6GSNS

Co-funded by the European Union

SAFE-6G

SAFE-6G

A Smart and Adaptive Framework for Enhancing Trust in 6G Networks

Deliverable D3.4: Infrastructure Openness, Monitoring and Programmability

Date: 17/07/2026

Version: V1.0

DISCLAIMER

Grant Agreement	101139031
Document number	D3.4
Document title	Infrastructure openness, monitoring and programmability
Lead Beneficiary	NCSRD
Editor(s)	Panagiotis Pavlidis (NCSRD) Spyridon Georgoulas (NCSRD) Harilaos Koumaras (NCSRD)
Author(s)	Panagiotis Pavlidis (NCSRD) Spyridon Georgoulas (NCSRD) Dimitrios Roidis (NCSRD) Harilaos Koumaras (NCSRD) Vagelis Anagnostopoulos (INF) Vaios Koumaras (INF) Alejandro Fornés (UPV) Charles Bailly (IMM) Jose Costa-Requena (CMC) Pelayo Torres (TID) José María Ramón (TID)
Dissemination level	Public
Contractual date of delivery	30/06/2026
Status	Final
File name	SAFE-6G_D3.4_V1.0.pdf

Revision History

Version	
V0.1	Initial table of contents, document structure and writing guidelines
V0.2	First draft with initial partner contributions
V0.3	Integration of technical contributions and API descriptions
V0.4	Added deployment sections
V0.5	Internal review and incorporation of partners' comments
V0.6	Updated figures, tables and API endpoint descriptions
V0.7	Quality review and consistency improvements across sections
V0.8	Homogenization with Deliverable D3.3, cross-reference verification, formatting refinements and final internal review
V0.9	Final technical revisions and editorial corrections, Internal review, sent to final edition
V1.0	Final version

GLOSSARY

Abbreviations/Acronym	Description
3GPP	3rd Generation Partnership Project
5G	Fifth-Generation Mobile Network
6G	Sixth-Generation Mobile Network
AF	Application Function
API	Application Programming Interface
CAPIF	Common API Framework
CEF	CAPIF Exposure Function
CMC	Cumucore
CNC	Cumucore Network Configuration
CoCo	Cognitive Coordinator
DB	Database
DNN	Data Network Name
DT	Digital Twin
GBR	Guaranteed Bit Rate
gNB	Next Generation Node B
HLO	High-Level Orchestrator
HMD	Head-Mounted Display
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
JSON	JavaScript Object Notation
JWT	JSON Web Token
K3s	Lightweight Kubernetes Distribution
K8s	Kubernetes
ML	Machine Learning
MLOps	Machine Learning Operations
NEF	Network Exposure Function
NF	Network Function
NRF	Network Repository Function
NWDAF	Network Data Analytics Function
O-NEF	Open Network Exposure Function
OAM	Operations, Administration and Maintenance
OCF	OpenCAPIF
OVR	Oculus Virtual Reality
QoS	Quality of Service
REST	Representational State Transfer
SBA	Service-Based Architecture
SCS/AS	Service Capability Server/Application Server
SM	Session Management
SMF	Session Management Function
S-NSSAI	Single Network Slice Selection Assistance Information

SUPI	Subscription Permanent Identifier
TA	Tracking Area
TF	Trust Function
TLS	Transport Layer Security
TS	Technical Specification
UC	Use Case
UDM	Unified Data Management
UDR	Unified Data Repository
UE	User Equipment
UPF	User Plane Function
URL	Uniform Resource Locator
vApp	Vertical Application
WP	Work Package
XR	Extended Reality

EXECUTIVE SUMMARY

This document belongs to the second round of WP3 deliverables. In contrast to the first round where one single document contained the report of all WP tasks, now each of them has produced their own deliverable. Specifically, D3.4: Infrastructure Openness, Monitoring and Programmability, reports the final outcomes of Task 3.3. This deliverable extends the software outcomes reported through the SAFE-6G GitLab repositories by providing a comprehensive technical description of the developed components. While the software implementations are maintained in the project's GitLab, this document presents their architecture, deployment, interfaces and operational details, offering a consolidated overview of the solutions developed within Task 3.3. The work focuses on exposing network capabilities through standardized and project-specific APIs, enabling secure interaction between network functions, applications and external consumers while supporting the user-centric operation of the SAFE-6G platform. The document presents the software components developed around the 5G Core, including API exposure, monitoring and programmability mechanisms based on the Cumucore and Open5GS platforms. It provides a consolidated reference of the available APIs, including internal interfaces, NWDAF services, QoS Session APIs and Network Exposure Function (NEF) interfaces, together with their supported endpoints and deployment status. Furthermore, the deliverable presents the software developed to support monitoring and programmability across the edge–cloud continuum, including the Meta-OS framework that enable dynamic orchestration of distributed services. Finally, it describes the programmable interfaces supporting the SAFE-6G Metaverse framework, enabling interaction with digital twins, immersive applications and XR services. Overall, the software components presented in this deliverable constitute the infrastructure layer that enables secure API exposure, monitoring and programmable operation of the SAFE-6G ecosystem, providing the necessary foundations for user-centric service deployment and intelligent network management.

KEYWORDS

Network Exposure APIs, OpenCAPIF, 6G Core Network, Open5GS, Edge–Cloud Continuum, Metaverse

TABLE OF CONTENTS

List of FIGURES

List of TABLES

1 INTRODUCTION

Figure 1 highlights in green the building blocks of the SAFE-6G architecture implemented in WP3. Within the SAFE-6G framework, the Cognitive Coordinator and the Trust Functions interact with the underlying infrastructure through standardized interfaces to monitor the state of the system and dynamically configure network, computing and application resources according to trust-related user intents. This deliverable focuses on the infrastructure openness, monitoring and programmability capabilities of the SAFE-6G framework. Specifically, it presents the final implementation of the APIs exposed by the user-centric 5G Core, the Edge-Cloud Continuum and the Metaverse platform, together with the integration of OpenCAPIF as the common API exposure framework. Furthermore, it introduces the CAPIF Exposure Functions (CEF), extending the OpenCAPIF ecosystem to simplify the integration of API providers. Collectively, these components provide a secure and standardized interface through which authorized SAFE-6G components and external applications can discover, access and interact with the underlying infrastructure and services. The components and modules addressed in this deliverable are highlighted in yellow below. This deliverable complements the software repositories developed within Task 3.3 by providing additional technical documentation of the reported software components. While the software developed within Task 3.3 includes both commercial and open-source components, this deliverable places particular emphasis on the open-source software developed as part of the SAFE-6G project. The commercial CUMUCORE developments are documented at a high level where appropriate, whereas the open-source components are presented in greater detail, accompanied by references to their corresponding GitLab repositories. The corresponding software repositories are referenced in the respective chapters, allowing the reader to directly access the implementation associated with each presented component.

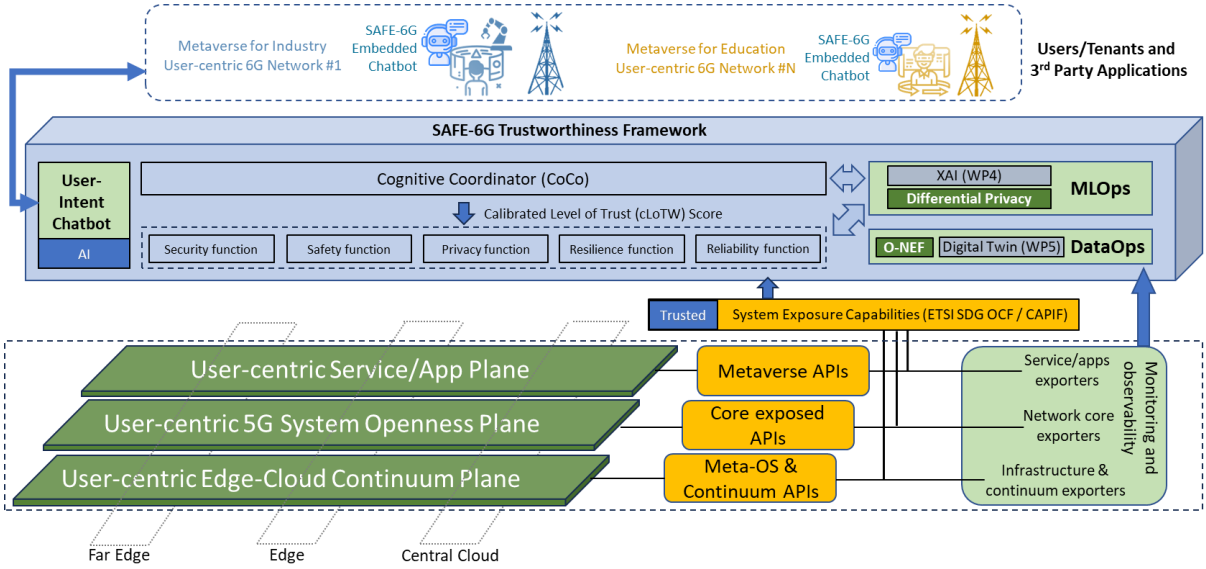


Figure 1: Building blocks and components of the SAFE-6G architecture

1.1 THE RATIONALE BEHIND THE STRUCTURE

The structure of this deliverable is as follows. After this introduction, Section 2 presents the APIs exposed by the different SAFE-6G infrastructure planes, namely the user-centric 6G Core, the Edge-

Cloud Continuum and the Metaverse platform. For each API family, a brief overview of its role within the SAFE-6G architecture is provided, followed by a description of the exposed endpoints, deployment aspects and the project requirements addressed by its implementation. Section 3 focuses on the OpenCAPIF framework, describing its architecture, deployment within the SAFE-6G environment, the workflow tools developed to simplify API onboarding and management, the APIs exposed through CAPIF and the CAPIF Exposure Functions (CEF) introduced to facilitate the integration of API providers into the OpenCAPIF ecosystem. Finally, Section 4 summarizes the main conclusions of the deliverable.

2 AVAILABLE APIS FOR INFRASTRUCTURE OPENNESS

This section presents the final set of APIs exposed by the SAFE-6G infrastructure domains, namely the user-centric 5G Core, the Edge-Cloud Continuum and the Metaverse platform. Building upon the initial architecture and API specifications introduced in Deliverable D3.1, it reports the APIs that have been implemented, refined and integrated during the project, highlighting the capabilities they expose, their deployment within the SAFE-6G framework and their role in enabling infrastructure openness, monitoring and programmability.

2.1 5G CORE APIS

The user-centric 5G Core exposes a set of APIs that provide secure access to network capabilities and management functions, enabling the SAFE-6G framework to monitor, configure and interact with the underlying core network. This section presents the final APIs implemented within the 5G Core, covering both the Cumucore and Open5GS deployments, together with their exposed functionalities and deployment aspects.

The specific development and implementation activities related to the 5G Core APIs have been guided by the following requirements stated in D2.1:

- **REQ-COM-ETH-NF-M-03**: Compliance with 5G and industry standards (use of API REST and standard Open API descriptors).
- **REQ-COM-DES-F-M-04**: Following Cloud Native paradigm (virtualized, ready for K8s).
- **REQ-COM-DES-F-M-19**: Access to performance metrics from 5G/6G core infrastructure
- **REQ-UCEN-COMP-F-M-05**: Core network exposure must be performed via the OpenCAPIF.

2.1.1 CUMUCORE CORE

The Cumucore 5G Core provides the network programmability capabilities required by the SAFE-6G framework through a set of APIs that expose network information and management functions in a controlled and secure manner. These APIs enable trusted SAFE-6G components to retrieve monitoring information, access network configuration data and interact with core network functions, thereby supporting the infrastructure openness and programmability objectives introduced in Deliverable D3.1. As seen in Figure 2, to distinguish between internal and external API consumers, Cumucore adopts a two-layer exposure model consisting of the Cumucore Network Controller (CNC) for internal access and the Network Exposure Function (NEF) for secure exposure to external application functions.

Cumucore core includes an internal function called Cumucore Network Controller (CNC) that provides an API for accessing information in the core network functions (NFs) such as UDM, UDR and NWDAF. The core configuration information and UE profiles, operator information and data monitoring are collected in these NFs. The CNC exposes a REST API for internal application functions (AFs) to access the core information. In SAFE-6G in order to bring flexibility to the access of the core internal information, the CNC API has been exposed through the Network Exposure Function (API) which facilitates its exposure as CAPIF API.

The Cumucore core includes the CNC for internal AFs to access all the information and configuration, and the NEF that exposes and extends CNC API to be accessible for external AFs as shown in the figure below.

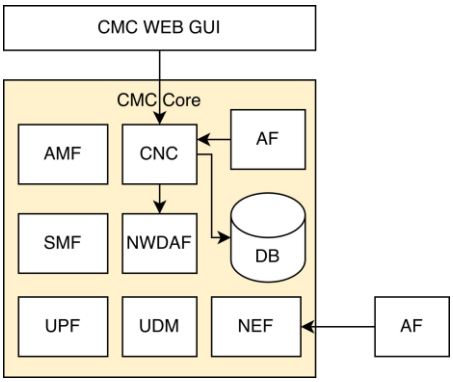


Figure 2: CUMUCORE architecture with CNC and NEF API exposure

As the implementation details of the 5G Core APIs are described in Deliverable D3.3, they are not repeated here from an implementation perspective. However, for consistency and to provide a single reference containing all APIs exposed across the SAFE-6G infrastructure, the corresponding API endpoints are presented again in this deliverable, accompanied by a brief description of their functionality and intended usage.

The user-centric 5G Core exposes a set of internal APIs that enable communication between trusted Application Functions (AFs) and Network Functions (NFs) deployed within the same core environment. These interfaces provide access to network configuration, monitoring information and management capabilities required for the operation of the SAFE-6G framework. Access to the internal APIs is restricted to trusted components and is protected through Transport Layer Security (TLS)-based authentication, ensuring secure communication between the interacting entities.

Table 1 presents the internal APIs exposed through the Cumucore Network Controller (CNC), together with their supported methods and a brief description of their functionality.

API prefix: /api/v1.0		
Endpoint	***Method/s***	***Description***
/cnc-subscriber-management?order={int=1 or -1}	GET	Subscriber's profiles
/cnc-subscriber-management/by-group?gname={{USER_GROUP_ID}}&order={{1 or -1}}	GET	Subscriber's info by group
/smf-cnc/qos-monitoring-report/	GET	QoS monitoring report from the UPF
/smf-cnc/qos-monitoring-report?report-type=delay&tgt_ue=\"{{TARGET_SUPI}}\"&time=10	GET	UE delays measurement
/cnc-configuration/sm-policy-config?ueId={{TARGET_SUPI}}&snssai={sst:{{TARGET_SST}},sd:"{{TARGET_SD}}"&dnn="{{TARGET_DNN}}"}	GET	SM policy of UE (SUPI)
/cnc-nwdaf/analytic-info-request?event-id={{event type}}	GET	NF list with features at NRF
/nnwdaf-analyticsinfo/v1/analytics?event-id=LOAD_LEVEL_INFORMATION&event-filter={"snssais":"[{"sst":{{TARGET_SST}},"sd":"{{TARGET_SD}}"}]}	GET	Slice usage statistics

/cnc-slice-capacity/per-interface-stats/{{sst+sd}}	GET	Slice usage capacity
/cnc-tac-geolocatiom/Polygon	GET	Polygons and slices in TA
/cnc-config/af-qos-profile	GET	QoS profile templates

Table 1: Cumucore CNC API

The Network Data Analytics Function (NWDAF) APIs provide access to network analytics and performance information collected from the 5G Core Network Functions (NFs). Through these interfaces, authorized internal applications can retrieve operational metrics, network statistics and analytics that support monitoring, optimization and intelligent decision-making within the SAFE-6G framework.

Table 2 presents the NWDAF APIs exposed by the user-centric 5G Core, together with their supported methods and a brief description of their functionality.

API prefix: /nwdaf-analyticsinfo/v1/analytics?eventid=		
Endpoint	***Method/s***	***Description***
LOAD_LEVEL_INFORMATION&eventfilter={"snssais":"[{"sst":{{TARGET_SST}},"sd":"{{TARGET_SD}}"}]}	GET	Analytics of network slice
UE_COMM&tgtue={"supi":["{{TARGET_SUPI}}"]}	GET	Analytics of UE (TS 23.288)
NF_LOAD&tgtue={"anyUe":true}	GET	NF load analytics predictions
UE_MOBILITY&tgt-ue={"supi":["{{TARGET_ SUPI}}]}	GET	UE mobility analytics (SUPI)
USER_DATA_CONGESTION&tgt-ue={{TARGET_ SUPI}}	GET	UE data congestion analytics/ predictions (SUPI)
NETWORK_PERFORMANCE&tgt-ue={"anyUe": true}	GET	GNB Analytics/predictions

Table 2: Cumucore NWDAF API

The QoS Session APIs enable authorized applications to establish, modify and terminate Quality of Service (QoS) sessions through the Network Exposure Function (NEF). By exposing standardized QoS management capabilities of the 5G Core, these interfaces allow external and trusted SAFE-6G components to request application-specific network treatment based on their service requirements. This functionality is essential for supporting user-centric and intent-driven services that require dynamic QoS provisioning.

Table 3 presents the QoS Session APIs exposed by the user-centric 5G Core, together with their supported methods and a brief description of their functionality.

API prefix: /3gpp-as-session-with-qos/v1/0001/subscriptions		
Endpoint	***Method/s***	***Description***
/	POST	Add secondary flow with QOS on top of default flow with best effort
/{{FLOW_ID}}	DELETE	Delete the secondary flow

Table 3: Cumucore QoS Session API

The Network Exposure Function (NEF) provides a secure northbound interface through which selected 5G Core capabilities are exposed to external Application Functions (AFs). Acting as the gateway between the core network and external consumers, the NEF enables authorized applications to access network information and request network services without direct interaction with internal Network Functions (NFs). This approach preserves the security and integrity of the core while providing controlled and standardized access to its capabilities. Within the SAFE-6G framework, the NEF exposes

both standard 3GPP interfaces and project-specific extensions developed to support the project's trustworthiness and user-centric service requirements. Access to these APIs is restricted to authenticated and authorized consumers and is protected through secure communication mechanisms.

Table 4 presents the NEF APIs exposed by the user-centric 5G Core, together with their supported methods and a brief description of their functionality.

API prefix: /		
Endpoint	***Method/s***	***Description***
/oam-data/operator-info	POST	Create operator
/oam-data/operator-info?id={id}	PUT	Update operator
/oam-data/operator-info	GET	Get operators
/oam-network-slice/slice-instance	POST	Create a slice meta data
/oam-configuration/ue-subscription-profile	GET	Get Slice
/oam-network-slice/slice-instance/{sliceName}	DELETE	Delete a slice
/oam-configuration/ue-subscription-profile	POST	Create qos profile
/oam-configuration/ue-subscription-profile/{profile_id}	PUT	Update qos profile
/oam-configuration/user-group	POST	Create Group
/oam-configuration/user-group/{groupId}	GET	Get Group Info
/oam-configuration/user-group/{groupId}	PUT	Update Group Info
/oam-network-slice/upf-dnn-list	GET	List available UPFs
/oam-config/upf-profile-update/{nfInstanceId}	PUT	Update UPF configuration
/oam-subscriber	POST	Create subscriber
/oam-subscriber	PUT	Update subscriber
/oam-preemption/preemption-config	POST	Update Qos on the run
/oam-settings/single-ue-am-status/{gpsi}	GET	Get gNB IP
/oam-settings/connectivity-per-gnb?gnb-ip={ip_address}	GET	Get gNB load
/oam-monitoring-info/core-capacity/interface-stats?id=goupf	GET	Get user plane load in Bare metal deployment
/oam-settings/core-capacity	GET	Get signalling load
/oam-data/{nfType}/nf-analytics-request	GET	Get computing platform load
/oam-data/v1/oam-monitoring-info/core-capacity/k8-interface-stats?instanceName={UPF}&namespace={5g}	GET	Get Computing platform load
/3gpp-analyticsexposure/v1/{afId}/fetch	POST	Get UE Usage
/oam-preemption/preemption-config	GET	Subscribe to alarm events
/oam-alarms/v1/alarm-event/history-information/time-range?start-time={start-time}&end-time={end-time}	GET	Get history of alarms
/api/v1/alarm-event/history-information?eventId=latest	GET	Get the latest specific alarm item

Table 4: NEF API Exposed to External AFs

2.1.2 DEPLOYMENT

The CUMUCORE 5G Core APIs were deployed and validated within the SAFE-6G testbed at NCSRD. The deployment includes both the Cumucore Network Controller (CNC), responsible for exposing the internal management interfaces, and the Network Exposure Function (NEF), which provides secure exposure of selected network capabilities to external Application Functions (AFs).

The CNC and NEF services operate as independent system services within the Cumucore deployment. During initialization, the NEF registers the exposed services, establishes connectivity with the supporting database and discovers the available network functions, while the CNC subscribes to network events and provides access to configuration and monitoring information. Figure 3, Figure 4, Figure 5, and Figure 6 present snapshots from the deployed services, illustrating their successful initialization and runtime operation.

```
-- Boot 320731407a584583bdfc6aa78a29c373 --
Jul 10 13:20:50 ubuntu systemd[1]: Starting Cumucore Network Configuration...
Jul 10 13:20:55 ubuntu systemd[1]: Started Cumucore Network Configuration.
Jul 10 13:20:55 ubuntu CNC-API[1330]: 2026/07/10 13:20:55 INFO-CUMUCORE NETWORK CONFIGURATION API
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="No DNS Configuration lookup : no such host"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="NF SUBSCRIPTION ID ==> map[bcc63bd8-986e-4d79-ae3b-6ee2450e98a1:00101-55503]"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="SUBSCRIPTION FOR nwdaf SUCCESS =>201"
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance Name:    goupf
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance Type:    UPF
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance IP:      [127.0.0.1]
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="UPF discovery successfull"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="NWDAF DISCOVERY SUCCESSFULL"
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance Name:   NWDAF-1
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance Type:   NWDAF
Jul 10 13:20:55 ubuntu CNC-API[1330]: Instance IP:     127.0.0.1
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="NWDAF discovery successfull"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="NF SUBSCRIPTION ID ==> map[bcc63bd8-986e-4d79-ae3b-6ee2450e98a1:00101-42797]"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="INFO-CNC SUBSCRIPTION UPDATE  SUCCESS =>201"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="upfinfo not deleted no such event =====>NF_PROFILE_CHANGED"
Jul 10 13:20:55 ubuntu CNC-API[1330]: v5.6.2=info msg="CNC UPF UPDATE  SUCCESS =>200"
Jul 10 13:20:58 ubuntu CNC-API[1330]: v5.6.2=info msg="NOTIFICATION EVENT: NF_REGISTERED NEW UPF ID 6d1e8053-3e3c-4c8f-9d7e-e7470b98ab0d"
Jul 10 13:20:58 ubuntu CNC-API[1330]: v5.6.2=info msg="upfinfo not deleted no such event =====>NF_PROFILE_CHANGED"
Jul 10 13:20:58 ubuntu CNC-API[1330]: v5.6.2=info msg="CNC UPF UPDATE  SUCCESS =>200"
Jul 10 13:21:00 ubuntu CNC-API[1330]: v5.6.2=warning msg="FAILED TO DELETE PDU SESS=>NO MATCH FOUND [0] [204]"
Jul 10 13:21:20 ubuntu CNC-API[1330]: v5.6.2=warning msg="FAILED TO DELETE GNB WITH ID [1-2-3] STATUS: [404]"
Jul 10 13:21:20 ubuntu CNC-API[1330]: v5.6.2=info msg="INFO-REMOVED ENTRIES: 0"
```

Figure 3: Runtime logs of the Cumucore CNC

```
-- Boot 320731407a584583bdfc6aa78a29c373 --
Jul 10 13:20:50 ubuntu systemd[1]: Started Network Exposure Function.
Jul 10 13:20:51 ubuntu NEF[1107]: DEBUG APIClient initialized
Jul 10 13:20:51 ubuntu NEF[1107]: DEBUG Swagger UI embedded and available at http://192.168.100.30:20080/swagger/
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ================================================================================
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Database Connection Established (MongoDB)!
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ================================================================================
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Network Function     : Network Exposure Function (NEF)
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Status               : Initializing server...
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Sbi Listening on     : 127.0.0.1:20035
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Nbi Listening on     : 192.168.100.30:20080
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Start Time           : 2026-07-10T13:20:51Z
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Version              : 5.6
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ PLMN-IDs             : [{"mcc":"001","mnc":"01"}]
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Services             : ["3gpp-analyticsexposure","3gpp-as-session-with-qos"]
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Redundancy Mode      : true
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ✔ Nbi API TLS Enabled  : false
Jul 10 13:20:51 ubuntu NEF[1107]: DEBUG ✔ SBI pprof           : https://127.0.0.1:20035/debug/pprof/
Jul 10 13:20:51 ubuntu NEF[1107]: INFO ================================================================================
Jul 10 13:20:51 ubuntu NEF[1107]: INFO [put_req]: Profile registration request
Jul 10 13:20:51 ubuntu NEF[1107]: INFO [put_resp]: Profile registration successful, response statusCode = [201 Created], received hbt-timer = 83secs
Jul 10 13:20:51 ubuntu NEF[1107]: INFO [subscr_req]: Create profile_status subscription request, subscrCond = {"nfSetId":"set01.pcfset.5gc.mnc001.mcc001"}
Jul 10 13:20:51 ubuntu NEF[1107]: INFO [subscr_resp]: Profile_Status subscription successfuly created, subscriptionId = 00101-9319, validityTime = 2027-07-10T13:20:51Z
Jul 10 13:20:53 ubuntu NEF[1107]: INFO [subscr_req]: Create profile_status subscription request, subscrCond = {"nfSetId":"set01.custom_cncset.5gc.mnc001.mcc001"}
Jul 10 13:20:53 ubuntu NEF[1107]: INFO [subscr_resp]: Profile_Status subscription successfuly created, subscriptionId = 00101-50807, validityTime = 2027-07-10T13:20:53Z
Jul 10 13:20:55 ubuntu NEF[1107]: INFO [subscr_req]: Create profile_status subscription request, subscrCond = {"nfSetId":"set01.nwdafset.5gc.mnc001.mcc001"}
Jul 10 13:20:55 ubuntu NEF[1107]: INFO [subscr_resp]: Profile_Status subscription successfuly created, subscriptionId = 00101-54155, validityTime = 2027-07-10T13:20:55Z
Jul 10 13:20:56 ubuntu NEF[1107]: DEBUG Created new NF type bucket: CUSTOM_CNC
Jul 10 13:20:56 ubuntu NEF[1107]: DEBUG Storing new discovery item: CUSTOM_CNC-1 (CUSTOM_CNC)
Jul 10 13:20:56 ubuntu NEF[1107]: INFO [notify_event]: CUSTOM_CNC-1 Profile Status Notification Received, event = NF_REGISTERED
Jul 10 13:20:56 ubuntu NEF[1107]: INFO         param(s): nfSetIdList = [set01.custom_cncset.5gc.mnc001.mcc001], subscriptionId = 00101-50807, conditionEvent = NF_ADDED
Jul 10 13:22:14 ubuntu NEF[1107]: DEBUG Heartbeat sent successfully
Jul 10 13:23:37 ubuntu NEF[1107]: DEBUG Heartbeat sent successfully
Jul 10 13:25:00 ubuntu NEF[1107]: DEBUG Heartbeat sent successfully
Jul 10 13:26:23 ubuntu NEF[1107]: DEBUG Heartbeat sent successfully
```

Figure 4: Runtime logs of the Cumucore NEF

Figure 5: Status of the deployed CNC service

Figure 6: Status of the deployed NEF service

2.1.3 Open5GS Core

In addition to the commercial Cumucore platform, SAFE-6G leverages an extended Open5GS deployment to provide an open-source implementation of the 5G Core with support for secure network exposure. The Open5GS platform has been enhanced with project-specific Network Exposure Function (NEF) capabilities, enabling the controlled exposure of standardized northbound APIs to authorized external Application Functions (AFs). These APIs allow external components of the SAFE-6G framework to access selected network capabilities while preserving compliance with the 3GPP Service-Based Architecture (SBA) principles. As the details of the Open5GS 5G Core and the developed NEF APIs are presented in Deliverable D3.3, this section focuses on the exposed APIs, their supported endpoints and their role in enabling infrastructure openness, monitoring and programmability within the SAFE-6G framework.

2.1.3.1 NEF APIs

2.1.3.1.1 NEF Monitoring Event API

The Monitoring Event API enables the secure exposure of network monitoring capabilities through the Open5GS Network Exposure Function (NEF). Within SAFE-6G, the API provides a standardized interface that allows authorized external applications to consume monitoring information from the 5G Core. The main capabilities of the Monitoring Event API include:

- Secure exposure of network monitoring information through the NEF.
- Subscription to monitoring events generated by the 5G Core.
- Retrieval of User Equipment (UE) location-related information.
- Delivery of monitoring notifications to authorized external Application Functions (AFs).
- Standards-compliant implementation based on the 3GPP Monitoring Event API specification.

The implementation details and operational workflow of the Monitoring Event API are presented in Deliverable D3.3 and in GitLab. Table 5 summarizes the endpoints exposed by the API together with their supported HTTP methods and functionality.

API Prefix: /3gpp-monitoring-event/v1/		
Endpoint	***Method/s***	***Description***
{scsAsId}/subscriptions	POST	Create a MonitoringEvent subscription
{scsAsId}/subscriptions/	GET	Read all of the active subscriptions for the AF
{scsAsId}/subscriptions/{subscriptionId}	GET	Read an active subscription for the AF and the subscription Id
{scsAsId}/subscriptions/{subscriptionId}	DELETE	Deletes an already existing subscription

Table 5: NEF Monitoring Event API supported by Open5GS

2.1.3.1.2 NEF AsSessionWithQos API

The AsSessionWithQoS API enables the controlled exposure of Quality of Service (QoS) management capabilities through the Open5GS Network Exposure Function (NEF). Within SAFE-6G, the API provides a standardized interface for authorized external applications to request and manage QoS-aware communication services.

The main capabilities of the AsSessionWithQoS API include:

- Secure exposure of QoS management capabilities through the NEF.
- Establishment and management of application sessions with QoS requirements.
- Request and modification of QoS-related session parameters.
- Support for QoS-aware service provisioning by authorized external Application Functions.
- Standards-compliant implementation based on the 3GPP AsSessionWithQoS API specification.

The implementation details and operational workflow of the AsSessionWithQoS API are presented in Deliverable D3.3 and in GitLab. Table 6 summarizes the endpoints exposed by the API together with their supported HTTP methods and functionality.

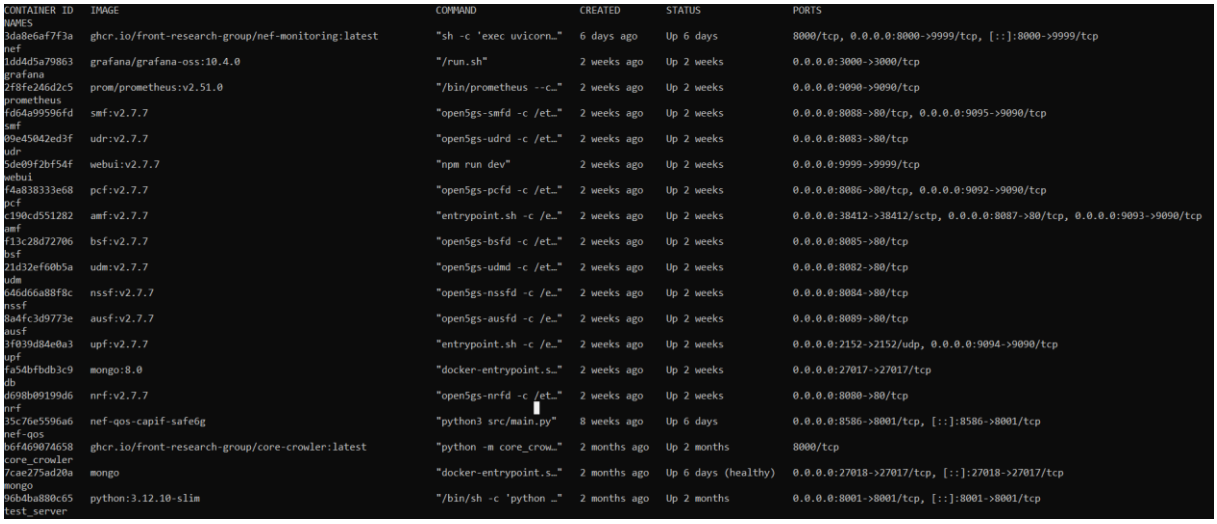
API Prefix: /3gpp-as-session-with-qos/v1/		
Endpoint	***Method/s***	***Description***
{scsAsId}/subscriptions	GET	Retrieve all active QoS subscriptions for a specific SCS/AS.
{scsAsId}/subscriptions	POST	Create a new QoS subscription resource for a specific SCS/AS.
{scsAsId}/subscriptions/{subscriptionId}	GET	Retrieve one active QoS subscription using the SCS/AS ID and subscription ID.
{scsAsId}/subscriptions/{subscriptionId}	DELETE	Delete an existing QoS subscription using the SCS/AS ID and subscription ID.

Table 6: NEF AsSessionWithQoS API supported by Open5GS

2.1.4 Deployment

The Open5GS deployment was realized as a containerized environment using Docker, allowing the different network functions and supporting services to be deployed and managed independently. Figure 7 presents the running Docker containers that compose the SAFE-6G Open5GS testbed at NCSRD. Besides the core Open5GS network functions, the deployment includes the monitoring

infrastructure (Prometheus and Grafana), MongoDB for data persistence, the SAFE-6G NEF extension implementing the Monitoring Event and Application Session with QoS APIs, the Core Controller and auxiliary services required for experimentation. This containerized architecture facilitates modular deployment, simplified management and reproducible experimentation.



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
3da8e6af7f3a	ghcr.io/front-research-group/nef-monitoring:latest	"sh -c 'exec uvicorn…"	6 days ago	Up 6 days	8000/tcp, 0.0.0.0:8000->9999/tcp, [::]:8000->9999/tcp	nef
1dd4d5a79863	grafana/grafana-oss:10.4.0	"/run.sh"	2 weeks ago	Up 2 weeks	0.0.0.0:3000->3000/tcp	grafana
2f8fe246d2c5	prom/prometheus:v2.51.0	"/bin/prometheus --c…"	2 weeks ago	Up 2 weeks	0.0.0.0:9090->9090/tcp	prometheus
fd64a99596fd	smf:v2.7.7	"open5gs-smfd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8088->80/tcp, 0.0.0.0:9095->9090/tcp	smf
09e45042ed3f	udr:v2.7.7	"open5gs-udrd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8083->80/tcp	udr
5de09f2bf54f	webui:v2.7.7	"npm run dev"	2 weeks ago	Up 2 weeks	0.0.0.0:9999->9999/tcp	webui
f4a838333e68	pcf:v2.7.7	"open5gs-pcfd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8086->80/tcp, 0.0.0.0:9092->9090/tcp	pcf
c190cd551282	amf:v2.7.7	"entrypoint.sh -c /e…"	2 weeks ago	Up 2 weeks	0.0.0.0:38412->38412/sctp, 0.0.0.0:8087->80/tcp, 0.0.0.0:9093->9090/tcp	amf
f13c28d72706	bsf:v2.7.7	"open5gs-bsfd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8085->80/tcp	bsf
21d32ef60b5a	udm:v2.7.7	"open5gs-udmd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8082->80/tcp	udm
646d66a88f8c	nssf:v2.7.7	"open5gs-nssfd -c /e…"	2 weeks ago	Up 2 weeks	0.0.0.0:8084->80/tcp	nssf
8a4fc3d9773e	ausf:v2.7.7	"open5gs-ausfd -c /e…"	2 weeks ago	Up 2 weeks	0.0.0.0:8089->80/tcp	ausf
3f039d84e0a3	upf:v2.7.7	"entrypoint.sh -c /e…"	2 weeks ago	Up 2 weeks	0.0.0.0:2152->2152/udp, 0.0.0.0:9094->9090/tcp	upf
fa54bfbdb3c9	mongo:8.0	"docker-entrypoint.s…"	2 weeks ago	Up 2 weeks	0.0.0.0:27017->27017/tcp	db
d698b09199d6	nrf:v2.7.7	"open5gs-nrfd -c /et…"	2 weeks ago	Up 2 weeks	0.0.0.0:8080->80/tcp	nrf
35c76e5596a6	nef-qos-capif-safe6g	"python3 src/main.py"	8 weeks ago	Up 6 days	0.0.0.0:8586->8001/tcp, [::]:8586->8001/tcp	nef-qos
b6f469074658	ghcr.io/front-research-group/core-crowler:latest	"python -m core_crow…"	2 months ago	Up 2 months	8000/tcp	core_crowler
7cae275ad20a	mongo	"docker-entrypoint.s…"	2 months ago	Up 6 days (healthy)	0.0.0.0:27018->27017/tcp, [::]:27018->27017/tcp	mongo
96b4ba880c65	python:3.12.10-slim	"/bin/sh -c 'python …"	2 months ago	Up 2 months	0.0.0.0:8001->8001/tcp, [::]:8001->8001/tcp	test_server

Figure 7: Docker containers composing the Open5GS-based SAFE-6G deployment at the NCSRD testbed along with the APIs

2.2 Edge-Cloud Continuum APIs

The edge-cloud continuum APIs allow the SAFE-6G framework and more concretely, its Trust Functions to gather contextual, real-time information from the state of the distributed computing ecosystem as well as managing the lifecycle of the services deployed on top of it. With the three family of APIs that are part of this layer, the continuum can thus be observed and managed by authorized systems or users, lowering the barriers related to the fact of having the computing resources delocalized (i.e., distributed on different geographical locations) or making use of various containerization technologies:

- **Context Broker APIs**: Part of the leveraged Meta-Operating System (an updated version of aeriOS Meta-OS), it offers contextual information of the managed distributed computing continuum for several kinds of entities (based on a continuum data model[1]): Domain, Infrastructure element, Service, ServiceComponent, Low-Level Orchestrator, Power Source, etc. This information can be fed to TF's models for real-time actuation or persisted to be used for their training.
- **Orchestration/HLO API:** Also part of the Meta-OS, it enables the lifecycle management of services on top of the managed computing resources (which can be communicated among domains/clusters either through exposed interfaces or direct communication for specific cases thanks to the service mesh implementation). With these APIs, TF agents can deploy different vApp/nApp flavors, upscale or downscale resources, upgrade or downgrade versions or change configurations, among other possibilities. In addition, components from the SAFE-6G framework itself (e.g., CoCo, TF agents) can also be managed by it.

[1] aeriOS data model documentation: https://gitlab.com/safe-6g/development/continuum/-/tree/main/documentation/datamodel?ref_type=heads

- **DataOps API**: Parallel to the Meta-OS implementation, the DataOps APIs enables the retrieval of stored metrics, not only from the continuum or the Meta-OS, but also from the other underlying layers – network (e.g., 5G/6G cores, gNB performance), applications and services (e.g., Metaverse use cases like those from the project).

The specific requirements that have been considered for the components' implementation can be found in D3.2. Specifically for the APIs, their implementation has been guided by the following ones:

- **REQ-COM-ETH-NF-M-03**: Compliance with 5G and industry standards (use of API REST and standard Open API descriptors).
- **REQ-COM-DES-F-M-04**: Following Cloud Native paradigm (virtualized, ready for K8s).
- **REQ-COM-RES-F-M-07**: Data availability (via DataOps and Context broker APIs).
- **REQ-COM-DES-F-M-08**: APIs interaction through Open CAPIF (APIs exposed through it).
- **REQ-COM-DES-F-M-19**: Access to performance metrics from 5G/6G core infrastructure (collected by dedicated exporters; exposed via DataOps).
- **REQ-COM-DES-F-M-20**: Access to performance metrics from the cloud continuum (direct from the Context Broker, extended by dedicated exporters; exposed via DataOps).
- **REQ-COM-DES-F-M-21**: Access to performance metrics from user application (collected by dedicated exporters; exposed via DataOps).
- **REQ-INFRA-META-NF-M-04**: Exposure capabilities of the meta-operating system (two dedicated Meta-OS APIs).
- **REQ-INFRA-META-NF-M-05**: Data accessibility across the computing continuum (APIs are exposed to the outside of the managed domain, accessible by authorized services/users).

2.2.1 Description and Analysis

In the following table, the three APIs are presented. Since they are exposed via a common API Gateway, they are unified under a common prefix. CAPIF has been used as a prefix due to the reasons explained in D3.2 – Section 2.1.4.

API Prefix: /capif/		
Endpoint	**Method/s**	**Description**
entities	GET	Returns the entities' metadata/attributes managed by the Context Broker. Filters can be applied
entities/{entity}	GET	Returns the attributes' metadata/metrics related to a specific entity of the broker. Filters can be applied
subscriptions	GET/POST	Check/create subscriptions in the broker, to receive a notification upon a change in a given entity/attribute
subscriptions/ {subscription_id}	DELETE	Remove a specific subscription from the Context broker
hlo_fe/services/ {service_id}	POST/ PATCH/ PUT/ GET/ DELETE	Deploys, upgrades, reallocates, removes or provides information related to a managed service. Reallocation (PUT) deploys/reuses a previously deallocated (DELETE) service
hlo_fe/services/ {service_id}/component/{service_component_id}	PATCH	Used for updating a single service component (e.g., if a service is composed by 2 Helm charts, it only updates one of them)

hlo_fe/services/{service_id}/ purge	DELETE	Completely removes the service metadata from the Context broker (only enabled if not running)
/thanos/api/v1/labels	GET	Returns the list of metrics' labels managed by the DataOps
/thanos/api/v1/query	GET	Evaluates a PromQL expression, in a given point in time (instant query)
/thanos/api/v1/ query_range	GET	Evaluates a PromQL expression, over a period of time (range query)
/thanos/api/v1/label/ {{label_name}}/values	GET	Gets the metrics associated to a given label

Table 7: Edge-Cloud Continuum APIs

Additional information and supporting tools can be found in the following links:

- Meta-OS endpoints: https://gitlab.com/safe-6g/development/continuum/-/tree/main/documentation/api/openapi?ref_type=heads
- DataOps endpoints: https://gitlab.com/safe-6g/development/dataops/monitoring/-/blob/main/docs/api.md?ref_type=heads

The following diagram describes the enabled flows to interact with the Meta-OS endpoints. The main one is through OpenCAPIF. API invokers with a token granted by the CAPIF core can make requests to the aeriOS CAPIF API Gateway, which validates the token with the CAPIF core before processing the received request (forwarding it towards the orchestration API, the Context Broker API or the DataOps API). This flow is valid for external services, yet for serving other internal aeriOS services, CAPIF is not involved (a token is shared among the aeriOS services, validated via its native Identity management system). A Kubernetes ingress is in charge of balancing the traffic by checking the presence of the prefix "/capif" in the received requests.

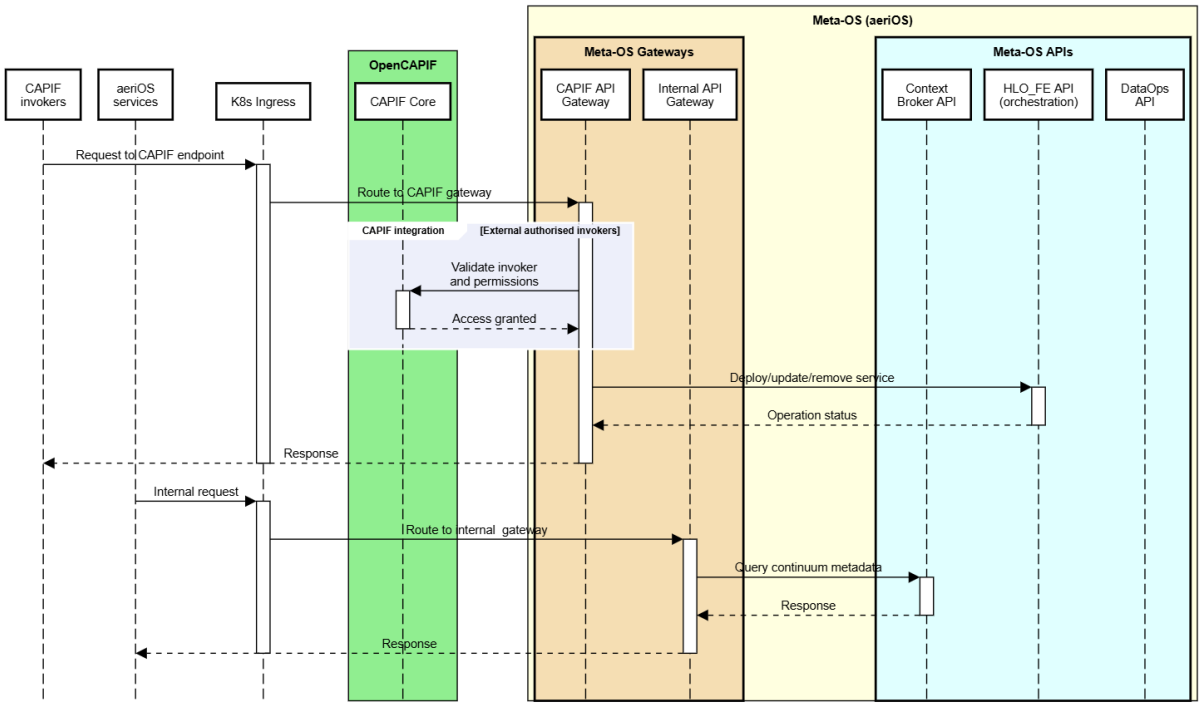


Figure 8: Edge-Cloud continuum API Gateway flows

2.2.2 DEPLOYMENT

The 3 API families are exposed through the Meta-OS' API Gateway and therefore are available in the 3 domains considered in SAFE-6G: UPV, NCSRD & UNIWA; except for the DataOps endpoints (/thanos/…), as this component is only available in UNIWA. Because of the federated nature of the Meta-OS, the endpoints of both the Context broker (/entities/…, /subscriptions/) and the Orchestration (/hlo_fe/…) can be operated equally regardless of the gateway consumed.

Figure 9 presents snapshots of the Meta-OS API Gateway deployments across the three SAFE-6G domains: UPV (top), NCSRD (middle), and UNIWA (bottom).

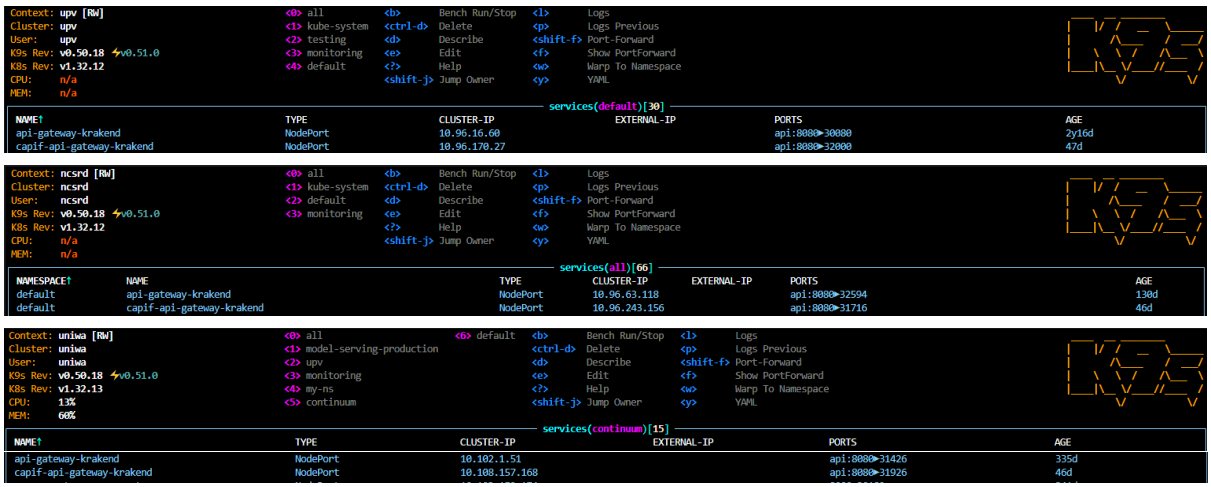


Figure 9: Snapshots of the Meta-OS' API Gateway deployed on each domain

2.3 METAVERSE APPLICATION APIS

The Metaverse manager exposes several APIs to let other SAFE-6G components get data from the Unity UC applications and trigger effects on them. This allows for instance TFs to directly impact the end user application in addition to their actions targeting the network. The Metaverse APIs are regrouped into three categories: chatbot-related APIs, system APIs and UC-specific APIs.

While Metaverse APIs are reported below, the final version of the Metaverse manager and Unity UC applications are reported in D3.6.

Chatbot APIs are simple endpoints allowing to add a textual or image message to the virtual Chatbot frontend on the Unity CloudXR client side. There are useful to provide additional information to the end users, for instance for xAI purposes. **System APIs** regroup generic services useful for both use-cases. They include ways of managing end-users (seeing connected users, blacklisting/whitelisting them etc.) in addition to ways of getting real-time metrics from XR headsets. Finally, two **use-case specific APIs** are available to perform actions that only concern the factory DT scene or the formation room scene. They respectfully allow to enable/disable factory DT updates for UC1 and to suggest stopping sharing the view of the XR headset for UC2.

timestamp_iso	timestamp	framerate	GPU_percent_usage	GPU_frequency	CPU_percent_usage	available_memory_MB	battery_percent_level	battery_temperature	power_wattage	app_pss	screen_tear_...
2026-05-26T12:10:07.2...	17797972370...	60	64	545	81	2437	54	45	1764	1348	0
2026-05-26T12:10:08.2...	0	0	0	0	0	0	0	0	0	0	0
2026-05-26T12:10:09.3...	17797974110...	62	22	285	43	2981	54	41	8961	1351	4
2026-05-26T12:10:10.6...	17797974120...	58	70	285	32	2981	54	41	8639	1351	0
2026-05-26T12:10:12.0...	17797974141...	57	82	285	25	2795	54	40	8100	1394	0
2026-05-26T12:10:14.1...	17797974150...	64	78	285	33	2795	54	40	5934	1394	0
2026-05-26T12:10:15.4...	17797974170...	41	54	285	20	2795	54	40	6161	1394	0
2026-05-26T12:10:16.1...	17797974180...	72	52	285	26	2795	54	40	6556	1394	0

Figure 10: Example of QoS metrics retrieved from an XR headset through a monitor-perf API subscription. Overall, a total of 92 metrics can be retrieved through the integration of the OVR metrics tool plugin with the CloudXR client Unity app

The specific development and implementation activities related to the Metaverse components have been guided by the requirements stated in D2.3 and reported in D3.6. More specifically, the Metaverse API contribute to the following:

- **REQ-COM-ETH-NF-M-03**: Compliance with 5G and industry standards (use of API REST and standard Open API descriptors).
- **REQ-COM-RES-F-M-07**: Data availability (via Metaverse manager API).
- **REQ-COM-DES-F-M-08**: APIs interaction through Open CAPIF (API exposed through it).
- **REQ-COM-DES-F-M-21**: Access to performance metrics from user application.
- **REQ-USER-CATEGORY-Both-NF-O-4**: Understandable SAFE-6G system decisions for novice end-users (high-level answers and additional content that can be transmitted to end-users through the *chatbot* Metaverse APIs).
- **REQ-USER-UC1-F-R-2**: Role-based access to factory data (Metaverse manager end-user management, *system* Metaverse APIs related to connected users and blacklisting).
- **REQ-USER-UC2-F-R-2**: Control of privacy about shared data (the *HMD-view-notif API* allow to suggest end users to enable or disable their shared video feed with other users).

2.3.1 Description and Analysis

The Metaverse Manager exposes a set of APIs that enable authorized SAFE-6G components to interact with the Metaverse platform in a standardized manner. These APIs allow external components to retrieve application information, monitor the state of the running use cases and trigger actions within the virtual environment. While the implementation details of the Metaverse platform are presented in Deliverable D3.6, this section focuses on the APIs exposed by the platform and their role in enabling interaction with the SAFE-6G framework.

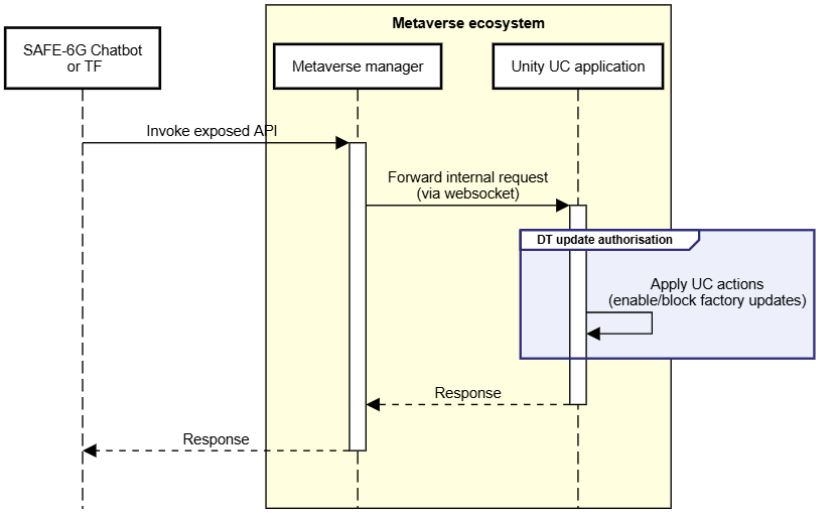


Figure 11: Example of flow diagram for the DT update authorization Metaverse API

The available endpoints exposed by the Metaverse Manager APIs are summarized in Table 8.

API Prefix: /safe6g-imm/v1		
Endpoint	**Method/s**	**Description**
/chatbot/new-bot-txt-msg	POST	Add a new txt msg from the chatbot. It will appear in the XR chatbot window in the Unity client app
/chatbot/new-bot-img-msg	POST	Add a new image msg from the chatbot. It will appear in the XR chatbot window in the Unity client app
/system/change-netcode-lobby-status	POST	Enable/Disable the Netcode lobby, which handles the synchronisation of Unity scenes between the connected users. When disabled, users are isolated from each other and can only work locally
/system/list-users	GET	Get the list of users who are connected to the Metaverse application
/system/get-blacklisted	GET	Get the list of users who are currently blacklisted.
/system/monitor-connections	POST	Request to be notified each time a user tries to connect to the metaverse app
/system/blacklist-user	POST	Blacklist or whitelist a given user, preventing him/her from accessing the shared XR scenes in case of suspicious activity or security issue. Blacklisted users are thus isolated from other users and cannot access the metaverse app features until being whitelisted.
/system/monitor-perf	POST	Request to be periodically notified of the real-time performance metrics from the XR headsets. Useful for real-time monitoring of the end-user application, for instance for the Reliability TF.
/v1/uc1/dt-update-authorization	POST	Authorize/Forbid the users to push updates on the factory DT. When DT updates are forbidden, users can only work on their local versions of the production lines models and cannot make any change on the factory DT. The main usage is to protect the factory DT in case of suspicious activity.
/uc2/hmd-view-notif	POST	Propose a user to start/stop sharing the view of the XR headset. This is useful in case of temporary QoS degradations.

Table 8: Endpoints exposed by the Metaverse Manager APIs

2.3.2 DEPLOYMENT

The Metaverse APIs are exposed by the Metaverse manager, which is deployed at NCSRD premises through a Kubernetes cluster. The Metaverse manager follows the OpenCAPIF API provider process to make the Metaverse API discoverable for the other SAFE-6G components.

Figure 12: Overview of the Metaverse manager deployment on NCSRD K8s cluster

3 OpenCAPIF

3.1 Introduction

CAPIF stands for Common API Framework. It is a 3GPP-specified framework for managing the interaction among API providers and API invokers. Its key components for interacting with external applications are API providers, API invokers and CAPIF Core Function.

API providers provide services to other applications through their APIs while API invokers are applications that consume those APIs. The CAPIF Core Function is the actual API manager that manages the onboarding, publishing, discovery, authentication and authorization mechanisms among API providers and invokers. CAPIF is described thoroughly on the 3GPP specifications TS 23.222[2], TS 29.222[3], and TS 33.122[4].

The open source implementation of the CAPIF, OpenCAPIF, is based on those 3GPP specifications and acts as the main entity while referring to CAPIF's actual implementation. The specific development and implementation activities related to the OpenCAPIF have been guided by the following requirements stated in D2.1:

- **REQ-COM-DES-F-M-08**: APIs interaction through Open CAPIF.
- **REQ-COM-DES-F-M-10**: CAPIF registration to interact with other SAFE-6G components in the framework.
- **REQ-COM-DES-F-M-17**: API Consumption Requirement.

SAFE-6G leverages the official open-source implementation of OpenCAPIF, which serves as the foundation for the API exposure framework used throughout the project. The public repository adopted in SAFE-6G is presented below.

Category	***Status***
Version	1.0.1
Documentation	https://ocf.etsi.org/documentation/latest/
Source code	https://labs.etsi.org/rep/ocf
License	Apache 2.0

Table 9: OpenCAPIF final release summary

3.2 OpenCAPIF Scope in SAFE-6G

Within SAFE-6G, OpenCAPIF serves as the common API exposure framework, providing a trusted and standardized mechanism for API discovery, registration, authentication, authorization and monitoring. APIs that need to be securely accessed by external applications, such as the Trust Functions and other authorized Application Functions (AFs), are exposed through OpenCAPIF, enabling controlled access to capabilities offered by the 5G Core, the Edge-Cloud Continuum and the application layer. In contrast, APIs intended solely for communication between internal SAFE-6G components remain private and are accessed directly without being exposed through CAPIF.

[2] https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3337
[3] https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3450
[4] https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3420

3.2.1 ARCHITECTURE

A sequence diagram that corresponds to the operations that need to be performed in the context of an API provider or API invoker onboarding and thus authorization of their communication is presented in Figure 13.

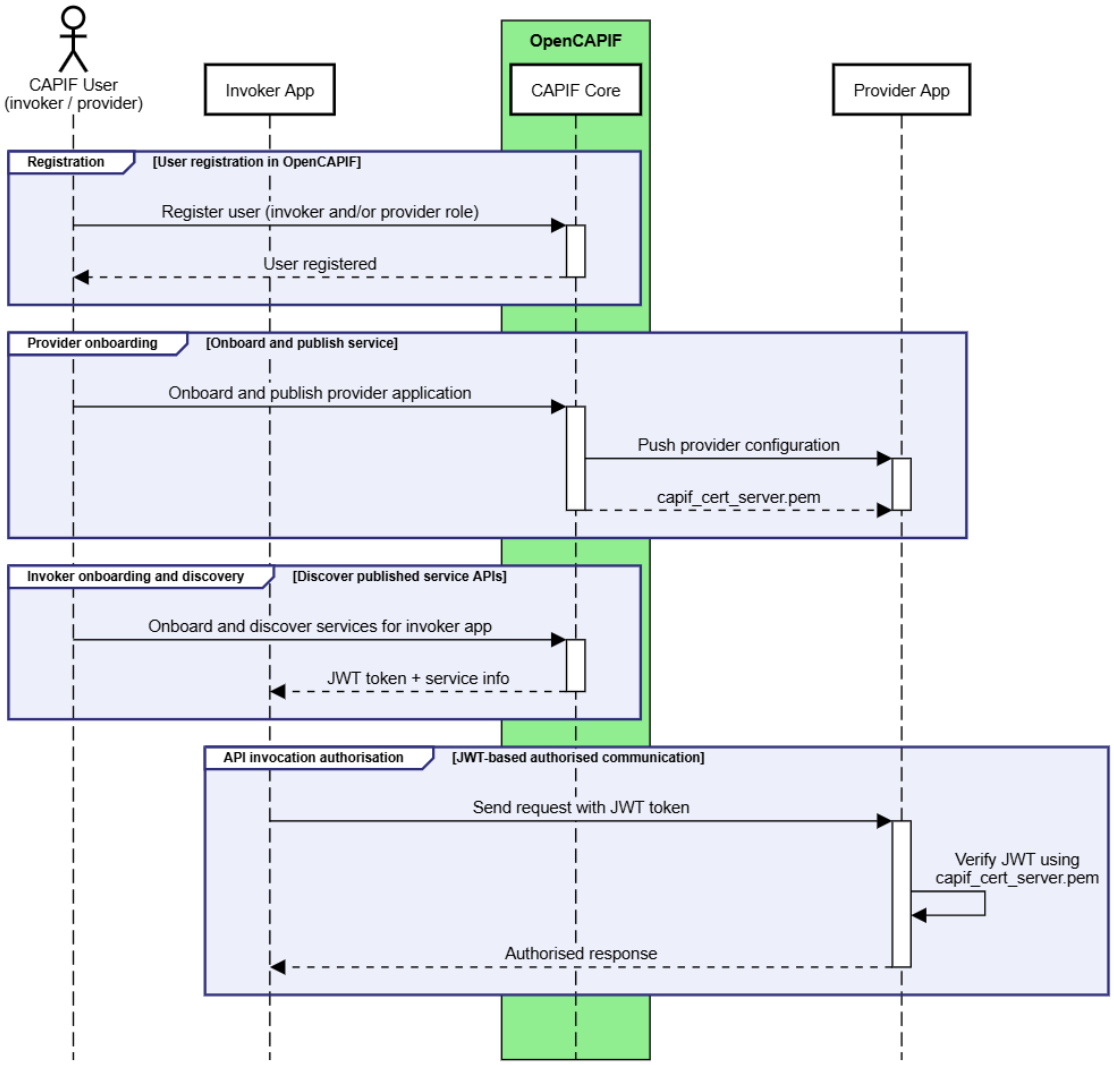


Figure 13: CAPIF Architecture Sequence Diagram

The procedure is as follows:

1. A user registration is performed within OpenCAPIF to have a CAPIF user that can act either as CAPIF Provider or CAPIF Invoker.
2. When the user registration is completed, the onboarding and publishing of the provider application take place. In that step, CAPIF sends back to the provider application the capif_cert_server.pem when publishing is completed while provider app should use it, to authorize the incoming requests from external applications.
3. As a next step, the onboarding of the invoker should be done alongside the discovery of the published services through CAPIF. By finishing the discovery of the published APIs from the CAPIF, a JSON Web Token (JWT) is sent back to the invoker to use it on future requests to be authorized from provider applications like any 3GPP native API or Metaverse Manager.

4. Lastly, an invoker (external application) performs a HTTP request with the JWT token that was retrieved earlier integrated in its HTTP Request Headers to be authorized successfully from provider.

3.3 Workflow Tools

This section presents the workflow tools accompanying the OpenCAPIF framework. The toolkit comprises a set of configuration files and automation scripts that support the complete API lifecycle, including user registration, provider and invoker onboarding, API publication and discovery, and the provisioning of the certificates and access tokens required for secure API consumption.

3.3.1 Configuration Files:

- <provider/invoker>_config_sample.json
  - Input for onboarding scripts (provider/invoker). Configuration options for connection to the CAPIF, certificate generation, provider folder option for storing created certificates.
- capif_sdk_register.json:
  - Input for user registration like connection to CAPIF, CAPIF's admin credentials and credentials for CAPIF's user registration.
- openapi.yaml:
  - The OpenAPI specification for the provider app service (used during service publication).

3.3.2 Scripts:

- provider_capif_connector.py:
  - Onboards the API provider to OpenCAPIF, publishes the provider's services using openapi.yaml, and stores the CAPIF-issued certificate (capif_cert_server.pem) in the directory specified in provider_config_sample.json.
- invoker_capif_connector.py:
  - Onboards the invoker into OpenCAPIF, discovers available services and retrieves a JWT token for consuming provider services that is stored under folder that specified in invoker_config_sample.json as txt.
- user_registration.py:
  - Registers a new user account in the OpenCAPIF Register Service using administrator credentials. The script reads its configuration from capif_sdk_register.json and provisions the CAPIF credentials required for subsequent provider and/or invoker onboarding procedures

3.4 OpenCAPIF Deployment in SAFE-6G Environment

OpenCAPIF has been deployed as part of the SAFE-6G infrastructure to provide the common API exposure framework for the platform. The deployment includes the CAPIF Core together with a MongoDB database used to persist information related to registered users, API providers, API invokers

and published APIs. To facilitate administration and inspection of the stored information, a web-based database management interface (Mongo Express) is also deployed, allowing authorized administrators to browse and manage the CAPIF database through a graphical user interface. Representative screenshots of the deployed OpenCAPIF environment are presented below.

Figure 14 presents representative screenshots of the deployed OpenCAPIF environment, including the Mongo Express interface used to inspect the registered users, API providers and API invokers.

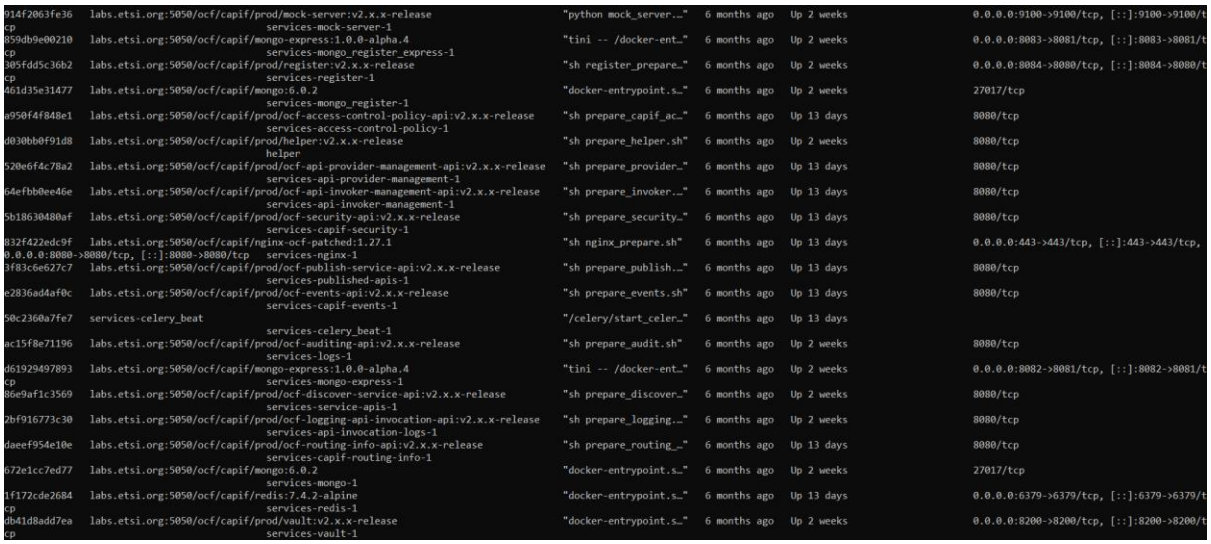


Figure 14: Deployed OpenCAPIF environment

3.5 EXPOSED APIS THROUGH CAPIF IN SAFE-6G

The following API families are exposed through OpenCAPIF to enable secure and standardized access by authorized external Application Functions. Detailed descriptions of these APIs are provided in the previous sections of this deliverable.

- Cumucore NEF APIs
- Open5GS NEF APIs
- Edge-Cloud Continuum APIs
- Metaverse APIs

To the context of CAPIF authentication and authorization mechanism, its API has been implemented authorization/authentication logic to combine the capif_certificate_server.pem that was retrieved during the publishing of its API to CAPIF to authorize incoming requests. This was a key point to the application so to be able to extract its certificate's public key and authorize the incoming JWT tokens that come from requests that have been performed by invokers. From the invokers side, the implementation that has been made is the integration of the onboarding to CAPIF and retrieval of the JWT token after the discovery of the available published API from the CAPIF to the core application of trust functions.

Both the providers and invokers integration to the CAPIF that has been described in the above sections can be found in the GitLab repository that contains all the scripts for the integration.

3.6 O-NEF Module as Part of the OCF Ecosystem

One of the key outcomes of the SAFE-6G project is the development of the Open Network Exposure Function (O-NEF), an extension of the OpenCAPIF ecosystem that bridges standardized API exposure with network-specific capabilities. This extension has been already approved by ETSI SDG OCF as a new module that will be included in the next release of OpenCAPIF. A more detailed reporting of this activity will be analyzed in the upcoming Deliverable D6.6 that discusses relevant standardization activities.

While OpenCAPIF provides a common framework for API discovery, registration, authentication and access control, SAFE-6G extends this functionality by introducing O-NEF as an abstraction layer that simplifies the integration of 5G/6G network services and enables their secure exposure to authorized applications and Trust Functions.

The O-NEF module addresses the need for a unified mechanism through which infrastructure providers can expose programmable network capabilities without requiring service developers to interact directly with heterogeneous network implementations. By adopting a provider-oriented integration model, O-NEF enables different Network Exposure Functions and infrastructure-specific APIs to be incorporated into the OpenCAPIF ecosystem through a common interface, improving interoperability and reducing integration complexity.

Within the SAFE-6G architecture, O-NEF plays a central role in supporting infrastructure openness and programmability by allowing network capabilities to be securely published, discovered and consumed by cognitive components such as the Cognitive Coordinator and the Trust Functions. This facilitates dynamic service orchestration and trust-aware network adaptation while maintaining compliance with standardized API exposure principles. The following sections describe the provider integration pattern adopted by O-NEF and the requirements for implementing compliant providers within the SAFE-6G ecosystem.

3.6.1 Provider Integration Pattern

As seen in Figure 16, the integration of any provider into the OCF module follows the same four-phase pattern defined by the 3GPP CAPIF specification.

- **Onboarding**: The provider registers itself with OpenCAPIF by sending a registration request that carries the security material needed for CAPIF to validate the request. CAPIF returns a provider domain identifier and the certificates the provider will use for subsequent interactions.
- **Publication**: Once registered, the provider publishes its service APIs through the CAPIF publication interface – the API names, endpoints, supported features and security requirements. From this point the APIs are visible in the CAPIF catalog.
- **Discovery**: Consumer applications query CAPIF to find providers that expose the APIs they need. CAPIF returns the service API descriptions along with the security material the consumer will use to authenticate against the provider.

- **Direct invocation**: With endpoints and credentials in hand, the consumer calls the provider directly – typically creating a subscription and registering a webhook callback URL. The provider then emits notifications to that callback when the relevant events occur.

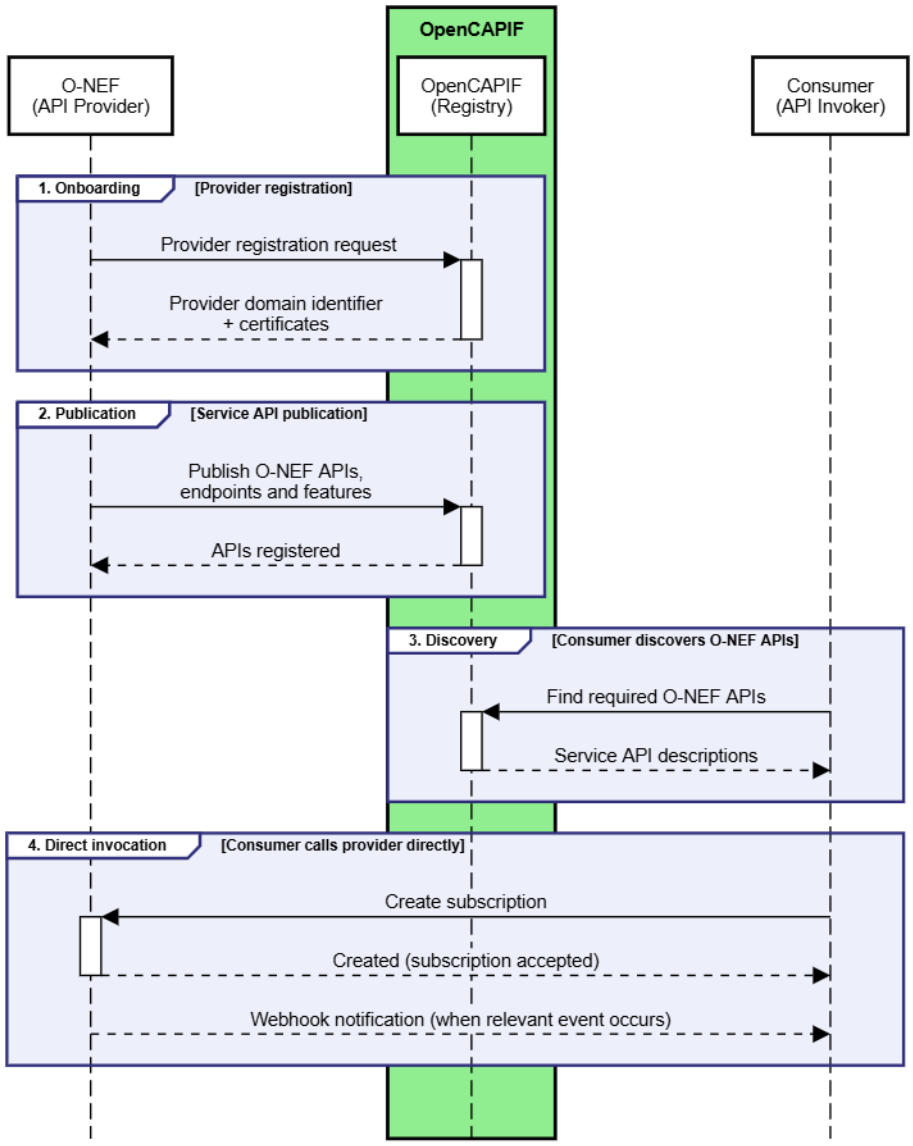


Figure 15: Four-phase provider integration sequence – onboarding, publication, discovery, direct invocation

3.6.2 Compliant Provider

O-NEF is a network function brought into the OCF module. The O-NEF module builds upon an existing open-source implementation of a 5G Network Exposure Function, leveraging previous development experience and extending it with additional capabilities and tighter integration within the SAFE-6G architecture. It exposes three core NEF API surfaces aligned with the 3GPP NEF specifications:

- Monitoring Events, applications can track device-level network events in real time. The main triggers we support are connection and disconnection events – when a UE comes online or drops out – along with mobility and location changes. Notifications are delivered through webhook callbacks that the consumer application registers when it creates a subscription.

- QoS Awareness applications can subscribe to GBR or Non-GBR sessions, set their targets and receive alerts when those targets cannot be met. Reports can be periodic or event-triggered, depending on what the application needs. Useful for real-time streaming, latency-sensitive workloads or any service that wants to react when network conditions change.
- Applications can retrieve UE location at cell granularity – that is, which cell the device is currently connected to. Two patterns are supported: a one-shot query – "give me the location of this device right now" – and a subscription model with continuous updates.

O-NEF is deployed on Kubernetes (K3s) within its own namespace and integrates with the OpenCAPIF instance running alongside it, following exactly the integration pattern shown in Figure 16.

4 CONCLUSION

This document reported the outcomes of Task 3.3, related to the final release of the software assets developed, tailored or updated for enabling infrastructure openness, monitoring and programmability within the SAFE-6G framework. In particular, it presented the APIs exposed by the user-centric 5G Core, the Edge-Cloud Continuum and the Metaverse platform, providing standardized access to network, infrastructure and application capabilities. Furthermore, it described the final integration of OpenCAPIF as the common API exposure framework, together with the CAPIF Exposure Functions (CEF) introduced to simplify the onboarding of API providers while preserving compatibility with the OpenCAPIF ecosystem. Collectively, these developments establish a secure, interoperable and standards-based interface through which the Cognitive Coordinator, the Trust Functions and other authorized applications can discover, access and interact with services across the different SAFE-6G infrastructure domains, providing the openness and programmability mechanisms required by the overall SAFE-6G architecture.