network (AN), Mobile Edge Computing (MEC), and gNB. The 5GC contains the resources for setting up and maintaining the traffic between the gNBs and Application Network can support multiple gNBs. The 5GC is capable to achieve throughput to 40Gbps and data plane latency less than 1ms. The 5G Core also provides interfaces to support Network OAM (Operations Administration and Maintenance) functions.

5G CONNI project also worked on the ETSI NFV-like instantiation and orchestration of 5G mobile core network components via OSM Release Eleven. With the designed VNFDs, it is possible to deploy a mobile core network with off-the-shelf, standard-compliant MANO implementations such as OSM and ONAP. 5G CONNI project focused on the implementation of the semantic of the Ve-Vnfm reference point, to make OSM able to directly configure the core network in an ETSI standard approach. A ProxyAPI module has been implemented and added into the VNF to allow 5GC management and configuration in a standard way. A simple Web interface displays the 5GC VNF configuration status and progress. In order to have a unique centralized point of monitoring, 5G CONNI project has updated the orchestration framework with OSM metric collection. Thus we performed the OSM approach to retrieve VIM metrics (CPU, memory, disk, and network usage) and to visualize them in its Prometheus/Grafana instances. As a complementary work, we developed a provisioning procedure of a Network Slice Subnet (NSS) modeled as a Network Service (NS), composed of several VNFs from potentially different vendors. It includes a Network Management System (NMS) conforming to the 3GPP Service-Based Management Architecture (SBMA), an ETSI MANO orchestrator, and a Network Function Virtualization Infrastructure (NFVI).

In task 4.3, two implementations of MEC are proposed by 5G-CONNI for the European and Taiwanese testbeds: the hybrid 5GC solution and the bump-in-the-wire solution. Hybrid 5GC solution consists in the splitting of CP and UP functions through OSM, giving more flexibility for the mobile network deployment. The 5GC CP network functions (e.g., AMF, UDM, SMF, etc.) is deployed in the central NFVI server, while the 5GC UPF, the only component acting as UP function, is deployed in the edge NFVI server. This approach allows the support of edge computing, with the consequent possibility of installing a MEC platform. MEC 5G SA based on a bump-in-the-wire architecture, developed handover, multi-PDU sessions and multi-QoS flows functionalities The bump-in-the-wire solution proposes a MEC platform with virtualization capability and VNF management. The virtualization capability is responsible for virtualizing edge computing functions, industrial applications and any needed applications to the MEC platform to realize flexible deployment. The VNF management supports the essential management demands of ETSI NFV specifications and adjusts the performance of VNFs to achieve the required transmission delay, execution performance and resource management.

In Task 4.4, 5G CONNI project worked on three vertical use cases, namely (1) Process Diagnostics by CNC and Sensing Data Collection, (2) process diagnostics using Augmented/Virtual Reality, (3) cloud based controller for CNC. Among these implemented use cases, (1) & (2) were implemented on a five-axis machine tool and (3) was implemented on a flexible fixture system, which is a specialized machine to test the cloud-based controller. The multi-site use case is basically an extension of use case 2. For use case (1) the implementation was completed, production data, machine operation data and sensor data can be access by application of use case 2 or other data analysis modules. For use case (2), 3D model of the target machine have been created and deployed in a remote rendering server, which will then linked with MEC. Machine operator and remote expert now use the same application to conduct process diagnosis, these two instances of application share the same 3D machine model and synchronized by the machine data from the WebAPI server developed by IMTC. The prototype of cloud CNC (i.e. use case 3) has been developed and tested on the flexible fixturing system by sending vibration suppression motion commands from cloud controller to the ground controller to evaluate overall performance. Experimental results of the use cases will be reported in D5.3.

5G CONNI project also investigated advanced functionalities for future private 5G networks. First, 5G CONNI proposes a dynamic resource allocations for URLLC. This novel dynamic decision maker framework enables a reliable, resource and delay-optimized scheduling suitable for dynamic URLLC scenarios (e.g., intermittent traffic source rate, time-varying channel). Based on Lyapunov optimization, it takes into account the traffic arrival at the network layer, the queue behaviors at the data link layer and the risk that the applied decision might trigger packet loss. The trade-off between the resource efficiency, latency and reliability is achieved by the timing and intensity of decisions.

Second, 5G CONNI proposes a dynamic strategy for resource allocation in the edge cloud, aimed at finding the optimal trade-off between energy consumption and service delay, taking into account both computational and radio resources. More specifically, we considered the service placement, incorporating also the storage resources necessary to run virtual machines in the edge cloud. The solution is based on Lyapunov stochastic optimization and extended to a distributed learning problem, exploiting federated learning as an effective mechanism to achieve globally optimal solutions without requiring all the exchange of data between peripheral devices and edge server, but only exchange of parameter updates.

Finally, 5G CONNI proposes a goal-oriented communication scheme, a very novel communication strategy that has the potentials to outperform the conventional Shannon-based approach in all cases where communication occurs to fulfill a common goal between source and destination. We generalizes the approach exploiting the information bottleneck principle as a driving principle, by incorporating convolutional neural networks, used as encode/decode pairs, trained offline and used adaptively online during the communication process.

References

[3GPP19] 3GPP, “Study on channel model for frequencies from 0.5 to 100 GHz, Release 16 (v16.1.0),” 2019.

[3GPP21] 3GPP, “Physical layer procedures for data, release 16,” 2021.

[Bat22] C. Battiloro, M. Merluzzi, P. Di Lorenzo, S. Barbarossa, "Lyapunov-based Optimization of Edge Resources for Energy-Efficient Adaptive Federated Learning", IEEE Transactions on Green Communications and Networking, 2022

[Bou19] E. Bourtsoulatze, D. B. Kurka, and D. Gunduz, "Deep joint source channel coding for wireless image transmission". IEEE Transactions on Cognitive Communications and Networking, pp. 567–579, 2019.

[CDD+21] M. Chahbar, G. Diaz, A. Dandoush, C. Cerin, and K. Ghoumid, “A comprehensive survey on the E2E 5G network slicing model,” in IEEE Trans. Netw. Service Manag., vol. 18, no. 1, pp. 49-62, Mar. 2021

[Chi19] C.-H. Lee, J.-W. Lin, P.-H. Chen, and Y.-C. Chang. "Deep learning constructed joint transmission-recognition for Internet of Things. IEEE Access, 7, pp. 76547–76561, 2019.

[ETSIMEC] https://docbox.etsi.org/isg/mec/open (draft).

[GSMA20] Generic Network Slice Template, GSMA NG.116 v4.0, Nov. 2020.

[GSNFV] Network Functions Virtualisation (NFV); Architectural Framework, ETSI GS NFV 002 v1.2.1, Dec. 2014.

[GSNFVIFA] Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Network Service Templates Specification, ETSI GS NFVIFA 014 v3.4.1, May 2021.

[GSNFVIFA+] Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; VNF Descriptor and Packaging Specification, ETSI GS NFV-IFA 011 v3.5.1, May 2021.

[GSNFVSOL] Network Functions Virtualisation (NFV) Release 3; Protocols and Data Models; RESTful protocols specification for the Os-Ma-nfvo Reference Point, ETSI GS NFV-SOL 005 v3.3.5, Mar. 2021.

[HCF20] D. Han, W. Chen, and Y. Fang, “Joint channel and queue aware scheduling for latency sensitive mobile edge computing with power constraints,” IEEE Transactions on Wireless Communications, vol. 19, no. 6, pp. 3938–3951, 2020.

[Huang21] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, “An efficiency-boosting client selection scheme for federated learning with fairness guarantee,” IEEE Trans. on Parallel & Distributed Systems, vol. 32, no. 07, pp. 1552–1564, July,2021.

[HZS+22] B. Han, Y. Zhu, M. Sun, V. Sciancalepore, Y. Hu, and H. D. Schotten, “CLARQ: A dynamic ARQ solution for ultra-high closed-loop reliability,” IEEE Transactions on Wireless Communications, vol. 21, 2022.

[ISH22] A. Ishaq et al., “Service-based management architecture for on-demand creation, configuration, and control of a network slice subnet,” 2022 IEEE International Conference on Network Softwarization (NetSoft), 2022.

[ITU00] Telecommunication Management Network; TMN management functions, ITU-T Recommendation M.3040, Feb. 2000.

[ITU19] ITU, “The prediction of the time and the spatial profile for broadband and mobile services using UHF and SHF bands,” Aug. 2019

[JAB17] T. Jacobsen, R. Abreu, G. Berardinelli, K. Pedersen, P. Mogensen, I. Z. Kovacs, and T. K. Madsen, “System Level Analysis of Uplink Grant- Free Transmission for URLLC,” in 2017 IEEE Globecom Workshops (GC Wkshps), pp. 1–6, 2017.

[Jan20] Mikolaj Jankowski, Deniz Gunduz, and Krystian Mikolajczyk, "Wireless image retrieval at the edge". IEEE Journal on Selected Areas in Communications, 39(1), pp. 89–100, 2020.

[JBSL16] M. Jabi, M. Benjillali, L. Szczecinski, and F. Labeau, “Energy Efficiency of Adaptive HARQ,” IEEE Transactions on Communications, vol. 64, no. 2, pp. 818–831, 2016.

[LDE+21] Y. Liu, Y. Deng, M. Elkashlan, A. Nallanathan, and G. K. Karagiannidis, “Analyzing Grant-Free Access for URLLC Service,” IEEE Journal on Selected Areas in Communications, vol. 39, no. 3, pp. 741–755, 2021.

[LSK21] T.-K. Le, U. Salim, and F. Kaltenberger, “An Overview of Physical Layer Design for Ultra-Reliable Low-Latency Communications in 3GPP Releases 15, 16, and 17,” IEEE Access, vol. 9, pp. 433–444, 2021.

[LWE+20] S. Lagen, K. Wanuga, H. Elkotby, S. Goyal, N. Patriciello, and L. Giupponi, “New Radio Physical Layer abstraction for System-Level Simulations of 5G Networks,” in ICC 2020 - 2020 IEEE International Conference on Communications (ICC), pp. 1–7, June 2020

[MAA16] H. Mukhtar, A. Al-Dweik, and M. Al-Mualla, “Content-aware and occupancy-based hybrid ARQ for video transmission,” in 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWS- CAS), pp. 1–4, 2016.

[MDC21] M. Maman, L. N. Dinh, and E. Calvanese Strinati, “Procédé et dispositif d’orchestration de l’exécution de mécanismes dans un réseau sans fil,” FR Patent 2103542, 2021.

[Mer22] M. Merluzzi, N. Di Pietro, P. Di Lorenzo, E. Calvanese Strinati, S. Barbarossa, "Discontinuous Computation Offloading for Energy-Efficient Mobile Edge Computing", IEEE Transactions on Green Communications and Networking, 2022.

[NEE14] M. J. Neely, “A Simple Convergence Time Analysis of Drift-Plus- Penalty for Stochastic Optimization and Convex Programs,” 2014.

[OL+18] J. Ordonez-Lucena et al., “The creation phase in network slicing: From a service order to an operative network slice,” in Proc. EuCNC, Ljubljana, Slovenia, June 2018

[OTR20] J. Ordonez-Lucena, C. Tranoris, and J. Rodrigues, “Modeling network slice as a service in a multi-vendor 5G experimentation ecosystem,” in Proc. IEEE ICC Workshops, Dublin, Ireland, June 2020.

[PDN14] S. Pfletschinger, D. Declercq, and M. Navarro, “Adaptive HARQ With Non-Binary Repetition Coding,” IEEE Transactions on Wireless Communications, vol. 13, no. 8, pp. 4193–4204, 2014.

[Pez22] F. Pezone, S. Barbarossa, P. Di Lorenzo, "Goal-Oriented Communication for Edge Learning Based on the Information Bottleneck", Proc. of IEEE ICASSP 2022, pp. 8832-8836.

[PLBG19] N. Patriciello, S. Lagen, B. Bojovic, and L. Giupponi, “An E2E simulator for 5G NR networks,” Simulation Modelling Practice and Theory, vol. 96, p. 101933, Nov. 2019.

[PLGB19] N. Patriciello, S. Lagen, L. Giupponi, and B. Bojovic, “The Impact of NR Scheduling Timings on End-to-End Delay for Uplink Traffic,” in 2019 IEEE Global Communications Conference (GLOBECOM), 2019

[POU20] K. Poularakis, J. Llorca, A. M. Tulino, I. Taylor, and L. Tassiulas, “Service placement and request routing in mec networks with storage, computation, and communication constraints,” IEEE/ACM Transactions on Networking, vol. 28, no. 3, pp. 1047–1060, 2020

[Sun20] Y. Sun, S. Zhou, and D. Gunduz, “Energy-aware analog aggregation for federated learning with redundant data,” in Proc. of IEEE ICC, Virtual, 7-11 June 2020, pp. 1–7.

[SYQ17] C. She, C. Yang, and T. Q. S. Quek, “Radio Resource Management for Ultra-Reliable and Low-Latency Communications,” IEEE Communications Magazine, vol. 55, no. 6, pp. 72–78, 2017.

[Tis00] N. Tishby, F. C Pereira, and W. Bialek, "The information bottleneck method", arXiv preprint physics/0004057, 2000.

[TS23501] System architecture for the 5G System (5GS), 3GPP TS 23.501 v16.7.0, Jan. 2021.

[TS28500] Management concept, architecture and requirements for mobile networks that include virtualized network functions, 3GPP TS 28.500 v16.0.0 Aug. 2020.

[TS28530] Management and orchestration; Concepts, use cases and requirements, 3GPP TS 28.530 v16.2.0, Aug. 2020.

[TS28531] Management and orchestration; Provisioning, 3GPP TS 28.531 v16.6.0, Aug. 2020.

[TS28533] Management and Orchestration; Architecture Framework, 3GPP TS28.533 v16.7.0, Apr. 2021.

[TS28541] Management and orchestration; 5G Network Resource Model (NRM); Stage 2 and stage 3, 3GPP TS 28.541 v16.6.2, Nov. 2020.

[TS32101] Telecommunication Management; Principles and High Level Requirements Release 16, 3GPP TS 32.101 v. 16.0.0, Aug. 2020

[WLCE19] M. Wang, J. Liu, W. Chen, and A. Ephremides, “Joint queue-aware and channel-aware delay optimal scheduling of arbitrarily bursty traffic over multi-state time-varying channels,” IEEE Transactions on Communications, vol. 67, no. 1, pp. 503–517, 2019.

[Xie21] H. Xie, Z. Qin, X. Tao, and K. B Letaief, "Task-oriented multiuser semantic communications", arXiv preprint arXiv:2112.10255, 2021.

[YUAN06] W. Yuan and K. Nahrstedt, “Energy-efficient cpu scheduling for multimedia applications,” ACM Trans. Comp. Syst., vol. 24, no. 3, pp. 292–331, aug 2006.

[Zhou20] S. Zhou, Z.and Yang, L. Pu, and S. Yu, “Cefl: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes,” IEEE Int. of Things Jour., vol. 7, no. 10, pp. 9341–9356, March 2020.